

PIMD: User's guide

version 2.7

M. Shiga (*Japan Atomic Energy Agency*)

shiga.motoyuki@jaea.go.jp

May 12, 2026

PIMD is an open-source software package written in Fortran 90 (f90) with Message Passing Interface (MPI) support, designed to perform molecular simulations efficiently on parallel computing architectures. It provides a broad spectrum of simulation methods, ranging from fundamental techniques such as static calculations, normal mode analysis, minimum energy path searches, and classical molecular dynamics in various ensembles, to more advanced approaches including replica exchange hybrid Monte Carlo, path integral molecular dynamics, metadynamics, and nonadiabatic dynamics simulations. The software supports simulations based on classical force fields as well as *ab initio* and hybrid potentials (e.g., ONIOM and QM/MM) through interfaces with external electronic structure codes. PIMD is distributed under the Apache License, Version 2.0.

Copyright 2016--2026 M. Shiga. All rights reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

To the best of our knowledge, the code functions correctly. If you encounter any bugs, please do not hesitate to contact us. However, we cannot be held responsible for any consequences arising from errors in specific applications. We also ask for your understanding that, due to limited resources, we may not be able to respond to all questions or requests for detailed advice.

In version 2.7.0, an interface to N2P2 was implemented, enabling simulations using machine-learning-based neural network potentials. An interface to VASP version 6 was also added, allowing simulations based on *ab initio* density functional theory. In version 2.7.0.r2, the build system was modified following the discontinuation of the Intel *ifort* compiler.

In version 2.7.1, a local parallel computation scheme (XMPI) was introduced to accelerate path integral simulations. An interface to PFP was implemented to support simulations using universal machine

learning potentials. In addition, the GAL21 force field for metal–water interfaces [49] was added. Hardware-dependent behavior of random number generators was also fixed for Brownian chain molecular dynamics and thermostatted ring-polymer molecular dynamics simulations.

In version 2.7.2, an interface with MACE has been implemented to enable simulations using MACE machine-learning interatomic potentials. The interface supports GPU acceleration through CUDA-enabled PyTorch, making it suitable for efficient molecular dynamics and path-integral simulations.

Contents

1	Changes from old versions	13
2	Installation	14
2.1	Docker	15
3	Methods	15
3.1	Static (STATIC)	16
3.2	Geometry optimization (GEOOPT)	18
3.3	Restarts and interruptions	19
3.4	Normal mode analysis (NMA)	20
3.5	Steepest descent (SD)	21
3.6	Box optimization (BOXOPT)	21
3.7	Full optimization (FULLOPT)	22
3.8	Static elastic constants (ELASTIC)	22
3.9	Classical molecular dynamics (MD)	23
3.9.1	NVE molecular dynamics	23
3.9.2	NVT molecular dynamics	24
3.9.3	Constrained molecular dynamics	26
3.9.4	NPT molecular dynamics in a cubic box	26
3.9.5	NPT and NtT molecular dynamics in a hexahedron box	29
3.10	Path integral simulations (PI)	31
3.10.1	Path integral molecular dynamics (PIMD)	31
3.10.2	Path integral hybrid Monte Carlo (PIHMC)	33
3.10.3	Brownian chain molecular dynamics (BCMD)	35
3.10.4	Centroid molecular dynamics (CMD)	35
3.10.5	Ring polymer molecular dynamics (RPMD)	38
3.10.6	Thermostatted ring polymer molecular dynamics (TRPMD)	39
3.10.7	Centroid Intrinsic Reaction Coordinate (CIRC)	39
3.11	Replica exchange hybrid Monte Carlo (REHMC)	40
3.11.1	Temperature replica exchange hybrid Monte Carlo	40
3.11.2	Hamiltonian replica exchange hybrid Monte Carlo	42
3.12	Metadynamics (MTD)	44
3.13	Mean force dynamics (AFED)	47
3.13.1	Common keywords used in all the AFED methods (Free energy gradient)	48
3.13.2	Descent search for free energy minima	52
3.13.3	Ascent search for free energy saddles	52
3.13.4	Automated search for free energy stationary points	53
3.13.5	Temperature accelerated molecular dynamics	54
3.13.6	Logarithmic mean force dynamics	56
3.14	String method (STRING)	58
3.15	Phonon calculation (PHONON)	59
3.16	Optimization of OM action (OMOPT)	60

3.17	User-made potentials	62
3.17.1	Testing forces	63
3.17.2	Testing virial	63
3.17.3	Testing Ewald parameters	64
3.18	Mechanochemistry	65
3.19	Nonadiabatic dynamics	65
3.19.1	Surface hopping dynamics	65
3.19.2	Mean field dynamics	68
3.20	Molecular dynamics of rigid molecules	69
3.20.1	Gentlest ascent dynamics (GAD)	71
3.21	Consecutive static calculations (SCAN)	72
4	Potentials	72
4.1	Empirical force fields	73
4.1.1	Molecular mechanics (MM)	73
4.1.2	TIP4P	76
4.1.3	Embedded atom method (EAM)	77
4.1.4	Angular dependent potential (ADP)	78
4.1.5	Pair potential	81
4.1.6	Tersoff potential	82
4.1.7	GAL21 potential	82
4.2	User-defined potentials	82
4.3	Born–Oppenheimer potentials	82
4.3.1	SMASH	83
4.3.2	ABINIT-MP	84
4.3.3	CP2K	84
4.3.4	DFTB – Internal	87
4.3.5	DFTB - External	90
4.3.6	GAMESS	91
4.3.7	GAUSSIAN	93
4.3.8	MOLPRO	94
4.3.9	MOPAC	96
4.3.10	NTCHEM	97
4.3.11	ORCA	98
4.3.12	PHASE/0	99
4.3.13	QUANTUM ESPRESSO	101
4.3.14	TURBOMOLE	104
4.3.15	VASP6 Interface	106
4.3.16	VASP - General Words of Caution	107
4.3.17	VASP5	107
4.4	Machine learning potentials	109
4.4.1	AENET	110
4.4.2	MTP	113
4.4.3	N2P2	114
4.4.4	MACE	116
4.4.5	ASE calculator interface	121
4.4.6	PFP	123

5	Hybrid potentials	123
5.1	QM/MM	123
5.1.1	QM/MM setup	123
5.1.2	QM/MM with SMASH	125
5.1.3	QM/MM with other codes	126
5.1.4	QM/MM with BEST method	126
5.1.5	Multiple time scale QM/MM method	127
5.2	ONIOM	127
5.2.1	ONIOM setup	128
5.2.2	ONIOM with any codes	130
5.2.3	ONIOM with BEST method	130
6	Data analysis	131
6.1	Visualization of trajectory	131
6.2	Statistical analysis	132
7	Codes	133
7.1	Main codes	133
7.2	Codes for post-processing	134
7.3	Subsidiary codes	134
7.4	Utility codes	135
7.5	External calls	135
8	Classical force field	135
8.1	How to convert TINKER files	136
8.2	Preparation of initial structure	138
8.3	How to use CHARMM force field	140
9	Keywords	141
9.1	Necessary keywords in <i>input.dat</i>	141
9.1.1	<ensemble>	141
9.1.2	<ipotential>	142
9.1.3	<method>	143
9.1.4	<natom>	144
9.1.5	<nspec>	144
9.2	Important keywords in <i>input.dat</i>	144
9.2.1	<bath.type>	144
9.2.2	<dt>	145
9.2.3	<iboundary>	145
9.2.4	<iprint_rest>	146
9.2.5	<iprint_std>	146
9.2.6	<iprint_trj>	146
9.2.7	<iread_exit>	146
9.2.8	<nbead>	146
9.2.9	<nstep>	146
9.2.10	<temperature>	147
9.3	Frequently used keywords in <i>input.dat</i>	147
9.3.1	<beadsread>	147
9.3.2	<corrections>	147
9.3.3	<input_style>	148
9.3.4	<iprint_dcd>	148
9.3.5	<iprint_xsf>	148

9.3.6	<iprint_xyz>	148
9.3.7	<ncolor>	148
9.3.8	<nnhc>	148
9.3.9	<np_beads>	148
9.3.10	<np_force>	148
9.3.11	<nref>	149
9.3.12	<nsymbol>	149
9.3.13	<nys>	149
9.3.14	<pimd_command>	149
9.3.15	<pressure>	149
9.3.16	<time_bath>	150
9.4	Specific keywords in <i>input.dat</i>	150
9.4.1	<abinit_mp_exe_command>	150
9.4.2	<abinit_mp_input>	150
9.4.3	<abinit_mp_output>	150
9.4.4	<abinit_mp_reuse_mo>	150
9.4.5	<alchem_dat_dir>	151
9.4.6	<alchem_potential>	151
9.4.7	<alchem_scr_dir>	151
9.4.8	<auto_string>	151
9.4.9	<box_anis>	151
9.4.10	<cells_phonon>	151
9.4.11	<cluster>	152
9.4.12	<cp2k_lib_calcforce>	152
9.4.13	<cp2k_lib_chargeout>	152
9.4.14	<cp2k_lib_output>	152
9.4.15	<cut_rec_3d>	152
9.4.16	<delay_aenet>	152
9.4.17	<dftb_exe_command>	152
9.4.18	<dftb_lib_output>	153
9.4.19	<dftb_version>	153
9.4.20	<dir_save_aenet>	153
9.4.21	<dir_save_n2p2>	153
9.4.22	<dir_save_mace>	153
9.4.23	<dir_train_aenet>	153
9.4.24	<dir_train_mace>	153
9.4.25	<dosrange_phonon>	153
9.4.26	<dt_conv_gad>	154
9.4.27	<dt_om>	154
9.4.28	<dt_poly>	154
9.4.29	<dtemp_meta>	154
9.4.30	<dual_dat_dir>	154
9.4.31	<dual_potential>	154
9.4.32	<dual_scr_dir>	154
9.4.33	<efei>	154
9.4.34	<ends_poly>	154
9.4.35	<ends_string>	155
9.4.36	<eps_best>	155
9.4.37	<equation_om>	155
9.4.38	<fc_best>	155
9.4.39	<fdiff>	155
9.4.40	<g03_command>	156

9.4.41	<g03_dip>	156
9.4.42	<g03_grad>	156
9.4.43	<g09_command>	156
9.4.44	<g09_dip>	156
9.4.45	<g09_grad>	156
9.4.46	<g09_oniom>	156
9.4.47	<g16_command>	156
9.4.48	<g16_dip>	156
9.4.49	<g16_grad>	157
9.4.50	<g16_oniom>	157
9.4.51	<g98_command>	157
9.4.52	<g98_dip>	157
9.4.53	<g98_grad>	157
9.4.54	<gamess_command>	157
9.4.55	<gamma_gad>	157
9.4.56	<gamma_om>	157
9.4.57	<generate_aenet>	157
9.4.58	<gh_meta>	158
9.4.59	<guess_poly>	158
9.4.60	<gw_meta>	158
9.4.61	<iatom_qtst>	158
9.4.62	<iformat_trj>	158
9.4.63	<iformat_xyz>	158
9.4.64	<igamma>	159
9.4.65	<ikind_best>	159
9.4.66	<ikind_xyz>	159
9.4.67	<integrator_bcmd>	159
9.4.68	<integrator_trpmd>	159
9.4.69	<iobest>	159
9.4.70	<ioption_best>	159
9.4.71	<ioption_meta>	159
9.4.72	<ioption_xsf_aenet>	160
9.4.73	<ioption_xsf_n2p2>	160
9.4.74	<ioption_xsf_mace>	160
9.4.75	<iorder_hmc>	160
9.4.76	<iprint_akin>	160
9.4.77	<iprint_alc>	160
9.4.78	<iprint_best>	161
9.4.79	<iprint_bond>	161
9.4.80	<iprint_box>	161
9.4.81	<iprint_cavg>	161
9.4.82	<iprint_cons>	161
9.4.83	<iprint_cv_meta>	161
9.4.84	<iprint_cv_tass>	161
9.4.85	<iprint_dip>	161
9.4.86	<iprint_dual>	162
9.4.87	<iprint_eavg>	162
9.4.88	<iprint_hfx_aenet>	162
9.4.89	<iprint_hfx_n2p2>	162
9.4.90	<iprint_mech>	162
9.4.91	<iprint_meta>	162
9.4.92	<iprint_mfe>	162

9.4.93	<iprint_minfo>	162
9.4.94	<iprint_mom>	162
9.4.95	<iprint_nac>	163
9.4.96	<iprint_oniom>	163
9.4.97	<iprint_poly>	163
9.4.98	<iprint_qmmm>	163
9.4.99	<iprint_rdf>	163
9.4.100	<iprint_rdfbead>	163
9.4.101	<iprint_rdfcent>	163
9.4.102	<iprint_rec_meta>	163
9.4.103	<iprint_rgy>	164
9.4.104	<iprint_str>	164
9.4.105	<iprint_tfs>	164
9.4.106	<iprint_xsf_aenet>	164
9.4.107	<iprint_xsf_n2p2>	164
9.4.108	<iprint_xsf_mace>	164
9.4.109	<irandom>	164
9.4.110	<irem_type>	164
9.4.111	<istate_init>	165
9.4.112	<istep_adjust_hmc>	165
9.4.113	<istep_ax_hmc>	165
9.4.114	<istep_hmc>	165
9.4.115	<istep_max_hmc>	165
9.4.116	<istep_mul_hmc>	165
9.4.117	<iprint_rec_tass>	165
9.4.118	<istep_save_aenet>	165
9.4.119	<istep_save_n2p2>	165
9.4.120	<istep_save_mace>	166
9.4.121	<iprint_tass>	166
9.4.122	<istep_train_aenet>	166
9.4.123	<istep_train_n2p2>	166
9.4.124	<istep_train_mace>	166
9.4.125	<ivar_qmmm>	166
9.4.126	<joption_meta>	166
9.4.127	<kdisp_phonon>	166
9.4.128	<kdos_phonon>	166
9.4.129	<kickstep_bcmd>	167
9.4.130	<lstep_train_aenet>	167
9.4.131	<lstep_train_n2p2>	167
9.4.132	<lstep_train_mace>	167
9.4.133	<mace_generate_command>	167
9.4.134	<ase_infer_options_file>	167
9.4.135	<mace_infer_options_file>	167
9.4.136	<mace_train_options_file>	167
9.4.137	<mech_type>	168
9.4.138	<mg_meta>	169
9.4.139	<minxsf_train_aenet>	169
9.4.140	<minxsf_train_n2p2>	169
9.4.141	<minxsf_train_mace>	169
9.4.142	<model_water>	169
9.4.143	<molpro_command>	169
9.4.144	<mopac_command>	169

9.4.145	<n2p2.en_unit>	170
9.4.146	<n2p2.len_unit>	170
9.4.147	<ncons>	170
9.4.148	<ncycle_poly>	170
9.4.149	<neighbor_list_skin>	171
9.4.150	<ngrid_poly>	171
9.4.151	<ngrid.string>	171
9.4.152	<nmeta>	171
9.4.153	<nmulti>	172
9.4.154	<np_poly>	172
9.4.155	<nph_type>	172
9.4.156	<npoly>	172
9.4.157	<npt_type>	172
9.4.158	<nref_meta>	173
9.4.159	<nref_tass>	173
9.4.160	<nstep_modes>	173
9.4.161	<nstate>	173
9.4.162	<ntass_rec>	173
9.4.163	<nth_type>	173
9.4.164	<ntt_type>	173
9.4.165	<ntype_aenet>	174
9.4.166	<oniom.dat_dir>	174
9.4.167	<oniom.hi_potential>	174
9.4.168	<oniom.lo_potential>	174
9.4.169	<oniom.scr_dir>	174
9.4.170	<orca_command>	174
9.4.171	<params_cons>	174
9.4.172	<params_dual>	175
9.4.173	<params_lbfgs>	175
9.4.174	<params_rdf>	175
9.4.175	<params_rec_meta>	175
9.4.176	<phase0_proc>	175
9.4.177	<projcmf_poly>	176
9.4.178	<qe_input_file_name>	176
9.4.179	<qe_output>	176
9.4.180	<qmmm_embedding>	176
9.4.181	<qmmm.dat_dir>	176
9.4.182	<qmmm.potential>	176
9.4.183	<qmmm.scr_dir>	176
9.4.184	<ratio_max_hmc>>	176
9.4.185	<ratio_min_hmc>>	177
9.4.186	<scale.bcnd>	177
9.4.187	<scale.trpmd>	177
9.4.188	<scan.exe_shell>	177
9.4.189	<skin_aenet>	177
9.4.190	<smash_command>	177
9.4.191	<smash_dip>	177
9.4.192	<smash_grad>	177
9.4.193	<smash_guess>	177
9.4.194	<smash_options>	178
9.4.195	<smash_threads>	178
9.4.196	<temp_tamd_tass>	178

9.4.197	<temprange_phonon>	178
9.4.198	<temprange_rem>	178
9.4.199	<tension>	178
9.4.200	<time_baro>	178
9.4.201	<time_cv_bath>	179
9.4.202	<time_cv_meta>	179
9.4.203	<time_fc_meta>	179
9.4.204	<time_limit_meta>	179
9.4.205	<time_mode>	179
9.4.206	<train_aenet>	179
9.4.207	<turbo_command>	179
9.4.208	<turbo_guess>	179
9.4.209	<user_command>	180
9.4.210	<user_input_file>	180
9.4.211	<user_output_file>	180
9.4.212	<vasp_command>	181
9.4.213	<vasp_energy>	181
9.4.214	<vasp_keyword_energy>	181
9.4.215	<vasp_keyword_gradient>	181
9.4.216	<vasp_keyword_stress>	181
9.4.217	<vasp_output>	182
9.4.218	<vasp_reuse_wavefunction>	182
9.4.219	<xmpi>	182
9.4.220	<xtb_exe_command>	182
9.5	Optional keywords in <i>input.dat</i>	182
9.5.1	<atom_change>	182
9.5.2	<charge_libnnp>	182
9.5.3	<components>	183
9.5.4	<file_dms_libnnp>	183
9.5.5	<file_ip_libnnp>	183
9.5.6	<file_pes_libnnp>	183
9.5.7	<ioption_dms_libnnp>	183
9.5.8	<molecules>	183
9.5.9	<natom_ip_libnnp>	183
9.6	Keywords in <i>calc.dat</i>	184
9.6.1	<ncalc>	184
9.6.2	<iprint_calc>	186
9.6.3	<iprint_std_calc>	186
9.6.4	<iconf_fin_calc>	186
9.6.5	<iconf_ini_calc>	186
9.6.6	<jformat_xyz>	186
9.6.7	<jorigin_xyz>	186
9.6.8	<jprint_xyz>	186
9.6.9	<jspec_xyz>	186
9.6.10	<jxyz_atom>	187
9.6.11	<jxyz_bead>	187
9.6.12	<jxyz_origin>	187
9.6.13	<params_lin_dens>	187
9.6.14	<params_adiff_dens>	187
9.6.15	<params_angl_dens>	187
9.6.16	<params_diff_dens>	187
9.6.17	<params_dih_dens>	187

9.6.18	<3d_range_cube>	187
9.7	Keywords for MM potential in <i>mm.dat</i>	187
9.7.1	<linear_bonds>	188
9.7.2	<genlin_bonds>	189
9.7.3	<angular_bonds>	189
9.7.4	<improper_bonds>	190
9.7.5	<dihedral_bonds>	190
9.7.6	<ncmap>, <nkind_cmap>	191
9.7.7	<lennard-jones>	191
9.7.8	<charges>, <nbcpr>, <ewald>	192
9.7.9	<ewald_type>, <pme_ewald>	193
9.7.10	<charges>, <damping>, <ewald>, <ewpol>	194
9.7.11	<morse>	195
9.7.12	<buckingham>	195
9.8	Keywords for EAM potential in <i>eam.dat</i>	196
9.8.1	<nref_eam>	197
9.8.2	<rcut_eam>	197
9.8.3	<rho_eam>	198
9.8.4	<frho_eam>	198
9.8.5	<phir_eam>	198
9.8.6	<ur_adp>	199
9.8.7	<wr_adp>	199
9.9	Keywords for the Tersoff potential in <i>tersoff.dat</i>	199
9.10	Common keywords shared with constrained MD and metadynamics	201
9.10.1	<ncons>	201
9.10.2	<nref>	201
9.10.3	<params_cons>	201
9.11	Common keywords shared in all types of AFED	201
9.11.1	<afed_type>	201
9.11.2	<fenergy_max_afed>	201
9.11.3	<mdcycle_pro_afed>	201
9.11.4	<niter_afed>	202
9.11.5	<nstep_pre_afed>	202
9.11.6	<nstep_pro_afed>	202
9.11.7	<params_afed>	202
9.12	Additional keywords in DESCENT, ASCENT and AUTO	202
9.12.1	<algo_ascent_afed>	202
9.12.2	<ascent_sampling_afed>	203
9.12.3	<dt_ascent_afed>	203
9.12.4	<dt_conv_afed>	203
9.12.5	<dt_damp_afed>	203
9.12.6	<dt_descent_afed>	203
9.12.7	<fdiff_sampling_afed>	203
9.12.8	<gamma_ascent_afed>	203
9.12.9	<niter_gad2evf_afed>	203
9.12.10	<niter_refresh_afed>	204
9.12.11	<nmiss_auto_afed>	204
9.12.12	<ioption_eigen_afed>	204
9.12.13	<nshot_auto_afed>	204
9.12.14	<radius_auto_afed>	204
9.12.15	<iprint_xyz_afed>	204
9.12.16	<iprint_test_afed>	204

9.13	Additional keywords in TAMD and LogMFD	205
9.13.1	<alpha_logmfd>	205
9.13.2	<dt_logmfd>	205
9.13.3	<dt_tamd>	205
9.13.4	<fenergy_ini_logmfd>	205
9.13.5	<fenergy_ini_tamd>	205
9.13.6	<gamma_logmfd>	205
9.13.7	<ioption_afed>	205
9.13.8	<logmfd_type>	205
9.13.9	<mass_afed>	206
9.13.10	<niter_bath_afed>	206
9.13.11	<tamd_type>	206
9.13.12	<temperature_tamd>	206
9.13.13	<start_afed>	206
9.13.14	<start_therm_afed>	207
10	Files	207
10.1	Input files	207
10.1.1	<i>input.dat</i>	207
10.1.2	<i>input_default.dat</i>	207
10.1.3	<i>centroid.dat</i>	207
10.1.4	<i>eam.dat</i>	207
10.1.5	<i>tersoff.dat</i>	208
10.1.6	<i>gamess.dat</i>	208
10.1.7	<i>g03.dat</i>	208
10.1.8	<i>g98.dat</i>	208
10.1.9	<i>mm.dat</i>	208
10.1.10	<i>molpro.dat</i>	208
10.1.11	<i>mopac.dat</i>	208
10.1.12	<i>orca.dat</i>	208
10.1.13	<i>structure.dat</i>	208
10.1.14	<i>turbo.dat</i>	209
10.1.15	TAMD	209
10.1.16	LogMFD	210
10.2	Output files	210
10.2.1	<i>afed.out</i>	210
10.2.2	<i>afed.xyz</i>	211
10.2.3	<i>afed_weight.out</i>	211
10.2.4	<i>alc.out</i>	212
10.2.5	<i>best.out</i>	212
10.2.6	<i>bond.out</i>	212
10.2.7	<i>box.dcd</i>	213
10.2.8	<i>box.out</i>	213
10.2.9	<i>calc.xyz</i>	214
10.2.10	<i>charges.dcd</i>	215
10.2.11	<i>cons.out</i>	215
10.2.12	<i>cv.out</i>	215
10.2.13	<i>dip.dcd</i>	216
10.2.14	<i>dipole.out</i>	216
10.2.15	<i>dipole_scan.out</i>	216
10.2.16	<i>dual.out</i>	216
10.2.17	<i>eavg.out</i>	217

10.2.18	<i>forces.out</i>	217
10.2.19	<i>forces_scan.out</i>	218
10.2.20	<i>force.dcd</i>	218
10.2.21	<i>force_high.dcd</i>	218
10.2.22	<i>force_low.dcd</i>	218
10.2.23	<i>hessian.out</i>	218
10.2.24	<i>logmfd.out</i>	219
10.2.25	<i>mech.out</i>	219
10.2.26	<i>meta.out</i>	219
10.2.27	<i>mfe.out</i>	220
10.2.28	<i>nac.out</i>	220
10.2.29	<i>oniom.out</i>	220
10.2.30	<i>phonon_dos.out</i>	220
10.2.31	<i>phonon_energy.out</i>	221
10.2.32	<i>phonon_kdisp.out</i>	221
10.2.33	<i>phonon_kdos.out</i>	221
10.2.34	<i>pot.dcd</i>	221
10.2.35	<i>potential_scan.out</i>	222
10.2.36	<i>qmmm.out</i>	222
10.2.37	<i>rdf.out</i>	222
10.2.38	<i>rec.out</i>	222
10.2.39	<i>rgy.out</i>	223
10.2.40	<i>standard.out</i>	223
10.2.41	<i>string.out</i>	224
10.2.42	<i>string.xyz</i>	225
10.2.43	<i>string_final.out</i>	225
10.2.44	<i>template.xyz</i>	225
10.2.45	<i>tfs.out</i>	225
10.2.46	<i>trj.dcd</i>	225
10.2.47	<i>trj.out</i>	227
10.2.48	<i>trj.xyz</i>	227
10.2.49	<i>vel.dcd</i>	227
10.3	Restart files	228
10.3.1	<i>afed.ini</i>	228
10.3.2	<i>auto.ini</i>	229
10.3.3	<i>averages.ini</i>	230
10.3.4	<i>bath.ini</i>	230
10.3.5	<i>box.ini</i>	230
10.3.6	<i>cstate.ini</i>	230
10.3.7	<i>cv.ini</i>	230
10.3.8	<i>gad.ini</i>	230
10.3.9	<i>geometry.ini</i>	230
10.3.10	<i>hills.ini</i>	231
10.3.11	<i>step.ini</i>	231
10.3.12	<i>string.ini</i>	231
11	Theoretical background	232
11.1	Mechanochemistry	233
11.2	Steepest descent	233
11.3	Box optimization	234
11.3.1	Zero pressure condition	234
11.3.2	Finite pressure condition	234

11.3.3	Finite tension condition	234
11.4	Static elastic constants	235
11.5	Normal mode analysis	235
11.6	Phonon calculations	236
11.7	String method	237
11.8	OM action	237
11.9	NVE molecular dynamics	238
11.9.1	Velocity-Verlet algorithm	238
11.10	NVT molecular dynamics	239
11.10.1	Nosé-Hoover method	239
11.10.2	Nosé-Hoover chain method	240
11.10.3	Massive Nosé-Hoover chain method	242
11.11	Path integral methods	244
11.11.1	Fictitious mass	245
11.11.2	Thermostats	246
11.11.3	Barostats	248
11.11.4	How to set up RPMD and CMD	249
11.11.5	Dual-level hybrid Monte Carlo	249
11.11.6	Fourth order path integral hybrid Monte Carlo	249
11.11.7	Translational and rotational corrections	250
11.12	Replica exchange hybrid Monte Carlo	251
11.12.1	Temperature replica exchange	251
11.12.2	Hamiltonian replica exchange	252
11.13	Metadynamics	252
11.14	Conventional and well-tempered metadynamics	254
11.14.1	Coordination number	255
11.15	Well-sliced metadynamics	255
11.16	Mean force dynamics (AFED)	257
11.16.1	Steepest descent	258
11.16.2	Gentlest ascent dynamics	260
11.16.3	Automated search	263
11.16.4	Temperature accelerated molecular dynamics	264
11.16.5	Logarithmic mean force dynamics	266
11.17	Nonadiabatic dynamics	267
11.18	Hybrid potentials: ONIOM and QM/MM	268
11.19	Multiple time scale method	269
11.20	BEST method	271
11.21	Ewald sum	271
11.22	Polarizable force field	272
11.23	Dihedral bond potential	274

12 Publications

275

1 Changes from old versions

Starting from version 2.2.0, a new initial geometry format has been introduced. The previous file, *centroid.dat*, has been replaced by a new file called *structure.dat*. The format of *structure.dat* follows an xyz-like format, which is explained in Section 10.1.13, and includes information on the number of atoms and their atomic species. As a result, the keywords `<natom>` and `<nspec>` are no longer required in the *input.dat* file, except in certain special cases.

It is still possible to use the old input format, *centroid.dat*, together with the keywords `<natom>` and `<nspec>`, by specifying the keyword `<input_style> = OLD`. However, the default input style is now `<input_style> = NEW`, which uses the new *structure.dat* file format.

This change was implemented to facilitate the use of hybrid potentials (QM/MM, ONIOM) and methods that employ collective variables (constrained MD, metadynamics, AFED).

