MDynaMix Package
version 5.1
User manual

Alexander Lyubartsev

Division of Physical Chemistry
Stockholm University

12 June 2008

Contents

1 Introduction

MDynaMix is a general purpose molecular dynamics code for simulations of mixtures of rigid or flexible molecules, interacting by AMBER-like force field in a periodic cell. The program is well suited for simulations of flexible molecules based on the double time step algorithm. Alternatively, rigid bonds can be treated by the SHAKE algorithm. Algorithms for NVE, NVT, NPT and anisotropic NPT ensembles are employed, as well as Ewald summation for treatment of the electrostatic interactions. The program can be run both in sequential and parallel execution, in the latter case the MPI parallel environment is required. From v.5.1 possibility for taking into account quantum motion of nuclei using Path Integral approach is implemented. The program is written in standard Fortran-77 and thus can be run on any computer system having a Fortran compiler.

The program is originally based on the MOLDYN program by Aatto Laaksonen, available from the CCP5 program library, Daresbury Lab, UK. Since 1993 many additional changes were made by Alexander Lyubartsev. The first parallelized version of the program (v.4.0) was published in:

A.P.Lyubartsev, A.Laaksonen, "*MDynaMix - a scalable portable parallel MD simulation package for arbitrary molecular mixtures*" Computer Physics Communications, v.128(3), pp.565-589 (2000)

Please use this reference in any work made with the help of this package.

Apart the main MDynaMix block, the package includes the following components:

- **makemol** utility which provides some help in creation of files describing molecular structure and the force field
- **tranal** - a suite of utilities for analyzing trajectories
- **mdee** - a version of the program which implement expanded ensemble method for computation of free energy / chemical potential. MDEE program is not parallelized

1.1 Changes from previous versions

The changes made from v.5.0 to 5.1:

- A module to perform path integral - molecular dynamics simulations is included
- An option for separate barostats in XY-plane and Z-axis
- Correction of a bug in anisotropic constant pressure calculations for rigid molecular models
- Internal reorganization of data structures: arrays for non-bonded interaction types
- Additional utilities and trajectory analysis suite (tranal)

2 Description of the molecular structure and the force field

2.1 The force field

The general form of the force field implemented in the program is:

$$U = U_{LJ} + U_{el} + U_{bond} + U_{ang} + U_{tors} + U_{impr} \tag{1}$$

where

$$U_{LJ} = \sum_{non-bonded} 4\epsilon_{ij}\Big(\big(\frac{\sigma_{ij}}{r_{ij}}\big)^{12} - \big(\frac{\sigma_{ij}}{r_{ij}}\big)^{6}\Big) \tag{2}$$

is the sum of Lennard-Jones interactions taken over non-bonded atom pairs. A pair of atoms is considered as non-bonded if they are on different molecules or if they are on the same molecule but separated by more than two covalent bonds. A case when a pair of atoms is separated by exactly 3 bonds (the so-called 1-4 neighbors) is treated separately, see below. r_{ij} is the distance between atoms i and j.

$$U_{el} = \sum_{non-bonded} \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \tag{3}$$

is the sum of electrostatic interactions

The remaining terms in (1) are inramolecular interactions due to covalent bonds, covalent angles and torsion angles. They are described below.

For each molecule type used in the simulation a file describing molecular structure and parameters of the force field is needed. The file must have extension *.mmol . Some examples of .mmol files are given in `moldb` directory.

.mmol files consist of several parts which are described below. Lines, beginning with "#" are commentaries and they are ignored by the program. Other lines contain parameters in free format.

2.2 Molecular geometry and non-bonded interaction parameters

The first non-commentary line of a `.mmol` file is the number of atoms in the molecule. After it the corresponding number of lines follows, one line per atom. Each line contains 8 compulsory parameters. They are: 1) atom name in the program; 2),3) and 4) are the initial X,Y,Z coordinates of the atom in the molecular coordinate system, 5) mass in atom units, 6) charge, 7) Lennard-Jones parameter σ (effective atom diameter) in Å, 8) Lennard-Jones parameter ϵ in kJ/M. Two optional columns may present, the 9-th column with the chemical types of atoms according to the chosen force field and the 10th, with the atom numbers.

This part is concluded by a reference to the molecular structure/ force field which is read by the program and printed in the output. It consists of the first

line with a number specifying the number of lines for the reference, followed by the corresponding number of lines with the reference itself.

2.3 Bonds

Two types of bonds are implemented, standard harmonic bond (Note: no factor 1/2 in the potential):

$$U_{harm}(r) = k(r - r_0)^2 \tag{4}$$

and Morse potential:

$$U_{morse}(r) = D\big(1 - \exp(-\rho(r - r_0))\big)^2 \tag{5}$$

Harmonic bonds may be also used for description of the so-called Urey-Bradley term which is a harmonic potential between the 1-3 neighbors (atoms separated by two covalent bonds). This term is included for example into CHARMM force field as a part of interaction potential for covalent angles.

The first line of the bond section of a `.mmol` file is the number of bonds. Then lines, one per each bond, follow with parameters for each bond. Each line consists of the following fields:

- The first is the type of the bond potential. Available values are 0 (harmonic bond), 1 (Morse potential) and 2 (Urey-Bradley term). In the case of the Urey-Bradley term, the interaction is calculated exactly as in the case of a harmonic bond (type 0), but the corresponding bond is not accounted for definition of 1-3 and 1-4 neighbors. It is also not accounted for in constraint dynamics (that is, constraints are not applied for these bonds)
- The second and third parameters are atom numbers which are linked by the bond.
- The fourth field is the equilibrium bond length r_0 in Å. It may be set negative, in such a case the equilibrium bond length will be taken from the corresponding coordinates in the fisrt section of the `.mmol` file.
- The 5-th parameter is the force constant k in $kJ/mol/\mathring{A}^2$
- The 6-th and 7-th parameters are used only for the Morse-type bonds and define the D (kJ/mol) and ρ ($\mathring{A}^{-1}$) parameters of the Morse potential respectively

2.4 Covalent angles

Covalent angles are described by harmonic potential of the angle:

$$U_{ang}(\theta) = k_\theta(\theta - \theta_0)^2 \tag{6}$$

where θ_0 is the equilibrium angle and k_θ is the force constant.

The first line specifies the number of angles.

Next lines give parameters of angles. The fields of each line have the follow meanings:

1),2) and 3) define atom numbers

4) is the equilibrium angle in degrees

5) is the force constant in kJ/mol/rad**2

Note. The CHARMM force field has the so-called Urey-Bradley term which is a harmonic interaction between 1-3 neighbors. This term can be introduced in the list of bonds.

2.5 Dihedral angles

.

In this section, parameters for dihedral angles are introduced for the standard type of torsion potential:

$$U_{tors}(\phi) = K_\phi(1 + \cos(M\phi - \Delta)) \tag{7}$$

Torsional angle ϕ is defined as having value 180^o for a trans-conformation.

Other types of dihedral angle potential are supported, they can be introduced in other, optional, sections.

The first line of this section specifies the number of torsion angles of this type. Lines describing torsion potential have the following meaning for the fields:

1)-4) define atoms for the torsion angle

5) is parameter Δ in degrees

6) is the force constant K_ϕ in kJ/mol

7) is the multiplicity factor M

Multiple torsions are supported. Any torsion angle can be repeated any number of times with different parameters, and all they will be summed up in the calculations of forces and energies.

2.6 Optional features of the force field

Description of the optional features of the force field begins with the line defining the type of additional potential features. These are:

- `tors1`

  describes torsion angles in terms of MM3 potential:

  $$U_{tors}(\phi) = K_1(1 + \cos\phi)/2 + K_2(1 - \cos 2\phi)/2 + K_3(1 + \cos 3\phi)/2 \tag{8}$$

  The first line of this section contains the number of torsions of this type followed by lines in which fields 1)-4) define atoms and fields 5)-7) constants K_1, K_2 and K_3 in kJ/M for each torsion angle of this type.