2 Installation

First, decompress the tar file:

```
> tar xvfz pimd-x.x.x.tar
```

In the main directory, the following subdirectories should be present:

```
> ls
examples webpage lib makefiles manual source tools
>
```

Next, create a new directory in which the compilation will be performed:

```
> mkdir compile
```

and place all necessary files there. First, copy all source files from the *source* directory to the *compile* directory:

```
> cp source/* compile/
```

Here, *makefile* and *makefile.inc* are required. For specific platforms, suitable versions are available in the *makefiles* directory. For other platforms, they must be prepared by the user. Second, BLAS and LAPACK libraries are required. They can be downloaded from the NETLIB website: Download the appropriate versions for your computer system. These libraries are mainly used for matrix diagonalization in the *diag.F* routine.

To compile the code, type:

```
> cd compile; make veryclean; make; cd ..
```

If successful, the serial executable *pimd.x* and the parallel executable *pimd.mpi.x* will be created. If not, edit the *makefile* as needed and try again.

NOTE: If you encounter the following type of error, it is likely that the LAPACK library is not linked correctly:

```
diag.o:diag.F(.test+0x488): undefined reference to 'zggev'
diag.o:diag.F(.test+0x9f8): undefined reference to 'dsyev'
```

If linking the LAPACK library is difficult, you may use an internal routine for diagonalization of real symmetric matrices by including `LAPACK = -Dnolapack` in the *makefile*. In this case, however, diagonalization of complex matrices is not available, as it is not supported by the internal routine. Consequently, some methods that require diagonalization of complex matrices, such as phonon calculations and nonadiabatic dynamics, cannot be used when the “LAPACK = -Dnolapack” option is employed.

NOTE: If an error occurs while compiling *fftpack.F*, use the “-w” option in *OPTS* or enable *PME* = *-Dnopme*.

Finally, copy all executable files with names ending in **.x* (including *pimd.x* and *pimd.mpi.x*) to a directory (*/xxx*) that is included in your PATH. In bash, this can be done as follows:

```
echo "export PATH=$PATH:/xxx" >> ~/.bashrc
```

You are now ready to run calculations.

2.1 Docker

This section describes how to use a preinstalled PIMD Docker image linked with QE and AENET.

- **Step 1:** Download Docker from the website appropriate for your operating system, for example, macOS:

```
https://hub.docker.com/editions/community/docker-ce-desktop-mac/
```

- **Step 2:** Create a working directory for Docker, for example, `test`:

```
mkdir ~/test
cd ~/test
```

- **Step 3:** Download the PIMD Docker image from the following website:

```
https://hub.docker.com/r/cometscome/pimd\_aenet\_qe
```

To download the image, open a terminal and type:

```
docker pull cometscome/pimd_aenet_qe
```

- **Step 4:** Start the Docker container by typing:

```
docker run --shm-size=2gb -v ~/test:/home/pimder/home --rm -it cometscome/pimd_aenet_qe
```

The shared memory size should be adjusted depending on the number of parallel processes you plan to use. After starting the container, the `test` directory can be accessed at `/home/pimder/home` inside Docker.

- **Step 5:** You are ready to go. Confirm that PIMD runs correctly by executing the following examples:

```
cd ~/pimd/examples/SiO2/qe_md/
mpirun -np 2 pimd.mpi.x
```

```
cd ~/pimd/examples/SiO2/aenet_pimd_nvt/
mpirun -np 2 pimd.mpi.x
```

```
cd ~/pimd/examples/SiO2/qe_slhmc/
mpirun -np 2 pimd.mpi.x
```

3 Methods

To verify that the setup is correct, it is recommended to begin with simple test calculations. In this section, the keywords required for each simulation method are explained through a series of input examples. In most examples, a single water molecule described by the flexible simple point charge (flexible-SPC) potential [43] is used.

The flexible-SPC potential can be specified using the keyword:

```
<ipotential>
WATER
```

This is a special case in which the potential is implemented internally in the PIMD code. For all other choices of `<ipotential>`, additional files defining the potential must be prepared by the user. These files and their formats are explained in Section 4.

3.1 Static (STATIC)

As an example, a set of input files for a static calculation of flexible-SPC water is available. Therefore, let us copy it to a new directory called *run*:

```
> mkdir run
> cp -r examples/H2O/water_static run
```

and navigate to that directory:

```
> cd run/water_static/
```

Here, three files are available: *input.dat*, *input_default.dat*, and *structure.dat*. In ALL calculations, two input files, *input.dat* and *input_default.dat*, must be prepared in the execution directory. The file *input_default.dat* is always available from the *examples* directory. The input files contain the main information needed to set up the calculation. It is expected that *input.dat* is edited by the user, while *input_default.dat* is kept unchanged. The file *input_default.dat* includes all default values that are not specified in *input.dat*. This means that when running the calculation, the code first looks for the necessary data in *input.dat*, and if the data is not found there, the code searches for it in *input_default.dat*. Therefore, *input.dat* does not have to contain the full set of data if the default setup is satisfactory.

Some additional files with the name **.dat* may be necessary depending on the conditions specified mainly by *input.dat*. The **.dat* files should also be placed in the execution directory unless otherwise specified. For instance, in order to start a new calculation, the initial-geometry file called *structure.dat* must be prepared. The file *structure.dat* consists of a list of Cartesian coordinates of atoms:

```
N
unit
a1, x1 y1 z1 n1
a2, x2 y2 z2 n2
...
```

where *N* is the number of atoms, *unit* is the unit of the atomic coordinates (“BOHR” or “ANGSTROM”), and *ai*, *xi*, *yi*, *zi*, *ni* are the chemical element, the *x*, *y*, and *z* components of the atomic coordinates, and the atomic kind of atom *i*, respectively.

Before running, let us examine the contents of *input.dat*. You will find a list of keywords enclosed in brackets *<...>*, followed by a few lines that contain the essential data. The remaining parts of this file are just comments, so you can keep them, remove them, or modify them as you prefer. The order of appearance of the keywords is arbitrary. The details of each keyword are listed in Section 9.

Let us examine the keywords one by one. First,

```
<method>
STATIC
```

This keyword sets the simulation method. In this case, “STATIC” corresponds to a static calculation.

```
<ipotential>
WATER
```

This keyword sets the potential model. In this case, “WATER” corresponds to the flexible-SPC water potential (the order should be O, H, H in this special case).

```
<iboundary>
0
```

This keyword sets the boundary condition. In this case, “0” means the free boundary condition, i.e., an isolated system. To begin the calculation, type


```
> pimd.x
```

Then the following message will appear on the screen:

```
+++++
```

PIMD version x.x.x

Author: M. Shiga

Last Updated: Xxx. xx, 20xx.

Copyright(C) 20xx M. Shiga All rights reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

```
+++++
```

Method: static.

Model: water.

Ensemble: none.

Atomic positions read from structure.dat. Molecular configuration initialized.

Free boundary condition - isolated system.

```
=====
```

bead	potential energy values		
	hartree	kcal/mol	kJ/mol
1	0.00036497	0.229023	0.958230

```
-----
```

Normal termination of pimd.

To start the calculation again, remove the output files **.out*, trajectory files **.xyz*, and the restart files **.ini* before executing. To run the calculation as a background job, type

```
> pimd.x >> monitor.out &
```

The job will end by creating an output file named *forces.out* that contains energy and force data, as explained in Section 10.2.

3.2 Geometry optimization (GEOOPT)

Geometry optimization is performed using the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [2, 3].

As an example, a set of input files for the flexible-SPC water model is provided. First, copy the example to the directory *run*:

```
> cp -r examples/H2O/water_geoopt run
```

and move to that directory:

```
> cd run/water_geoopt/
```

To prepare the file *input.dat*, all keywords described in Section 3.1 are required. In addition, the following keywords are important.

```
<method>  
GEOOPT
```

This keyword sets the simulation method. In the present case, **GEOOPT** specifies a geometry optimization.

```
<nstep>  
10
```

This keyword sets the maximum number of optimization steps. Here, the maximum number of geometry optimization steps is set to 10.

```
<iread_exit>  
1
```

This keyword sets the interval (in steps) at which the code checks for the existence of the file *exit.dat* to allow for a soft exit (see Section 3.3). In this example, the check is performed at every step.

```
<iprint_std>  
1
```

This keyword sets the print interval of the standard output file *standard.out*. Here, output is printed at every step.

```
<iprint_rest>  
1
```

This keyword sets the interval for generating (and overwriting) the restart files **.ini*. In this case, restart files are written at every step.

```
<iprint_trj>  
1
```

This keyword sets the print interval of the standard trajectory file *trj.out*. Here, the trajectory is written at every step.

```
<iprint_xyz>  
1
```

This keyword sets the print interval of the trajectory file in XYZ format, *trj.xyz*. In this example, the XYZ trajectory is written at every step.

```
<params_lbfgs>
  1.d-4, 1.d-5, 1.d-5, 1.d-6
```

This keyword specifies the convergence criteria for the L-BFGS algorithm. The four values correspond to:

- the maximum atomic displacement (bohr),
- the root-mean-square (RMS) atomic displacement (bohr),
- the maximum force component (hartree/bohr),
- the RMS force (hartree/bohr).

Before starting the calculation, make sure that the files *input.dat*, *input_default.dat*, and *structure.dat* are present in the execution directory. To run the geometry optimization, type:

```
> pimd.x
```

During execution, information similar to the following will be printed to the screen:

```
=====
```

step	energy [au]	max displ	rms displ	max force	rms force
		0.00001	0.00000	0.000100	0.000010

0	0.00036497	0.00000	0.00000	0.009221	0.001357
1	0.03310297	0.75483	0.05153	0.038251	0.006037
2	0.00021197	0.71993	0.04915	0.008308	0.001669
3	0.00005349	0.01012	0.00117	0.003593	0.000618
4	0.00000292	0.01351	0.00151	0.000759	0.000114
5	0.00000005	0.00335	0.00040	0.000184	0.000026
6	0.00000000	0.00046	0.00004	0.000007	0.000001
7	0.00000000	0.00001	0.00000	0.000000	0.000000

Upon convergence, restart files such as *geometry.ini* and *step.ini* are written, and the program terminates normally.

The output files **.out* and the restart files **.ini* are described in Section 10.2 and Section 10.3, respectively.

NOTE: In some cases, the GEOOPT calculation may stop with an error message indicating that the convergence criteria are too strict. If this happens, it is recommended to relax the convergence thresholds by increasing the values specified in the keyword `<params_lbfgs>`.

3.3 Restarts and interruptions

For each run, restart files are automatically created at the end of the calculation, unless this feature is intentionally disabled by setting the keyword `<iprint_rest>` to -1. To prepare for unexpected interruptions, such as a sudden power outage, restart files can also be written periodically during the run. This is controlled by the keyword `<iprint_rest>`, which specifies the print interval of the restart files.

It is strongly recommended to save restart files as well as output files from time to time, for example:

```
> cp *.ini *.out *.xyz somewhere
```

IMPORTANT: Restart files are always overwritten, replacing older ones.

If you wish to stop a job during execution, the *soft exit* mechanism can be used. To do this, simply create an empty file named *exit.dat* in the execution directory:

```
> touch exit.dat
```

During the run, the code periodically checks for the existence of *exit.dat*, with a frequency specified by the keyword `<iread_exit>`. When *exit.dat* is detected, the job terminates as soon as possible after generating the complete set of restart files. The file *exit.dat* is automatically removed at the end of the job, so that restarting the next run is straightforward.

NOTE: In some cases, especially for MM potentials, frequent disk access to check *exit.dat* may slow down the calculation. This behavior can be controlled by adjusting the keyword `<iread_exit>`. For example, setting `<iread_exit> = 100` causes the check to be performed every 100 steps. The soft exit mechanism can be completely disabled by setting `<iread_exit> = -1`.

The current step number is written to a restart file called *step.ini*. To restart a calculation, the number of steps specified by the keyword `<nstep>` in the file *input.dat* must always be larger than the current step recorded in *step.ini*. Therefore, to extend a trajectory after a job has completed, `<nstep>` must be increased accordingly. If this is not done, the restarted job will terminate immediately, since the target number of steps has already been reached.

To restart the job, simply execute:

```
> pimd.x
```

or, to run it in the background,

```
> pimd.x >> monitor.out &
```

At the beginning of the run, the code automatically searches for restart files such as *geometry.ini*, *step.ini*, and others in the execution directory. If these files are not found, the default initial conditions are used for the restarted calculation.

3.4 Normal mode analysis (NMA)

In normal mode analysis, the Hessian matrix is obtained numerically as the second derivative of the potential energy surface with respect to atomic positions.

As an example, a set of input files for flexible-SPC water is available, so copy it to the *run* directory using the following command:

```
> cp -r examples/H2O/water_nma run
```

Then, navigate to that directory:

```
> cd run/water_nma/
```

Before running the simulation, the optimized structure should be prepared. This can be done either manually by creating a *structure.dat* file or by using the restart file *geometry.ini*, which is produced at the end of a GEOOPT calculation.

To prepare the input file *input.dat*, all the keywords in Section 3.1 are required. In addition, the following keywords must be specified:

```
<method>
NMA
```

This keyword sets the simulation method; in this case, it corresponds to normal mode analysis.

```
<fdiff>
1.d-4
```

This keyword sets the finite difference shift of the geometry in bohr, which is used to compute the Hessian matrix. Here, a value of 10^{-4} bohr is used. The results are written to the output file named *nma.out* in a GAUSSIAN 98-like format.

3.5 Steepest descent (SD)

From a given starting point, the steepest descent trajectory is computed as the minimum energy path in the mass-weighted coordinate space. When the starting geometry is a saddle point, the trajectory corresponds to the intrinsic reaction coordinate (IRC). The theoretical background of this method is briefly explained in Section 11.2.

As an example, a set of input files for flexible-SPC water is available. To use it, copy the files to the directory *run*:

```
> cp -r examples/H2O/water_sd run
```

and enter that directory:

```
> cd run/water_sd
```

To prepare the file *input.dat*, all the keywords in Sections 3.1 and 3.2 are important. In addition, the following keywords are also required:

```
<method>  
SD
```

This keyword sets the simulation method. In the present case, "SD" corresponds to the steepest descent method.

```
<nstep>  
10
```

This keyword sets the number of steps. In an SD calculation, this corresponds to the number of steepest descent steps.

```
<dt>  
0.01d0
```

This keyword sets the SD step size in units of hartree^{0.5} femtoseconds. The output files named **.out* and the restart files named **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.6 Box optimization (BOXOPT)

Box optimization can be performed for a system with periodic boundary conditions. The theoretical background of this method is briefly explained in Section 11.3. The limited-memory Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [2, 3] is used.

As an example, a set of input files for the face-centered-cubic nickel crystal using the EAM potential is available. Let us copy it to the directory *run*:

```
> cp -r examples/Ni/eam_boxopt run
```

and enter that directory:

```
> cd run/eam_boxopt
```

To prepare the file *input.dat*, all the keywords in Sections 3.1 and 3.2 are important. In addition, the following keywords are also important.

```
<method>  
BOXOPT
```

This keyword sets the simulation method. In the present case, "BOXOPT" corresponds to box optimization.

```
<iprint_box>
1
```

This keyword sets the print interval of the box matrix to the file `box.out`. In the present case, “1” means every step.

```
<iprint_rest>
1
```

This keyword sets the print interval of the restart files `*.ini`. In the present case, “1” means every step.

```
<fdiff>
1.d-4
```

This keyword sets the finite difference shift of the box matrix elements in Bohr, which is used to compute the virial matrix. In the present case, 10^{-4} Bohr is used.

```
<iboundary>
1
33.20000000  0.00000000  0.00000000
 0.00000000 33.20000000  0.00000000
 0.00000000  0.00000000 33.20000000
```

This keyword sets the boundary condition and the initial box. In the present case, “1” means periodic boundary conditions, and the following three lines define the matrix of the initial box. Here, a cubic box with a side length of 33.2 Bohr is used. The output files `*.out` and the restart files `*.ini` will be explained in Section 10.2 and Section 10.3, respectively.

3.7 Full optimization (FULLOPT)

To optimize both the box and geometry, one can use the following method:

```
<method>
FULLOPT
```

3.8 Static elastic constants (ELASTIC)

The static elastic constants can be computed using the following method:

```
<method>
ELASTIC
```

This is usually done after optimizing both the box and geometry (FULLOPT). In this case, the files *geometry.ini* and *box.ini* are read at the beginning of the run. The static elastic constants are computed as the numerical derivatives of the stress tensor. The accuracy can be controlled by setting the finite difference parameter:

```
<fdiff>
1.d-4
```

In PIMD, the elastic constants can be computed for orthorhombic boxes only.

3.9 Classical molecular dynamics (MD)

In the PIMD code, classical molecular dynamics (MD) simulations are implemented for a variety of statistical ensembles, including the microcanonical (NVE), canonical (NVT), isobaric–isothermal (NPT), and isotensional–isothermal (NtT) ensembles.

In the NVT, NPT, and NtT ensembles, temperature control is achieved using Nosé–Hoover (NH), Nosé–Hoover chain (NHC), or massive Nosé–Hoover chain (MNHC) thermostat methods. These approaches enable efficient and stable sampling of the target ensemble by extending the system with additional thermostat degrees of freedom.

For simulations of isotropic liquids under pressure, a cubic simulation box can be employed in the NPT ensemble. For anisotropic solids, a general parallelepiped (triclinic) hexahedral simulation box is available, allowing independent fluctuations of the box vectors. This formulation is suitable for treating crystals and low-symmetry materials.

In simulations of anisotropic solids under external mechanical loading, the NtT ensemble can be used. In this case, the external thermodynamic tension $\overleftrightarrow{\mathbf{t}}$ is specified as a real symmetric 3×3 matrix, enabling the application of general stress states beyond hydrostatic pressure.

Details of the equations of motion and numerical integration schemes used for each ensemble are provided in the corresponding subsections, together with representative input examples.

3.9.1 NVE molecular dynamics

In the NVE ensemble, the equation of motion is given by Newton’s equation. In the PIMD code, it is integrated using the velocity Verlet algorithm. The theoretical background is briefly explained in Section 11.9.

As an example, a set of input files for flexible-SPC water is available. Copy it to the directory *run* by typing

```
> cp -r examples/H2O/water_md_nve run
```

and then move to the directory:

```
> cd run/water_md_nve/
```

To prepare the file *input.dat*, all the keywords described in Sections 3.1 and 3.2 are required. In addition, the following keywords are important.

```
<method>
MD
```

This keyword sets the simulation method. In the present case, MD corresponds to classical molecular dynamics.

```
<ensemble>
NVE
```

This keyword sets the statistical ensemble. Here, NVE denotes the microcanonical ensemble.

```
<dt>
0.25d0
```

This keyword sets the time step in femtoseconds. In the present case, the time step is 0.25 fs.

```
<nstep>
100
```

This keyword sets the total number of MD steps.

```
<temperature>
300.d0
```

This keyword sets the temperature in Kelvin. If the restart file *geometry.ini* exists, the atomic positions and velocities are read from this file. Otherwise, the atomic positions are read from *structure.dat*, and the velocities are generated according to the Maxwell–Boltzmann distribution at this temperature. Note that, in NVE molecular dynamics, the temperature is not controlled during the run and is only used for the initialization of velocities.

```
<corrections>
1 1
```

This keyword sets the translational and rotational corrections applied to the system. In the present case, both the translational and rotational corrections are applied at the initial step only.

```
<iprint_dip>
1
```

This keyword sets the print interval of the dipole moment. Here, the dipole moment is printed at every step.

```
<iprint_rdf>
10
```

This keyword sets the print interval of the atomic pair distribution function. In the present case, it is printed every ten steps. When `<iprint_rdf>` is activated, it is also useful to specify

```
<params_rdf>
1.d0 5.d0 0.01d0
```

This keyword sets the mesh parameters for the atomic pair distribution function. In the present case, the mesh points range from 1.0 bohr to 5.0 bohr with an increment of 0.01 bohr.

The output files **.out* and the restart files **.ini* are explained in Sections 10.2 and 10.3, respectively.

3.9.2 NVT molecular dynamics

In the PIMD code, molecular dynamics for the NVT ensemble is based on massive Nosé-Hoover chain thermostats [8, 10, 11]. The equations of motion are solved using the reversible RESPA algorithm [12, 13]. The theoretical background of NVT molecular dynamics is briefly explained in Section 11.10.

As an example, a set of input files for flexible-SPC water is available. Copy it to the directory *run*:

```
> cp -r examples/H2O/water_md_nvt run
```

Then navigate to that directory:

```
> cd run/water_md_nvt
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are required:

```
<method>
MD
```

This keyword sets the simulation method. In this case, “MD” corresponds to classical molecular dynamics.

```
<ensemble>
NVT
```

This keyword sets the statistical ensemble. Here, “NVT” corresponds to the NVT ensemble.

```
<dt>
0.25d0
```


This keyword sets the step size in femtoseconds.

```
<nstep>  
100
```

This keyword sets the number of steps.

```
<temperature>  
300.d0
```

This keyword sets the temperature in kelvin. If the restart file *geometry.ini* exists, atomic positions and velocities are read from this file. Otherwise, positions are read from *centroid.dat*, and velocities are generated at this temperature for the initial run. The temperature is controlled during the simulation.

```
<corrections>  
0 0
```

This keyword sets the translational and rotational corrections. In this case, the corrections are not applied at the initial step.

```
<iprint_dip>  
10
```

This keyword sets the print interval of the dipole moment. Here, it is printed every ten steps.

```
<iprint_rdf>  
10
```

This keyword sets the print interval of the atomic pair distribution. Here, it is printed every ten steps.

```
<params_rdf>  
1.d0 5.d0 0.01d0
```

This keyword is valid when `<iprint_rdf>` is activated. It sets the mesh parameters for the atomic pair distribution. Here, the mesh points range from 1.0 Bohr to 5.0 Bohr with an increment of 0.01 Bohr.

```
<bath_type>  
MNHC
```

This keyword sets the type of thermostats. When NHC is chosen, a Nosé-Hoover thermostat chain (NHC) is attached to the whole system. When MNHC is chosen, massive Nosé-Hoover thermostat chains (MNHC) are attached to each degree of freedom.

```
<nnhc>  
4
```

This keyword sets the chain length of the MNHC thermostats to four.

```
<ncolor>  
1
```

This keyword sets the multiplicity of the MNHC thermostats to one.

```
<time_bath>  
10.d0
```

This keyword controls the mass of the thermostats, which is related to the characteristic time scale of the system in femtoseconds, as designated by this keyword. The mass of the thermostats can be set close to the fastest vibrational time scale of the system. Here, it is set to 10 femtoseconds, which is close to the oscillation period of 8.881 femtoseconds for the anti-symmetric OH stretching mode of water, corresponding to 3756 cm^{-1} .

```
<nys>
5
```

This keyword controls the step size for the MNHC thermostats, with the thermostats updated five times per step.

The output files **.out* and the restart files **.ini* are explained in Section 10.2 and Section 10.3, respectively.

3.9.3 Constrained molecular dynamics

In NVT and NVE molecular dynamics simulations, harmonic constraints can be added to calculate free energy gradients, which are the negative mean forces imposed on each constraint. The free energy profile can be obtained by integrating with respect to the shifts in the constraints. To accomplish this, independent trajectories (beads) with different constraint equilibrium positions are calculated. The following example is for a water molecule:

```
<nbead>
128
```

This keyword sets the number of independent trajectories (beads) with different constraints; in this case, “128”.

```
<ncons>
1
1 1 2 1.3 2.5
```

This keyword sets the number of constraints, followed by the parameters of each constraint. In this case, there is one constraint of type “1” (interatomic distance) between atoms “1” and “2”. As a function of the bead number, the position of the constraint is shifted linearly from 1.3 bohr to 2.5 bohr.

```
<params_cons>
1 1.d0
2 4.d-4
3 1.d-4
4 1.d0
5 1.d0
6 1.d0
7 1.d0
8 1.d0
```

This keyword sets the force constants for each type of constraint. In this case, the first line sets the force constant of type “1” (interatomic distance) to be 1.0 hartree/bohr².

```
<iprint_cons>
10
```

This keyword sets the print interval of constraint information to the file *cons.out*. In this case, it is every “10” steps.

3.9.4 NPT molecular dynamics in a cubic box

In the PIMD code, molecular dynamics in the NPT ensemble is implemented using massive Nosé–Hoover chain thermostats [8, 10, 16, 17, 11]. The equations of motion are integrated using the reversible RESPA algorithm [12, 13, 14, 15]. The theoretical background of NPT molecular dynamics is briefly described in Section 11.11.

As an example, a set of input files for liquid water is available. First, copy them to the directory *run*:

```
> cp -r examples/water_256/mm_md_npt run
```

and move to that directory:

```
> cd run/mm_md_npt
```

To prepare the file *input.dat*, all keywords described in Sections 3.1 and 3.2 are required. In addition, the following keywords are important.

<method>

PIMD

This keyword sets the simulation method. In general, “PIMD” corresponds to path integral molecular dynamics. In the special case of a single bead, **<nbead>=1** (the default), it reduces to classical molecular dynamics.

<ensemble>

NPT

This keyword specifies the statistical ensemble. Here, “NPT” denotes the isothermal–isobaric ensemble.

<npt_type>

CUBIC2

This keyword defines the shape of the simulation box and the equations of motion for the barostat. For a cubic box, either Andersen’s formulation (“CUBIC1”) or Martyna’s formulation (“CUBIC2”) can be used. In this example, a cubic box with Martyna’s equation of motion is chosen.

<dt>

0.25d0

This keyword sets the time step in femtoseconds.

<nstep>

100

This keyword sets the total number of MD steps.

<temperature>

300.d0

This keyword sets the temperature in kelvin. If the restart file *geometry.ini* exists, atomic positions and velocities are read from it. Otherwise, atomic positions are read from *structure.dat*, and velocities are generated from a Maxwell–Boltzmann distribution at this temperature. The temperature is controlled throughout the simulation.

<corrections>

0 0

This keyword controls translational and rotational corrections. In the present case, no corrections are applied at the initial step.

<bath_type>

MNHC

This keyword specifies the thermostat type. When “NHC” is selected, a Nosé–Hoover chain thermostat is attached to the whole system. When “MNHC” is selected, massive Nosé–Hoover chain thermostats are attached to each degree of freedom.

<nnhc>

4

This keyword sets the chain length of the MNHC thermostats. Here, a chain length of four is used.

<ncolor>

1

This keyword sets the multiplicity of the MNHC thermostats. In this example, the multiplicity is one.

<time_bath>

10.d0

This keyword controls the thermostat mass through the characteristic time scale of the system, given in femtoseconds. A reasonable value is close to the fastest vibrational time scale of the system. Here, it is set to 10 fs, which is comparable to the oscillation period (8.881 fs) of the anti-symmetric OH stretching mode of water (3756 cm^{-1}).

<nys>

5

This keyword controls the number of thermostat updates per MD step. In this case, the thermostats are updated five times per step.

<pressure>

100.d0

This keyword sets the external pressure in megapascals. Here, the pressure is set to 100 MPa.

<time_baro>

1000.d0

This keyword controls the mass of the barostat through the characteristic time scale of volume fluctuations, in femtoseconds. In this example, it is set to 1000 fs.

<iprint_box>

10

This keyword sets the print interval of the box matrix to the output file *box.out*. Here, the box matrix is written every ten steps.

<iboundary>

1

```
37.2276d+00,  0.0000d+00,  0.0000d+00
 0.0000d+00, 37.2276d+00,  0.0000d+00
 0.0000d+00,  0.0000d+00, 37.2276d+00
```

This keyword specifies the boundary condition and the initial simulation box. Here, “1” indicates periodic boundary conditions with box vectors in bohr. The following three lines define the initial box matrix. A cubic box with a side length of 37.2276 bohr is used.

<iprint_trj>

10

This keyword sets the print interval of the trajectory to the output file *trj.out*. In this case, the trajectory is written every ten steps.

The output files **.out* and the restart files **.ini* are explained in Sections 10.2 and 10.3, respectively.

3.9.5 NPT and NtT molecular dynamics in a hexahedron box

In the PIMD code, molecular dynamics for the NPT and NtT ensembles (NtT: isotensional ensemble) are based on massive Nosé-Hoover chain thermostats [8, 10, 11]. The equations of motion are solved using the reversible RESPA algorithm [12, 13, 14, 15]. The theoretical background of NtT molecular dynamics is briefly explained in Section 11.11.

Setups for NPT MD simulation with a hexahedron box. As an example, a set of input files for a nickel crystal is available. Copy it to the directory *run*:

```
> cp -r examples/Ni/eam_pimd_ntt run
```

and navigate to that directory:

```
> cd run/eam_pimd_ntt
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are required.

```
<method>
PIMD
```

This keyword sets the simulation method. Generally, “PIMD” corresponds to path integral molecular dynamics, but in the special case of a single bead (**nbead=1** by default), it corresponds to classical molecular dynamics.

```
<ensemble>
NPT
```

This keyword sets the statistical ensemble. In this case, “NPT” corresponds to the NPT ensemble.

```
<npt_type>
PPHEX
```

This keyword sets the shape of the simulation box. Here, the box is set to a parallel-piped hexahedron.

```
<dt>
0.25d0
```

This keyword sets the step size in femtoseconds.

```
<nstep>
100
```

This keyword sets the number of steps.

```
<temperature>
300.d0
```

This keyword sets the temperature in kelvin. If the restart file *geometry.ini* exists, atomic positions and velocities are read from this file. Otherwise, positions are read from *structure.dat*, and velocities are generated at this temperature for the initial run. The temperature is controlled during the simulation.

```
<corrections>
0 0
```

This keyword sets the translational and rotational corrections. In this case, the corrections are not applied at the initial step.

<bath_type>

MNHC

This keyword sets the type of thermostats. When “NHC” is chosen, a Nosé-Hoover thermostat chain (NHC) is attached to the whole system. When “MNHC” is chosen, massive Nosé-Hoover thermostat chains (MNHC) are attached to each degree of freedom.

<nnhc>

4

This keyword sets the chain length of the MNHC thermostats, which is four in this case.

<ncolor>

1

This keyword sets the multiplicity of the MNHC thermostats; here, it is one.

<time_bath>

10.d0

This keyword controls the mass of the thermostats. The mass of the thermostats is related to the characteristic time scale of the system, in femtoseconds, which is specified by this keyword. This parameter can be set close to the fastest vibrational time scale of the system. Here it is set to 10 femtoseconds, which is close to the oscillation period of 8.881 femtoseconds for the anti-symmetric OH stretching mode of water, which corresponds to 3756 cm^{-1} .

<nys>

5

This keyword controls the step size for the MNHC thermostats. In this case, the thermostats are updated five times per step.

<pressure>

100.d0

This keyword sets the pressure in megapascals; here, it is 100 MPa.

<time_baro>

1000.d0

This keyword controls the mass of the barostat. The mass of the barostat is related to the characteristic time scale of the volume fluctuation, in femtoseconds, which is specified by this keyword. Here it is set to 1000 femtoseconds.

<iprint_box>

10

This keyword controls the print interval of the box matrix to the output file *box.out*. Here, it is set to every ten steps.

<iboundary>

1

33.20000000	0.00000000	0.00000000
0.00000000	33.20000000	0.00000000
0.00000000	0.00000000	33.20000000

This keyword sets the boundary condition and the initial box. Here, “1” means periodic boundary, and the following three lines are the matrix of the initial cubic box with side length 33.2 bohr.

<iprint_trj>

10

This keyword controls the print interval of the trajectory to the output file *trj.out*. Here, it is set to every ten steps.

Setups for NtT MD simulation with a hexahedron box. The setups in NtT MD are the same as those in NPT MD, except for the following keywords. The keyword `<pressure>` should be replaced by

```
<tension>
  0.0    0.0    0.0
  0.0    0.0    0.0
  0.0    0.0 1000.0
```

This keyword sets the tension tensor in megapascals; here, 1000 MPa is the zz element. Note that the tension tensor must be provided as a 3×3 real symmetric matrix. Then, designate the keyword

```
<iboundary>
1
33.20000000  0.00000000  0.00000000
 0.00000000 33.20000000  0.00000000
 0.00000000  0.00000000 33.20000000
```

This keyword sets the boundary condition and the initial box. In the NtT ensemble, this will be the reference box matrix under zero tension. To provide the reference box, one must run a preliminary simulation under the same conditions (same temperature, etc.) with zero tension. The reference box matrix is evaluated as the average box matrix from this preliminary simulation. Within this simulation, the reference box matrix is arbitrary.

The output files **.out* and the restart files **.ini* are explained in Section 10.2 and Section 10.3, respectively.

3.10 Path integral simulations (PI)

In the PIMD code, the following path-integral methods are implemented:

- Path integral molecular dynamics (PIMD) with the second-order Suzuki–Trotter expansion [23].
- Brownian chain molecular dynamics (BCMD) [33].
- Centroid molecular dynamics (CMD) [26].
- Ring polymer molecular dynamics (RPMD). [24].
- Thermostatted ring polymer molecular dynamics (TRPMD) [34].
- Path integral hybrid Monte Carlo (PIHMC) with the second-order [23] and fourth-order Suzuki–Trotter expansion. [35]
- Centroid intrinsic reaction coordinate (CIRC). [32]

The theoretical background of these methods is briefly explained in Section 11.11.

3.10.1 Path integral molecular dynamics (PIMD)

As an example, a set of input files for flexible-SPC water is available. To use it, copy it to the directory *run*:

```
> cp -r examples/H2O/water_pimd_nvt run/
```

and navigate to that directory:

```
> cd run/water_pimd_nvt/
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. Additionally, the following keywords are important:

```
<method>  
PIMD
```

This keyword sets the simulation method. In this case, “PIMD” corresponds to path integral molecular dynamics.

```
<ensemble>  
NVT
```

This keyword sets the statistical ensemble. In this case, “NVT” corresponds to the NVT ensemble.

```
<dt>  
0.25d0
```

This keyword sets the step size in femtoseconds. In this case, the step size is 0.25 femtoseconds.

```
<nstep>  
100
```

This keyword sets the number of steps. In this case, the simulation runs for 100 steps.

```
<temperature>  
300.d0
```

This keyword sets the temperature in Kelvin. In this case, the temperature is 300 K. The temperature is controlled during the PIMD simulation.

```
<corrections>  
0 0
```

This keyword sets the translational and rotational corrections. In this case, the corrections are not applied at the initial step.

```
<iprint_dip>  
10
```

This keyword sets the print interval for the dipole moment. In this case, the dipole moment is printed every ten steps.

```
<iprint_rdf>  
10
```

This keyword sets the print interval for the atomic pair distribution. In this case, the atomic pair distribution is printed every ten steps.

```
<params_rdf>  
1.d0 5.d0 0.01d0
```

This keyword sets the mesh parameters for the atomic pair distribution. In this case, the mesh points range from 1.0 bohr to 5.0 bohr with an increment of 0.01 bohr.

```
<bath_type>  
MNHC
```

This keyword sets the type of thermostats. When NHC is chosen, a Nosé–Hoover thermostat chain (NHC) is attached to the whole system. When “MNHC” is chosen, massive Nosé–Hoover thermostat chains (MNHC) are attached to each degree of freedom.

```
<nnhc>  
4
```


This keyword sets the chain length of the MNHC thermostats. In this case, the chain length is four.

```
<ncolor>  
1
```

This keyword sets the multiplicity of the MNHC thermostats. In this case, the multiplicity is one.

```
<time_bath>  
10.d0
```

This keyword controls the mass of thermostats. The mass of thermostats is related to the characteristic time scale of the system, in femtoseconds, which is designated by this keyword. Here it is set to 10 femtoseconds.

```
<nys>  
5
```

This keyword controls the step size for the MNHC thermostats. In the present case, thermostats are updated 5 times per step.

```
<nbead>  
32
```

This keyword sets the number of beads, in the present case, 32 beads.

```
<nref>  
1
```

This keyword controls the step size for updating the harmonic forces in path integrals. In the present case, “1” means the harmonic forces are updated once per step.

```
<beadspread>  
500.d0
```

This keyword controls the bead spread for the initial configuration.

If the restart file *geometry.ini* exists, the positions and velocities of the beads are read from this file. Otherwise, the centroid positions being read from *structure.dat*, all the positions of beads are created randomly around the centroid positions with the standard deviation controlled by the keyword `<beadspread>`, while the velocities are generated at this temperature as an initial run.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.10.2 Path integral hybrid Monte Carlo (PIHMC)

As an example, a set of input files for a water molecule is available. So, let us copy it to the directory *run*:

```
> cp -r examples/H2O/water_pihmc_nvt_2 run
```

and enter that directory:

```
> cd run/water_pihmc_nvt_2
```

Let us look at the keywords one by one.

```
<method>  
PIHMC
```

This keyword sets the simulation method, in the present case, path integral hybrid Monte Carlo.

```
<ensemble>  
NVT
```

This keyword sets the statistical ensemble, in the present case, the NVT ensemble.

```
<iorder_hmc>  
2
```

This keyword sets the order of the Suzuki–Trotter expansion, in the present case, the second order.

```
<istep_hmc>  
1
```

This keyword sets the number of MD updates per MC step in the hybrid Monte Carlo method.

```
<ipotential>  
WATER
```

This keyword sets the potential, in the present case, the flexible SPC potential of water.

```
<iboundary>  
0
```

This keyword sets the boundary condition. In the present case, “0” means the free boundary condition.

```
<nbead>  
32
```

This keyword sets the number of beads to 32.

```
<nstep>  
1000
```

This keyword sets the number of steps of PIHMC to 1000.

```
<dt>  
1.5d0
```

This keyword sets the step size of PIHMC to 1.5 femtoseconds.

```
<temperature>  
300.d0
```

This keyword sets the temperature to 300 K.

```
<beadspread>  
500.d0
```

This keyword sets the initial bead spread to 500 K.

```
<iread_exit>  
100
```

This keyword sets the interval for reading *exit.dat* to every 100 steps.

```
<iprint_std>  
100
```

```
<iprint_bond>  
100
```

```
<iprint_eavg>  
100
```

```
<iprint_rest>  
100
```

```
<iprint_rdf>  
100
```

```
<iprint_trj>  
100
```

```
<iprint_xyz>  
100
```

```
<iprint_dip>  
100
```

These keywords set the intervals for printing the standard output, average bond distances, energy averages, restart files, radial distributions, trajectories in regular and xyz formats, and dipole moments, respectively.

The output files **.out* and restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.10.3 Brownian chain molecular dynamics (BCMD)

The following keywords are important:

```
<method>  
BCMD
```

This keyword sets the simulation method. In the present case, “BCMD” corresponds to Brownian chain molecular dynamics.

```
<ensemble>  
NVE
```

This keyword sets the statistical ensemble. In the present case, “NVE” corresponds to the NVE ensemble.

```
<irandom>  
1
```

This keyword sets the random number seed. In the present case, “1” sets a time-dependent seed.

3.10.4 Centroid molecular dynamics (CMD)

As an example, a set of input files for the flexible-SPC water model is available. Let us copy it to the directory *run*:

```
> cp -r examples/H2O/water_cmd run
```

Then, navigate to that directory:

```
> cd run/water_cmd
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are also important:

```
<method>  
CMD
```

This keyword sets the simulation method. In the present case, “CMD” corresponds to centroid molecular dynamics.

```
<ensemble>  
NVE
```

This keyword sets the statistical ensemble. In the present case, “NVE” corresponds to the NVE ensemble.

```
<dt>  
0.025d0
```

This keyword sets the step size in femtoseconds. In the present case, the step size is 0.025 femtoseconds. Note that in centroid molecular dynamics, the step size should be smaller than usual to deal with the fast motion of non-centroid modes. It is recommended to set `<dt>` to be the original step size divided by the factor `<igamma>`.

```
<nstep>  
1000
```

This keyword sets the number of steps. In the present case, it is set to 1000 steps.

```
<temperature>  
300.d0
```

This keyword sets the temperature in Kelvin. In the present case, it is set to 300 K.

```
<corrections>  
2 2
```

This keyword sets the translational and rotational corrections. In the present case, both translation and rotational corrections are applied every step.

```
<iprint_dip>  
100
```

This keyword sets the print interval of the dipole moment. In the present case, it is set to every 100 steps.

```
<iprint_rdf>  
100
```

This keyword sets the print interval of the atomic pair distribution. In the present case, it is set to every 100 steps.

```
<params_rdf>  
1.d0 5.d0 0.01d0
```

This keyword is valid when `<iprint_rdf>` is activated. It sets the mesh parameters for the atomic pair distributions. In the present case, the mesh points range from 1.0 bohr to 5.0 bohr with an increment of 0.01 bohr.

```
<bath_type>  
MNHC
```

This keyword sets the type of thermostats. When MNHC is chosen, massive Nosé–Hoover thermostat chains (MNHC) are attached to each degree of freedom. In the case of CMD, the MNHC thermostats are attached only to the non-centroid modes, and they are not attached to the centroids.

```
<nnhc>  
4
```

This keyword sets the chain length of the MNHC thermostats. In the present case, it is set to four.

```
<ncolor>  
1
```

This keyword sets the multiplicity of the MNHC thermostats. In the present case, it is set to one.

```
<time_bath>  
10.d0
```

This keyword controls the mass of the thermostats. The mass of the thermostats is related to the characteristic time scale of the system, in femtoseconds, which is designated by this keyword. Here it is set to 10 femtoseconds.

```
<nys>  
5
```

This keyword controls the step size for the MNHC thermostats. In the present case, thermostats are updated 5 times.

```
<nbead>  
32
```

This keyword sets the number of beads. In the present case, 32 beads are used.

```
<nref>  
1
```

This keyword controls the step size for updating the harmonic forces in path integrals. In the present case, “1” means the harmonic forces are updated once per step.

```
<beadspread>  
500.d0
```

This keyword controls the bead spread for the initial configuration.

```
<igamma>  
10
```

This keyword controls the adiabaticity in centroid molecular dynamics. The accuracy of CMD will increase as the value of `<igamma>` becomes larger. Note that the adiabaticity parameter, γ , is given by the inverse of the square of `<igamma>`; in the present case, it is 0.01.

If the restart file *geometry.ini* from a previous PIMD run exists, the atomic positions and velocities are read from this file (recommended in CMD). If the restart file *bath.ini* from a previous PIMD run also exists, the bath positions and velocities are read from this file (recommended in CMD). Otherwise, the centroid positions are read from *structure.dat*, and all bead positions are generated randomly around the centroid positions with a standard deviation controlled by the keyword `<beadspread>`. The velocities are generated at this temperature as an initial run (not recommended in CMD). The temperature is not controlled with respect to the centroid coordinates in the CMD method. Temperature control is applied to the non-centroid modes only.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.10.5 Ring polymer molecular dynamics (RPMD)

As an example, a set of input files for the flexible-SPC water is available. So, let us copy it to the directory *run*:

```
> cp -r examples/H2O/water_rpmd run
```

and enter that directory:

```
> cd run/water_rpmd
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are also important.

```
<method>
RPMD
```

This keyword sets the simulation method. In the present case, “RPMD” corresponds to ring polymer molecular dynamics.

```
<ensemble>
NVE
```

This keyword sets the statistical ensemble. In the present case, “NVE” corresponds to the NVE ensemble.

```
<dt>
0.25d0
```

This keyword sets the step size in femtoseconds. In the present case, the step size is 0.25 femtoseconds.

```
<nstep>
100
```

This keyword sets the number of steps. In the present case, there are 100 steps.

```
<temperature>
300.d0
```

This keyword sets the temperature in Kelvin.

```
<corrections>
2 2
```

This keyword sets the translational and rotational corrections. In the present case, both translation and rotation corrections are applied at each step.

```
<iprint_dip>
10
```

This keyword sets the print interval of the dipole moment. In this case, the dipole moment is printed every ten steps.

```
<iprint_rdf>
10
```

This keyword sets the print interval of the atomic pair distribution. In this case, the atomic pair distribution is printed every ten steps.

```
<params_rdf>
1.d0 5.d0 0.01d0
```

This keyword is valid when `<iprint_rdf>` is activated. It sets the mesh parameters for the atomic pair distributions. In the present case, the mesh points range from 1.0 bohr to 5.0 bohr with an increment of 0.01 bohr.

```
<nbead>
32
```

This keyword sets the number of beads. In the present case, there are 32 beads.

```
<beadsread>
500.d0
```

This keyword controls the bead spread used to generate the initial configuration.

If the restart file *geometry.ini* from a previous PIMD run exists, the atomic positions and velocities are read from this file (recommended in RPMD). Otherwise, the centroid positions are read from *structure.dat*, and all bead positions are generated randomly around the centroid positions with a standard deviation controlled by the keyword `<beadsread>`, while the velocities are generated at this temperature as an initial run (not recommended in RPMD). The temperature is not controlled during the RPMD run since it is an NVE dynamics.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.10.6 Thermostatted ring polymer molecular dynamics (TRPMD)

The following keywords are important:

```
<method>
TRPMD
```

This keyword sets the simulation method. In the present case, “TRPMD” corresponds to thermostatted ring polymer molecular dynamics.

```
<ensemble>
NVT
```

This keyword sets the statistical ensemble. In the present case, “NVT” corresponds to the NVT ensemble.

3.10.7 Centroid Intrinsic Reaction Coordinate (CIRC)

The CIRC calculations can be carried out using the code *polymers.x*. In addition to the PIMD setups mentioned in Subsection 3.10.7, the following keywords are also important.

```
<npoly>
16
```

This keyword sets the number of centroid images. In the present case, there are 16 images.

```
<np_poly>
16
```

This keyword sets the parallel number with respect to centroid images. In the present case, all images are computed in parallel with a parallel number of 16.

```
<ends_poly>
FIXED
```

This keyword sets the option for the end centroid images. In this case, the end centroid images are fixed.

```
<dt_poly>  
0.04d0
```

This keyword sets the displacement of centroid images per update cycle.

```
<ncycle_poly>  
1
```

This keyword sets the number of update cycles of centroid images by the string method. In this case, there is only one cycle.

```
<iprint_poly>  
10
```

This keyword sets the print interval of the centroid mean force during the PIMD runs. In this case, the centroid mean force is printed every 10 PIMD steps.

```
<guess_poly>  
LINE
```

This keyword sets the initial guess for the centroid images. In this case, the initial guess is a linear interpolation of the two ends read from the file *structure.dat*.

```
<projcmf_poly>  
OFF
```

This keyword sets the projection of mean forces applied to centroid images. In this case, no projection is applied.

```
<ngrid_poly>  
10
```

This keyword sets the number of grid points used for interpolation in the string method. In this case, 10 grid points are used.

3.11 Replica exchange hybrid Monte Carlo (REHMC)

In the PIMD code, the following replica exchange hybrid Monte Carlo (REHMC) methods are implemented:

- Temperature replica exchange (TX) between systems with different temperatures.
- Hamiltonian replica exchange (HX) between systems with different alchemical intermediates.

The theoretical background of these methods is briefly explained in Section 11.12.

3.11.1 Temperature replica exchange hybrid Monte Carlo

For example, a set of input files for a water molecule is available. To use it, copy the files to the directory *run* by executing the following command:

```
> cp -r examples/H2O/water_rehmc_tx run
```

Then enter that directory:

```
> cd run/water_rehmc_tx
```

Next, let's examine the keywords one by one:

```
<method>  
REHMC
```


This keyword sets the simulation method to replica exchange hybrid Monte Carlo.

```
<ensemble>  
NVT
```

This keyword sets the statistical ensemble to NVT.

```
<irem_type>  
TX
```

This keyword sets the type of replica exchange method to temperature exchange.

```
<istep_hmc>  
1
```

This keyword sets the number of MD updates per MC step in the hybrid Monte Carlo method.

```
<ipotential>  
WATER
```

This keyword sets the atomic species of the system. In this case, the system is a water molecule.

```
<iboundary>  
0
```

This keyword sets the boundary condition. Here, “0” means a free boundary condition.

```
<nstep>  
100000
```

This keyword sets the number of steps to 100000.

```
<dt>  
2.d0
```

This keyword sets the step size to 2 femtoseconds.

```
<nbead>  
4
```

This keyword sets the number of replicas to four.

```
<temprange_rem>  
300.d0 2400.d0
```

This keyword sets the temperature range by specifying the lowest temperature T_1 and the highest temperature T_n among the replicas. In this case, the temperature range is set between 300 K and 2400 K, with the first, second, third, and fourth replicas set to 300 K, 600 K, 1200 K, and 2400 K, respectively.

```
<beadspread>  
500.d0
```

This keyword sets the initial bead spread to 500 K.

```
<iread_exit>  
1000
```

This keyword specifies the read interval for the file *exit.dat*.

```

<iprint_std>
1000

<iprint_rest>
1000

<iprint_rdf>
1000

<iprint_trj>
1000

<iprint_xyz>
1000

```

These keywords specify the print intervals for the standard output, the restart files, the radial distribution functions, and the trajectories in the regular and xyz formats, respectively.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.11.2 Hamiltonian replica exchange hybrid Monte Carlo

As an example, a set of input files for an $M^+(H_2O)$ cluster ($M = K$ and Cs) is available. To use these input files, copy them to the directory *run*:

```
> cp -r examples/CsH2O/mm_rehmc_hx run
```

and navigate to that directory:

```
> cd run/mm_rehmc_hx
```

Now, let's examine the keywords one by one:

```

<method>
REHMC

```

This keyword sets the simulation method, which in this case is replica exchange hybrid Monte Carlo.

```

<ensemble>
NVT

```

This keyword sets the statistical ensemble, which in this case is NVT.

```

<irem_type>
HX

```

This keyword sets the type of replica exchange method, which in this case is Hamiltonian exchange.

```

<istep_hmc>
1

```

This keyword sets the number of molecular dynamics updates per Monte Carlo step in the hybrid Monte Carlo method.

```

<ipotential>
ALCHEM

```

This keyword specifies the use of potentials of alchemical intermediates.

```
<iboundary>  
0
```

This keyword sets the boundary condition, where “0” indicates a free boundary condition.

```
<nstep>  
100000
```

This keyword sets the number of steps to 100000.

```
<dt>  
2.d0
```

This keyword sets the step size to 2 femtoseconds.

```
<nbead>  
4
```

This keyword sets the number of replicas with different alchemical intermediates, which in this case is four.

```
<alchem_dat_dir>  
dat_1  dat_2
```

This keyword sets the input directories for the alchemical potentials.

```
<alchem_scr_dir>  
scr_1  scr_2
```

This keyword sets the temporary scratch directories for the alchemical potentials.

```
<beadspread>  
500.d0
```

This keyword sets the initial bead spread to 500 K.

```
<iread_exit>  
1000
```

This keyword specifies the read interval for the file *exit.dat*.

```
<iprint_std>  
1000
```

```
<iprint_rest>  
1000
```

```
<iprint_rdf>  
1000
```

```
<iprint_trj>  
1000
```

```
<iprint_xyz>  
1000
```

These keywords specify the print intervals for the standard output, restart files, radial distribution functions, and trajectories in the regular and xyz formats, respectively.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.12 Metadynamics (MTD)

The theoretical background of metadynamics (MTD) is briefly explained in Section 11.13.

In PIMD, MTD in the NVT ensemble based on massive Nosé-Hoover thermostats is implemented.

So far, eight types of collective variables have been implemented, and the set of collective variables should be chosen from the following list:

- type 1: bond distance (between two atoms)
- type 2: bond angle (among three atoms)
- type 3: dihedral angle (among four atoms)
- type 4: bond difference (among three atoms)
- type 5: coordination number (for atomic species)
- type 6: difference in coordination numbers (for atomic species)
- type 7: center of mass (of atomic species)
- type 8: difference in centers of mass (of two atomic species)

It is possible to choose one, two, or three collective variables, corresponding to one-, two-, or three-dimensional metadynamics, respectively. It is also possible to employ two or three collective variables of the same type, such as two bond distances.

In the PIMD implementation, the Gaussian width is set for each type of collective variable, rather than for each individual collective variable. For example, if two bond distances are specified, the same Gaussian width value is shared by both.

As an example, a set of input files for one-dimensional MTD of a water molecule is provided. In this case, the H–H bond distance is chosen as the collective variable. Copy the input files to the directory *run* by executing:

```
> cp -r examples/H2O/water_mtd_1d run
```

Then enter the directory:

```
> cd run/water_mtd_1d
```

To prepare the file *input.dat*, all keywords described in Sections 3.1 and 3.2 are required. In addition, the following keywords are important:

```
<method>  
MTD
```

This keyword sets the simulation method, which in this case is metadynamics.

```
<ensemble>  
NVT
```

This keyword sets the statistical ensemble, here the NVT ensemble.

```
<nstep>  
10000
```

This keyword sets the total number of steps, in this case 10,000.

```
<dt>  
0.25d0
```

This keyword sets the time step in femtoseconds, here 0.25 fs.

```
<temperature>  
300.d0
```

This keyword sets the system temperature in Kelvin, here 300 K.

```
<bath_type>  
MNHC
```

This keyword sets the type of thermostats. In metadynamics, only MNHC thermostats are implemented.

```
<ncolor>  
1
```

This keyword sets the multiplicity of the MNHC thermostats, in this case one.

```
<nnhc>  
4
```

This keyword sets the chain length of the MNHC thermostats, here four.

```
<time_bath>  
10.d0
```

This keyword controls the mass of the thermostats attached to the system. The thermostat mass is related to the characteristic time scale of the system, in femtoseconds, specified by this keyword. In the present case, a time scale of 10 fs is chosen.

```
<nys>  
5
```

This keyword controls the time integration of the MNHC thermostats. In the present case, the thermostats are updated five times per `<dt>`.

```
<nbead>  
1
```

This keyword sets the number of walkers, i.e., the number of beads. In this case, a single walker is used.

```
<beadsread>  
500.d0
```

This keyword sets the temperature controlling the initial bead spread of the walkers. In the present case, 500 K is chosen.

The keywords specific to metadynamics are listed below.

```
<mg_meta>  
50000
```

This keyword sets the maximum number of history-dependent Gaussian hills, here 50,000.

```
<nmeta>  
1  
1, 2 3
```

This keyword specifies the parameters of the collective variables. The first line gives the dimensionality of the collective variables, which is one in this case. The following line defines the collective variable(s); here, a one-dimensional collective variable corresponding to the bond distance (type 1) between atoms #2 and #3.

```
<nref_meta>
10
```

This keyword controls the update frequency of the forces arising from the harmonic potential and Gaussian hills. In this case, the forces are updated ten times within one `<dt>`.

```
<gh_meta>
150.d0
```

This keyword sets the height of the Gaussian hills. In the present case, a height of 150 K is chosen.

```
<gw_meta>
1 0.100d0
2 5.000d0
3 10.000d0
4 0.100d0
5 0.100d0
6 0.100d0
7 0.100d0
8 0.100d0
```

This keyword defines the Gaussian widths for each type of collective variable. In the present case, only the first line is relevant, which specifies a Gaussian width of 0.1 bohr for the bond distance (type 1).

```
<time_cv_meta>
75.0
```

This keyword sets the diffusion time scale of the collective variables, which determines the mass of the fictitious particle. Here, a time scale of 75 fs is chosen.

```
<time_fc_meta>
30.d0
```

This keyword sets the oscillation period of the harmonic potential between the fictitious particle and the collective variables. In this case, 30 fs is chosen.

```
<time_limit_meta>
225.0
```

This keyword sets the minimum time interval for adding a new Gaussian hill. In the present case, 225 fs is chosen, which corresponds to three times the value of `<time_cv_meta>`.

```
<time_cv_bath>
1000.0
```

This keyword sets the time scale of the thermostat attached to the fictitious particle, which controls the thermostat mass. Here, it is set to 1000 fs.

```
<iprint_cv_meta>
1
```

This keyword sets the print interval for the collective variables and the fictitious particle to the output file *cv.out*. In this case, they are printed at every step.

```
<iprint_meta>
1
```

This keyword sets the print interval of the energy components in metadynamics to the output file *meta.out*. In this case, the data are printed at every step.

```
<iprint_rec_meta>
100
```

This keyword sets the print interval of the reconstructed free energy to the output file *rec.out*. In the present case, it is printed every 100 steps.

```
<params_rec_meta>
1  1.0    6.0   0.02
2  0.0   180.0  1.00
3  0.0   360.0  2.00
4 -3.0    3.0   0.02
5  0.0    4.0   0.02
6  0.0    4.0   0.02
7  1.0    6.0   0.02
8  1.0    6.0   0.02
```

This keyword defines the mesh points used to reconstruct the free energy landscape for each type of collective variable. In the present case, only the first line is relevant, specifying a range from 1.0 bohr to 6.0 bohr with an increment of 0.02 bohr.

```
<cut_rec_3d>
100.0
```

This keyword controls the numerical accuracy of the reconstruction of the free energy surface in three-dimensional metadynamics. In the present case, this option is inactive because the metadynamics is one-dimensional.

```
<joption_meta>
0
```

This keyword controls the timing of adding new Gaussian hills. In the present case, a new hill is added when the walker is sufficiently displaced.

To perform well-tempered metadynamics, one must additionally specify:

```
<dtemp_meta>
1000.0
```

This keyword sets the temperature parameter of well-tempered metadynamics, here 1000 K. If this keyword is not specified, `<dtemp_meta> = 0.0` corresponds to conventional metadynamics.

The output files **.out* and the restart files **.ini* are described in Sections 10.2 and 10.3, respectively.

3.13 Mean force dynamics (AFED)

The theoretical background of mean force dynamics is briefly explained in Section 11.16. Throughout this manual, “adiabatic free energy dynamics (AFED)” is used to indicate “mean force dynamics”. In PIMD, the following AFED-related techniques are implemented.

- GRAD: calculation of the free energy gradient with a constant constraint.
- DESCENT: descent search for free energy minimum points using the descent method.
- ASCENT: ascent search for free energy saddle points using ascent dynamics.
- AUTO: automated search for free energy stationary points by combining DESCENT and ASCENT.
- TAMD: temperature-accelerated molecular dynamics. The options are NVE, NVT, and velocity scaling.
- LOGMFD: logarithmic mean force dynamics. The options are NVE, NVT, and velocity scaling.

Eight types of collective variables have been implemented (so far). The set of collective variables should be chosen from the list below.

- type 1: bond distance (between two atoms)
- type 2: bond angle (among three atoms)
- type 3: dihedral angle (among four atoms)
- type 4: bond difference (among three atoms)
- type 5: coordination number (for atomic species)
- type 6: difference in coordination numbers (for atomic species)
- type 7: center of mass (of atomic species)
- type 8: difference in centers of mass (of two atomic species)

3.13.1 Common keywords used in all the AFED methods (Free energy gradient)

As an example, a set of input files for one-dimensional AFED of a butene molecule is available. In this case, the collective variable is chosen to be the C-C-C-C bond dihedral. Let us copy it to the directory *run*:

```
> cp -r examples/C4H10/mm_afed_grad run
```

and enter that directory:

```
> cd run/mm_afed_grad
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords, which can be categorized into three groups, are also important.