- `tors5`

  describes torsion angles given by the Ryckaert-Bellemans potential:

$$U_{tors}(\phi) = \sum_{i=1}^{5} K_i \cos^i(\phi - 180) \tag{9}$$

  The first line of this section contains the number of torsions of this type followed by lines in which fields 1)-4) define atoms and fields 5)-9) constants $K_1, ..., K_5$ in kJ/M for each torsion angle of this type.

- `improper`

  improper dihedral potential:

$$U_{impr}(\phi) = k_\phi(\phi - \phi_0)^2 \tag{10}$$

  field 5) is the equilibrium angle in degrees and 6) is the force constant

- `special`

  introduces a list, over-reading Lennard-Jones parameters for 1-4 interactions for the specified atoms. This list consists of the following:

  the first line - number of atoms in the list;

  other lines: the first field defines atom, the second is sigma (Å) and the third parameter is epsilon in kJ/M.

  Scaling parameter for 1-4 LJ interactions, eventually specified by `slj14` keyword (see below), is applied to these modified LJ parameters also.

- `no_14`

  This parameter tells that all 1-4 pairs must be excluded from the intramolecular Lennard-Jones and electrostatic interactions

- `noi15`

  This parameter tells that all intramolecular electrostatic and LJ interactions are switched off

- `sel14 <factor>`

  Scale 1-4 electrostatic interactions by this factor

- `slj14 <factor>`

  Scale 1-4 Lennard-Jones interactions by this factor

- `fSPC`

  This parameter is exclusively for the flexible SPC model. It tells to the program to compute water intramolecular potential according to the paper by Toukan and Rahman, Phys. Rev. B, 31(2) 2643 (1985)

Examples of `.mmol` files are included into `moldb` directory. For big molecules, procedure of writing `.mmol` files may be somewhat automatized by the utility `makemol`.

3 The main input file

3.1 General notes

Input file consists of keywords followed by parameter(s) Lines beginning with "#" are considered as commentaries. Keywords are case sensitive. Lines which are not recognized as keywords are considered as commentaries, but a warning is given in the output if the first symbol of such line is not "#". The order of keywords is almost unimportant. The only exception is that the information on the number of molecular types (keyword `Mol_types`) should be given before any keyword involving reference to specific molecules and molecular types. For missing keywords, default values are used. Repeated reading of the same keyword overwrites its previous value

3.2 Basic setup

- `Main_filename <name>`

  `<name>` is the base file name for the given simulation. Other files requested or created by the program have this name with various extensions

  This keyword is compulsory.

- `Verbose_level <number>`

  `<number>` - Output control parameter (integer). Suitable values 2-10. The less number, the less you see in the output. Parameters higher 7 used mostly for debug purposes

  default: 5

- `Path_DB <name>`

  `<name>` - Path to the molecular database containing files describing molecular structures and force field parameters (.mmol files)

  default: . (current directory)

- `Visual yes/no`

  Run with visual shell (not included)

  Default: no

3.3 Restart control

- `Read_restart yes/no [ASCII]`

  "yes" - read restart file `<Main_filname>.dmp` and continue old simulation from the restart file. .

  "no" - start a new simulation

  default: yes (this is to avoid non-intentional startup of a new simulation which may rewrite an old restart file possibly containing results of a valuable simulation)

  "ASCII" is an optional parameter. If it is specified, restart file will be read in ASCII format

- `Dump_restart <nsteps> [ASCII]`

  Dump restart file after every `<nsteps>` of the simulation

  "ASCII" is an optional parameter. If it is specified, the restart file will be written in ASCII format.

  Note: Binary format of restart file is preferable since it takes less space and do not lose any precision. ASCII restart file should be used only if you want to continue the simulation on a computer with another architecture

  Default: do not dump restart file

- `Change_T yes/no`

  Change temperature after restart. This option sets up a new value for the target temperature which is specified in the `Nose_thermostat` or `Velocity_scaling` keywords. Otherwise ("no"), the old value of the temperature saved in the restart file is used

  If the target temperature is changed, all the velocities are rescaled by the corresponding factor.

  Default: no

- `Change_V yes/no`

  Change volume after restart. This option sets up a new value for the box sizes which are specified in the Box or Density keyword. Otherwise ("no") the old values of the box sizes saved in the restart file are used

  If the box sizes are changed, all the distances between the molecules are rescaled by the corresponding factor.

  Default: no

- `Check_only yes/no`

  "yes" - Do not run actual simulation but only check all input information. If new simulation, only check input. If keyword `Read_restart yes` is specified, type out results saved in the restart file.

"no" - Run simulation as specified

Default: no

- `Zero_CPU yes/no`

  If "yes", set counter of the CPU time to zero

  Default: no

- `Zero_vel yes/no`

  If "yes", set all velocities to zero (at start-up or at restart)

  Default: no

- `Zero_average yes/no`

  If "yes", set all accumulators of averages to zero and clear the history of averages

  Default: no

3.4 System

- `Molecule_types <num_types>`
  `<name> <number> [fixed]`
  `...`
  `<name> <number> [fixed]`

  Which molecules to simulate. `<num_types>` is the number of molecule types. This keyword must be followed by `<num_types>` lines each of which contains at least two parameters: `<name>` and `<number>`

  `<name>` - name of the molecule. File `<name>.mmol` , describing the molecular structure and the force field parameters, must be present in directory specified by parameter `Path_DB`.

  `<number>` - number of molecules of this type.

  "fixed" - Option "fixed" signals that these molecules are fixed and will not move in the MD procedure.

  This keyword and its parameters (except "fixed") are compulsory

- `PBC <type>`

  Type of periodic boundary conditions

  `<type>` may be:

  `rect` - rectangular (default)

  `hexa` - hexagonal

  `octa` - truncated octahedron

- `Box <box-x> <box-y> <box-z>`

  Specifies simulation box size (in Å). In case of hexagonal boundary conditions, only `box-x` and `box-z` have a meaning. For octahedral boundary conditions only `<box-x>` has a meaning. The keyword cannot be used simultaneously with "Density"

- `Density <value>`

  Setup initial density. `<value>` is density in g/cm^3. This option cannot be used simultaneously with setting up box sizes. In case of rectangular periodic boundary conditions, a cubic cell is implied. Zero density implies "vacuum simulation" regime with the box size automatically set to 1000 Å

- `El_field <ampl> <freq>`

  Put the system in a homogeneous time-dependent electrostatic field

$$E(t) = A\cos(2\pi\omega t) \tag{11}$$

  where `<ampl>`$= A$ is given in V/cm and `<freq>` $= \omega$ in Hz.

3.5 Ensemble

- `Nose_thermostat <temp> <relax.time>`

  Specify whether to use the Nose thermostat. The option is incompatible with "velocity scaling". Two parameters are compulsory:

  `<temp>` - target temperature

  `<relax.time>` - relaxation time in femtoseconds

  Note: if simulation is restarted from a restart file, the old value of temperature specified in the restart file is normally used, ignoring value of this keyword. To override this, use keyword `Change_T`.

- `Velocity_scaling <temp> <dt>`

  Specify whether the temperature is regulated by the velocity scaling. The option is incompatible with "Nose thermostat". Two parameters are compulsory:

  `<temp>` - target temperature

  `<dt>` - admissible deviation. If the temperature deviates from the target by more that "dt", velocities are rescaled to fit the target temperature.

  Note: if simulation is restarted from a restart file, the old value of temperature specified in the restart file is normally used, ignoring value of this keyword. To override this, use keyword `Change_T` .

- `Separate_thermostating yes/no`

  Specify whether to apply temperature control to the whole system (no) or separately to each molecular type (yes). The option works both for Nose thermostate and for temperature control by velocity scaling

  default: no

  Note. At this option, the total momentum is not conserved. Use with care. See also `COM_check` keyword.