First group: Keywords for molecular dynamics This keyword sets the number of atoms in the system. In the present case, there are 14 atoms in the C₄H₁₀ molecule.

```
<ipotential>
MM
```

This keyword sets the potential model. In the present case, “MM” corresponds to the molecular mechanics potential.

```
<iprint_std>
100
```

This keyword sets the print interval of the standard output *standard.out*. In the present case, every 100 steps.

```
<ensemble>
NVT
```

This keyword sets the statistical ensemble for molecular dynamics. In the PIMD code, the “NVT” ensemble must always be chosen for AFED (the NPT ensemble for AFED is currently not available).

```
<bath_type>
MNHC
```

This keyword sets the type of thermostat. In the PIMD code, the massive Nosé-Hoover chain thermostat, “MNHC”, must always be chosen for AFED. The thermostat settings can be controlled by the keywords `<time_bath>`, `<nnhc>`, `<ncolor>`.


```
<temperature>
500.d0
```

This keyword sets the system temperature in kelvin; in the present case, “500” K.

```
<dt>
0.25d0
```

This keyword sets the molecular dynamics time step in femtoseconds; in the present case, 0.25 fs.

```
<nys>
8
```

This keyword controls the step size for the MNHC thermostat. In the present case, the thermostat is updated “8” times per step.

Second group: Keywords for constraints

```
<nref>
8
```

This keyword controls the step size for the harmonic constraint. In the present case, the harmonic constraint is updated “8” times per step.

```
<iprint_cons>
-1
```

This keyword sets the print interval for constraint information to the file *cons.out*; in the present case, it is not printed.

```
<ncons>
1
3 1 5 8 11
```

This keyword sets the number of constraints followed by the parameters of each constraint. In the present case, there is “1” constraint of type “3” (dihedral angle) among atoms “1, 5, 8, 11”.

```
<params_cons>
1 5.0d+0
2 5.0d-2
3 5.0d-3
4 2.5d+0
5 1.0d+0
6 1.0d+0
7 5.0d+0
8 5.0d+0
```

This keyword sets the force constant for each type of constraint. In the present case, only the third line is important, which sets the force constant of type “3” (dihedral angle) to be 5.0×10^{-3} hartree/degree².

Third group: Keywords for adiabatic free energy dynamics

```
<method>
AFED
```

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

```
<afed_type>
GRAD
```

This keyword sets the type of AFED. In the present case, “GRAD” means free energy gradient calculation.

```
<nbead>
8
```

This keyword sets the number of beads (replicas). In the present case, “8” beads.

```
<params_afed>
1    0.08d0    3.0d0    10.0d0
2    3.00d0    0.0d0    180.0d0
3    5.00d0 -180.0d0    180.0d0
4    0.04d0    -3.0d0    3.0d0
5    0.02d0    0.0d0    4.0d0
6    0.02d0    0.0d0    4.0d0
7    0.08d0    3.0d0    10.0d0
8    0.08d0    3.0d0    10.0d0
```

This keyword sets the AFED parameters for all types of collective variables. In the present case, the active ones are the first four numbers in the third line. For the collective variable of type “3” (dihedral angle), the reference shift $\Delta\mathcal{R}_d^{\text{ref}}$ is “5.0” degrees, the lower bound $\mathcal{R}_d^{\text{min}}$ is “-180” degrees, and the upper bound $\mathcal{R}_d^{\text{max}}$ is “180” degrees. All the remaining values are inactive.

```
<nstep_pre_afed>
1000
```

This keyword sets the number of preliminary MD steps to equilibrate the system. In the present case, “1000” steps.

```
<nstep_pro_afed>
2000
```

This keyword sets the number of productive MD steps to collect averages of the free energy gradient. In the present case, “2000” steps.

To start the calculation, type

```
> pimd.x
```

Then the following message will appear on the screen:

```

.....
.....
1700    1.32412595    0.03728593    485.62    2017-02-23 THU 12:06:50.47
1800    1.32412021    0.03356013    469.85    2017-02-23 THU 12:06:50.92
1900    1.32411461    0.03331202    527.77    2017-02-23 THU 12:06:51.49
2000    1.32412219    0.03664502    416.04    2017-02-23 THU 12:06:52.05

=====
step st cv  r-ideal  r-mean  f-energy  -df/dr  dr/dt  dt    n  xi
-----
   0 GR  1  179.010  179.016  0.000000  0.000094  0.00000  0.500  1.00  0
```

Restart files: geometry.ini, step.ini, bath.ini, cv.ini, afed.ini.

Normal termination of pimd.

The last line provides the following information:

- **step**: AFED step number (iteration number).
- **st**: AFED status.
 - **GR**: free energy gradient.
 - **TE**: convergence test.
 - **AS**: ascent trajectory.
 - **DE**: descent trajectory.
 - **D1**: first descent trajectory from a saddle point.
 - **D2**: second descent trajectory from a saddle point.
 - **EQ**: minimum point.
 - **TS**: saddle point.
 - **TA**: temperature-accelerated molecular dynamics.
 - **L0**: logarithmic mean force dynamics.
- **r-ideal**: position of the fictitious particle.
- **r-mean**: mean position of the collective variables.
- **f-energy**: free energy (potential of mean force).
- **-df/dr**: mean force (negative of the free energy gradient).
- **dr/dt**: velocity of the collective variables.
- **dt**: AFED step size.
- **n**: unit vector.
 - For ascent trajectories, vector **n**.
 - For descent trajectories, vector parallel to the mean force.
- **xi**: angle ξ between the force vectors of the last two AFED steps, used to judge convergence.

To test the convergence of mean forces, one can use

```
<afed_type>  
TEST
```

In this case, the run is controlled by the following keywords:

```
<nstep_pro_afed>  
20000
```

This keyword sets the length of the production MD run.

```
<iprint_test_afed>  
1000
```

This keyword sets the print interval of the mean forces during the production MD run.

The output files **.out*, **.xyz*, and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.13.2 Descent search for free energy minima

To prepare the file *input.dat*, all the keywords given in Sections 3.1, 3.2, and 3.13.1 are important. In addition, the following keywords are also important:

```
<method>
AFED
```

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

```
<afed_type>
DESCENT
```

This keyword sets the type of AFED. In the present case, “DESCENT” means descent search.

```
<niter_afed>
200
```

This keyword sets the number of iterations (updates of the fictitious particle) in adiabatic free energy dynamics. In the present case, “200” iterations.

```
<dt_descent_afed>
0.5d0
```

This keyword sets the initial AFED step size for the descent search. In the present case, “0.5” (dimensionless).

```
<dt_damp_afed>
0.7d0
```

This keyword sets the damping factor of the AFED step size as convergence is approached. In the present case, “0.7” (dimensionless).

```
<dt_conv_afed>
0.05d0
```

This keyword sets the AFED step size at convergence. In the present case, “0.05” (dimensionless).

3.13.3 Ascent search for free energy saddles

To prepare the file *input.dat*, all the keywords given in Sections 3.1, 3.2, and 3.13.1 are important. In addition, the following keywords are also important.

```
<method>
AFED
```

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

```
<afed_type>
ASCENT
```

This keyword sets the type of AFED. In the present case, “ASCENT” means ascent search.

```
<niter_afed>
200
```

This keyword sets the number of iterations (updates of the fictitious particle) in adiabatic free energy dynamics. In the present case, “200” iterations.

```
<dt_ascent_afed>
1.0d0
```

This keyword sets the initial AFED step size for the ascent search. In the present case, “1.0” (dimensionless).

```
<dt_damp_afed>  
0.7d0
```

This keyword sets the damping factor of the AFED step size as convergence is approached. In the present case, “0.7” (dimensionless).

```
<dt_conv_afed>  
0.05d0
```

This keyword sets the AFED step size at convergence. In the present case, “0.05” (dimensionless).

```
<gamma_ascent_afed>  
1.0d0
```

This keyword sets the parameter γ^{gad} for AFED ascent search in hartree. In the present case, “1.0” hartree.

```
<hessian_ascent_afed>  
NUMERICAL
```

This keyword sets the option for evaluating the Hessian term in AFED ascent search. In the present case, “NUMERICAL” evaluation.

```
<fdiff_sampling_afed>  
5.0
```

This keyword sets the finite difference parameter for the numerical evaluation of the Hessian term in AFED ascent search. In this case, “5.0”.

3.13.4 Automated search for free energy stationary points

To prepare the file *input.dat*, all the keywords given in Sections 3.1, 3.2, and 3.13.1 are important. In addition, the following keywords are also important.

```
<method>  
AFED
```

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

```
<afed_type>  
AUTO
```

This keyword sets the type of AFED. In the present case, “AUTO” means automated search by combining descent and ascent methods.

```
<niter_afed>  
200
```

This keyword sets the number of iterations (updates of the fictitious particle) in adiabatic free energy dynamics. In the present case, “200” iterations.

```
<dt_descent_afed>  
0.5d0
```

This keyword sets the initial AFED step size for descent search. In the present case, “0.5” (dimensionless).

```
<dt_ascent_afed>  
1.0d0
```

This keyword sets the initial AFED step size for ascent search. In the present case, “1.0” (dimensionless).

```
<dt_damp_afed>  
0.7d0
```

This keyword sets the damping factor of the AFED step size as convergence is approached. In the present case, “0.7” (dimensionless).

```
<dt_conv_afed>  
0.05d0
```

This keyword sets the AFED step size at convergence. In the present case, “0.05” (dimensionless).

```
<gamma_ascent_afed>  
1.0d0
```

This keyword sets the gamma parameter for AFED ascent search in hartree. In the present case, “1.0” hartree.

```
<hessian_ascent_afed>  
NUMERICAL
```

This keyword sets the option for evaluating the Hessian term in AFED ascent search. In the present case, “NUMERICAL” evaluation.

```
<fdiff_sampling_afed>  
5.0
```

This keyword sets the finite difference parameter for the numerical evaluation of the Hessian term in AFED ascent search. In this case, “5.0”.

```
<nshot_auto_afed>  
10
```

This keyword sets the maximum number of shots in AFED ascent search from a minimum in automated calculations.

```
<nmiss_auto_afed>  
5
```

This keyword sets the maximum number of consecutive misses in AFED ascent search from a minimum in automated calculations.

3.13.5 Temperature accelerated molecular dynamics

To prepare the file *input.dat*, all the keywords given in Sections 3.1, 3.2, and 3.13.1 are important. In addition, the following keywords are also important.

```
<method>  
AFED
```

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

```
<afed_type>  
TAMD
```

This keyword sets the type of AFED. In the present case, “TAMD” means temperature accelerated molecular dynamics.

```
<tamd_type>
NVE
```

This keyword sets the type of TAMD. In the present case, “NVE” means NVE-type TAMD. NVT (Nosé-Hoover dynamics) and VS (velocity scaling dynamics) are also available.

```
<niter_afed>
200
```

This keyword sets the number of AFED steps (total updates of the fictitious particle) in TAMD. In the present case, “200” iterations.

```
<dt_tamd>
1.0
```

This keyword sets the AFED step size, Δt_{cv} , for TAMD. In the present case, “1.0” (since the dynamics of TAMD is physically meaningless, the unit can be arbitrarily chosen; e.g., fs).

```
<start_afed>
MANUAL
0.2  1.3e-2
1.67 -6.3e-1
...
```

This keyword sets the initial position and velocity of each fictitious CV particle. If “MANUAL” is specified in the first line, the initial position and velocity of each CV particle should be given for each line as in the example above. In this case, the initial position of the first CV particle is 0.2 and its velocity is 1.3e-2. Those of the second CV particle are 1.67 and -6.3e-1, respectively. If “AUTOMATIC” is instead specified (default setting), the initial positions are automatically estimated from the initial configuration of the MD system, and the initial velocities are set by random numbers corresponding to the physical temperature given by the keyword `<temperature>`.

```
<start_therm_afed>
MANUAL
0  0
```

This keyword sets the initial position and velocity of the Nosé-Hoover thermostat variable η . If “MANUAL” is specified, its initial position and velocity should be given; in this case, $\eta(0) = 0$ and $v_\eta(0) = 0$ (default setting). If “AUTOMATIC” is instead specified, η is set to “0” and v_η is automatically set according to the preset physical temperature.

```
<mass_afed>
MANUAL
1
4
...
```

This keyword sets the parameters that re-adjust the mass of the fictitious CV particles. If “MANUAL” is set, the mass determined by `<params_afed>` (see Sec.3.13.1) is multiplied by the number given below “MANUAL” (the i th line is for the mass of the i th particle). In this example, the mass of the first CV particle is multiplied by “1” and that of the second CV particle is multiplied by “4”. If “AUTOMATIC” is instead specified (default setting), the masses are not re-adjusted.

```
<temperature_tamd>
3000.0
```

This keyword sets the temperature of TAMD, T_{tamd} . In the present case, “3000.0” kelvin.

<niter_bath_afed>
100

This keyword sets the number of AFED steps, n_b , associated with the mass of the Nosé-Hoover thermostat, Q . In the present case, “100” (no unit); i.e., $Q = Dk_B T(n_b \Delta t_{cv})^2$, where D is the number of fictitious particles and T is the physical temperature given by the keyword <temperature>.

<ioption_afed>
1 or 2

This keyword sets the weight W_l in parallel dynamics discussed in Sec.11.16. With “1”, $W_l = 1$ for all beads (replicas), while W_l is set by Eq.(237) (see Sec.11.16) with “2”.

<iprint_weight>
10

This keyword sets the print interval of the weight W_l in parallel dynamics discussed in Sec.11.16. In the present case, W_l are output every 10 steps to *afed.weight.out*. No W_l will be output with “-1” (default).

3.13.6 Logarithmic mean force dynamics

To prepare the file *input.dat*, all the keywords given in Sections 3.1, 3.2, and 3.13.1 are important. In addition, the following keywords are also important.

<method>
AFED

This keyword sets the simulation method. In the present case, “AFED” means adiabatic free energy dynamics.

<afed_type>
LOGMFD

This keyword sets the type of AFED. In the present case, “LOGMFD” means logarithmic mean force dynamics.

<logmfd_type>
NVE

This keyword sets the type of LogMFD. In the present case, “NVE” means NVE-type LogMFD. NVT (Nosé-Hoover dynamics) and VS (velocity scaling dynamics) are also available.

<niter_afed>
200

This keyword sets the number of AFED steps (total updates of the fictitious particle) in LogMFD. In the present case, “200” iterations.

<dt_logmfd>
1.0

This keyword sets the AFED step size, Δt_{cv} , for LogMFD. In the present case, “1.0” (since the dynamics of LogMFD is physically meaningless, the unit can be arbitrarily chosen; e.g., fs).

<alpha_logmfd>
4.0

This keyword sets the parameter $\alpha_{\log}(> 0)$ in LogMFD. In *input.dat*, $\alpha' (= \alpha_{\log} k_B T)$ should instead be given by this keyword, where T is the physical temperature set by the keyword <temperature>. In the present case, $\alpha_{\log} = 4/k_B T$ hartree⁻¹.


```
<gamma_logmfd>
0.25
```

This keyword sets the parameter γ_{\log} in LogMFD. In *input.dat*, γ' ($= \gamma_{\log}/k_B T$) should instead be given by this keyword. Normally, $\gamma_{\log} = 1/\alpha_{\log}$ works well (default setting). In the present case, $\gamma_{\log} = 0.25k_B T$ hartree.

```
<start_afed>
MANUAL
0.2  1.3e-2
1.67 -6.3e-1
...
```

This keyword sets the initial position and velocity of each fictitious CV particle. If “MANUAL” is specified in the first line, the initial position and velocity of each CV particle should be given for each line as in the example above. In this case, the initial position of the first CV particle is 0.2 and its velocity is 1.3e-2. Those of the second CV particle are 1.67 and -6.3e-1, respectively. If “AUTOMATIC” is instead specified (default setting), the initial positions are automatically estimated from the initial configuration of the MD system, and the initial velocities are set by random numbers corresponding to the physical temperature given by the keyword `<temperature>`.

```
<start_therm_afed>
MANUAL
0  0
```

This keyword sets the initial position and velocity of the Nosé-Hoover thermostat variable η . If “MANUAL” is specified, its initial position and velocity should be given; in this case, $\eta(0) = 0$ and $v_\eta(0) = 0$ (default setting). If “AUTOMATIC” is instead specified, η is set to “0” and v_η is automatically set according to the preset physical temperature.

```
<mass_afed>
MANUAL
1
4
...
```

This keyword sets the parameters that re-adjust the mass of the fictitious CV particles. If “MANUAL” is set, the mass determined by `<params_afed>` (see Sec.3.13.1) is multiplied by the number given below “MANUAL” (the i th line is for the mass of the i th particle). In this example, the mass of the first CV particle is multiplied by “1” and that of the second CV particle is multiplied by “4”. If “AUTOMATIC” is instead specified (default setting), the masses are not re-adjusted.

```
<niter_bath_afed>
100
```

This keyword sets the number of AFED steps, n_b , associated with the mass of the Nosé-Hoover thermostat, Q . In the present case, “100” (no unit); i.e., $Q = Dk_B T(n_b \Delta t_{cv})^2$, where D is the number of fictitious particles and T is the physical temperature given by the keyword `<temperature>`.

```
<fenergy_ini_logmfd>
0.01
```

This keyword sets the initial value of the free energy estimated on-the-fly during the LogMFD run. This defines the origin of the energy scale. In the present case, the initial free energy is “0.01” hartree.

```
<ioption_afed>
1 or 2
```

This keyword sets the weight W_l in parallel dynamics discussed in Sec.11.16. With “1”, $W_l = 1$ for all beads (replicas), while W_l is set by Eq.(237) (see Sec.11.16) with “2”.

```
<iprint_weight>
10
```

This keyword sets the print interval of the weight W_l in parallel dynamics discussed in Sec.11.16. In the present case, W_l are output every 10 steps to *afed_weight.out*. No W_l will be output with “-1” (default).

3.14 String method (STRING)

The theoretical background of the string method (STRING) is briefly described in Section 11.7.

As an example, a set of input files for the water dimer is provided. Copy the example files to the directory *run* as follows:

```
> cp -r examples/H402/mm_string run
```

and move to that directory:

```
> cd run/mm_string
```

Let us examine the keywords in *input.dat* one by one.

```
<method>
STRING
```

This keyword sets the simulation method to the string method.

```
<ipotential>
MM
```

This keyword specifies the potential model. In this example, a classical molecular mechanics (MM) force field is employed.

```
<iboundary>
0
```

This keyword sets the boundary condition. In the present case, “0” corresponds to the free boundary condition.

```
<nbead>
32
```

This keyword sets the number of images in the string, which is equal to the number of beads. In this example, 32 images are used.

```
<nstep>
1000
```

This keyword sets the maximum number of string updates. Here, the string is evolved for up to 1000 steps.

```
<dt>
0.01d0
```

This keyword sets the step size for updating the string. In this case, the step size is 0.01 hartree^{1/2} femtoseconds.

```
<ends_string>
FIXED
```

This keyword specifies the boundary condition for the end points of the string. Here, both end points are fixed during the string optimization.

```
<iprint_std>
100
```

This keyword sets the print interval of the standard output. In the present case, output is written every 100 steps.

The output files (*.out) and restart files (*.ini) are explained in Sections 10.2 and 10.3, respectively.

When starting a calculation from scratch, the initial string can be generated automatically as a straight-line interpolation between the reactant and product structures provided in the file *structure.dat*. Alternatively, the user may prepare a restart file *string.ini*, which explicitly contains the atomic coordinates of all images (e.g., 32 images in this example). If both *string.ini* and *structure.dat* are present in the execution directory, the restart file *string.ini* always takes priority.

3.15 Phonon calculation (PHONON)

The theoretical background of phonon calculation (PHONON) is briefly explained in Section 11.6.

As an example, a set of input files for nickel FCC crystal is available. So, let us copy it to the directory *run*:

```
> cp -r examples/Ni/eam_phonon run
```

and enter that directory:

```
> cd run/eam_phonon
```

Let us look at the keywords one by one.

```
<method>
PHONON
```

This keyword sets the simulation method, in the present case, the phonon calculation.

```
<ipotential>
EAM
```

This keyword sets the potential, in the present case, the embedded atom method.

```
<iboundary>
1
33.25917740  0.00000000  0.00000000
0.00000000 33.25917740  0.00000000
0.00000000  0.00000000 33.25917740
```

This keyword sets the boundary condition. In the present case, “1” means that the periodic boundary condition is applied, and the following nine numbers indicate that the simulation box is cubic with a side length of 33.25917740 bohr.

Now, the keywords specific to the phonon calculation are listed below.

```
<cells_phonon>
5 5 5
```

This keyword sets the number of unit cells in the box along the a , b , c axes. In the present case, there are $125 = 5 \times 5 \times 5$ unit cells in the box with the size $\vec{a}_{u.c.} = (L, 0, 0)$, $\vec{b}_{u.c.} = (0, L, 0)$, and $\vec{c}_{u.c.} = (0, 0, L)$, where $L = 33.25917740/5 = 6.65183548$.

```
<kdisp_phonon>
20
0.00000 0.0 0.0
0.05000 0.0 0.0
0.10000 0.0 0.0
0.15000 0.0 0.0
0.20000 0.0 0.0
0.25000 0.0 0.0
0.30000 0.0 0.0
0.35000 0.0 0.0
0.40000 0.0 0.0
0.45000 0.0 0.0
0.50000 0.0 0.0
0.55000 0.0 0.0
0.60000 0.0 0.0
0.65000 0.0 0.0
0.70000 0.0 0.0
0.75000 0.0 0.0
0.80000 0.0 0.0
0.85000 0.0 0.0
0.90000 0.0 0.0
0.94458 0.0 0.0
```

This keyword sets the k -points for the calculation of the phonon dispersion curve. In the present case, 20 points are computed up to $k_{\max} = 2\pi/L = 0.94458$. The results are displayed in the output file *phonon_kdisp.out*.

```
<kdos_phonon>
3 3 3
```

This keyword sets the k -point sampling for the calculation of the phonon density of states. In the present case, $3 \times 3 \times 3$ is chosen.

```
<dosrange_phonon>
0.0 500.0 5.0
```

This keyword sets the frequency range of the phonon density of states, which is printed to the output file *phonon_dos.out*. In the present case, the range is set from 0.0 cm^{-1} to 500.0 cm^{-1} with an increment of 5.0 cm^{-1} .

```
<temprange_phonon>
0.0 900.0 5.0
```

This keyword sets the temperature range of the phonon energies to be displayed in the output file *phonon_energy.out*. In the present case, the range is set from 0.0 K to 900.0 K with an increment of 5.0 K .

The output files **.out* will be explained in Section 10.2 and Section 10.3, respectively.

3.16 Optimization of OM action (OMOPT)

The theoretical background of the OM action is briefly explained in Section 11.8.

As an example, a set of input files for the water dimer is available. Copy it to the directory *run*:

```
> cp -r examples/H402/mm_omopt run
```

and enter that directory:

```
> cd run/mm_omopt
```

Let us look at the keywords one by one.

```
<method>
OMOPT
```

This keyword sets the simulation method, the string method.

```
<ipotential>
MM
```

This keyword sets the potential. In the present case, the MM force field is chosen.

```
<iboundary>
0
```

This keyword sets the boundary condition. Here, “0” means the free boundary condition.

```
<nbead>
120
```

This keyword sets the number of images, i.e., the number of beads in the string method. In the present case, 120 images are chosen.

```
<nstep>
10000
```

This keyword sets the maximum number of updates of the string. Here, 10000 steps are chosen.

```
<dt>
0.625d0
```

This keyword sets the step size for updates of the string. In the present case, 0.625 femtoseconds.

```
<iprint_std>
100
```

This keyword sets the print interval of standard output. Here, it is set to every 100 steps.

```
<gamma_om>
0.00625d0
```

This keyword sets the friction constant in femtoseconds⁻¹. In the present case, it is 0.00625 femtoseconds⁻¹.

```
<fdiff>
1.d-6
```

This keyword sets the maximum shift of atomic coordinates, δ , used to obtain the numerical gradient of the OM action.

The output files **.out* and the restart files **.ini* are explained in Section 10.2 and Section 10.3, respectively.

To run from scratch, one can start from a straight-line interpolation of the atomic positions of the reactant and product ends, which are given in the file *structure.dat*. Alternatively, one can prepare the restart file *string.ini*, in which all the atomic positions of the 120 images are listed. When both the restart file (*string.ini*) and the input file (*structure.dat*) are present, the restart file always has priority.

When the keyword

```
<method>
TESTOM
```

is used instead of `<method>=OMOPT`, the numerical gradient of the OM action is tested.

3.17 User-made potentials

The PIMD code can be modified to implement user-defined potentials. The simplest approach is to edit the source files *force_user.F* and *force_user_MPI.F*, which are used by the serial and parallel versions of the code, respectively.

```
C*****
      subroutine force_user
C*****

C-----
C      /*   shared variables                               */
C-----

      use common_variables, only :
      &   x, y, z, fx, fy, fz, pot, natom, nbead

C-----
C      /*   local variables                               */
C-----

      implicit none

      .....
      .....

      return
      end

C*****
      subroutine force_user_MPI
C*****

C-----
C      /*   shared variables                               */
C-----

      use common_variables, only :
      &   x, y, z, fx, fy, fz, pot, natom, nbead, myrank, nprocs

C-----
C      /*   local variables                               */
C-----

      implicit none

      .....
      .....

      return
      end
```

After modifying these routines, compile the code in the usual way, as described in Section 2. To use the user-defined potential, set `<ipotential> = USER` in *input.dat*.

To verify the correctness of user-made potentials, it is strongly recommended to compare analytical forces and virial values with numerical ones obtained by finite differences. For this purpose, the following test methods are available:

- **<method> = TESTFORCE:** Tests analytical forces against numerical derivatives obtained by displacing atomic positions. This test works for both free boundary conditions (**<iboundary> = 0**) and periodic boundary conditions (**<iboundary> = 1**).
- **<method> = TESTVIRIAL:** Tests the analytical virial matrix by comparing it with numerical derivatives of the potential energy with respect to the box matrix. This test is valid only for periodic boundary conditions (**<iboundary> = 1**).

3.17.1 Testing forces

Starting from a static calculation setup, replace **<method> = STATIC** with

```
<method>
TESTFORCE
```

The precision of the numerical differentiation is controlled by the keyword

```
<fdiff>
1.e-4
```

which sets the displacement δ used to compute numerical forces:

$$-\frac{\partial V}{\partial r_{i\alpha,l}} = -\frac{V(r_{i\alpha,l} + \delta) - V(r_{i\alpha,l} - \delta)}{2\delta}. \quad (1)$$

During execution, output similar to the following is printed:

=====						
atom	bead	F (analytical)	F (numerical)	stat	wall	clock time

1	1	0.2275176522	0.2275176520	OK	2015-05-19	TUE 10:08:23.N
1	1	-0.0197918981	-0.0197918968	OK	2015-05-19	TUE 10:08:23.N
1	1	0.0547412167	0.0547412165	OK	2015-05-19	TUE 10:08:23.N
...

The third and fourth columns show the analytical and numerical values of $-\partial V/\partial r_{i\alpha,l}$ for each Cartesian component. The status column reports “OK” if the difference is within the tolerance; otherwise it is marked as “X”.

3.17.2 Testing virial

To test the virial, replace **<method> = STATIC** with

```
<method>
TESTVIRIAL
```

The numerical differentiation step size is controlled by

```
<fdiff>
1.e-3
```

which sets the displacement δ of the box matrix:

$$-\frac{\partial V}{\partial h_{\alpha\beta}} = -\frac{V(h_{\alpha\beta} + \delta) - V(h_{\alpha\beta} - \delta)}{2\delta}. \quad (2)$$

The analytical derivatives are obtained from the virial matrix:

$$-\frac{\partial V}{\partial h_{\alpha\beta}} = \sum_{\gamma=1}^3 A_{\alpha\gamma} h_{\beta\gamma}^{-1}, \quad (3)$$

where $A_{\alpha\gamma}$ is the virial matrix.