- `COM_check yes/no [num]`

  Specify whether to remove motion of the total center of mass. If parameter "num" is zero or negative, motion of the center of mass is removed only at the program start. If this parameter is positive, it gives the number of MD-steps after which the removal of center of mass motion is repeated

  Default: yes 0

- `Barostate_NH <pres> <relax_time>`

  Specify whether to use the Nose-Hoover barostate. Option works only if Nose thermostat is specified. Without the keyword, constant volume simulations are implied. Two parameters are compulsory:

  `<pres>` - pressure in bar (atm)

  `<relax_time>` - relaxation time in femtoseconds

- `Barostate_anisotropic yes/no/XY`

  Specify whether to apply pressure control isotropically (no), separately to each direction (yes), or separately in XY-plane and along Z-axis (XY). The option has effect only if constant-pressure simulations (barostate) is specified

  default: no

3.6 Molecular Dynamics

- `Time_step <value>`

  Time step in femtoseconds. This is the long time step in case of the double time step algorithm.

  Default: 2 fs.

- `Number_steps <number>`

  How many MD steps to run.

- `Double_timestep <number>`

  Double time step algorithm by Tuckerman et al. `<number>` is the number of short time steps in one long. The keyword cannot be used simultaneously with `Constrain`.

- `Constrain <toler> <i_1> <i_2> <i_3> .... <i_ntypes>`

  Use SHAKE algorithm to constrain the bond lengths. The option cannot be used with `Double_timestep`. `<toler>` is the tolerance level.

  `<i_1> <i_2> ... <i_ntypes>` are numbers 0 or 1, which specify whether (1) or not (0) to apply constraints to molecules of each species.

- `R_cutoff <value>`

  Cut-off (in Å) for the Lennard-Jones and real-space part of the electrostatic forces.

  Default: 12 Å

- `R_short <value>`

  Cut-off (in Å) for Lennard-Jones forces computed each short-time step (has an effect only in double-time step algorithm)

  Default: 5 Å

- `Neighbour_list <number>`

  Update the list of neighbors (Verlet list) every `<number>` steps

  Default: 10

- `Electrostatics <type> [<A> <B>]`

  How to treat electrostatics

  `<type>` may be Ewald (default), RF (reaction field) and Cutoff

  For Ewald: A is α/R_{cutoff}, where α is the Ewald convergence parameter. Precision of the real-space Ewald part is determined by $erfc(A)$. B defines the number of terms in the reciprocal part. It cuts the reciprocal series when expression in the *exp* of the reciprocal part exceeds B. Recommended values A = 2.5 – 3; B = 7 – 10.

  For reaction field, A is the dielectric permittivity and B is the Debye screening length in Å. Setting the Debye length to 0 means an infinite Debye length, i.e non-conducting solution.

  If `<type>` is "Cutoff", parameters A and B are not necessary, and no special treatment of electrostatic forces out of R_{cutoff} takes place.

  Default: Ewald method with A=2.8 and B=9.

- `Cut_forces <value>`

  If this option is specified and the absolute force acting on any atom exceeds the specified `<value>`, the force will be cut to this level while maintaining the direction. The value is given in Å and has a sense of the maximum allowed additional displacement during one time step (F*dt**2/m) caused by this force

  Default: Do not cut forces

- `Combination_rule <type>`

  Use another (than Lorentz-Berthelot) combination rule for Lennard-Jones parameters for different types

  `type` may be one of the following:

  `Sigma_geom` - LJ σ parameter is calculated as a geometrical average of the both types (as for example in the OPLS force field)

  `Kong` - use Kong rules (J.Kong, J.Chem.Phys., 59, 2464, (1973))

  Default: use Lorentz-Berthelot combination rules

- `Bind_atoms <file> <deviation>`

  If this keyword is specified, atoms defined in file `<file>` will be bound to corresponding positions (given in the same file) by a harmonic potential with characteristic deviation given by `<deviation>`

3.7 Path Integral mode

Path integral mode can be run only in parallel using number of processors equal to the number of beads in the path integral representation of quantum particles. The number of beads should be even.

- `PIMD <num. of beads>`

  Use PIMD mode with the given number of beads

- `PI_thermo <temp> <tau> <chain>`

  Use Nose chain of thermostats for each atom. This is preferable option instead of the common Nose thermostat for all atoms. `<temp>` is the target temperature, `<tau>` is the relaxation time for the thermostats and `<chain>` is the number of thermostats in each chain. If this option is specified, Nose_thermostat option is ignored.

3.8 Startup

- `Startup <type> [<optional parameters>]`

  How to set up initial coordinates. This option has an effect only if `Read_restart` was specified as "no"

  `<type>` may be one of the following:

  `xmol` (default) - the input configuration is taken from file `<Main_filename>.start` (see keyword "`Main_filname`") which should be in "XMOL" format The second (commentary) line of this file may contain box sizes (which follow after keyword `BOX:`). They will be used as actual box sizes if keyword "`Change_V`" is not set or set to "no". If keyword "`Change_V`" is set to "yes", the box sizes specified by keywords "`Box`" or "`Density`" will be used.

`xyz` - the input configuration is taken from coordinates written as three columns of X Y Z coordinates in `<Main_filename>.start` file

`Mol_COM` - molecular center-of mass coordinates are taken from

`<Main_filename>.start` file, as three columns of X Y Z coordinates.

`FCC` - molecular center of mass coordinates are put on a FCC lattice

`Cubic` - molecular center of mass coordinates are put on a cubic lattice

`Cyl_hole <radius>`

`Sph_hole <radius>`

Molecules, except those defined in option `Start_rot` are distributed outside sphere or cylinder of radius `<radius>`

In all cases, except "xmol" and "xyz", atom coordinates are build around molecular center-of-mass using local coordinates specified in the corresponding .mmol files.

- `Start_rot  <i_1> <i_2> <i_3> ... <i_ntypes>`

  This option has an effect only at Startup with options:

  Startup option "FCC" or "cubic":

  Tells which molecules rotate randomly (parameter `i_n` =0) and which not (`i_n` = 1)

  If Startup options "xmol" or "xyz" are specified, keyword `Start_rot` do not have any effect.

  Startup options "`Cyl_hole`" and "`Sph_hole`":

  Molecules which should be placed inside cylindrical or spherical hole, are marked by parameter `i_n` = 1. Initial coordinates of atoms of such molecules are taken from the corresponding .mmol file and it can be only a single molecule of this type. Other molecules (with `i_n = 0`) are distributed outside the spherical or cylindrical hole.

  Default: all `i_n`=1 (rotate molecules randomly)

- `Gather`

  Gather each molecule in one place (if its atoms were spread over different periodical images). Works both at start-up and at restart.

  Default: use coordinates as they are.

3.9 Properties

- `Output <number>`

  At output level >= 5, type line(s) with some data (time step, energies, etc) each `<number>` of steps

  Default: 1

- `Serie_average <number>`

  Compute and remember intermediate averages over series of `<number>` steps

  Default: 10000

- `Average_from  <number>`

  Begin final averaging over series (see `Serie_average` keyword) from series with the given `<number>`

- `Average_internal  yes/no`

  Whether to compute average values of all bond lengths, angles and their energies

  Default: no

- `Dump_XMOL  yes/no`

  Dump the final configuration in "XMOL" format

  Default: no

- `Trajectory  <format> <step> <num_conf> <list>`

  Dump trajectory.

  `<format>` may be one of:

  `bincrd` - binary with coordinates only

  `binvel` - binary with coordinates and velocities

  `asccrd` - ACSII (XMOL-format) with coordinates only

  `ascvel` - ACSII (XMOL-format) with coordinates and velocities

  `<step>` is time interval (in femtoseconds) between the configurations saved in the trajectory file

  `<num_conf>` - number of configurations in each trajectory file. After filling `<num_conf>` configuration in a trajectory file, the program begins to write configurations in the next file. Trajectory files acquire extensions .001, .002, .003, etc

  `<list>` - list (in the form of numbers 0 or 1 for each molecule type) which signals whether to include molecules of this type into trajectory. `<list>` may be set also to "all" (all molecules)

  Units for velocities in the trajectory file are Å/fs.