Typical output is as follows:

```
=====
      i      j  -dV/dH (analy)  -dV/dH (numer)  stat  wall clock time
-----
      1      1      0.15682910      0.15682911      OK  2015-05-19 TUE 16:21:53.N
      1      2     -0.11805355     -0.11805355      OK  2015-05-19 TUE 16:21:54.N
      ...
=====
```

Here, “OK” indicates agreement within tolerance. By definition, virial tests are only valid for periodic boundary conditions.

3.17.3 Testing Ewald parameters

For non-polarizable MM force fields (`<ipotential> = MM`) or polarizable force fields (`<ipotential> = POL, OSS`), the Ewald parameters can be tested. These include the precision parameter ϵ and the ratio between real and reciprocal space contributions.

Starting from a static calculation, replace `<method> = STATIC` with

```
<method>
TESTEWALD
```

for charge–charge interactions, or

```
<method>
TESTEWPOL
```

for charge–dipole and dipole–dipole interactions in polarizable force fields.

During execution, output similar to the following is produced:

```
=====
epsilon  ratio n1 n2 n3 l1 l2 l3      energy [au]      wall clock time
-----
0.1E-07  0.100  3  3  3  5  5  5  -339.755346433  2020-05-10 Sun 09:11:53.93
0.1E-06  0.100  3  3  3  5  5  5  -339.755347885  2020-05-10 Sun 09:11:55.52
      ...
=====
```

Here, `n1 n2 n3` denote real-space cutoffs, and `l1 l2 l3` denote reciprocal-space cutoffs, which are automatically determined from ϵ and the ratio. Initial values of these parameters are controlled by the keywords `<ewald>` and `<ewpol>` in *mm.dat*. As with virial tests, Ewald parameter tests are valid only under periodic boundary conditions.

3.18 Mechanochemistry

The theoretical background of mechanochemical calculations (MECH) is briefly explained in Section 11.1.

The external force explicitly included (EFEI) method for mechanochemical calculations [40] is implemented in the PIMD code. The EFEI method can, in principle, be combined with any other methods. An example for a water molecule can be found in the directory *examples/H2O/water_mech*. Enter that directory:

```
> cd run/water_mech
```

The following keywords specify the inclusion of the external force.

```
<mech_type>
EFEI
```

This keyword sets the type of mechanochemical calculation to the external force explicitly included (EFEI) method.

```
<efei>
2 3 0.01
```

This keyword sets the parameters of the EFEI method. In this example, a repulsive force is applied between the pair of atoms 2 and 3 with an amplitude of $F = 0.01$ hartree/bohr.

```
<iprint_mech>
1
```

This keyword sets the print interval for the energy components to the output file, *mech.out*.

When the PIMD code is executed, the output file, *mech.out*, will be generated. See Section 10.2 for the data format of the output file.

3.19 Nonadiabatic dynamics

The following methods for nonadiabatic dynamics are implemented in the PIMD code:

- TFS: surface hopping dynamics (Tully's fewest switches).
- MFE: mean field dynamics (Ehrenfest mean field).

3.19.1 Surface hopping dynamics

The theoretical background of surface hopping dynamics is briefly explained in Section 11.17.

As an example, a set of input files for the LiH molecule is available. To use it, copy the files to the directory *run*:

```
> cp -r examples/LiH/molpro_tfs run
```

and enter that directory:

```
> cd run/molpro_tfs/
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are also required.

```
<method>
TFS
```

This keyword sets the simulation method. In the present case, surface hopping dynamics.

```
<nbead>
4
```

This keyword sets the number of independent trajectories, i.e., the number of beads; in the present case, four trajectories.

```
<dt>  
0.25d0
```

This keyword sets the step size, in the present case, 0.25 femtoseconds.

```
<nstep>  
400
```

This keyword sets the number of steps, in the present case, 400 steps.

```
<nref>  
100
```

This keyword controls the time increment of the state coefficients; in the present case, 1/100 of <dt>.

```
<nstate>  
2
```

This keyword sets the number of adiabatic electronic states. In the present case, two states are chosen: the ground state and the first excited state.

```
<istate_init>  
2
```

This keyword sets the state occupied initially. In the present case, the second lowest state (the first excited state) is chosen.

```
<temperature>  
300.d0
```

This keyword sets the initial temperature of the system. The velocities are generated randomly according to the Maxwell-Boltzmann distribution at this temperature; in the present case, 300 kelvin.

```
<iprint_tfs>  
1
```

This keyword sets the print interval of the output file *tfs.out*; in the present case, every step.

```
<iprint_nac>  
1
```

This keyword sets the print interval of the output file *nac.out*; in the present case, every step.

Input data for MOLPRO. In the PIMD code, nonadiabatic dynamics, i.e., surface hopping dynamics and mean field dynamics, can be carried out with the MOLPRO code. The MOLPRO input file, *molpro.dat*, should be present in the execution directory. The following is an example for the LiH molecule using the CASSCF method and the 6-31g(d,p) basis set. The input file sets up the static calculation of the potential energy, the dipole moment, the forces, and the nonadiabatic coupling elements. It outputs the data to the file *table.molpro*.

```
> cat molpro.dat  
  
***,title  
memory,100,m  
basis=6-31g(d,p)
```

```

geomtyp=xyz
geometry={nosym;noorient;
2
comment
li  0.0000  0.0000  0.0000
h   5.0000  0.0000  0.0000
}
int;
multi;
occ,3;frozen,0;closed,1;wf,4,1,0;state,2;weight,1,1
cpmcscf,nacm,1.1,2.1,accu=1.d-7,record=5101.1
cpmcscf,grad,1.1,accu=1.d-7,record=5102.1
cpmcscf,grad,2.1,accu=1.d-7,record=5103.1

e=energy
table,e                      !print a table with results
save,table.molpro           !save to file
title,<energy>               !title for the table
format,(e24.16)             !explicit format

table,dmx,dmy,dmz           !print a table with results
save,table.molpro           !save to file
title,<dipole>               !title for the table
format,(3e24.16)           !explicit format

force
samc,5101.1
varsav
table,gradx,grady,gradz     !print a table with results
save,table.molpro           !save to file
title,<nacme> 1 2            !title for the table
format,(3e24.16)           !explicit format

force
samc,5102.1
varsav
table,gradx,grady,gradz     !print a table with results
save,table.molpro           !save to file
title,<grad> 1              !title for the table
format,(3e24.16)           !explicit format

force
samc,5103.1
varsav
table,gradx,grady,gradz     !print a table with results
save,table.molpro           !save to file
title,<grad> 2              !title for the table
format,(3e24.16)           !explicit format

```

This file can be tested by running MOLPRO directly:

```
> molpro < molpro.dat
```

If this works correctly, the PIMD code is ready to run.

Upon executing PIMD, the output file *tfs.out* is generated.

```
> less tfs.out
```

```
.....
=====
      step  bead  s occ      pop/s  ratio/s      potential/s  potential/occ
-----
        0     1   1  2  0.000000  0.000000      -7.92966939      -7.86722151
        0     1   2  2  1.000000  1.000000      -7.86722151      -7.86722151
        0     2   1  2  0.000000  0.000000      -7.92966939      -7.86722151
        0     2   2  2  1.000000  1.000000      -7.86722151      -7.86722151
        0     3   1  2  0.000000  0.000000      -7.92966939      -7.86722151
        0     3   2  2  1.000000  1.000000      -7.86722151      -7.86722151
        0     4   1  2  0.000000  0.000000      -7.92966939      -7.86722151
        0     4   2  2  1.000000  1.000000      -7.86722151      -7.86722151
        1     1   1  2  0.000004  0.000000      -7.92988085      -7.86730272
        1     1   2  2  0.999996  1.000000      -7.86730272      -7.86730272
        1     2   1  2  0.000003  0.000000      -7.92947539      -7.86714613
        1     2   2  2  0.999997  1.000000      -7.86714613      -7.86714613
        1     3   1  2  0.000004  0.000000      -7.92988085      -7.86730272
        1     3   2  2  0.999996  1.000000      -7.86730272      -7.86730272
        1     4   1  2  0.000003  0.000000      -7.92947539      -7.86714613
        1     4   2  2  0.999997  1.000000      -7.86714613      -7.86714613
.....
```

The first three columns are the step number, the bead number, and the state number. The fourth column is the occupied state of the corresponding bead. The fifth column is the population of the corresponding state for each bead. The sixth column is the occupancy ratio of the corresponding state among all the beads. The seventh and eighth columns are the potential energies of the corresponding state and the occupied state, respectively, for each bead.

3.19.2 Mean field dynamics

The theoretical background of mean field dynamics is briefly explained in Section 11.17.

As an example, a set of input files for the LiH molecule is available. Copy it to the *run* directory:

```
> cp -r examples/LiH/molpro_mfe run
```

Then enter that directory:

```
> cd run/molpro_mfe/
```

The keywords for the file *input.dat* are basically the same as those in Section 3.19.1, except that the keyword `<method> = TFS` is replaced by

```
<method>
MFE
```

and the keyword `<iprint_tfs>` is replaced by

```
<iprint_mfe>
1
```

which controls the print interval of the output file, *mfe.out*.

When PIMD is executed, the output file *mfe.out* is generated.

```
> less mfe.out
```

```
.....
```

=====								
step	bead	s	occ	pop/s	occ/s	pot/state	pot/occ	

0	1	1	0	0.000000	0.000000	-7.92966939	-7.86722151	
0	1	2	0	1.000000	1.000000	-7.86722151	-7.86722151	
0	2	1	0	0.000000	0.000000	-7.92966939	-7.86722151	
0	2	2	0	1.000000	1.000000	-7.86722151	-7.86722151	
0	3	1	0	0.000000	0.000000	-7.92966939	-7.86722151	
0	3	2	0	1.000000	1.000000	-7.86722151	-7.86722151	
0	4	1	0	0.000000	0.000000	-7.92966939	-7.86722151	
0	4	2	0	1.000000	1.000000	-7.86722151	-7.86722151	
1	1	1	0	0.000004	0.000003	-7.92988085	-7.86730277	
1	1	2	0	0.999996	0.999997	-7.86730272	-7.86730277	
1	2	1	0	0.000003	0.000003	-7.92947539	-7.86714619	
1	2	2	0	0.999997	0.999997	-7.86714613	-7.86714619	
1	3	1	0	0.000004	0.000003	-7.92988085	-7.86730277	
1	3	2	0	0.999996	0.999997	-7.86730272	-7.86730277	
1	4	1	0	0.000003	0.000003	-7.92947539	-7.86714619	
1	4	2	0	0.999997	0.999997	-7.86714613	-7.86714619	

```
.....
```

The first three columns are the step number, the bead number, and the state number. The fourth column is always zero (not meaningful, since the occupied state is mixed). The fifth column shows the population of the corresponding state for each bead. The sixth column shows the occupancy ratio of the corresponding state among all beads. The seventh and eighth columns are the potential energies of the corresponding state and the mixed state, respectively, for each bead.

3.20 Molecular dynamics of rigid molecules

As an example, a set of input files for the aqueous solution of NaCN is available. To use it, copy the files to the directory *run*:

```
> cp -r examples/"NaCN(aq)"/mm_rotor run
```

and enter that directory:

```
> cd run/mm_rotor/
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are also required.

```
<method>
ROTOR
```

This keyword sets the simulation method, which in the present case is molecular dynamics of rigid molecules.

```
<ensemble>
NVT
```

This keyword sets the statistical ensemble to NVT.

```
<dt>
0.25d0
```

This keyword sets the step size, which in the present case is 0.25 femtoseconds.

```
<nstep>  
400
```

This keyword sets the number of steps, which in the present case is 400.

```
<temperature>  
300.d0
```

This keyword sets the temperature of the system, which in the present case is 300 kelvin.

```
<iboundary>  
1  
37.237981 0.000000 0.000000  
0.000000 37.237981 0.000000  
0.000000 0.000000 37.237981
```

This keyword sets the boundary condition and the initial box. Here, “1” indicates a periodic boundary, and the following three lines define the matrix of the initial box. In the present case, a cubic box with a side length of 37.237981 bohr is used.

```
<iread_exit>  
10
```

This keyword sets the interval for reading *exit.dat*, which in the present case is every 10 steps.

```
<iprint_std>  
10
```

```
<iprint_rest>  
10
```

```
<iprint_rdf>  
10
```

```
<iprint_trj>  
10
```

```
<iprint_xyz>  
10
```

```
<iprint_dip>  
10
```

These keywords set the print intervals for the standard output, the restart files, the radial distribution functions, the trajectories in the regular and xyz formats, and the dipole moments, respectively.

Now, the keywords specific to `<method> = ROTOR` are introduced.

```
<components>  
3  
H2O 4 GENERAL  
-1.430429 -0.984140 0.000000  
1.430429 -0.984140 0.000000  
0.000000 0.123017 0.000000  
0.000000 -0.160441 0.000000
```

```

CN      4      LINEAR
0.000000  0.000000 -1.020452
0.000000  0.000000  0.805023
0.000000  0.000000  1.190527
0.000000  0.000000  1.610047
Na      1      MONOATOM
0.000000  0.000000  0.000000

```

This keyword defines the molecular components. There are three components: H₂O, CN, and Na. The H₂O and CN molecules each have four interaction sites, while Na has a single interaction site. Since the CN molecule is linear, its atoms must be arranged along the *z* axis.

```

<molecules>
256
H2O      1  255  509  763
H2O      2  256  510  764
...      ...  ...  ...  ...
...      ...  ...  ...  ...
H2O     254  508  762 1016
CN     1017 1018 1019 1020
Na     1021

```

This keyword sets the number of molecules, which in the present case is 256. The following lines assign the site numbers for each molecule.

3.20.1 Gentlest ascent dynamics (GAD)

In the PIMD code, gentlest ascent dynamics is implemented [79].

As an example, a set of input files for C₂H₄O is available. Let us copy it to the directory *run*:

```
> cp -r examples/C2H4O/smash_gad run
```

and enter that directory:

```
> cd run/smash_gad
```

To prepare the file *input.dat*, all the keywords given in Sections 3.1 and 3.2 are important. In addition, the following keywords are also required.

```

<method>
GAD

```

This keyword sets the simulation method. In the present case, “GAD” corresponds to gentlest ascent dynamics.

```

<dt>
0.01d0

```

This keyword sets the step size in hartree^{0.5} femtoseconds.

```

<nstep>
1000

```

This keyword sets the number of GAD steps.

```

<iprint_xyz>
1

```

This keyword sets the print interval for the trajectory in xyz format. In the present case, every ten steps.

```
<nbead>  
8
```

This keyword sets the number of GAD trajectories computed in parallel. In the present case, eight trajectories.

```
<fdiff>  
1.d-3
```

This keyword sets the finite difference parameter used in GAD. In the present case, 10^{-3} bohr.

```
<irandom>  
0
```

This keyword sets the random number seed. In the present case, the random numbers start from a predefined seed.

```
<gamma_gad>  
0.1
```

This keyword sets the gamma value for GAD. In the present case, “0.1” atomic units.

```
<energy_max_gad>  
-150.6
```

This keyword sets the maximum energy value for GAD. In the present case, “-150.6” hartree.

The output files **.out* and the restart files **.ini* will be explained in Section 10.2 and Section 10.3, respectively.

3.21 Consecutive static calculations (SCAN)

Consecutive static calculations are performed using `<method> = SCAN`. A series of molecular configurations is read from the file *structure.dat*. Upon executing the PIMD code, the potential energy, forces, and dipole moment are computed for each molecular configuration based on `<ipotential>`. The results are printed to the following files: *coord_scan.out* (atomic coordinates), *potential_scan.out* (potential energy), *forces_scan.out* (forces), and *dipole_scan.out* (dipole moment).

In addition, an external shell script can be executed for each structure. The file name of the shell script is specified by the keyword `<scan_exe_shell>`.

4 Potentials

This section explains the data contained in the potential files. The file names and required data depend on the potential being used. Input files for external codes (e.g., *ab initio* electronic-structure programs) must be prepared by the user. In principle, the PIMD code reuses the same input files that are valid for the corresponding external codes. For simplicity, most examples presented below use a water molecule with different potential models.

IMPORTANT: Only the geometry information provided in the PIMD input files—*structure.dat*, *geometry.ini*, or *string.ini*—is used by the PIMD code. Consequently, any geometry specified directly in the input files of external codes is ignored during PIMD simulations.

4.1 Empirical force fields

In the PIMD code, the following empirical force fields are implemented:

- MM: molecular mechanics (classical force field).
- EAM: embedded atom method.
- ADP: angular dependent potential.
- PAIR: pair potential.
- TersOFF: Tersoff potential.

4.1.1 Molecular mechanics (MM)

In the PIMD code, the molecular mechanics potential is given by the sum of the following terms:

- V_{lin} : the linear bonds in harmonic form.
- V_{genlin} : the linear bonds in general anharmonic form.
- V_{angl} : the angular bonds.
- V_{dih} : the dihedral angular bonds.
- V_{impr} : the improper angular bonds.
- V_{cmap} : the CMAP correction for the CHARMM force field.
- V_{lj} : Lennard-Jones (LJ) interaction of 6-12 type.
- V_{es} : the electrostatic interaction.
- V_{mrs} : the Morse interaction.
- V_{buck} : the Buckingham interaction.

The V_{es} term can be either polarizable (MM) or non-polarizable (POL). Parameters for each term must be specified in the input file called *mm.dat* in the execution directory. See Section 9.7 for details.

Example of non-polarizable liquid water. The following is an example of the input file *mm.dat* for liquid water composed of 256 non-polarizable water molecules (768 atoms in total). The order of atoms is O(1), ..., O(256), H(257), ..., H(512), H(513), ..., H(768). The parameters are taken from the flexible TIP3P model [41, 42].

```
> less examples/water/water_static/mm.dat
```

```
<linear_bonds>
```

```
512
1 257 0.180884D+01 0.401926D+00
1 513 0.180884D+01 0.401926D+00
... ..
... ..
256 512 0.180884D+01 0.401926D+00
256 768 0.180884D+01 0.401926D+00
```

```
<angular_bonds>
```

```
256
```

```

257  1 513  0.104520D+03  0.534400D-04
258  2 514  0.104520D+03  0.534400D-04
... ..
... ..
511 255 767  0.104520D+03  0.534400D-04
512 256 768  0.104520D+03  0.534400D-04

```

<lennard-jones>

```

293760
0.153900D+02  0.182250D+02
  1  2  0.242560D-03  0.595369D+01
  1  3  0.242560D-03  0.595369D+01
... ..
... ..
257 258 0.733600D-04  0.755910D+00
257 259 0.733600D-04  0.755910D+00
... ..
... ..
256 766 0.133400D-03  0.335480D+01
256 767 0.133400D-03  0.335480D+01

```

<charges>

```

768
  1 -0.834000D+00
  2 -0.834000D+00
. ....
. ....
767  0.417000D+00
768  0.417000D+00

```

<nbcpr>

```

768
  1 257  0.0
257 513  0.0
... ..
512 768  0.0
768 256  0.0

```

<ewald>

```

1.d-08  11.d+00  0

```

- **<linear_bonds>**: There are two OH linear bonds per water molecule, $O(n)$ - $H(n + 256)$ and $O(n)$ - $H(n + 512)$ for $1 \leq n \leq 256$. Thus, there are $256 \times 2 = 512$ OH linear bonds in total. Each line lists the two atoms involved in the linear bond, together with the equilibrium bond lengths (1.80884 bohr) and the force constants (0.401926 hartree/bohr²).
- **<angular_bonds>**: There is one HOH angular bond per water molecule, $H(n + 256)$ - $O(n)$ - $H(n + 512)$ for $1 \leq n \leq 256$. Thus, there are 256 HOH angular bonds in total. Each line lists the three atoms involved in the angular bond, together with the equilibrium bond angles (104.52 degrees) and the force constants (0.5344×10^{-4} hartree/degrees²).

- **<lennard-jones>**: There are $256 \times 255/2 = 32640$ intermolecular OO pairs, $512 \times 511/2 - 256 = 130560$ intermolecular HH pairs, and $256 \times 512 - 256 = 130560$ intermolecular OH pairs. Thus, there are $32640 + 130560 + 130560 = 293760$ LJ interactions in total. The LJ cutoff distances are set to 15.390 bohrs (inner) and 18.225 bohrs (outer). Within the range $15.39 < r < 18.225$, all LJ interactions are smoothly damped using a switching function. Each line lists the two atoms involved together with the LJ parameters, i.e., $\epsilon = (0.24256, 0.07336, 0.13340) \times 10^{-3}$ hartrees and $\sigma = (5.95369, 0.75591, 3.35480)$ bohrs for the intermolecular (OO, HH, OH) pairs, respectively.
- **<charges>**: There are 768 atoms, each with an atomic charge. Each line gives the atomic charge, -0.834 for O atoms and 0.417 for H atoms.
- **<nbc>**: The electrostatic interaction is neglected (the scaling factor is zero) for all 1-2 and 1-3 bonded pairs. Such bonded charge pairs correspond to three intramolecular pairs (two OH and one HH) per water molecule. Thus, there are 768 pairs in total. Each line lists these pairs together with the scaling factor, 0.0.
- **<ewald>**: For the Ewald sum of charge-charge interactions, the precision is set to 10^{-8} hartrees. The ratio of computational time between the real and reciprocal space contributions is set to 11.0. The dipole correction is not included (0).

Example of polarizable liquid water. The following is an example of the input file *mm.dat* for liquid water composed of 96 polarizable water molecules and one chloride ion (289 atoms in total). The parameters are taken from the RPOL model [85]. The intra-molecular parameters, as well as the Lennard-Jones parameters, should be set in the same manner as above, while the inter-molecular parameters should be set as follows:

```
> less examples/water/water_static/mm.dat
```

```
<linear_bonds>
```

```
.....
```

```
<angular_bonds>
```

```
.....
```

```
<lennard-jones>
```

```
.....
```

```
<charges>
```

```
289
```

```
  1 -0.730000D+00  3.563119D+00  1
  2 -0.730000D+00  3.563119D+00  1
  . .....
  . .....
287  0.365000D+00  1.147216D+00  2
288  0.365000D+00  1.147216D+00  2
289 -1.000000D+00  3.698086D+01  3
```

```
<damping>
```

```
4
```

```
DD  1  3  EXP  1.111158D+01
CC  2  3  EXP  4.119602D+00
CD  2  3  EXP  4.119602D+00
CD  3  2  EXP  4.119602D+00
```

<nbc<div data-bbox="112 142 174 151" data-label="Text">

.....

<ewald>

.....

<ewpol>

1.d-8 0.1d0

- **<charges>**: There are 289 atoms, each with an atomic charge. Each line gives the atomic charge (-0.730, 0.365, and -1.000 for O, H, and Cl atoms, respectively), the atomic polarizability in bohr⁻³ (3.563119, 1.147216, and 36.98086 for O, H, and Cl, respectively), and the atomic kind (1, 2, and 3 for O, H, and Cl, respectively).
- **<damping>**: There are 4 non-zero damping functions in the Thole exponential (EXP) form. The dipole-dipole (DD) damping function between atomic kinds 1 and 3 (O and Cl), the charge-charge (CC), and charge-dipole (CD) damping functions between atomic kinds 2 and 3 (H and Cl). The damping parameters (11.11158 and 4.119602) are given in bohr units. Note that the CD damping functions must be specified separately for kinds *x* and *y*, and for kinds *y* and *x*.
- **<ewpol>**: The parameters for the Ewald sum of charge-dipole and dipole-dipole interactions are given by this keyword. The precision is 10⁻⁸, and the ratio of computational time between the real and reciprocal space is 0.1.

4.1.2 TIP4P

A special routine for TIP4P potentials has been implemented. This routine can be used not only for pure water but also for aqueous solutions. To use this routine, one has to take care of three points, which are mentioned below.

- Set the keyword

<ipotential>

TIP4P

in the file *input.dat*.

- Set the keyword

<TIP4P>

O H 0.744

in the file *mm.dat*. The first two characters, O and H, are the symbols of oxygen and hydrogen atoms. These symbols should correspond to those designated in the file *structure.dat* (in the case of the old input style, those designated by the keyword **nspec** in the file *input.dat*). In this case, all O and H atoms should belong to TIP4P water molecules. For each O atom, two H atoms must exist within a distance of 1.4 Å, and they must be unshared with any other O atoms. Otherwise, the run will stop with an error message. The last value is the γ value, which determines the position of the M site, i.e., $\vec{r}_M = \gamma \vec{r}_O + \frac{1-\gamma}{2} (\vec{r}_{H'} + \vec{r}_{H''})$.

- Set all parameters in *mm.dat* as in the case of a TIP3P aqueous solution. A slight difference is that one assumes that the M charge belongs to the nearest O atom. The keyword **<charge>** designates the charge values of M in TIP4P, where those of O would have been designated in TIP3P. The keyword **<nbc<div data-bbox="487 918 510 931" data-label="Page-Footer">

76**

4.1.3 Embedded atom method (EAM)

The embedded atom method can be used by designating the keyword

```
<ipotential>
EAM
```

The theoretical background is briefly described in Section 9.8.

Example: hydrogen in FCC nickel crystal. The following is an example of the input file *eam.dat* for a nickel-hydrogen system composed of 500 nickel atoms and 1 hydrogen atom (501 atoms in total). The order of atoms is Ni(1), \dots , Ni(500), H(501). The parameters are taken from the paper by Baskes et al. [47].

```
<nref_eam>
1000
```

This keyword specifies the number of reference points for the table functions $\varrho(r)$, $F(\rho)$, and $\phi(r)$, which is 1000 in the present case.

```
<rcut_eam>
5.7d0
```

This keyword specifies the cutoff distance, which is 5.7 Å in this case.

```
<rhoeam>
1
0.0000000000000000E+00  0.0000000000000000E+00
0.5678391991991992E-02  0.1353993104251454E-10
.....
.....
2
0.0000000000000000E+00  0.0000000000000000E+00
0.5678391991991992E-02  0.8576998932607242E-09
.....
.....
```

This keyword provides the table for the electron density functions ϱ as a function of interatomic distance r in Å. The data shown here are

- “1”, meaning the first atomic species, Ni.
- 1000 lines of r and $\varrho_{\text{Ni}}(r)$.
- “2”, meaning the second atomic species, H.
- 1000 lines of r and $\varrho_{\text{H}}(r)$.

```
<frhoeam>
1
0.0000000000000000E+00  0.0000000000000000E+00
0.1300722992992993E-01 -0.1332662793615358E+00
.....
.....
2
0.0000000000000000E+00  0.0000000000000000E+00
0.1300722992992993E-01 -0.1535560579048078E+00
.....
.....
```

This keyword provides the table for the embedding functions F in eV as a function of the atomic electron density ρ . The data shown here are

- “1”, meaning the first atomic species, Ni.
- 1000 lines of ρ and $F_{\text{Ni}}(\rho)$.
- “2”, meaning the second atomic species, H.
- 1000 lines of ρ and $F_{\text{H}}(\rho)$.

```
<phir_eam>
1 1
0.5678391959798995E-02 0.6631473037261000E+02
0.1135678395182324E-01 0.6516245220935400E+02
.....
.....
1 2
0.5678391959798995E-02 0.2625695627813300E+03
0.1135678395182324E-01 0.2552302121997200E+03
.....
.....
2 2
0.5678391959798995E-02 0.3685598530752200E+02
0.1135678395182324E-01 0.3572904277414200E+02
.....
.....
```

This keyword provides the table for the pair functions ϕ in eV as a function of the interatomic distance r in Å. The data shown here are

- “1 1”, meaning the atomic species Ni and Ni.
- 1000 lines of r and $\phi_{\text{Ni-Ni}}(r)$.
- “1 2”, meaning the atomic species Ni and H.
- 1000 lines of r and $\phi_{\text{Ni-H}}(r)$.
- “2 2”, meaning the atomic species H and H.
- 1000 lines of r and $\phi_{\text{H-H}}(r)$.

4.1.4 Angular dependent potential (ADP)

The angular dependent potential, developed by Mishin et al. [50], can be used by designating the keyword

```
<ipotential>
ADP
```

The theoretical background is briefly described in Section 9.8.

Example: Iron-Nickel crystal. The following is an example of the input file *eam.dat* for an iron-nickel system composed of 3 nickel atoms and 1 iron atom (4 atoms in total). The order of atoms is Fe(1), Ni(2), Ni(3), Ni(4). The parameters are taken from the paper by Mishin et al. [50].

```
<nref_eam>
  3000
```

This keyword specifies the number of reference points for the table functions $\varrho(r)$, $F(\rho)$, $\phi(r)$, $u(r)$, and $w(r)$, which is 3000 in the present case.

```
<rcut_eam>
  5.1683020
```

This keyword specifies the cutoff distance, which is 5.1683020 Å in this case.

```
<rhoeam>
  1
  1.50118494266664  0.135029877700000
  1.50236988533328  0.135019407000000
  .....
  .....
  2
  1.50122276733329  7.97532702200000D-002
  1.50244553466657  7.97200488300000D-002
  .....
  .....
```

This keyword provides the table for the electron density functions ϱ as a function of interatomic distance r in Å. The data shown here are

- “1”, meaning the first atomic species, Fe.
- 3000 lines of r and $\varrho_{\text{Fe}}(r)$.
- “2”, meaning the second atomic species, Ni.
- 3000 lines of r and $\varrho_{\text{Ni}}(r)$.

```
<frhoeam>
  1
  1.61583175409239D-003  -0.157213669000000
  3.23166350818479D-003  -0.258334267800000
  .....
  .....
  2
  9.70543866871617D-004  -3.02907373000000D-002
  1.94108773374323D-003  -5.06347214800000D-002
  .....
  .....
```

This keyword provides the table for the embedding functions F in eV as a function of the atomic electron density ρ . The data shown here are

- “1”, meaning the first atomic species, Fe.
- 3000 lines of ρ and $F_{\text{Fe}}(\rho)$.