- `Bond_list yes/no`

  Generate list of all bonds and put it in file bond.list

  Default:no

- `RDF_calc <restart_option> <RDFcutoff> <nbins>`

  Calculate RDFs.

  `<restart_option>` may be one of the following

  RW - read restart rdf file and dump updated rdf-restart file

  RO - only read restart RDF file but do not dump updated rdf file

  (these two options have an effect only if simulation is continued from restart)

  WO - start a new collection of RDFs (that is, ignore restart file even if it is present) and dump a new RDF-restart file.

  NOR - do not read and do not write restart-RDF file

  `<RDFcutoff>` - Cutoff for RDF

  `<nbins>` - number of bins to compute RDFs

  This keyword must be followed by a description which RDFs need to be calculated. This part consists of the following lines (commentary symbols are allowed):

  `<n>` - number of RDFs

  followed by a list of atom pairs (site numbers) for RDFs. Symbol `&n` can be used for averaging of RDFs from several atom pairs. Such a group is counted as a single RDF.

- `TCF_calc <restart_option> <N-steps> <jump>`

  Calculate TCFs.

  `<restart_option>` may be one of the following

  RW - read restart TCF file and dump updated tcf-restart file

  RO - only read restart TCF file but do not dump updated tcf file

  (these two options have an effect only if simulation is continued from restart)

  WO - start new collection of TCFs (that is, ignore restart file even if it is present) and dump new restart TCF file

  NOR - do not read and do not write restart TCF file

  `<N-steps>` - number of steps to compute TCFs

  `<jump>` - number of (long) MD time steps for one step of TCF

  This means that TCF will be calculated with step "`dt*jump`" during "`dt*jump*<N-steps>`" time

  This keyword must be followed by a description which TCFs need to be calculated. This part consists of the following lines (commentary symbols are allowed):

  A line of 12 symbols "0", "1" or "2" which specify which of 12 types of TCFs are to compute:

```
1 - velocity autocorellations
2 - angular velocity autocorellations
3 - 1 order Legendre polynom for dipole moment
4 - 2 order Legendre polynom for dipole moment
5 - 1 order Legendre polynom for reorintational tcf for a
      specific vector
6 - 2 order Legendre polynom for reorintational tcf for a
      specific vector
7 - X projection of velocity TCF
8 - Y projection of velocity TCF
9 - Z projection of velocity TCF
10 - X projection of angular velocity TCF
11 - Y projection of angular velocity TCF
12 - Z projection of angular velocity TCF
```

"0" means do not compute, "1" or "2" signals that this type of TCF must be computed. "2" can be specified for tcf 7-9 or 10-12, then tcf projections are calculated in the molecular principal coordinate system; otherwise (1) they are calculated in the laboratory's coordinate system

If TCF of type 5 or 6 are given for computation, then two additional lines should be given which specify two atom number on each molecular type which define the unit vectors for reorientational TCF.

Note. TCF calculations are not parallelized and therefore not recommended for parallel computations. Use the trajectory analysis instead.

- `End`

  end of the input file. This line is compulsory

4 Compilation

The program can be installed in any user-defined directory. After copying the distribution file into chosen directory, unpack it:

`tar xfvz md50.tar.gz`

A new directory `md50` arises. Change to this directory.

Before compiling, you may need to change sizes of working arrays defined in the file `dimpar.h`. The values of parameters depends both of molecular system you want to simulate and on the computer in hands.

The are several Makefiles for different architectures. Choose the most suitable and edit it if necessary. Usually options for optimization can be tuned to improve performance. Then run "make" which invoke one of these Makefiles. The Makefiles differ mostly by compiler used and subroutine to count cpu time.

4.1 Sequential execution

These files produce executable "md"

`make default`

calls `Makefile.unknown` . This should work for any standard f77 compiler, with exception of accounting cpu time which is substituted by a "dumb" subroutine (substitute your own counter of cpu time in file `cpu_dummy.f`)

`make linux`

calls `Makefile.gfortran`. This can be used on Linux or other systems with gfortran fortran

`make intel`

calls `Makefile.ifort`. This can be used on Linux with free (at least now) Intel compiler `ifort`. Performance options, such as `-xP` for processors with sse3-instructions, may substantially speed up the program.

`make pgi`

calls `Makefile.pgi` Calls Portland Group Fortran compiler `pgi`.

`make risc`

calls `Makefile.risc` This works for IBM RISC workstations using `xlf` compiler.

4.2 Parallel execution

Note that these makefiles are strongly dependent on how the MPI library is installed. The name of executable is `mdp`

`make mpi`

Calls `Makefile.mpi` . This can be used at "standard" MPI installation which uses mpif77 script to call the correct Fortran compiler and link MPI-libraries.

`make t3e`

Used for Cray T3E with MPI

`make sp2`

Used for IBM SP2 parallel machine

4.3 Other makefiles

`Makefile.DEC` - DEC alpha

`Makefile.pgi_mpi` - Portland Group Fortran with MPI library

`Makefile.ifc` - old Intel compiler (v.7.0 and older)

`Makefile.g77` - old GNU g77 compiler

`Makefile.deb` - Intel fortran `ifort` with debug flags. Produces executable `mdd`

`Makefile.static` - Produces static executable

5 Execution

The program compiled for one processor can be started from a command line:

`md < md.in > md.out`

where `md.in` is the main input file and `md.out` is output. Names of input and output files may be arbitrary

In the case of parallel execution, the input file must be always called `md.input`. The program starts for example as:

```
mpirun -np 4 mdp > md.out
```

where number of processes is 4 and `md.out` is name of the output file. Other arguments for `mpirun` command can be given. Often `mpirun` can be invoked only from a queue batch script, see the rules in your computer center. Sometimes command `poe` is used to start a parallel program

5.1 Files used by the program

Table 1: Files used by the program. "<fname>" is the same for all files in the simulation

file	I/O	required	description
standard input	I	yes	the main input file
standard output	O	yes	the main output file
`*.mmol`	I	yes	files defining molecular structures and the force field
`<fname>.start`	I	opt	start configuration
`<fname>.dmp`	IO	opt	restart file
`<fname>.rdf`	IO	opt	restart RDF file
`<fname>.tcf`	IO	opt	restart TCF file
`<fname>.xmol`	O	opt	final configuration
`<fname>.00n`	O	opt	trajectory files

5.2 Memory considerations

Parameters defining arrays boundaries are given in `dimpar.h` file. They define also the required memory. They can be decreased to save memory or increased if they are not enough for the simulated system.

One of the most memory-consuming arrays is the list of the atom pairs within the cutoff distance. Its size is defined by parameters NTOT (maximum number of atoms) and NBLMX - maximum number of neighbors (within cutoff) for each atom. Since the list of neighbors is distributed among the available nodes, this parameter can be decreased in the case of parallel execution.

Another "memory consuming" parameters is MAXCF which defines the number of point for calculation of time correlation functions. It can be set to 1 if calculation of time correlation functions is not carried out during the program run. Note also that calculation of TCF is not parallelized, that is why it is often more appropriate for TCF calculations to dump the trajectory and recalculate TCFs afterwords using the `tranal` utility.

Much memory can be also taken by the array with intermediate averages for different parameters. Its size is NRQS*LHIST, where NRQS is maximum number of calculated different averages and LHIST is the maximum number of

series for which these averages are calculated. It may be really big for large macromolecules, if keyword `Average_internal` is set to `yes`. In this case, all bond lengths, covalent and torsion angles are calculated and remembered.

The program checks correspondence of array boundaries to the input data and stops if something is wrong. After correction of `dimpar.h`, the code must be recompiled.

5.3 Program structure

The program consists of several Fortran files, each contains one or several Fortran units. The files are:

- `main.f` - the main module.

  This is the main program unit taking care of all what follows.