- “2”, meaning the second atomic species, Ni.
- 3000 lines of ρ and $F_{\text{Ni}}(\rho)$.

<phir_eam>

```

1 1
1.50118494266664 81.8259297900000
1.50236988533328 81.1895109900000
.....
.....
1 2
1.50122786633336 50.0535601700000
1.50245573266673 49.6656730200000
.....
.....
2 2
1.50122276733329 2.92413458500000
1.50244553466657 2.91770869700000
.....
.....
```

This keyword provides the table for the pair functions ϕ in eV as a function of the interatomic distance r in Å. The data shown here are

- “1 1”, meaning the atomic species Fe and Fe.
- 3000 lines of r and $\phi_{\text{Fe-Fe}}(r)$.
- “1 2”, meaning the atomic species Fe and Ni.
- 3000 lines of r and $\phi_{\text{Fe-Ni}}(r)$.
- “2 2”, meaning the atomic species Ni and Ni.
- 3000 lines of r and $\phi_{\text{Ni-Ni}}(r)$.

<ur_adp>

```

1 1
0.1501184943E+01 0.8546158795E-01
0.1502369885E+01 0.8547910647E-01
.....
.....
1 2
1.50122786633336 4.31348913400000D-002
1.50245573266673 4.31657669700000D-002
.....
.....
2 2
0.1501222767E+01 -0.3653576403E-01
0.1502445535E+01 -0.3648116307E-01
.....
.....
```

This keyword provides the table for the pair functions u in $\text{eV}^{1/2} \text{Å}^{-1}$ as a function of the interatomic distance r in Å. The data shown here are

- “1 1”, meaning the atomic species Fe and Fe.
- 3000 lines of r and $u_{\text{Fe-Fe}}(r)$.
- “1 2”, meaning the atomic species Fe and Ni.
- 3000 lines of r and $u_{\text{Fe-Ni}}(r)$.
- “2 2”, meaning the atomic species Ni and Ni.
- 3000 lines of r and $u_{\text{Ni-Ni}}(r)$.

```
<wr_adp>
  1   1
0.1501184943E+01  0.1951794978E+00
0.1502369885E+01  0.1948689593E+00
.....
.....
  1   2
1.50122786633336  0.111232678200000
1.50245573266673  0.111050965000000
.....
.....
  2   2
0.1501222767E+01  0.1084737450E+00
0.1502445535E+01  0.1082973717E+00
.....
.....
```

This keyword provides the table for the pair functions w in $\text{eV}^{1/2} \text{\AA}^{-1}$ as a function of the interatomic distance r in \AA . The data shown here are

- “1 1”, meaning the atomic species Fe and Fe.
- 3000 lines of r and $w_{\text{Fe-Fe}}(r)$.
- “1 2”, meaning the atomic species Fe and Ni.
- 3000 lines of r and $w_{\text{Fe-Ni}}(r)$.
- “2 2”, meaning the atomic species Ni and Ni.
- 3000 lines of r and $w_{\text{Ni-Ni}}(r)$.

4.1.5 Pair potential

The tabulated pair potential can be used by designating the keyword

```
<ipotential>
PAIR
```

The input is provided by the file *eam.dat* with the keywords `<nref_eam>`, `<rcut_eam>`, and `<phir_eam>` in the same format as in the case of EAM.

4.1.6 Tersoff potential

The Tersoff potential can be used by specifying the keyword

```
<ipotential>  
TERSOFF
```

The input is provided in the file *tersoff.dat* using the keywords `<tersoff>`. The theoretical background is briefly described in Section 9.9.

4.1.7 GAL21 potential

For water-metal interfaces, the GAL21 potential can be used by specifying the keyword

```
<ipotential>  
METALWATER
```

for seven metals: Pt, Ag, Au, Co, Cu, Ni, and Pd. See reference [49] for details.

4.2 User-defined potentials

User-defined potentials can be used by designating the keyword

```
<ipotential>  
USER
```

along with the execution command, input file, and output file, which are designated by the keywords `<user_command>`, `<user_input_file>`, and `<user_output_file>`, respectively. Refer to Section 9.4 for further details.

4.3 Born–Oppenheimer potentials

In the PIMD code, Born–Oppenheimer potentials are evaluated by calling external electronic-structure programs based on *ab initio*, density functional theory (DFT), or semiempirical methods. At present, the PIMD code can be coupled with the following packages:

- ABINIT-MP [69] (ab initio; fragment molecular orbital method).
- CP2K [55] (ab initio).
- DFTB+ [54] (density functional tight binding).
- MOPAC [53] (semiempirical).
- GAMESS [61] (ab initio, density functional theory, semiempirical).
- GAUSSIAN 98, 03, 09 [57, 58, 59] (ab initio, density functional theory, semiempirical).
- MOLPRO [63] (ab initio).
- ORCA [65] (ab initio, density functional theory).
- PHASE/0 [56] (density functional theory).
- QUANTUM ESPRESSO [66] (density functional theory).
- SMASH [68] (ab initio, density functional theory).
- TURBOMOLE [62] (ab initio, density functional theory).
- VASP 6 [67] (density functional theory).

Among these, CP2K, DFTB+, QUANTUM ESPRESSO, SMASH, and VASP 6 can be internally linked to the PIMD code, enabling efficient and tightly coupled simulations.

4.3.1 SMASH

SMASH is an open-source electronic structure code released under the Apache License 2.0.

The ab initio SMASH code is linked to the PIMD program through the subroutine *force_smash.F*. To use this interface, SMASH must be compiled together with the PIMD code.

To enable the linkage, modify the PIMD *makefile* as follows:

- Add the compiler option `-Dsmash` to activate the SMASH interface.
- Specify the options `SMASHOPT` and `SMASHOPTMPI` for compiling the modified SMASH code (serial and MPI versions).

Prepare the original SMASH source code, for example in the directory *lib/smash/smash.1.1.0.src*. Then copy it to a new directory for modification:

```
> cd lib/smash
> cp -r smash.1.1.0.src smash.1.1.0.modified
```

Next, compile the code by executing:

```
> cd ../../compile
> make veryclean
> make
```

During this process, the following steps are carried out automatically:

- The shell script *lib/smash/smash.1.1.0.modified/modify_smash.sh* is executed to apply the required modifications to the SMASH source code.
- The modified SMASH code is compiled using the options `SMASHOPT` and `SMASHOPTMPI`.
- A static library named *lib/libsmash.a* is created.
- Finally, the PIMD code is compiled with the `-Dsmash` option and linked against *lib/libsmash.a*.

NOTE: In SMASH version 1.1.0, force calculations are available only for Hartree–Fock and density functional theory (DFT) methods. MP2 forces are not supported. In addition, dipole moments are not available. These limitations are inherited by the PIMD–SMASH interface.

NOTE: The script *modify_smash.sh* requires the command `sed -i` to be available on the system. This option is not enabled by default on some systems, such as standard macOS installations.

Setups of SMASH. To perform calculations using SMASH within PIMD, prepare the PIMD input file *input.dat* and set:

```
<ipotential>
SMASH
```

This keyword instructs PIMD to call the subroutine *force_smash.F* for force evaluations.

The initial guess for molecular orbitals can be controlled by:

```
<smash_guess>
PREVIOUS
```

In this example, the molecular orbitals from the previous step are reused as the initial guess.

The standard SMASH input options should be provided using the keyword `<smash_options>` as follows:

```
<smash_options>
job method=b3lyp basis=6-31g(d) memory=1gb charge=0
control iprint=1
```

Here, the SMASH input lines beginning with `job`, `control`, and similar directives should be specified explicitly. Exceptionally, the `geom` section and the nuclear coordinates do not need to be included, as atomic coordinates are provided by the PIMD code.

Execution of PIMD. The PIMD calculation with SMASH is executed in the usual way using:

```
> pimd.x
```

or, for parallel execution,

```
> pimd.mpi.x
```

4.3.2 ABINIT-MP

IMPORTANT: The user is solely responsible for all issues regarding the ABINIT-MP license. PIMD does not provide any ABINIT-MP license.

The ab initio ABINIT-MP code is linked externally at the subroutine *force_abinit_mp5.F* of the PIMD code. The ABINIT-MP code is linked internally at the subroutine *force_abinit_mp_MPI.F*. The ABINIT-MP code must be installed beforehand in order to use this link. The source code is complete, but it has not been tested with a sufficient number of examples. Therefore, the user's manual for this part is incomplete. To use these methods, the user is encouraged to carefully read the source code.

4.3.3 CP2K

IMPORTANT: The user is solely responsible for all issues related to the CP2K license. PIMD does not provide a CP2K license.

The CP2K code for quantum chemistry and solid-state physics [55] is linked directly to PIMD through the interface implemented in the subroutine *force_libcp2k.F*, which calls the F77 CP2K API. To use this interface, the CP2K API library must be compiled beforehand. This can be done, for example, by invoking `make ARCH=psmp libcp2k` in the directory where CP2K was compiled. Please consult the CP2K documentation for compilation instructions.

Linking the CP2K API (Cmake) This method was tested in December 2024 with the latest version of CP2K available at the time (2024.3). To link to the CP2K API, first ensure that CP2K is built with the API enabled. Then run CMake in a build directory (e.g., *pimd/build*) with the following flags:

```
cmake -DCP2K=ON -DCP2K_DIR=CP2KDIR -DCP2K_VER=CP2KVER ../.
```

Here, `CP2KDIR` is the directory from which you compiled CP2K (e.g., */home/g3/a184053/QM-Programs/cp2k_new*), and `CP2KVER` is the CP2K build configuration you used (e.g., *Linux-x86-64-intel-minimal/ssmp*). To find the configuration you compiled, look in the *obj/* or *lib/* directories under the CP2K build directory.

The keywords above can, of course, be combined with other options, such as those needed to install AENET or N2P2. Just ensure that all programs are compiled with the same toolchain.

Linking the CP2K API (Old method) This method was tested in February 2020 using a development version of CP2K and the *Linux-x86-64-intel-minimal.psm* architecture file. To link to the CP2K API, first ensure that CP2K is built with the API enabled. Then make a few changes to the PIMD makefile around the following lines.

```

#-----
#      Usage of CP2K as static library - without MPI
#-----

#      compile PIMD with CP2K as a static library
#CP2KFLAG = -Dlibcpgk

#      compile PIMD without CP2K as a static library
CP2KFLAG =

ifdef CP2KLIBFLAG
  CP2KDIR = /home/g3/a184053/QM-Programs/cp2k_new
  CP2KVER = Linux-x86-64-intel-minimal/ssmp

  CP2KINC = -I$(CP2KDIR)/obj/$(CP2KVER)/
  CP2KLIB = -L$(CP2KDIR)/lib/$(CP2KVER)/ \
            -lcp2k -L$(CP2KDIR)/lib/$(CP2KVER)/exts/dbcsr/ -ldbcsr
endif

#-----
#      Usage of CP2K as static library - with MPI
#-----

#      compile PIMD with CP2K as a static library
#CP2KMPIFLAG = -Dlibcpgkmpi

#      compile PIMD without CP2K as a static library
CP2KMPIFLAG =

ifdef CP2KMPIFLAG
  CP2KDIRMPI = /home/g3/a184053/QM-Programs/cp2k_new
  CP2KVER = Linux-x86-64-intel-minimal/psmp

  CP2KINCMPI = -I$(CP2KDIRMPI)/obj/$(CP2KVER)/
  CP2KLIBMPI = -L$(CP2KDIRMPI)/lib/$(CP2KVER)/ \
               -lcp2k -L$(CP2KDIRMPI)/lib/$(CP2KVER)/exts/dbcsr/ -ldbcsr
endif

```

Please note that support for CP2K must be enabled separately for the serial and MPI-parallel versions of the code. Do not attempt to link an MPI-parallel CP2K build to the serial PIMD code, or vice versa. To enable compilation with the CP2K API, comment out either the *CP2KFLAG=* or *CP2KMPIFLAG=* line and remove the comment marker from the line containing the actual flag. The *CP2KDIR* (or the corresponding MPI variable) should be set to the top-level directory where CP2K was compiled. The *CP2KVER* (and the corresponding MPI variable) should be set to the *architecture/parallel-version* corresponding to the *libcpgk* library you built.

CP2K input file. Before running the PIMD code, the CP2K input files [55] should be prepared. The CP2K input must correspond to a static calculation of the energy and gradient for the system of interest. The order of atoms in the CP2K input file must match that in the PIMD input files.

Here is an example of the *cp2k.dat* file for a system of 32 water molecules:

```

&FORCE_EVAL
  METHOD QS

```

```

&DFT
  BASIS_SET_FILE_NAME GTH_BASIS_SETS
  BASIS_SET_FILE_NAME BASIS_MOLOPT
  POTENTIAL_FILE_NAME POTENTIAL
  &PRINT
    &MOMENTS
      PERIODIC TRUE
    &END
  &END
  &MGRID
    CUTOFF 300
  &END MGRID
  &QS
    EPS_DEFAULT 1.0E-12
  &END QS
  &SCF
    SCF_GUESS ATOMIC
    EPS_SCF 1.0E-5
    &MIXING
      ALPHA 0.4
    &END
    &PRINT
      &RESTART OFF
      &END RESTART
      &END PRINT
    &END SCF
  &END SCF
  &XC
    &XC_FUNCTIONAL Pade
    &END XC_FUNCTIONAL
  &END XC
&END DFT
&SUBSYS
  &CELL
    ABC 9.8528 9.8528 9.8528
  &END CELL
# 32 H2O (TIP5P,1bar,300K) a = 9.8528
  &COORD
    O      2.280398      9.146539      5.088696
    . . .
    H      8.687134      8.667252      2.448452
  &END COORD
  &KIND H
    BASIS_SET DZVP-GTH
    POTENTIAL GTH-PADE-q1
  &END KIND
  &KIND O
    BASIS_SET DZVP-GTH
    POTENTIAL GTH-PADE-q6
  &END KIND
&END SUBSYS
&END FORCE_EVAL
&GLOBAL
  PROJECT H2O-32
  PRINT_LEVEL MEDIUM
&END GLOBAL

```

The coordinates have been omitted here but can be found in the *examples/water_32/cp2k_pimd_dip* directory. Using this input file, you can test CP2K execution by running:

```
cp2k.psmmp -i cp2k.dat
```

Note that the *RESTART OFF* option under *SCF* is essential to ensure that no unnecessary files are written. This is especially important if the input-suppression keyword `<cp2k_lib_output>` is used in the PIMD input file, as discussed below.

PIMD input file. The following keywords should be set in the *input.dat* file:

```
<ipotential>
CP2KLIB
```

This keyword declares the use of the CP2K code. Another useful option is the ability to control how much output from CP2K is printed for each calculation:

```
<cp2k_lib_output>
1
```

This can be set to -1, 0, or 1. The option -1 suppresses all CP2K output. The option 0 prints only the output of the first bead in the directory 001/. Setting it to 1 causes all beads to write their outputs in the numbered directories created in the main PIMD simulation directory. Note that if CP2K output is suppressed, it is especially important to ensure that CP2K does not create any restart or other data files as part of the single-point calculation described in the previous paragraph. This is because all CP2K processes launched will attempt to write to the same restart files located in the main simulation directory, which can cause file locking and severely reduce performance. Finally, if you want PIMD to output dipole moments, the CP2K calculation must also compute dipole moments. For example, to output dipoles every 10 steps, set:

```
<iprint_dip>
10
```

Execution of PIMD. Execution is performed in the usual way using *pimd.x* or *pimd.mpi.x*. Note that it may be necessary to set the environment variable *OMP_NUM_THREADS* to obtain correct parallelization of CP2K. If you are running the MPI version of the code, for example, *OMP_NUM_THREADS* times the number of MPI processes should be less than or equal to the number of hardware threads available on your system.

4.3.4 DFTB – Internal

IMPORTANT: The user is solely responsible for any issues related to the DFTB+ license. PIMD does not provide any DFTB+ license.

The semiempirical DFTB+ code [54] can be linked directly to the PIMD code via the subroutine *force_libdftb.F*, using calls to the DFTB+ application programming interface (API). Before using this functionality, the DFTB+ API library must be compiled in advance. For example, this can be done by running *make api* in the directory where DFTB+ has been compiled. At present, this feature has been tested only with the non-MPI version of DFTB+. Support for the MPI version of DFTB+ is currently under investigation, but is still experimental and not yet officially supported.

Linking the DFTB+ API To enable linking against the DFTB+ API, the DFTB+ code must be built with API support enabled. In addition, the CMake configuration files provided by DFTB+ must be discoverable by CMake. This is achieved by adding the DFTB+ CMake directories to the *CMAKE_PREFIX_PATH* environment variable.

For a typical installation, these directories are located under *\$HOME/opt/dftb+/lib64/cmake*. An example configuration is shown below:

```
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/dftbplus:$CMAKE_PREFIX_PATH
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/mctc-lib:$CMAKE_PREFIX_PATH
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/mstore:$CMAKE_PREFIX_PATH
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/s-dftd3:$CMAKE_PREFIX_PATH
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/test_drive:$CMAKE_PREFIX_PATH
export CMAKE_PREFIX_PATH=$HOME/opt/dftb+/lib64/cmake/oml-f:$CMAKE_PREFIX_PATH
```

After setting the environment variables, run CMake in the *build* directory of PIMD with the following option enabled:

```
cmake -DDFTB+=ON ..
```

PIMD can then be built in the usual way using CMake, and it will be automatically linked against the DFTB+ library. This option may be combined with other CMake flags, provided that all components are built using the same toolchain.

DFTB+ input file Before running PIMD, the DFTB+ input file must be prepared. The DFTB+ input should correspond to a static calculation of the energy and gradients for the system of interest. The ordering of atoms in the DFTB+ input must exactly match that in the PIMD input files.

The following rules apply to the DFTB+ input file:

- The file name must be *dftb.dat*.
- The geometry must be specified using *GenFormat*.
- The unit of length must be Å.

An example *dftb.dat* file for a water molecule is shown below:

```
Geometry = GenFormat {
3 C
0 H
  1  1  0.000000000000E+00 -0.100000000000E+01  0.000000000000E+00
  2  2  0.000000000000E+00  0.000000000000E+00  0.783064000000E+00
  3  2  0.000000000000E+00  0.000000000000E+00 -0.783064000000E+00
}

Driver = SteepestDescent {
  MaxSteps = 0
}

Hamiltonian = DFTB {
  SCC = Yes
  SCTolerance = 1.0e-5
  MaxSCCIterations = 1000
  Mixer = Broyden {
    MixingParameter = 0.2
  }
  SlaterKosterFiles = {
    O-O = "O-O.skf"
    O-H = "O-H.skf"
    H-O = "O-H.skf"
    H-H = "H-H.skf"
  }
  MaxAngularMomentum = {
    O = "p"
    H = "s"
  }
}
```



```

Charge = 0.0
SpinPolarisation = {}
Filling = Fermi {
    Temperature [Kelvin] = 0.0
}
}

Options = {
    WriteDetailedOut = No
    RestartFrequency = 0
}

Analysis = {
    WriteBandOut = No
}

ParserOptions = {
    ParserVersion = 3
}

```

The required Slater–Koster files (**.skf*), such as *O-O.skf*, *O-H.skf*, and *H-H.skf*, must be present in the execution directory. Alternatively, their location can be specified using the *Prefix* keyword inside the *SlaterKosterFiles* block.

To test the DFTB+ input independently, execute:

```
> dftb+ < dftb.dat
```

When using the MPI version of PIMD, it is essential to include the options *WriteDetailedOut = No*, *RestartFrequency = 0*, and *WriteBandOut = No*, since multiple processes may otherwise attempt to write to the same output files, leading to race conditions and performance degradation.

PIMD input file To use DFTB+ via the internal API, set the following keyword in *input.dat*:

```
<ipotential>
DFTBLIB
```

The printing of DFTB+ output can be controlled using:

```
<dftb_lib_output>
0
```

This keyword may take the values -1, 0, or 1:

- -1: suppress all DFTB+ output.
- 0: print output from a single bead only.
- 1: print output from all beads.

In the serial (non-MPI) version of PIMD, options 0 and 1 both cause output to be written to the directory *001*, since all beads share a single DFTB+ instance. In the MPI version, 0 prints output only for the first bead in *001*, whereas 1 prints output for all beads into their respective numbered directories. Note that in the MPI version, each MPI process runs an independent instance of DFTB+.

Execution of PIMD Execution proceeds in the usual manner using *pimd.x* or *pimd.mpi.x*. It may be necessary to set the environment variable *OMP_NUM_THREADS* appropriately to ensure correct parallelization of the DFTB+ code. In MPI runs, the product of *OMP_NUM_THREADS* and the number of MPI processes should not exceed the number of available hardware threads.

Furthermore, the number of MPI processes must not exceed the number of beads in the simulation. If this condition is violated, PIMD will terminate with an error. In the current implementation, parallelization of DFTB+ is assumed to take place at the OpenMP level, while bead parallelization in PIMD assigns one DFTB+ calculation to each MPI process.

4.3.5 DFTB - External

IMPORTANT: The user is solely responsible for all issues related to the DFTB license. PIMD does not provide a DFTB license.

The semiempirical DFTB code [54] is linked in the subroutine *force_dftb.F* via system calls. The DFTB code must be installed beforehand in order to use this link.

DFTB input file. Before running the PIMD code, the DFTB+ input files [54] should be prepared. The DFTB+ input files should correspond to a static calculation of the energy and gradient for the system of interest. The order of atoms in the DFTB+ input files must match that in the PIMD input files. There are some rules for the format of the DFTB+ input file:

- The file name should be *dftb.dat*.
- “GenFormat” should be used for the specification of geometry.
- The unit should be in Å.

Here is an example of a *dftb.dat* file for a water molecule:

```

Geometry = GenFormat {
3 C
0 H
    1    1    0.000000000000E+00  -0.100000000000E+01    0.000000000000E+00
    2    2    0.000000000000E+00    0.000000000000E+00    0.783064000000E+00
    3    2    0.000000000000E+00    0.000000000000E+00   -0.783064000000E+00
}

Driver = SteepestDescent{
    MaxSteps = 0
}

Hamiltonian = DFTB {
    SCC = Yes
    SCCTolerance = 1.0e-5
    MaxSCCIterations = 1000
    Mixer = Broyden {
        MixingParameter = 0.2
    }
    SlaterKosterFiles = {
        O-O = "O-O.skf"
        O-H = "O-H.skf"
        H-O = "O-H.skf"
        H-H = "H-H.skf"
    }
}

```

```

MaxAngularMomentum = {
  O = "p"
  H = "s"
}
Charge = 0.0
SpinPolarisation = {}
Filling = Fermi {
  Temperature [Kelvin] = 0.0
}
}

Options = {}

ParserOptions = {
  ParserVersion = 3
}

```

The Slater-Koster files (**.skf*), in this case O-O.skf, O-H.skf, and H-H.skf, should be present in the execution directory. Using this input file, one can test the DFTB execution with the following command:

```
> dftb+ < dftb.dat
```

Note that the file “detailed.out” needs to be printed (which is the default option for DFTB+) in order for the code to work. Therefore, the setting

```

Options = {
  WriteDetailedOut = No
}

```

should NOT be used in the DFTB+ input.

PIMD input file. The following keyword should be set in the file *input.dat*.

```

<ipotential>
DFTB

```

This keyword declares the usage of the DFTB+ code.

```

<dftb_command>
'dftb+'

```

This keyword sets the DFTB+ execution command.

Execution of PIMD. The execution is done in the usual way by the command *pimd.x* or *pimd.mpi.x*. When the subroutine *force_dftb.F* is called, the following procedure is undertaken automatically. First, *dftb.dat* in the execution directory is read, then the lines where the geometry is specified are detected and replaced by the current geometry. Then, the DFTB calculation is executed using system calls. Finally, the energy and gradient values are read from the DFTB output, and the run continues.

4.3.6 GAMESS

IMPORTANT: The user is solely responsible for all issues related to the GAMESS license. PIMD does not provide any GAMESS license.

The ab initio GAMESS code [61] is linked in the subroutine *force_gamess.F* via system calls. The GAMESS code must be installed beforehand in order to use this link.

GAMESS input file. Before running the PIMD code, the GAMESS input file [61] must be prepared. The GAMESS input file should correspond to a static calculation of the energy and gradient for the system of interest. The order of atoms in the GAMESS input file must match that in the PIMD input file. There are some rules for the format of the GAMESS input file.

- The file name should be *gamess.dat*.
- There should be no symmetry in the molecule: C1.
- The SCF convergence should be tight enough so that the error in the energy/gradient is small: NCONV=9.
- The geometry lines should start from the line immediately after C1.
- Cartesian coordinates should be used: COORD=UNIQUE.
- Atomic units should be chosen: UNITS=BOHR.
- The gradient calculation should be included: RUNTYPE=GRADIENT.

Here is an example GAMESS input file for a water molecule:

```
$CONTRL SCFTYP=RHF MULT=1 RUNTYP=GRADIENT COORD=UNIQUE UNITS=BOHR $END
$SYSTEM TIMLIM=1 $END
$SCF DIRSCF=.TRUE. NCONV=9 $END
$BASIS GBASIS=N31 NGAUSS=6 NDFUNC=1 NPFUNC=0 $END
$GUESS GUESS=HUCKEL $END
$DATA
TEST H2O MOLECULE
C1
O 8.0 0.0000000000 0.0000000000 0.0000000000
H 1.0 1.4748637392 1.0778200416 0.0000000000
H 1.0 -1.4748637392 1.0778200416 0.0000000000
$END
```

Using this input file, one can test the GAMESS execution:

```
> cp gamess.dat gamess.inp; runrms gamess 00 1
```

PIMD input file. The following keywords should be set in the file *input.dat*:

```
<ipotential>
GAMESS
```

This keyword declares the use of the GAMESS code.

```
<gamess_command>
'runrms gamess 00 1'
```

This keyword sets the command to execute GAMESS.

Execution of PIMD. The execution is done in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_gamess.F* is called, the following procedure is performed automatically: First, *gamess.dat* in the execution directory is read, then the lines containing the geometry are detected and replaced by the current geometry. Next, the GAMESS calculation is executed using system calls. Finally, the energy and gradient values are read from the GAMESS output, and the run continues.

4.3.7 GAUSSIAN

IMPORTANT: The user is solely responsible for all issues related to the GAUSSIAN license. PIMD does not provide any GAUSSIAN license.

The ab initio GAUSSIAN 03 (98,09,16) code [57, 58, 59, 60] is linked in the subroutine *force_g03.F* (*force_g98.F*, *force_g09.F*, *force_g16.F*) via system calls. The GAUSSIAN code must be installed beforehand in order to use this link.

GAUSSIAN input file. Before running the PIMD code, the GAUSSIAN input file [57, 58, 59, 60] must be prepared. The GAUSSIAN input file should correspond to a static calculation of the energy and gradient for the system of interest. There are some rules for the format of the GAUSSIAN input file.

- The file name should be *g03.dat* (*g98.dat*, *g09.dat*, *g16.dat*).
- There should be no symmetry in the molecule: `Nosymm`.
- The SCF convergence should be tight enough so that the error in the energy/gradient is small: `SCF=Tight`.
- A check point file called *chk.g03* (*chk.g98*, *chk.g09*, *chk.g16*) should be produced: `%chk=chk.g03` (`%chk=chk.g98`, `%chk=chk.g09`, `%chk=chk.g16`).
- The geometry lines should start from the second line after the second blank line.
- Cartesian coordinates should be used.
- Atomic units should be chosen: `Units=au`.
- The memory size should be chosen appropriately: `%mem`.
NOTE: This may affect the calculation time.
- The gradient calculation should be included: `Force`.
- It is recommended to restart MO coefficients if possible: `GUESS=TCHECK`.

Here is an example input file for a water molecule called *g03.dat*:

```
%mem=20MB
%chk=chk.g03
#p HF/6-31G* SCF=TIGHT UNITS=AU NOSYMM FORCE GUESS=TCHECK

comment

0 1
8 0.0000000000 0.0000000000 0.0000000000
1 1.4891196813 1.1634278031 0.0000000000
1 -1.4891196813 1.1634278031 0.0000000000
```

NOTE: Do not forget the last blank line, as required by the GAUSSIAN input format. Using this input file, one can test the execution of *g03* (*g98*, *g09*, *g16*) and the *formchk* program [57, 58, 59, 60]:

```
> g03 < g03.dat

> formchk chk.g03

> cat chk.fchk
```

PIMD input file. The following keywords should be set in the file *input.dat*:

```
<ipotential>  
G03
```

```
<ipotential>  
G98
```

```
<ipotential>  
G09
```

```
<ipotential>  
G16
```

These keywords declare the use of g03, g98, g09, and g16 codes.

```
<g03_command>  
'g03'  
'formchk'
```

```
<g98_command>  
'g98'  
'formchk'
```

```
<g09_command>  
'g09'  
'formchk'
```

```
<g16_command>  
'g16'  
'formchk'
```

These keywords set the execution commands for g03, g98, g09, g16, and the *formchk* program.

Execution of PIMD. The execution is done in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_g98.F* (*force_g03.F*, *force_g09.F*, or *force_g16.F*) is called, the following procedure is performed automatically: First, *g98.dat* (*g03.dat*, *g09.dat*, or *g16.dat*) in the execution directory is read, then the lines containing the geometry are detected and replaced by the current geometry. Next, GAUSSIAN is executed using *g98* (*g03*, *g09*, or *g16*) and *formchk* commands via system calls. Finally, the energy and gradient values are read from the GAUSSIAN output, and the run continues.

4.3.8 MOLPRO

IMPORTANT: The user is solely responsible for all issues related to the MOLPRO license. PIMD does not provide any MOLPRO license.

The ab initio MOLPRO code [63] is linked via the subroutine *force_molpro.F* using system calls. The MOLPRO code must be installed beforehand in order to use this link.

MOLPRO input file. Before running the MOLPRO code, the MOLPRO input file [63] must be prepared. The MOLPRO input file should correspond to a static calculation of the energy and gradient for the system of interest. There are several rules regarding the MOLPRO input file format:

- The file name should be *molpro.dat*.
- Cartesian coordinates must be used by setting `geomtyp=xyz`.

- The input must be set up such that a table of potential energy is created after the keyword `<energy>`.
- The input must be set up such that a table of forces is created after the keyword `<grad>`.
- The input must be set up such that a table of dipole moments is created after the keyword `<dipole>`.

Here is an example of a MOLPRO input file for a water molecule called *molpro.dat*:

```
geomtyp=xyz
geometry={
3

O   0.0000000000    0.0000000000   -0.1302052882
H   1.4891244004    0.0000000000    1.0332262019
H  -1.4891244004    0.0000000000    1.0332262019
}
hf
e(1)=energy
force
varsav

table,e                ! print a table with results
save,table.molpro      ! save to file
title,<energy>          ! title for the table
format,(e24.16)        ! explicit format

table,gradx,grady,gradz ! print a table with results
save,table.molpro      ! save to file
title,<grad> 1          ! title for the table
format,(3e24.16)       ! explicit format

table,dmx,dmy,dmz      ! print a table with results
save,table.molpro      ! save to file
title,<dipole>          ! title for the table
format,(3e24.16)       ! explicit format
```

Using this input file, one can test the MOLPRO execution:

```
> molpro < molpro.dat
```

PIMD input file. The following keywords should be set in the file *input.dat*:

```
<ipotential>
MOLPRO
```

This keyword declares the use of the MOLPRO code.

```
<molpro_command>
'molpro'
```

This keyword sets the command used to execute MOLPRO.

Execution of PIMD. The execution is performed in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_molpro.F* is called, the wrapper code *run_molpro.x* is activated (NOTE: *run_molpro.x* should be placed in a directory accessible via PATH). The following procedure is undertaken automatically: First, *molpro.dat* in the execution directory is read, then the lines containing the geometry

are detected and replaced with the current geometry. Next, MOLPRO is executed via system calls. Finally, the energy and gradient values are read from the MOLPRO output, and the run continues.

Additional setup is necessary to perform nonadiabatic dynamics, i.e., surface hopping dynamics and mean field dynamics, with the MOLPRO code. See Section 3.19.1 for details.

4.3.9 MOPAC

IMPORTANT: The user is solely responsible for all issues related to the MOPAC license. PIMD does not provide any MOPAC license.

The semi-empirical MOPAC code [53] is linked via the subroutine *force_mopac.F* using system calls. The MOPAC code must be installed beforehand in order to use this link.

MOPAC input file. Before running the PIMD code, the MOPAC input file [53] must be prepared. The MOPAC input file should correspond to a static calculation of the energy and gradients (negative of the forces) for the system of interest. There are several rules regarding the MOPAC input file format:

- The file name should be *mopac.dat*.
- Perform one SCF cycle then stop: 1SCF.
- Include gradient calculation: GRADIENTS.
- Do not reorient the molecule: NOREOR.
- Do not use molecular symmetry: NOSYM.
- Ensure SCF convergence is tight enough to keep the error of energy/gradient values small: PRECISE.
- Use Cartesian coordinates: XYZ.
- The geometry lines should start from the fourth line.

Here is an example of a MOPAC input file for a water molecule:

```
PM3 1SCF XYZ PRECISE GRADIENTS NOSYM NOREOR GEO-OK
molecule
comment
8  0.0000000000 1  0.0000000000 1  0.0000000000 1
1  0.7804643317 1  0.5703578412 1  0.0000000000 1
1 -0.7804643317 1  0.5703578412 1  0.0000000000 1
```

Using this input file, one can test a MOPAC calculation:

```
> nohup /opt/mopac/mopac/MOPAC2007.out mopac.dat
```

PIMD input file. The following keywords should be set in the file *input.dat*:

```
<ipotential>
MOPAC
```

This keyword declares the use of the MOPAC code.

```
<mopac_command>
'nohup /opt/mopac/mopac/MOPAC2007.out'
```

This keyword sets the command for executing MOPAC.

Execution of PIMD. The execution is performed in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_mopac.F* is called, the following procedure is executed automatically: First, *mopac.dat* in the execution directory is read, then the lines containing the geometry are detected and replaced with the current geometry. Next, MOPAC is executed using system calls. Finally, the energy and gradient values are read from the MOPAC output, and the run continues.

4.3.10 NTCHEM

IMPORTANT: The user is solely responsible for all issues regarding the NTChem license. PIMD does not provide any NTChem license.

The MultiNTChem code is linked internally at the subroutine *force_ntchem.MPI.F*. To make use of the internal link, the MultiNTChem code must be installed together with the PIMD code. Note that the user must legally possess permission to use MultiNTChem. The instructions for the internal link, the setup, and the execution are shown below.

Internal link of NTChem. Follow the instructions below (an example for the H₂O molecule).

1) Compilation.

NTChem and MNTChem are binary codes. PIMD should be linked to the library of MNTChem, either without OpenMP (*libmntchem_mpi.a*) or with OpenMP (*libmntchem_mpiomp.a*).

In the makefile, set `NTCHEM = -Dntchem` for the former case, `NTCHEM = -Dntchem -Domp` for the latter case.

For Intel computers, sometimes one needs the following for both compilation and execution:

```
> module unload mpt; module load intel/cur intel-mpi/cur
```

2) Prepare INPUT - the NTCHEM input file.

To calculate the dipole moments, set:

```
&prop
Dipole_TxtOut=T
/
```

3) INPUT.MNTChem - the MNTCHEM input file.

Be sure to set the following values correctly: `NCorePerIO = n` and `NCorePerReplica = n`, where *n* is the number of processors (`nprocs`) divided by the number of beads (`nbead`). Also set `DoProp = T`.

4) INPUT.xyz - the MNTCHEM/PIMD interface input file.

```
3
O  0.000  0.000  0.000  8.0  8.0
H  1.489  1.163  0.000  1.0  1.0
H -1.489  1.163  0.000  1.0  1.0
```

5) To secure sufficient memory space, sometimes one needs the following on execution:

```
> ulimit -s unlimited; export OMP_STACKSIZE=1G
```

Setups of NTCHEM. In the PIMD input file, *input.dat*, set the following keywords:

```
<ipotential>
NTCHEM
```

This keyword sets the usage of NTChem.

Execution of PIMD. The execution is performed in the usual way using the *pimd.mpi.x* command.

4.3.11 ORCA

IMPORTANT: The user is solely responsible for all issues related to the ORCA license. PIMD does not provide any ORCA license.

The ab initio ORCA code [65] is linked to the PIMD code via system calls through the subroutine *force_orca.F*. The ORCA program must be installed in advance in order to use this interface.

ORCA input file. Before running PIMD with ORCA, an ORCA input file [65] must be prepared. The ORCA input file should correspond to a static calculation of the total energy and nuclear gradients for the system of interest. The order of atoms in the ORCA input file must exactly match that of the PIMD input files.

The following rules apply to the ORCA input file format:

- The file name must be *orca.dat*.
- The geometry specification must start on the line immediately following **xyz*.
- Cartesian coordinates must be used.
- Atomic units must be selected by specifying *Bohrs*.
- Gradient calculation must be enabled using *ENGRAD*.

An example ORCA input file for a water molecule is shown below:

```
! HF def2-SV(P) ENGRAD Bohrs
%elprop
  Dipole true
end
*xyz 0 1
O 0.0 0.0 0.0
H 0.0 1.0 0.2
H 0.0 0.0 1.1
*
```

Using this input file, the ORCA installation can be tested by running:

```
> cp orca.dat orca_test.inp; orca orca_test.inp
```

NOTE: The dipole moment is currently not supported in the PIMD-ORCA interface.

PIMD input file. The following keywords must be set in the file *input.dat* to use ORCA:

```
<ipotential>
ORCA
```

This keyword activates the ORCA interface.

```
<orca_command>
'orca'
```

This keyword specifies the command used to execute ORCA.

Execution of PIMD. The execution is performed in the usual way using *pimd.x* or *pimd.mpi.x*. When the subroutine *force_orca.F* is invoked, the following procedure is carried out automatically: First, the file *orca.dat* in the execution directory is read. The geometry section is detected and replaced with the current atomic coordinates provided by PIMD. Next, the ORCA calculation is launched via a system call. Finally, the total energy and nuclear gradients are extracted from the ORCA output files, and the PIMD simulation continues.

4.3.12 PHASE/0

IMPORTANT: The user is solely responsible for any issues related to the PHASE/0 license. PIMD does not provide any PHASE/0 license.

The electronic structure PHASE/0 code [56] is directly linked to the PIMD code via the subroutine *force_phase0.MPI.F* by calling the PHASE/0 libraries, *libphase.a* and *libesm.a*. The PHASE/0 libraries must be compiled beforehand to use this link. This can be done by invoking *make libphase.a libesm.a* in the directory where PHASE/0 was compiled (*src_phase* or *src_phase-3d*), for example. Please refer to the PHASE/0 documentation on compilation for further instructions.

Currently, the library version of PHASE/0 does not support stress tensor calculations. Therefore, it cannot be used for BOXOPT calculations or NPT simulations.

Linking the PHASE/0 libraries To link to PHASE/0, a few changes are needed in the PIMD makefile around the following lines.

```
#-----
#      Usage of PHASE0
#-----

#      compile PIMD with PHASE/0
PHASE0 = -Dphase0

ifdef PHASE0
LIBPHASE0 = -L../lib/phase0 -lphase -lesm #-L. -lfftw3
LIBMKL = -L../lib -L$(MKLR00T)/lib/intel64 -L$(HOME)/mkl \
-lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -lmkl_intel_lp64 \
-lmkl_sequential -lmkl_cdft_core -lfftw3x_cdft_lp64 -lmkl_core -lm -ldl
endif
```

PHASE/0 input file. Before running the PIMD code, prepare the PHASE/0 input files [56]. The PHASE/0 input files must correspond to a static calculation of the energy and gradient for the system of interest. The order of atoms in the PHASE/0 input files must match the order in the PIMD input files.

Here is an example for ice. The file *file_names.data* contains the following

```
&fnames
F_POT(1) = '../../../../../pp/H_ggapbe_paw_nc_01m.pp'
F_POT(1) = '../../../../../pp/O_ggapbe_paw_us_02m.pp'
/
```

The file *nfinp.data* contains the following:

```
Control{
  cpumax = 5 day
  condition = initial
}

accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 225.0 rydberg
  ksampling{
    method = monk ! {mesh|file|directin|gamma}
    mesh{ nx = 2, ny = 2, nz = 2 }
    kshift{ k1 = 0.5, k2 = 0.5, k3 = 0.0 }
  }
}

structure{
  unit_cell{
    a_vector = 14.36865687014571 0.0 0.0
    b_vector = -7.184328435072851 12.443621867807988 0.0
    c_vector = 0.0 0.0 13.498240952339659
  }
  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      #units angstrom
      #tag element rx ry rz
H 2.5154346140648998 0 1.403077739736
H -1.2577173070324492 2.1784302773389084 1.403077739736
H 2.5440658429675516 4.406451297820341 1.403077739736
H 5.0881316859351005 0 4.9745587397360005
H -2.544065842967549 4.406451297820341 4.9745587397360005
H 1.2577173070324505 2.1784302773389084 4.9745587397360005
H 3.4394504051060997 0 0.076043973452
H -1.7197252025530492 2.9786514258785615 0.076043973452
H 2.0820579474469505 3.6062301492806883 0.076043973452
H 4.164115894893899 0 3.647524973452
H -2.0820579474469487 3.6062301492806883 3.647524973452
H 1.7197252025530505 2.9786514258785615 3.647524973452
H 1.710372816004051 5.778529901873124 6.991866924814
H -2.0577569495521484 5.176842932824522 6.991866924814
H 0.3473841335481005 2.2143903156208533 6.991866924814
```

```

H 5.51215596600405 0.806351673286126 6.991866924814
H 1.7440262004478506 1.4080386423347273 6.991866924814
H 4.149167283548101 4.3704912595383965 6.991866924814
H 2.09141033399595 0.806351673286126 3.420385924814
H 5.8595400995521505 1.4080386423347273 3.420385924814
H 3.454399016451901 4.3704912595383965 3.420385924814
H -1.7103728160040483 5.778529901873124 3.420385924814
H 2.057756949552151 5.176842932824522 3.420385924814
H -0.3473841335480995 2.2143903156208533 3.420385924814
O 2.4836441033646 0 0.39756297899599996
O -1.2418220516822993 2.1508988874731676 0.39756297899599996
O 2.5599610983177015 4.433982687686083 0.39756297899599996
O 5.1199221966354 0 3.969043978996
O -2.559961098317699 4.433982687686083 3.969043978996
O 1.2418220516823006 2.1508988874731676 3.969043978996
O 5.0229539156115 0 6.63991890115
O -2.5114769578057485 4.350005692958076 6.63991890115
O 1.2903061921942507 2.2348758822011736 6.63991890115
O 2.5806123843885 0 3.06843790115
O -1.2903061921942494 2.2348758822011736 3.06843790115
O 2.511476957805751 4.350005692958076 3.06843790115
    }
  }
}

```

The atom arrangement is required by PHASE/0 for allocating arrays, among other purposes, but the actual coordinate values are passed from PIMD. Therefore, the atoms must be aligned, and the atomic species must be the same, but any coordinate and attribute values are acceptable.

PIMD input file. The following keywords should be set in the file *input.dat*.

```

<ipotential>
PHASE0

```

This keyword declares the use of the PHASE/0 code. Another useful option is to selectively print the output of a given PHASE/0 calculation.

```

<phase0_proc>
ne nk ng

```

The three integers, **ne nk ng**, correspond to the number of band parallels, k-point parallels, and G-point parallels, respectively. In the case of a two-dimensional version, 1 should be specified as the value corresponding to **ng**. Each value should be chosen so that the MPI parallel number (**np_force**) allocated to each replica by PIMD is equal to the product **ne nk ng**.

Execution of PIMD. The execution is done in the usual way using the *pimd.mpi.x* command.

4.3.13 QUANTUM ESPRESSO

IMPORTANT: The user is solely responsible for all issues related to the Quantum ESPRESSO (QE) license. PIMD does not provide any QE license.

The QE code is internally linked through the subroutine *force_qe-MPI.F*. To use this internal link, QE must be installed together with the PIMD code. Note that the user must legally possess the source code of QE version 6.2.1 or 6.3 and have permission to use and modify it. Instructions for the internal link, setup, and execution are provided below.

Setup using cmake. Follow the directions described below. Currently, only QE versions 6.2.1 and 6.3 are supported.

- **Step 0: Obtain the QE code (OPTIONAL).** This step is not necessary if you have internet access and the `wget` command available during the compilation process. If you need to download QE in advance, do not unpack the archive; keep the zip file and add the “-QEFILES=../lib/qe/q-e-qe-6.2.1.zip” flag to the `cmake` command in Step 1, where the part after the equal sign is the path to the zip file containing the source code.
- **Step 1: Setup the compilation with cmake** First, create the directory where the PIMD code will be compiled.

```
> mkdir build
> cd build
```

Then run `cmake` with at least the following flags:

```
> cmake -DMKLUSE=ON -DQE=ON -DQEVERSION=6.3 ..
```

Note that even if you have downloaded the QE code in advance, you still need to provide the correct version number to `cmake`, since it is required to select the appropriate patch file. You may also add other options, such as enabling compilation with AENET:

```
> cmake -DMKLUSE=ON -DQE=ON -DQEVERSION=6.3 -DAENET=ON ..
```

- **Step 2: Run make** Run the following command to compile both PIMD and QE:

```
> make -j 10
```

Here we compile with 10 parallel jobs; note that QE itself is compiled with a single job, so this step can be somewhat slow. Once the process finishes, PIMD will be compiled and ready to run using the binaries in the build directory.

Setups of QE. In addition to the PIMD input files (*input.dat*, *input_default.dat*, *structure.dat*), the QE input file (*qe.dat*) is also required. Refer to the QE manual for instructions on preparing these input files at https://www.quantum-espresso.org/Doc/INPUT_PW.html. In the PIMD input file, *input.dat*, set the following keywords:

```
<ipotential>
QE
```

This keyword enables the use of QE.

```
<qe_input_file_name>
si.md8.in
```

This keyword specifies the QE input file name.

```
<qe_output>
1 10
```

This keyword controls the printing of QE output. The first number should be set to ‘0’ or ‘1’ (integer). If it is set to ‘1’, QE output is printed for the first bead only; if it is set to ‘0’, QE output is printed for all beads. The second number specifies the print interval; in this example, output is printed every 10 steps.

It is recommended not to use “nosym=.false.” in *qe.dat*.

Execution of PIMD. Execution is performed in the usual way by running the *pimd.mpi.x* command.

Internal link with QE (Old method - Expert option). Follow the directions described below.

- **Step 1: Prepare the PIMD makefile.** Make a copy of the source directory named “compile”, where the PIMD code will be compiled.

```
> cp -r ~/pimd/source ~/pimd/compile
```

Activate the option “QE = -Dqe” in the PIMD makefile to link QE during compilation.

```
> vi ~/pimd/compile/makefile
```

Note that the libraries used must be consistent with those used to compile QE (see below).

- **Step 2: Prepare PIMD modules for QE** In the *~/pimd/compile* directory, generate the modules needed to compile the modified QE.

```
> make common_variables.o
```

- **Step 3: Download the QE source code.** Go to the QE library directory (*~/pimd/lib/qe*). Download and unzip the source code for QE version 6.2.1, *q-e-qe-6.2.1*.

```
> cd ~/pimd/lib/qe
> wget https://github.com/QEF/q-e/archive/qe-6.2.1.zip
> unzip q-e-qe-6.2.1.zip
```

Make sure the list of files is as follows:

```
> ls
README                               make_patch_qe.sh*           qe-6.2.1.zip
apply_patch_qe.sh*                  q-e-qe-6.2.1/              qe-6.2.1_to_pimdlb.patch
```

Also ensure that an FFTW library (fftw3 or equivalent) required by QE is available (see below).

- **Step 4: Apply a patch to the QE code.** In the QE library directory, apply the patch file as follows:

```
> patch -p0 < qe-6.2.1_to_pimdlb.patch
```

The QE source code will be modified to link with PIMD.

- **Step 5: Prepare the QE makefile.** Configure QE in the *q-e-qe-6.2.1* directory.

```
> cd ~/pimd/lib/qe/q-e-qe-6.2.1
> ./configure
```

- **Step 6: Compile QE** Go to the QE directory and compile it.

```
> cd ~/pimd/lib/qe/q-e-qe-6.2.1
> make pw
```

- **Step 7: Compile and link PIMD and QE codes.** Go to the PIMD compile directory and edit the makefile as needed:

```
> vi ~/pimd/compile/makefile
```

After editing the makefile, compile the code.

4.3.14 TURBOMOLE

IMPORTANT: The user is solely responsible for all issues related to the TURBOMOLE license. PIMD does not provide any TURBOMOLE license.

The ab initio TURBOMOLE code [62] is linked to the subroutine *force_turbo.F* via system calls. The TURBOMOLE code must be installed beforehand in order to make use of this link.

TURBOMOLE input file. Before running the PIMD code, the TURBOMOLE input files [62] should be prepared. The TURBOMOLE input files should correspond to a static calculation of the energy and gradient for the system of interest. The order of atoms in the TURBOMOLE input files must match that of the PIMD input files. In the TURBOMOLE code, the energy and the gradient are computed by two consecutive commands (*dscf,grad*), (*ridft,rdgrad*), (*dscf,mpgrad*), (*dscf,rimp2*), or (*dscf,ricc2*) [62]. Prepare the *control* file properly, as well as all other necessary files (*coord*, *mos*, *basis*, *auxbasis*, etc.). All of these can be obtained using the *define* command of the TURBOMOLE code [62]. Here is an example of the TURBOMOLE *control* file for a water molecule:

```
> cat control

$title
water molecule
$operating system unix
$symmetry c1
$coord      file=coord
$user-defined bonds      file=coord
$atoms
o 1 \
    basis =o def-SV(P)
h 2-3 \
    basis =h def-SV(P)
$basis      file=basis
$rundimensions
    dim(fock,dens)=211
    natoms=3
    nshell=10
    nbfc(AO)=19
    nbfc(AO)=18
    dim(trafo[SAO<-->AO/CAO])=21
    rhfshells=1
$scfmo      file=mos
$closed shells
a          1-5          ( 2 )
$scfiterlimit      30
$scfconv          8
$thize          0.10000000E-04
$thime          5
$scfdamp      start=0.300 step=0.050 min=0.100
$scfdump
$scfintunit
unit=30      size=0      file=twoint
$scfdiis
$scforbitalshift      automatic=.1
$drvopt
    cartesian on
    basis off
    global off
```



```

    hessian    on
    dipole     on
    nuclear polarizability
$interconversion off
    qconv=1.d-7
    maxiter=25
$optimize
    internal   off
    cartesian  on
    global     off
    basis      off    logarithm
$coordinateupdate
    dqmax=0.3
    interpolate on
    statistics  5
$forceupdate
    ahlrichs numgeo=0 mingeo=3 maxgeo=4 modus=<g|dq> dynamic fail=0.3
    threig=0.005 reseig=0.005 thrbig=3.0 scale=1.00 damping=0.0
$forceinit on
    diag=default
$energy    file=energy
$grad      file=gradient
$forceapprox file=forceapprox
$lock off
$end

```

There are some rules regarding the format of TURBOMOLE *control* files.

- There should be no symmetry in the molecule.
- The SCF convergence should be tight enough so that the error in the energy and gradient values is small.

Let us give an example of the (**dscf,grad**) combination. One can test the TURBOMOLE execution of the **dscf** command with the *control* file as well as all other input files.

```
> dscf
```

If it works correctly, copy the *control* file to the *control.1* file. Then test the TURBOMOLE execution of the **grad** command using the *control* file as well as all other input files.

```
> grad
```

If it works correctly, copy the *control* file to the *control.2* file.

PIMD input file. The following keywords should be set in the file *input.dat*.

```
<ipotential>
TURBOMOLE
```

This keyword declares the use of the TURBOMOLE code.

```
<turbo_command>
    dscf    control.1
    grad    control.2
```

This keyword sets the TURBOMOLE commands together with the control files. In the present case, the TURBOMOLE commands consist of the `dscf` and `grad` commands, and the names of the control files are *control.1* and *control.2*, respectively.

```
<turbo_guess>  
PREVIOUS
```

This keyword specifies the option for the initial guess of molecular orbital coefficients. In the present case, those from the previous step are reused.

Execution of PIMD. The execution is carried out in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_turbo.F* is called, the following procedure is performed automatically. First, the TURBOMOLE *coord* file is created using the current geometry. Then, the TURBOMOLE calculation is executed using system calls. Finally, the energy and gradient values are read from the TURBOMOLE output, and the run is continued. NOTE: When the subroutine *force_turbo.F* is called, this keyword is searched primarily in the *input.dat* file, secondarily in the *turbo.dat* file, and finally in the *input_default.dat* file.

4.3.15 VASP6 Interface

IMPORTANT: The user is solely responsible for all issues related to the VASP6 license. PIMD does not provide a VASP license.

Linking the VASP6 code Before compiling PIMD, you must compile a patched version of VASP. At present, the patch we provide supports VASP version 6.4.0. In the directory *lib/vasp6*, you can find the patch and a script to apply it, assuming you unpack the VASP6 source code in that directory. We do not provide support for compiling VASP; please refer to the official VASP documentation or other online resources for instructions.

Once VASP6 has been compiled, you can proceed to compile the PIMD program. When using CMake to configure the build in a separate directory, you need to set the following flags to enable the interface between PIMD and VASP:

```
-DVASP6=ON
```

The first flag enables compilation of the VASP interface. You must also specify the directory where VASP was compiled:

```
-DVASP6_DIR=../lib/vasp6/vasp.6.4.0.modified/build/std/
```

This flag sets the directory containing the VASP build, since PIMD needs to access VASP *.mod* files and the libraries required for linking. Here, we assume that CMake is run in the *build* directory under the main PIMD directory.

VASP6 files. Please refer to the VASP manual for details on how to set up the files mentioned here. The current interface automatically copies the files *INCAR*, *KPOINTS*, *POSCAR*, and *POTCAR* from the main calculation directory into a set of numbered directories, one for each bead. The standard VASP output files for each bead calculation are also written in these directories. Finally, a file named *VASPSTDOUT* in each bead directory contains the output that VASP would normally write to standard output.

PIMD input file. The following keywords should be set in the file *input.dat*.

```
<ipotential>  
VASP6
```

This keyword declares the use of the VASP6 interface.

4.3.16 VASP - General Words of Caution

IMPORTANT NOTE: The PIMD developers cannot provide specific support for VASP calculations. The advice given here is based on their experience and may not be sufficient in all cases.

As with all large-scale calculations, we recommend performing a test run, either as a single-point energy calculation or as a structural relaxation. A non-exhaustive list of things to check includes:

- Ensure that VASP is compiled correctly and that the executable is in your PATH.
- Ensure that the VASP input files are set up correctly.
- Ensure that the VASP output files are parsed correctly by the PIMD code.

One problem that can occur is that the order of ions in the POSCAR file does not match the order of ions in the POTCAR file. VASP may still be able to run the calculation, but the resulting relaxed structures can deviate significantly from what is expected. It is recommended to set *LCHARG* = *FALSE*. to prevent VASP from writing charge-density output, since trajectory output is better handled through PIMD's internal trajectory output functions.

4.3.17 VASP5

IMPORTANT: The user is solely responsible for any issues related to the VASP license. PIMD does not provide any VASP license.

The VASP code is linked externally to the subroutine *force_vasp5.F* of the PIMD code. To make use of the external link, the VASP code must be installed beforehand.

The VASP code is linked internally to the subroutine *force_vasp_MPI.F*. To make use of the internal link, the VASP code must be installed together with the PIMD code. Note that the user must legally possess the source code of VASP version 5.3.5, as well as permission to use and modify it. The directions for the internal link, the setup, and execution are shown below.

Internal link of VASP. Follow the directions below:

- **Step 1: Copy the source code.** Copy the source code of VASP version 5.3.5 and its library to the lib directory of PIMD.

```
> cp -r vasp.5.3.5 lib/vasp
> cp -r vasp.5.lib lib/vasp
```

- **Step 2: Compile the VASP library.** Starting from the main directory, go to the vasp.5.lib directory.

```
> cd lib/vasp/vasp.5.lib
```

Then, edit the Makefile appropriately with respect to the Fortran compiler, compilation options, and the LAPACK/BLAS libraries, depending on the machine.

```
> cp makefile.*** Makefile
> vi Makefile
```

Compile the VASP library.

```
> make
```

If successful, a new file called *libdmy.a* will be created. Copy it to the lib directory.

```
> cp libdmy.a ../../../../lib
```

Return to the main directory.

```
> cd ../../../../lib
```

- **Step 3: Compile the modified VASP code.** Go to the vasp directory.

```
> cd lib/vasp
```

Apply the patch to modify the VASP source:

```
> patch -u -p1 -d vasp.5.3.5/ < vasp.5.3.5_to_pimdlb.patch
```

If successful, the following output will appear:

```
patching file acfdt.F
patching file aedens.F
...
patching file xml.F
patching file zgemmtest.F
```

without any error message. Go to the vasp.5.3.5 directory.

```
> cd vasp.5.3.5
```

Then, edit the Makefile appropriately with respect to the Fortran compiler, compilation options, and the LAPACK/BLAS libraries, depending on the machine.

```
> vi Makefile
```

Here, do not overwrite the Makefile with the original makefile.***, because the Makefile is created by the patch. Compile the modified VASP code.

```
> make
```

If successful, a new file called *libvasp.a* will be created. Copy it to the lib directory.

```
> cp libvasp.a ../../../../lib
```

Return to the main directory.

```
> cd ../../../../
```

- **Step 4: Compile the PIMD code.** To compile the PIMD code, edit the *makefile* to activate the *-Dvasp* option.

```
> cp -r source compile
> cd compile
> vi makefile
> make
```

If successful, this generates *pimd.mpi.x*, an executable of PIMD internally linked to VASP.