- `dimpar.h` is the file defining maximum sizes of working arrays. The code must be recompiled after any change in this file.

- `prcm.h` is a header file common for most of program units. It contains static dimensioning of the working fields and definition of all global parameters. Practically all modules depend somehow on it.

- `input.f` This file contains subroutines which read the main input file and .mmol files.

- `setup.f` This file setup units, reference arrays and data structures

- `mdstep.f` This file contains MD integration algorithms, including thermostats and barostats as well as SHAKE algorithm for constraint dynamics

- `forces.f` This file contains subroutines responsible for calculations of different forces. Forces are divided into two groups: slow and fast forces. This division is essential only if the double time step algorithm is used. Procedure for updating the lists of neighbors is also included into this file

- `mpi.f` contains collection of MPI calls for creating parallel executable code. This is the standard version of this file, which uses real*4 size of data during communications.

  `mpi_cray.f` is Cray version of MPI calls for parallel execution

  `mpi_double.f` is a version of MPI calls for parallel execution with double precision for the data transfer

  `mpi_safe.f` is a version of MPI calls with some additional controls for communications.

  `scalar.f` is collection of dummy MPI calls which is used in a single-processor version of the code

- `restart.f`

  This file contains subroutines responsible for writing and reading restart file, dumping trajectories and other operations with input/output files

- `aver.f` collecting averages, including RDFs
- `pimd.f` Path Integral molecular dynamics
- `tcf.f` calculation of time correlation functions
- `service.f` is a collection of some procedures specific for molecular dynamics
- `util.f` is a collection of some other auxiliary procedures
- `cpu_*.f` are different variants of counting cpu time; `getcpu.c` is a C-function to count CPU-time

6 *Makemol* utility

6.1 General organization

The `makemol` utility is used to generate a `.mmol` file (containing information on both molecular structure and the force field) from two files: a force field (`.ff`) file, with information about force field parameters, and a `.smol` file, which contains only structural information about the molecule (initial atom coordinates, the list of covalent bonds, the partial atomic charges, and the chemical types of atoms (which should match the types in the force field file). The utility constructs the list of covalent angles and torsion angles, and pick up the force field parameters from the force field file. If some parameters are not found in the force field file, the corresponding line in the resulting `.mmol` file is marked with an exclamation mark, and a note is written on the screen and in the `makefile.log` file.

The list of Improper torsions is not generated automatically. If impropers exist for the given molecule, they should be added into `.mmol` file manually.

The utility is run in a self-explaining dialog regime. The format of the force field and the molecular structure files is explained below.

6.2 The force field (.ff) file

All the lines beginning with "#" are considered as commentaries.

The file consists of a number of sections; each beginning with a keyword. Each new keyword means the end of the old section and beginning of the new one corresponding to this keyword. All energies are given in kJ/mol and distances in $\mathring{A}$.

- `BONDS`

  Contains parameters for covalent bonds. Each line consists of four or six parameters. The first two parameters are the force field atom types. The third parameter is the force constant k of the harmonic potential (4) and the fourth parameter is the equilibrium distance r_0. The presence of 5-th and 6-th parameters means that the corresponding bond should be described by a Morse potential, they have a sense of D and ρ parameters of the Morse potential (5) correspondingly.

- `ANGLES`

  Contains parameters for covalent bonds. Each line consists of 5 or 7 parameters. The first three parameters are the force field atom types. The fourth parameter is the equilibrium angle, the 5th parameter is the force constant k. The 6-th and 7-th parameters are optional, if they are present, they define the Urey-Bradley (UB) potential which is in fact a distance-dependent harmonic potential between 1st and 3rd atoms. The equilibrium distance for UB potential is given as a 6-th parameter and the force constant as a 7-th. `makemol` puts the UB term into the "bond" section of the `.mmol` file.

- `TORSIONS`

  Contains parameters for torsion potentials of "standard" type (7). Each line consists of 7 parameters, the first four define the force field atom types, and parameters from 5-th to 7-th are Δ, K_ϕ and M correspondingly.

  This section allows definition of multiple torsions and wildcards (denotes as "X"). The wildcards can be used only for the 1st and 4-th atoms in the list. The following rules are applied. If an explicit torsion (without wildcards) is specified, lines with wildcards matching this torsion are ignored. If lines with explicit torsions are repeated for the same set of atom types, they define multiple torsions, that is the total torsion potential is a sum of several terms of type (7).

- `NONBONDED`

  Contains Lennard-Jones parameters for non-bonded interactions. Each line consists of 3 or 5 parameters. The first parameter is the force field atom type. The second and third parameters are σ and ϵ of the Lennard-Jones potential. If the 4-th and 5-th parameters are present, they define σ and ϵ parameters for 1-4 Lennard-Jones interactions.

6.3 The molecular structure (.smol) file

This file contains information on molecular structure, partial atomic charges and the force field atom types. All the lines beginning with "#" are considered as commentaries. The first non-commentary line defines the number of atoms in the molecule. Then follows the corresponding number of lines describing each

atom. Each line consists of 6 parameters. The first one is the name of the atom. Parameters 2 - 4 are the initial X, Y and Z coordinates of atoms. The 5-th parameter is the partial atom charge, and the 6th parameter is the force field atom type. `Makemol` just rewrites parameters 1-5 into corresponding sections of the `.mmol` file, and uses the force field atom type (6th parameter) in order to find the force field parameters from the `.ff` file.

`Makemol` defines masses of atoms from the one or two first letters in the atom name (the first column of `.smol` file). If the program cannot recognize the element, it complains about that; in such a case the mass of the atom must be put manually into resulting `.mmol` file.

After the description of atoms, a section describing the bond structure follows. The first line of this section is the number of bonds. Then the corresponding number of lines follows, with two numbers in each line defining numbers of atoms bound by the corresponding bond. From this list, `Makemol` builds lists of covalent angles and torsions angles, and substitute the corresponding force field parameters.

Besides the described above `.smol` format, `Makemol` can accept `alchemy` and CHARMM-coordinate formats which contain similar information in the corresponding fields.

7 *Tranal* utility

7.1 General organization

TRANAL package consists of a number of utilities for analyzing of atom trajectories generated by molecular dynamics programs. The package is constructed to work mainly with MDynaMix molecular dynamics program, but is able to accept trajectories written in some other formats: trajectories written in XMOL format, PDB trajectories from GROMACS and NAMD binary trajectories.

It is supposed that in all cases, the atoms and molecules are arranged in the following way: (the same as in MDynaMix program):

`<molecules of type 1><molecules of type 2>...`

in each molecule type:

`<molecule 1><molecules 2>...`

in each molecule:

`<atom1><atom2>...` (the same atom order must be in all molecules of this type)

Additionally, a term "site" is determined. If one chooses one molecule of each type and set them one after another (in the same order as in the trajectory), then the number of each atom in this sequence would correspond to the "site" number.

Each of utilities in the TRANAL suite consists of two parts: the main block `tranal_base.f`, which reads trajectories, and a specific part which makes required analysis. Correspondingly, the input file for each utility consists of two

parts: the first one which determines how to read trajectories, and the second part containing parameters for the required analysis.

Currently, the following analyzing utilities are included:

- **bileldens** - for membranes or bilayers: computation of mass density, electron density, and electrostatic potential across membrane
- **diel** - calculation of average dielectric permittivity
- **diffus** - calculation of the mean square displacement and the self-diffusion coefficients
- **dryrun** - just reading trajectories and optionally rewriting them to XMOL format
- **latdiff** - computation of 2D mean square displacement and lateral diffusion for membranes or bilayers
- **order** - computation of the order parameter and angular distribution of specific molecular vectors
- **rdf** - computation of radial distribution functions
- **sdf** - computation of spatial distribution functions
- **torsion** - computation of distributions of torsion angles
- **torsion2** - Ramachandra plot for two torsion angles
- **trtcf** - computation of some time correlation functions

Each of utilities must be compiled by a Fortran compiler together with **tranal_base** block. For instance, the program for RDF calculations can be compiled by:

```
f77 -O3 -o rdf rdf.f tranal_base.f
```

Eventually, sizes of arrays may be needed to adjust, by editing **tranal.h** file, as well as sizes of arrays in each specific procedure.

Contributions for analysis of other properties are welcomed.

7.2 tranal_base: reading trajectories

This part is common for all types of trajectory analysis. It reads the first part of the input file and then reads trajectories accordingly. The first part of the input file is written in "NAMELIST" format which looks like:

```
$TRAJ
parameter=value(s),
...
$END
```

"TRAJ" is the name of this NAMELIST section.
The following parameters must be defined:

- NFORM = <format>

  where <format> is one of:

  - MDYN - MDynaMix binary trajectory (default)
  - XMOL - XMOL trajectory. It is implied, that the commentary (second) line of each configuration is written in the format:

    (char) <time> (char-s) BOX: <box_x> <box_y> <box_z>

    where (char) is any character word, <time> is time in fs,
    <box_x> <box_y> <box_z> (following after keyword BOX) are the box sizes.
  - PDBT - PDB trajectory as generated by "trajconv" utility of GROMACS simulation package
  - DCDT - DCD trajectories generated by NAMD package

- FNAME = <file_name>

  set the base name of the trajectory files. The trajectory must be written as a sequence of files <file_name>.001 , <file_name>.002 and so on, the largest possible number being <file_name>.999 .

- PATHDB = <value>

  Directory with molecular description files (.mmol). Default is the current directory (.) .

- NTYPES = <value>

  Number of molecule types in the trajectory

- NAMOL = <name1> [,<name2>,...]

  NTYPES names of molecules. It is supposed that files <name1>.mmol, <name2>.mmol,... describing the molecules are present in the directory defined by PATHDB. Format of .mmol files is the same as for MDynaMix program. For analyzing trajectories generated by other programs, .mmol files are still needed. It is however enough to have only the first section of .mmol files containing names of atoms.

- NSPEC = <n1>[,<n2>,...]

  Number of molecules of each type (NTYPES numbers). This parameter is not necessary in MDynaMix binary trajectories.

- NFBEG = <value>

  Number of the first trajectory file (integer between 0 and 999)

- `NFEND = <value>`

  Number of the last trajectory file (integer between 0 and 999)

- `IPRINT = <value>`

  Defines how much you see in the intermediate output. The final output with analysis of results does not depend on it. Default value is 5.

- `BOXL = <x-box-size>`
  `BOYL = <y-box-size>`
  `BOZL = <z-box-size>`

  define the box size if it is not present in the trajectory (implies constant-volume simulation)

- `BREAKM = <value>`

  Defines which break in trajectory (in ps) is allowed to consider the trajectory as continuous. The parameter is used in analysis of time-dependent properties.

- `LVEL = <logical>`

  Accept a logical value. `.true.` signals that the trajectory may contain velocities, which will then be read for the analysis.

- `LXMOL = <logical>`

  Accept a logical value. If .true., rewrite trajectory in XMOL format

  `FXMOL = <name>`

  File name for output XMOL trajectory

- `ISTEP = <value>`

  Specifies that only each `ISTEP`-th configuration from the trajectory is taken for the analysis

- `LBOND = <logical>`

  Accept logical value. If .true., information about bonds is read from .mmol files

- `VFAC = <value>`

  Multiply velocities from the trajectory by the given `<value>`, in order to bring them to Å/fs (the units which are used in the analysis). Default is 1.

7.3 Computation of the electron density and electrostatic potential in bilayers: bileldens.f

This computation is made by `bileldens` utility. It is supposed that a membrane-like system is oriented in XY-plane. The utility computes the electron density, the mass density, the charge density, and the electrostatic potential across membrane in Z-direction.

Compilation:

```
f77 -O3 -o bileldens bileldens.f tranal_base.f
```

Input parameters for this utility follow after the trajectory parameters in the NAMELIST block `ELDEN`:

```
$ELDEN
parameter=value(s),
...
$END
```

The following parameters are used:

- `FDENS = <filename>`

  Defines name of the output file

- `IRT = <int.num>`

  Defines the type number of "lipid" molecules. The middle plane of the bilayer is calculated from the center-of-mass Z-coordinate of molecules of this type. Eventually the molecules are transfered across the periodic boundaries to keep the whole bilayer within the same periodic cell.

- `NA = <int.num>`

  Defines the number of "bins" for computation of densities

- `ZMAX = <value>`

  Defines the borders of the interval in Z-direction. The densities and the electrostatic potential are calculated in the range of z: $-ZMAX < z < ZMAX$ relative to the membrane middle plane

- `IATE = <list of values>`

  This optional parameter defines which sites to take into account while calculating the densities. Default is all sites. A value equal to 1 says that the given site is accounted while 0 says that this site is not accounted. For example, if the system consists of DMPC lipids (118 atoms in molecule), water (3 atoms), Na and Cl ions, this compute contributions from the lipids only:

  ```
  IATE = 118*1,5*0
  ```

  while this compute contribution from the water oxygen only:

  ```
  IATE = 118*0,1,0,0,0,0
  ```

- `LCHG = <logical>` This parameter tells whether to use a special file containing charges for computation of electron density. In case `LCHG = .true.`, a file defined by parameter `FCHG` is read; it determines contribution of each site into the electron density. If `LCHG = .false.`, contribution of each site is defined from the first letter of the site name (for H, C, N, O, P and S atoms only) and corrected by the partial atom charges.

- `FCHG = <file_name>`

  File name for the case `LCHG = .true.`

- `LSYM = <logical>`

  This parameter tells whether to symmetrize the computed densities relative to the membrane middle plane. Note that even slightest asymmetry in the charge distribution may lead to a potential difference from the both sides of membrane.

7.4 Dielectric constant: diel.f

This utility computes static dielectric permittivity (dielectric constant) from fluctuations of the total molecular momenta using expression (S.W.deLeeuw, J.W.Perram, E.R.Smith, Ann.Rev.Phys.Chem., 37, 245 (1986)):

$$\epsilon = 1 + \frac{4\pi}{3kTV}\langle|\sum_i \vec{\mu}_i|^2\rangle \tag{12}$$

where $\vec{\mu}_i$ is the dipole moment of molecule i and the sum is taken over all the molecules. Note that this formula is applicable only for small and rigid (or almost rigid) molecules. It is important that the molecules were not broken by the periodic boundary conditions.

Compilation:

```
f77 -O3 -o diel diel.f tranal_base.f
```

This utility needs only a single parameter - the temperature, which should be written directly after the trajectory parameters.

7.5 Diffusion: diffus.f

This procedure computes time dependence of the mean square displacement (MSD) and evaluates the self-diffusion coefficient.

Compilation:

```
f77 -O3 -o diffus diffus.f tranal_base.f
```

Input parameters for this utility follow after the trajectory parameters in the NAMELIST block `DIFF`:

```
$DIFF
parameter=value(s),
...
$END
```

The following parameters are used:

- `FILDIF = <filename>`

  Defines the name of the output file

- `IDF = <int.num>`

  Defined the type of molecules for which diffusion is calculated

- `DTT = <value>`

  Defines the time interval (in s) for MSD calculations. It is very recommended that this parameter is equal to the time step of the trajectory multiplied by `ISTEP` parameter defined in the trajectory (`TRAJ`) section of the input file. If the above requirement is not fulfilled, the program may still work but less accurately.

- `NTT = <int.value>`

  Defines the number of steps for MSD calculation. The total time of tracking the MSD will be thus `DTT*NTT`.

- `IAT = <int.value>`

  If `IAT=0`, MSD is calculated for the centers of masses of the selected molecules. Otherwise MSD is calculated for `IAT`-th atom of each molecule.

- `LCOM = <logical>`

  Specifies whether to correct for the total center-of-mass motion of the selected molecules (that is, of type `IDF`). If `.true.`, the COM for each molecule is computed relative to center of mass motion of the molecules of this type. The default is `.false.`. Note also, that correction for the center of mass motion of the whole system is not carried out (except the case of only one molecule type and `LCOM=.t.`).

- `FBEG=<value>`

  Defines the beginning of linear fitting of the MSD curve to evaluate the self-diffusion coefficient as: `FBEG*DTT*NTT` . The default value is 0.2 , that is initial 20% of the MSD vs time dependense is not included. It is always recommended to look at the computed MSD vs time dependence to evaluate acceptable value for this parameter.

Futher comments
For each time t the MSD is computed as:

$$\langle \Delta r^2(t) \rangle = \langle (\vec{r}(t_0 + t) - \vec{r}(t_0))^2 \rangle \tag{13}$$

where averaging is taken over all molecules of type `IDF` and all acceptable initial times t_0: $t_{beg} \leq t_0 \leq t_{end} - t$, where t_{beg} and t_{end} are the initial and final time of a *continuous* part of the whole trajectory respectively. A trajectory is

regarded as continuous if each next configuration differ from the previous no more than parameter `BREAKM` defined in the trajectory part (`TRAJ`) of the input file.

The output file consists of the following columns:

The first column- time t.

The second: for each t, evaluation of the diffusion coefficient as:

$$D_{av} = \frac{\langle \Delta r^2(t) \rangle}{6t}$$

The third column: for each t, evaluation of the diffusion coefficient as:

$$D_{dif} = \frac{1}{6} \frac{\partial \langle \Delta r^2(t) \rangle}{\partial t}$$

The 4-th column: root square of the MSD (average particle displacement)

5-th – 7-th columns: Evaluation of diffusion coefficient in X-, Y-, and Z-directions as:

$$D_X = \frac{\langle \Delta x^2(t) \rangle}{2t}$$

In all cases, diffusion is given in $10^{-5} cm^2/s$.

7.6 Dryrun: dryrun.f

This utility just reads the trajectories and type the coordinates of the first atom in the output. It can be used as a template to write new analyzing utilities

7.7 Lateral diffusion: latdiff.f

This utility computes two-dimensional mean square displacement in membrane-like systems and the lateral diffusion coefficients. It works essentially as `diffus` utility with a few exceptions:

- parameter

`IRT = <int.value>`

defines type of "lipid" molecules (that is molecules which build the membrane)

- if parameter

`LCOM = .true.`

is defined, then MSD of lipids are computed relative to the center-of mass motion of *each* monolayer.

- in definition of lateral diffusion, factor 1/4 is used instead of 1/6 in 3D-diffusion.

7.8 Order parameters: order.f

This utility computes the order parameters of selected molecular vectors relative to the Z-axis, as well as angular distributions of these vectors.

Compilation:

```
f77 -O3 -o order order.f tranal_base.f
```

Input parameters for this utility follow after the trajectory parameters in the following format:

- All the lines beginning with "#" after the end of trajectory section are considered as commentaries
- The first non-commentary line is the name of the output file
- The second line is an integer number.

  If this parameter is zero (or negative), angular distributions of the specified vectors are computed relative to the positive direction of the Z axis. If this parameter is positive, it defines the type of "lipid" molecules in membrane-like systems. The Z-coordinate of the center of mass of these molecules defines the membrane middle plane, and angular distributions of the molecules which are below the middle plane are calculated relative to the negative direction of the Z-axis.
- The third line is the number of order parameters to compute
- Description of molecular vectors:

  One line per each order parameter (their number is defined above) Each line can be in one of the following form:

  `<n1> <n2> 0`

  Here `<n1>` and `<n2>` are the site numbers (must belong to the same type of molecules) which define the molecular vector. Computation for this vector is done independently from others

  `<n1> <n2> 1 <nref1> <nref2>`

  followed immediately by

  `<n3> <n4>`

  Order parameters and angular distributions for molecular vectors determined by a pair of such lines are computed together. A scalar product of vectors defined by `<n1> <n2>` and `<nref1> <nref2>` is compared with scalar product defined by `<n3> <n4>` and `<nref1> <nref2>`. If the first scalar product is greater, then the vector `<n1> <n2>` is considered as going first and the vector `<n3> <n4>` next; if the second scalar product is greater, then the order in which these vectors follow is changed to opposite (as these lines change places). Note that this comparison is made for each molecule separately. Such arrangement has a sense if, for instance, two hydrogens, equivalent from the force field point of view, are distinguishable

in each specific configuration by relation to another, "reference" vector, defined by numbers `<nref1>,<nref2>`, like so-called G1R and G1S order parameters in the G1 glycerol atom of phospholipids.

- Resolution of the orientational distributions

  The last line is an integer. If 0, no orientational distributions are computed. If non-zero, it defined the number of "bins" into which the interval $[-1, 1]$ is divided for calculation of $Cos(\theta)$ distribution.

further comments.
The order parameter for a specific molecular vector is defined as

$$S = \langle \frac{3\cos^2\theta - 1}{2} \rangle$$

where θ is the angle between the molecular vector and Z-axis, and averaging is taken over all molecules and time frames. The output lists also NMR dipole coupling for the corresponding vectors as:

$$C = \langle \frac{3\cos^2\theta - 1}{2|r|^3} \rangle$$

where $|r|$ is the length of the corresponding molecular vector.

7.9 Radial distribution functions: rdf.f

This utility compute radial distribution functions between specified atom pairs.
Compilation:
`f77 -O3 -o rdf rdf.f tranal_base.f`
The input into `rdf` utility consists of a NAMELIST section `RDFIN` and subsequent lines defining sites for RDF calculation. The NAMELIST section

```
$RDFIN
parameter=value(s),
...
$END
```

contains the following parameters

- `FOUTRDF = <filename>`

  The name of the output file

- `NRDF = <int.value>`

  number of RDFs to compute.

- `RDFCUT = <value>`

  Cut-off distance for intermolecular RDF

- `NA = <int.value>`

  Resolution of intermolecular RDF (number of bins within the interval `[0,RDFCUT]`

- `RMI = <value>`
  `RMAX = <value>`

  Boundaries for intramolecular RDFs

- `NAI = <int.value>`

  Resolution of intramolecular RDF (number of bins within the interval `[RMI,RMAX]`

Then a number of lines follows defining the sites for RDFs. In this section all the lines beginning with "#" are considered as commentaries. The rules to specify RDFs are the same as in MDynaMix program.

Each (of `NRDF`) RDFs is specified by a single or by several pairs of sites. If RDF is specified by a single pair of sites, two number defining these sites are written in the input. If an RDF needs to be specified by several pair of sites (in order to average them), it is written in the following way:

```
&<num of pairs>
<n1-1>  <n2-1>
<n1-2>  <n2-2>
...

(<num of pairs> times)
```

RDFs between sites `<n1-1> <n2-1>` ... are averaged and counted as a single RDF. For example, for water there exists 3 RDFs (`NRDF=3`) and the list of sites may look like (assuming O as a first atom and hydrogens as 2 and 3):

```
1  1
&2
1  2
1  2
&3
2  2
2  3
3  3
```

7.10 Spatial distribution functions: sdf.f

This utility computes spatial distribution functions and creates a file which can be visualized using gOpenMol package (http://.www.csc.fi/gopenmol/).

Compilation:

```
f77 -O3 -o sdf sdf.f tranal_base.f
```

The input into `sdf` utility consists of a NAMELIST section `SDFIN` and subsequent lines defining sites which will be displayed for visualization of the molecule around which SDF is calculated

The NAMELIST section

```
$SDFIN
parameter=value(s),
...
$END
```

contains the following parameters

- `FILORI = <filename>`

  The name of the output file with SDF written as "formatted" plt-file for gOpenMol package. Before visualization, it must be converted to the binary using "pltfile" utility of gOpenMol (available also from gOpenMol menu).

- `FILCRD = <filename>` The name of file into which average coordinates of molecule, defining the local basis, are written. These coordinates are computed using the basis of the molecule itself, and they in fact used for visualization of the local basis. This file is written in "XMOL" format. The last column shows the variance (average deviations) of atoms around their average positions in the local coordinate system, which may serve also for evaluation of molecular flexibility.

- `NOI = <int.value>` Defines the number of equivalent local coordinate systems on the molecule (for example, for DNA SDFs can be defined around each of phosphate group used as a local basis)

- `IO1 = <n1_1>,<n1_2>,...,<n1_NOI>`
  `IO2 = <n2_1>,<n2_2>,...,<n2_NOI>`
  `IO3 = <n3_1>,<n3_2>,...,<n3_NOI>`

  `NOI` values in each of these three lines. They define `NOI` local basises. Each basis is defined by three sites (for example, `<n1_1>,<n2_1>,<n3_1>`. The center of local coordinate system is set at the first site, the X-axis goes as a mediana of 2-1-3 angle, and the Z-axis is perpendicular to the plane defined by these three sites.

- `NSOR = <int.value>`

  Number of sites for calculation of SDF.

- `ISOR = <n1>,<n2> ...`

  `NSOR` integer values. They define SDF of which sites to compute, and then take an average over them.

- `RXMAX = <value>`
  `RYMAX = <value>`
  `RZMAX = <value>`

  These parameters define the box inside which SDF is calculated (for example, from `-RXMAX` to `RXMAX` in X-direction)

- `NOMX = <value>`
  `NOMY = <value>`
  `NOMZ = <value>`

  resolution (number of bins) in each direction

After the NAMELIST section, there is a section in the input file which describes which atoms are displayed in the center of SDF as a structure defining the local basis. The first line of this section may be either an integer number or the word "`ALL`".

If an integer number is specified, it defines the number of atoms displayed in the central structure. Then this number of lines follows, each line per atom, each consisting of `NOI + 1` values.

The first position in each line is one of numbers 1,2,3. Positions from 2 to `NOI + 1` define sites which are to be displayed, for each of `NOI` equivalent local coordinate system.

The number in the first position of each line means the following:

1 - coordinates of this atom are averaged independently of the next atom

2 - average coordinates of this atom are calculated taking into account the next atom also (for example, if these are two hydrogens of water)

3 - average coordinates of this atom are calculated taking into account the next two atoms (for instance, treating hydrogens in CH3 group)

Example of these lines for water:

```
1 1
2 2
1 3
```

and for methanol (see the order of atoms in CH3OH.mmol file)

```
1  1
1  2
1  3
3  4
2  5
1  6
```

If parameter `ALL` is specified in the first (and the only) line of this section, then all the atoms of the molecule defining the local basis are displayed, without any special treatment of averaging coordinates of equivalent atoms.

7.11 Distribution of torsion angles: torsion.f

This utility computes distributions of torsion angles. Several torsion angles can be unified in a group, for which average distribution of the torsion angle is calculated. Distributions for several such groups can be built at a time. For each group, fraction of "gauche" and "trans" conformations is also printed in the output.

Compilation:

```
f77 -O3 -o torsion torsion.f tranal_base.f
```

Input parameters for this utility follow after the trajectory parameters in the following format:

- All the lines beginning with "#" after the end of trajectory section are considered as commentaries
- The first non-commentary line is the name of the output file
- The second line contains two integer numbers:

  1) number of groups of torsion angles 2) resolution (number of bins)
- The third line (a list of integer numbers) specifies how many torsion angles are in the each group
- In the next lines, site numbers of each torsion are given (four integer numbers for each torsion at each separate line). First, torsions of the first group are listed, then of the second one and so on.

7.12 Ramachandra plot for a pair of torsion angles: torsion2.f

This utility computes two-dimensional distribution for a couple of torsion angles (Ramachandra plot). Distribution of several pairs of torsions can be averaged.

Input parameters for this utility follow after the trajectory parameters in the following format:

- The first non-commentary line is the name of the output file followed by an optional key "tab". If "tab" key specified, the output is given as NxN table of the distribution. Otherwise, the output is given in three columns (as required for e.g. *gnuplot*):

  ```
  <first angle>   <second angle>  <probability>
  ```
- The second line contains two integer numbers:

  1) number of pairs of torsion angles 2) resolution (number of bins)
- Then site numbers of each torsion are given (four integer numbers for each torsion at each separate line). First, torsions of the first pair are listed, then of the second and so on.

7.13 Time correlation functions: trtcf.f

This utility computes some time correlation functions. For a molecular vector $\vec{n}(t)$, the time correlation function can be determined as:

$$c(t) = \frac{\langle \vec{n}(t_0) \cdot \vec{n}(t_0 + t) \rangle}{\langle \vec{n}(t_0)^2 \rangle} \tag{14}$$

where averaging is taken over all molecules of the given type and all acceptable initial times t_0: $t_{beg} \leq t_0 \leq t_{end} - t$, where t_{beg} and t_{end} are initial and final time of a *continuous* part of the whole trajectory respectively. A trajectory is regarded as continuous if each next configuration differs from the previous by no more than parameter `BREAKM` defined in the trajectory part (`TRAJ`) of the input file.

The program can compute the following time correlation functions:

- Molecular translational velocity (velocity of the center of mass) and their X,Y,Z components in the laboratory or in the molecular coordinate systems
- Molecular angular velocity and their X,Y,Z components in the laboratory or in the molecular coordinate systems
- Dipole moment and its second Legendre polynomial
- Of a chosen molecular vector and its second Legendre polynomial

The second Legendre polynomial TCF is defined as:

$$c(t) = \frac{\langle L_2(\vec{n}(t_0) \cdot \vec{n}(t_0 + t)) \rangle}{\langle L_2(\vec{n}(t_0)^2) \rangle} \tag{15}$$

where $L_2(x) = 0.5(3\cos^2 x - 1)$

Compilation:

```
f77 -O3 -o trtcf trtcf.f tranal_base.f
```

Note that compilation of `trtcf.f` requires, in addition to the common header file `tranal.h`, another header file `trtcf.h`, which may need to be edited in order to set array borders matching the studied system

Input parameters for this utility follow after the trajectory parameters in the NAMELIST block `TCF`:

```
$TCF
parameter=value(s),
...
$END
```

The following parameters are used:

- `FILTCF = <filename>`

  Defines the name of the output file

- `NSTEG = <int.num>`

  Number of time steps in the time correlation functions

- `DTCF = <value>`

  TCF time step

  It is recommended that TCF time step be equal to the trajectory time step multiplied by parameter `ISTEP` defined in the trajectory part of the input. The total time of tracking TCF is `NSTEG*DTCF`

- `ITCF = <i1>,<i2>,...,<i12>`

  Flags (integers) specifying which TCF to compute (zero value - do not compute) Meanings of the flags are following:

  - `<i1>` - linear velocity of molecular center of mass
  - `<i2>` - angular molecular velocity
  - `<i3>` - dipole moment
  - `<i4>` - second Legendre polynomial of dipole moment
  - `<i5>` - a specific molecular vector (see below)
  - `<i6>` - second Legendre polynomial of a specific vector
  - `<i7>,<i8>,<i9>` - X, Y, and Z projections of the linear molecular velocity. The projections are calculated in the laboratory frame if the flags are equal to 1, and they are calculated in the principal molecular frame determined by the axis of the inertia tensor if the corresponding flags are equal to 2.
  - `<i10>,<i11>,<i12>` - X, Y, and Z projections of the angular molecular velocity. The projections are calculated in the laboratory frame if the flags are equal to 1, and they are calculated in the principal molecular frame determined by the axis of the inertia tensor if the corresponding flags are equal to 2.

- `N1 = <n1-1>,<n2-2>,...`
  `N2 = <n2-1>,<n2-2>,...`

  `NTYPES` values in each of the two lines, one per each molecular type. These values specify sites (two per each molecule) which define the molecular vector whose TCF is calculated