Setups of VASP. In addition to the PIMD input files (*input.dat*, *input_default.dat*, *structure.dat*), the VASP input files are necessary (*INCAR*, *POSCAR*, *POTCAR*, *KPOINTS*). See the VASP manual for the preparation of these input files. In the *INCAR* file, the following keywords must always be set:

```
IBRION = 1
NSW = 100
ISIF = 3
ISYM = 0
```

On the other hand, in *input.dat*, the following keyword sets the use of VASP:

```
<ipotential>
VASP5
```

for the external link, and

```
<ipotential>
VASP
```

for the internal link.

```
<vasp_output>
0 10
```

This keyword controls the VASP output. The first number should be set to ‘0’ or ‘1’ (integer). The VASP output is printed for the first bead only if ‘1’, and for all beads if ‘0’. The second number is the print interval, in the present case, every 10 steps.

```
<vasp_reuse_wavefunction>
1
```

This keyword controls the initial guess for the Kohn–Sham orbitals. The Kohn–Sham orbitals are built from scratch if ‘0’, and taken from the previous step if ‘1’.

Execution of PIMD. PIMD can be executed using the command *pimd.x* or *pimd.mpi.x*.

4.4 Machine learning potentials

The PIMD code can also be combined with several types of machine learning interatomic potentials. These potentials enable efficient large-scale simulations with near *ab initio* accuracy, once properly trained. Currently, the following machine learning potentials are supported:

- AENET [84] (artificial neural network potential).
- ASE calculator interface (generic Python/ASE calculator bridge).
- MACE [89] (equivariant message-passing neural network potential).
- MTP [87, 88] (moment tensor potential).
- N2P2 [90] (high-dimensional neural network potential).

Each machine learning potential has its own requirements for training data, parameter files, and execution workflow. Details of the setup, input files, and execution procedures for each potential are described in the corresponding subsections.

4.4.1 AENET

IMPORTANT: The user is solely responsible for all issues related to the AENET license. PIMD does not provide any AENET license.

The artificial neural network code AENET [84] is linked via the subroutines *force_aenet.F* and *force_aenet.MPI.F*. To use this link, the AENET code must be installed properly.

Internal link of AENET. Follow the directions below.

- **Step 1: Copy the source code.** First, copy the source code of AENET version 2.0.3 and its libraries to the PIMD library directory.

```
> cp -r aenet-2.0.3/src lib/aenet
> cp -r aenet-2.0.3/lib lib/aenet
```

- **Step 2: Compile the AENET library.** Starting from the main directory, move to the L-BFGS library directory.

```
> cd lib/aenet/lib
```

Edit the Makefile as appropriate for your Fortran compiler.

```
> vi Makefile
```

Compile the AENET library.

```
> make
```

If successful, a file named *liblbfgsb.a* will be created. Copy it to the *lib* directory.

```
> cp liblbfgsb.a ../../
```

Return to the main directory.

```
> cd ../../../../
```

- **Step 3: Compile the modified AENET code.** Move to the AENET directory.

```
> cd lib/aenet
```

Create a copy of the source directory and apply the patch.

```
> cp -r src src_modified
> patch -u -p1 -d src_modified < src.patch
```

Move to the modified source directory.

```
> cd src_modified
```

Edit the Makefile options according to your system (Fortran compiler, LAPACK/BLAS libraries, etc.).

```
> vi makefiles/Makefile.options
```

Compile the modified AENET code.

```
> make -f makefiles/Makefile.options lib
```

If successful, a file named *libaenet.a* will be created. Copy it to the *lib* directory.

```
> cp libaenet.a ../../
```

Return to the main directory.

```
> cd ../../../../
```

- **Step 4: Compile the PIMD code.** Edit the *makefile* to activate the options `-Daenet2 -Dnlist`.

```
> cp -r source compile
> cd compile
> vi makefile
> make
```

If successful, this procedure generates *pimd.mpi.x*, an executable of PIMD internally linked to AENET.

PIMD input file to create xsf files. Use the PIMD, PIHMC, or REHMC method to create a set of xsf files. Set the following keyword in *input.dat*:

```
<iprint_xsf_aenet>
10
```

This keyword specifies the interval for printing AENET xsf files, in this case, every 10 steps.

If the PIHMC or REHMC method is used, the following keyword may also be set:

```
<ioption_xsf_aenet>
1
```

With this option, xsf files are printed for both accepted and rejected structures. The xsf files are created in the directory named *structures*.

PIMD input file to generate/train using xsf files. Use the PIMD, PIHMC, or REHMC method to generate training data and train AENET networks using the xsf files. Set the following keywords in *input.dat*:

```
<istep_train_aenet>
100
```

This keyword specifies the interval for generating and training AENET. In this example, training is performed every 100 steps.

```
<lstep_train_aenet>
3000
```

This keyword sets the final step for generating and training AENET. Here, training stops at step 3000.

```
<minxsf_train_aenet>
300
```

This keyword specifies the minimum number of xsf files required to start training. Training is skipped until at least 300 xsf files are found in the *structures* directory.

```

<generate_aenet>
OUTPUT SiO2.train
TYPES
2
Si -5.420474 | eV
O -4.939198 | eV
SETUPS
Si Si.fingerprint.stp
O O.fingerprint.stp

```

This block specifies the input for generating the AENET training set.

```

<train_aenet>
TRAININGSET SiO2.train
TESTPERCENT 10
ITERATIONS 5000
MAXENERGY 1.0
TIMING
METHOD
bfgs
NETWORKS
! atom   network           hidden
! types  file-name         layers nodes:activation
Si      Si.15-15.ann      2      15:tanh 15:tanh
O       O.15-15.ann      2      15:tanh 15:tanh

```

This block specifies the training parameters for AENET.

```

<dir_train_aenet>
"trained_networks"

```

This keyword specifies the scratch directory for trained networks. After training is completed, the final network files are copied to the execution directory.

PIMD input file to run simulations (predict). Set the following keywords in *input.dat*:

```

<ipotential>
AENET

```

This keyword specifies the use of the AENET potential.

```

<ntype_aenet>
2
Si Si.15-15.ann
O O.15-15.ann

```

This keyword defines the number of atomic types and the corresponding AENET network for each type.

PIMD input file to run simulations (predict/generate/train). In the PIHMC or REHMC method with a dual potential, the generate/train process can be performed on the fly when AENET is used as the low-level potential. For example:

```

<method>
PIHMC

```



```
<ipotential>
DUAL
```

```
<dual_potential>
VASP AENET
```

The following keywords are useful:

```
<istep_hmc>
8
```

```
<istep_max_hmc>
128
```

```
<istep_adjust_hmc>
100
```

```
<istep_mul_hmc>
2
```

```
<ratio_min_hmc>
0.05
```

```
<ratio_max_hmc>
0.20
```

These keywords define a variable number of MD steps per Metropolis step. In this example, the number of MD steps varies from 8 to 128 with a geometric progression of ratio 2. Every 100 Metropolis steps, the number of MD steps is increased if the acceptance ratio is below 0.05, and decreased if it is above 0.20.

Execution of PIMD. The execution is performed in the usual way using the *pimd.x* or *pimd.mpi.x* command. When the subroutine *force_aenet.F* is called, the input file *predict.in* is automatically generated.

4.4.2 MTP

IMPORTANT: The user is solely responsible for all issues related to the MLIP code [87, 88], which is distributed via the website

<https://mlip.skoltech.ru>

PIMD does not provide any license for the MLIP code.

In the PIMD code, the subroutines *force_mtp.F* and *force_mtp_MPI.F* have been reimplemented in Fortran 90 based on the original MLIP code written in C++.

The potential data file *pot.mtp*, in the native MLIP format, must be present in the execution directory. The file *pot.mtp* should be generated by the MTP training procedure using the MLIP code. Manually modifying the *pot.mtp* file is strongly discouraged, as it may lead to unexpected errors when the PIMD code reads the potential parameters.

To use the MTP potential, the following keyword must be specified in the file *input.dat*:

```
<ipotential>
MTP
```

This keyword activates the MTP potential in the PIMD code.

The execution is performed in the usual manner using either *pimd.x* or *pimd.mpi.x*.

As an example, a set of input files for a path integral simulation of hydrogen in bcc iron using the MTP potential is provided in the directory

examples/FeH/mtp_pimd_npt

4.4.3 N2P2

IMPORTANT: The user is solely responsible for all issues related to the N2P2 license. PIMD does not provide any N2P2 license.

The high-dimensional neural network N2P2 code [90] is linked through the subroutines *force_n2p2_MPI.F* and *predict_n2p2_MPI.F*. The N2P2 code must be properly installed in order to use this interface. Currently, only version 2.2.0 is supported; support for version 2.3.0 is planned for a future release. This means that, at present, only second-generation NNPs generated by N2P2 are supported in PIMD.

Internal link of N2P2. Follow the directions below.

- **Step 1: Obtain the N2P2 source code and apply the patch.** Go to the N2P2 directory.

```
> cd lib/n2p2/
```

Run the bash script to download the source code from GitHub and apply the patch.

```
> ./getandapply_patch.sh
```

This script downloads the source files for N2P2 version 2.2.0 and applies the patch required to interface with PIMD.

- **Step 2: Compile the modified N2P2 code.** Go to the directory containing the modified N2P2 source code.

```
> cd n2p2-2.2.0.modified
```

Compile the modified N2P2 code. Note that when using the Intel compiler, the `-ipo` option must be removed from the `PROJECT_CFLAGS` variable in `./src/makefile.intel` to avoid runtime errors. The following commands overwrite the `PROJECT_CFLAGS` variable by passing arguments directly to the make command.

```
> make -C ./src/libnnp COMP=intel PROJECT_CFLAGS="-O3 -xHost -std=c++11"
> make -C ./src/libnnptrain COMP=intel PROJECT_CFLAGS="-O3 -xHost -std=c++11"
> make -C ./src/libnnpif COMP=intel PROJECT_CFLAGS="-O3 -xHost -std=c++11"
```

If successful, new files named *libnnp.a*, *libnnptrain.a*, and *libnnpif.a* will be created. Copy them to the lib directory.

```
> cp ./lib/libnnp.a ../../
> cp ./lib/libnnptrain.a ../../
> cp ./lib/libnnpif.a ../../
```

Return to the main directory.

```
> cd ../../
```

- **Step 3: Compile the PIMD code.**

CMake method: Create a build directory and invoke CMake with the N2P2 flag enabled.

```

> mkdir build
> cd build
> cmake -DN2P2=ON ..
> make

```

As always, it is important to ensure that the same toolchain is used to compile both N2P2 and PIMD. Therefore, make sure to set the correct compilers consistently when building N2P2 and PIMD.

Makefile method: To compile the PIMD code using the makefile, edit the file *makefile* to activate the `-Dn2p2` option.

```

> cp -r source compile
> cd compile
> vi makefile
> make

```

If successful, this process generates *pimd.mpi.x*, the PIMD executable internally linked to N2P2.

N2P2 input files. In addition to the PIMD input files (*input.dat*, *input_default.dat*, *structure.dat*), the N2P2 input files are also required (*input.nn*, *scaling.data*, *weights.***.data*). See the N2P2 manual for instructions on preparing these input files.

PIMD input file to create xsf files. Use the PIMD, PIHMC, or REHMC method to create a set of xsf files. The following keyword should be set in the file *input.dat*.

```

<iprint_xsf_n2p2>
10

```

This keyword specifies the step interval for printing N2P2 xsf files; in this case, every “10” steps.

If the PIHMC or REHMC method is selected, one can also set

```

<ioption_xsf_n2p2>
1

```

This keyword controls the printing option for xsf files. In this case, “1” indicates that xsf files are printed for both accepted and rejected structures. The xsf files are created in the directory named “structures”.

PIMD input file to generate/train using xsf files. Use the PIMD, PIHMC, or REHMC method to generate input data and train networks using a set of xsf files. The following keywords should be set in the file *input.dat*.

```

<istep_train_n2p2>
100

```

This keyword specifies the step interval for generating and training N2P2. In this case, “100” means that generation and training are performed every 100 steps.

```

<lstep_train_n2p2>
3000

```

This keyword sets the final step for N2P2 generation and training. In this case, “3000” means that generation and training stop at step 3000.

```

<minxsf_train_n2p2>
300

```

This keyword specifies the minimum number of xsf files required to train N2P2. In this case, “300” means that the generation and training process is skipped until 300 xsf files are found in the “structures” directory.

PIMD input file to run simulations (predict). The following keywords should be set in the file *input.dat*.

```
<ipotential>
N2P2
```

This keyword declares the use of the N2P2 code.

PIMD input file to run simulations (predict/generate/train). In the PIHMC or REHMC method with a dual potential, where the low-level potential is N2P2, generation and training can be performed on the fly. For example, use the following combination:

```
<method>
PIHMC

<ipotential>
DUAL

<dual_potential>
VASP N2P2
```

The following keywords are useful.

```
<istep_hmc>
8

<istep_max_hmc>
128

<istep_adjust_hmc>
100

<istep_mul_hmc>
2

<ratio_min_hmc>
0.05

<ratio_max_hmc>
0.20
```

These keywords define a variable number of MD steps per Metropolis step. In this example, the number of MD steps per Metropolis step can vary from “8” (initial value) to “128” following a geometric progression with a common ratio of “2”. The following adjustment is performed every “100” Metropolis steps: The number of MD steps per Metropolis step is increased if the acceptance ratio is lower than “0.05”, and decreased if the acceptance ratio is higher than “0.20”.

Execution of PIMD. Execution is performed in the usual way using the *pimd.mpi.x* command.

4.4.4 MACE

IMPORTANT: The user is solely responsible for all issues related to the MACE license. PIMD does not provide any MACE license.

The MACE machine-learning potential [89] is linked through the subroutine *force_mace_MPI.F90*. The current interface is designed for the MPI version of PIMD. The recommended executable for production calculations is *pimd.mpi.x*. Two inference backends are available:

- **Python backend:** force evaluation is carried out through the Python helper runtime in *lib/mace*.
- **libtorch backend:** force evaluation is carried out directly in the linked C++/libtorch interface in *lib/mace/libtorch*. Python is still used for helper tasks such as model export and on-the-fly training.

Both backends require the helper scripts in *lib/mace*. For helper scripts and on-the-fly training, a Python environment with `torch`, `mace`, `ase`, and `PyYAML` is required.

Compilation and Python runtime. Follow the directions below.

- **Step 1: Compile the PIMD code with the MACE interface. CMake method:** Create a build directory and invoke CMake with the MACE option enabled. The backend is selected with `MACE_INFER_BACKEND`. For the Python backend:

```
> cmake -S . -B build-mace-python \
>   -DMACE=ON \
>   -DMACE_INFER_BACKEND=python
> cmake --build build-mace-python -j
```

For the libtorch backend:

```
> cmake -S . -B build-mace-libtorch \
>   -DMACE=ON \
>   -DMACE_INFER_BACKEND=libtorch \
>   -DLIBTORCH_ROOT=/path/to/libtorch
> cmake --build build-mace-libtorch -j
```

Here, `LIBTORCH_ROOT` must point to a prebuilt libtorch distribution containing *share/cmake/Torch/TorchConfig.cmake*. The libtorch package should be compatible with the PyTorch version used in the helper Python environment. If a dedicated Python virtual environment should also be prepared automatically, the following option can be added to either backend:

```
> cmake -S . -B build-mace-python \
>   -DMACE=ON \
>   -DMACE_INFER_BACKEND=python \
>   -DMACE_INSTALL_VENV=ON
```

In this case, the virtual environment is created below *build/lib/mace/venv*. The MACE backend can be combined with other CMake options. For example, to build the QE + MACE/libtorch executable:

```
> cmake -S . -B build-mace-libtorch-qe \
>   -DMACE=ON \
>   -DMACE_INFER_BACKEND=libtorch \
>   -DLIBTORCH_ROOT=/path/to/libtorch \
>   -DQE=ON \
>   -DQEVERSION=6.3
> cmake --build build-mace-libtorch-qe -j
```

Makefile method: The current MACE interface is maintained primarily through CMake. The makefile-based build is not the recommended method for MACE.

- **Step 2: Select the Python executable used by MACE.** The recommended method is to export the environment variable `PIMD_MACE_PYTHON` before running PIMD:

```
> export PIMD_MACE_PYTHON=/path/to/python
```

If `MACE_INSTALL_VENV=ON` was used at configuration time, the generated helper file may be sourced:

```
> source build/lib/mace/use_mace_python.sh
```

If `PIMD_MACE_PYTHON` is not set, the helper scripts fall back to `sys.executable` or the script shebang, depending on the helper. This Python environment is required not only for the Python inference backend, but also for libtorch export and on-the-fly MACE training.

- **Step 3: Prepare the MACE input files.** For inference, the file *mace_infer_options.yaml* is required. This file must define the model used in production calculations. For example:

```
mace:
  model: SiO2_MACE.model
  device: cuda
  device_assignment: bead_local_rank
  gpus_per_node: 4
  device_id: 0
  default_dtype: float64
  r_max: 6.0
```

Here, `device_assignment: fixed` assigns all local bead ranks to the same `device_id`. `device_assignment: bead_local_rank` assigns the local bead-owning MPI ranks on a node to GPU indices in local order. When `gpus_per_node` is smaller than the number of local bead-owning ranks, the GPU indices are reused cyclically. For on-the-fly learning, the file *mace_train_options.yaml* is also required. This file specifies the finetuning options and the foundation model. For example:

```
mace:
  foundation_model: mace_mp_0_small.model
  device: cuda
  train_gpus_per_node: 1
  max_num_epochs: 50
  patience: 10
```

Here, `train_gpus_per_node` specifies how many GPUs are used for on-the-fly training on the node that contains PIMD world rank 0. If `device: cuda` and `train_gpus_per_node > 1`, PIMD launches single-node distributed MACE training with `python -m torch.distributed.run --standalone`. If `train_gpus_per_node = 1`, training is carried out on a single GPU. Both YAML files may either use a top-level `mace:` mapping, or place the options directly at the top level. The files `model` and `foundation_model` are resolved relative to the directory containing the corresponding YAML file, unless absolute paths are given.

If successful, this procedure generates *pimd.mpi.x*, the PIMD executable internally linked to MACE.

MACE input files. In addition to the standard PIMD input files (*input.dat*, *input_default.dat*, *structure.dat*), the MACE interface uses the following files:

- *mace_infer_options.yaml*: inference settings.
- *mace_train_options.yaml*: training settings (only for on-the-fly learning).

- **.model*: the active MACE model and, optionally, the foundation model.
- *mace_libtorch.pt*: libtorch-exported model generated automatically for the libtorch backend.
- *mace_libtorch.runtime*: libtorch runtime settings generated automatically from *mace_infer_options.yaml*.

The keywords `<mace_infer_options_file>` and `<mace_train_options_file>` are resolved relative to the current run directory unless absolute paths are given. In contrast, `<dir_train_mace>` is used as a working directory for generated runtime and training files. PIMD creates this directory automatically when needed. During on-the-fly learning, PIMD automatically creates *structures*, *mace_training.xyz*, *mace.ini*, *mace_count_1.out*, *mace_count_2.out*, and the helper output directories below `<dir_train_mace>`.

PIMD input file to run simulations (predict). The following keyword should be set in *input.dat*:

```
<ipotential>
MACE
```

This keyword declares the use of the MACE potential. The following optional keywords are also available:

```
<mace_infer_options_file>
mace_infer_options.yaml
```

```
<dir_train_mace>
mace_work
```

The first keyword selects the YAML file used for inference. The second keyword sets the working directory where generated MACE runtime files are created. For the libtorch backend, the runtime control parameters should be set in *mace_infer_options.yaml* through `device`, `device_assignment`, `gpus_per_node`, `device_id`, and `r_max`. The legacy input keywords `<mace_use_cuda>`, `<mace_device_id>`, and `<mace_rmax>` remain as deprecated fallbacks and should not be used in new input files.

PIMD input file to create xsf files. Use the PIMD, PIHMC, or REHMC method to create a set of xsf files for MACE learning. The following keyword may be set in *input.dat*:

```
<iprint_xsf_mace>
10
```

This keyword specifies the interval for writing MACE xsf files; in this case, every 10 steps.

If the PIHMC or REHMC method is selected, the following keyword may also be set:

```
<ioption_xsf_mace>
1
```

This keyword controls whether xsf files are written only for accepted structures or for all trial structures. The xsf files are created in the directory named *structures* below `<dir_train_mace>`.

PIMD input file to generate/train using xsf files. Use the PIMD, PIHMC, or REHMC method to generate training data and train MACE using the xsf files. The following keywords are available in *input.dat*:

```
<istep_train_mace>
100
```

This keyword specifies the interval for MACE training.

```
<lstep_train_mace>
3000
```

This keyword sets the final step for MACE training.

```
<minxsf_train_mace>  
300
```

This keyword specifies the minimum number of xsf files required to start training.

```
<mace_train_options_file>  
mace_train_options.yaml
```

This keyword selects the YAML file used for training.

```
<mace_generate_command>  
" "
```

This keyword sets an optional shell command executed before the MACE training script is started.

```
<istep_save_mace>  
30
```

```
<dir_save_mace>  
saved_models
```

These keywords save snapshot models periodically.

The active production model is taken from the `model` entry in *mace_infer_options.yaml*. When training succeeds, this active model is replaced by the updated model, optional snapshots are written to `<dir_save_mace>`, and the libtorch backend automatically regenerates *mace_libtorch.pt* from the updated active model.

PIMD input file to run simulations (predict/generate/train). In the PIHMC or REHMC method with a dual potential, MACE can be used as the low-level potential while a higher-level potential such as QE or VASP is used to generate reference data. For example:

```
<method>  
PIHMC  
  
<ipotential>  
DUAL  
  
<dual_potential>  
QE MACE  
  
or  
  
<dual_potential>  
VASP MACE
```

In this setup, MACE is used for inexpensive trial moves, while the high-level potential is used for acceptance tests and for generating updated training data.

Remarks. For the Python backend, the MACE interface launches one Python daemon per MPI rank that is responsible for the local bead calculations. Therefore, the Python environment specified by `PIMD_MACE_PYTHON` must be available on all compute nodes. This is particularly important when running batch jobs on supercomputers. For the libtorch backend, force evaluation itself is performed in the linked C++/libtorch code. Python is still used for model export and on-the-fly training. The libtorch backend supports two GPU assignment modes configured in

mace_infer_options.yaml. With `device_assignment: fixed`, all local bead ranks use the same `device_id`. With `device_assignment: bead_local_rank`, the bead-owning ranks on the same node are assigned to GPU indices in local bead order. In this mode, `gpus_per_node` specifies the number of visible GPUs on each node. If `gpus_per_node` is smaller than the number of local bead-owning ranks, the GPU indices are reused cyclically (for example, 4 beads on 2 GPUs are assigned as 0, 1, 0, 1).

4.4.5 ASE calculator interface

The ASE calculator interface is linked through the subroutine *force_ase_MPI.F90*. This interface allows PIMD to evaluate energies, forces, and virials through Python calculators provided by ASE or ASE-compatible packages. The current implementation is intended for the MPI version of PIMD. The recommended executable is therefore *pimd.mpi.x*. The serial executable does not support the ASE calculator interface.

The PIMD executable should be compiled with the ASE option enabled:

```
> cmake -S . -B build-ase -DASE=ON
> cmake --build build-ase -j
```

If ASE and the target calculator are installed in a specific Python environment, set the environment variable `PIMD_ASE_PYTHON` before running PIMD:

```
> export PIMD_ASE_PYTHON=/path/to/python
```

If `PIMD_ASE_PYTHON` is not set, the helper scripts fall back to the local file *.pimd_ase_python* when it is present, and otherwise to the Python specified by the script shebang.

PIMD input file. To use the ASE calculator interface, set in *input.dat*:

```
<ipotential>
ASE
```

```
<ase_infer_options_file>
ase_infer_options.yaml
```

The keyword `<ase_infer_options_file>` selects the YAML file that describes which calculator is constructed in Python.

ASE inference YAML file. The file *ase_infer_options.yaml* must contain a `calculator:` section. Relative paths given in the YAML file are resolved relative to the directory containing that YAML file. The optional section `properties:` controls how the virial is constructed. The following values are available:

- `virial_mode: stress`: use the stress tensor returned by the calculator.
- `virial_mode: forces`: construct the virial from $F^T R$.
- `virial_mode: auto`: use stress when available and otherwise fall back to $F^T R$.

For calculators that provide stress, `virial_mode: stress` is recommended. In particular, the current MACE implementation through ASE has been checked against `TESTVIRIAL` in this mode.

Examples of ASE calculators. The ASE interface supports several styles of calculator specification.

- **MACE through ASE.** The dedicated MACE interface remains available through `<ipotential> = MACE`. However, the same model can also be used through the ASE interface:

```
calculator:
  type: mace
  kwargs:
    model: mace_mp_0_small.model
    device: cpu
    default_dtype: float64
```

```
properties:
  virial_mode: stress
```

- **CHGNet through ASE.** CHGNet is used as an ASE-compatible calculator through the generic custom type:

```
calculator:
  type: custom
  module: chgnet.model
  class: CHGNetCalculator
  kwargs:
    use_device: cpu
```

```
properties:
  virial_mode: stress
```

- **NequIP through ASE.** NequIP is used through a compiled model prepared in advance:

```
calculator:
  type: nequip
  kwargs:
    compile_path: mir-group__Allegro-MP-L__0.1.cpu.nequip.pth
    device: cpu
```

```
properties:
  virial_mode: stress
```

Files and examples. The ASE interface uses the standard PIMD input files (*input.dat*, *input.default.dat*, *structure.dat*) plus the calculator YAML file specified by `<ase_infer_options_file>`. Example directories are provided in *examples/SiO2* and *examples/TiO2*. For example:

```
examples/SiO2/ase_mace_static
examples/SiO2/ase_chgnet_static
examples/SiO2/ase_nequip_static
```

and the corresponding directories under *examples/TiO2*. For foundation or generic models distributed outside this repository, the example directories provide helper scripts such as *setup_env.sh* and *download_model.sh*. In the NequIP examples, the recommended procedure is:

```
> ./setup_env.sh
> ./download_model.sh
> mpirun -np 1 ../../../build-ase/pimd.mpi.x
```

GPU settings can be stored in separate files such as *input_gpu.dat* and *ase_infer_options_gpu.yaml*. In the current NequIP examples, the same compiled model file is used for both CPU and GPU execution, while the device is switched in the YAML file.

4.4.6 PFP

IMPORTANT: The user is solely responsible for all matters related to the PFP license. PIMD does not provide any PFP license.

The machine learning potential of Matlantis PFP [91] is linked through the subroutine *force_pfp.F*. Note that *force_pfp.F* does not support bead-parallel calculations, but single GPU acceleration is available. The MPI version (*force_pfp-MPI.F*), which will support multiple GPU acceleration, is currently under development. The PFP code must be properly installed in order to use this interface. This service is provided by the commercial cloud platform of The Matlantis Corporation: <https://matlantis.com/en/product/about-pfp/>. Please contact them for any inquiries.

5 Hybrid potentials

The hybrid potentials based on QM/MM and ONIOM methods can be used in the PIMD code.

5.1 QM/MM

The following QM/MM methods are implemented in the PIMD code.

- The mechanical embedding QM/MM method.
- The electronic embedding QM/MM method with SMASH.
- The multiple time scale method with mechanical embedding QM/MM method.

These methods can be used in combination with the link atom. The following methods are NOT implemented, although they might be future targets.

- The electronic embedding QM/MM method with codes other than SMASH.
- The multiple time scale method with electronic embedding QM/MM method.

The theoretical background of the QM/MM methods is briefly explained in Section 11.18.

IMPORTANT: The user is solely responsible for all the license issues of external software used to compute hybrid potentials, e.g., SMASH, ABINIT-MP, DFTB, GAMESS, GAUSSIAN, MOLPRO, MOPAC, ORCA, TURBOMOLE. PIMD does not provide any of these licenses.

5.1.1 QM/MM setup

In the two-layer QM/MM method, the whole system is composed of two layers, A and B. Layer A is the primary region (e.g., the solute), while layer B is the secondary region (e.g., solvent). If the boundary between layers A and B is chosen such that a bonded pair is separated between layers A and B, the link atom method should be used. The link atom L caps the bonds broken in layer A when layer A is separated from layer B. The following files are required in the QM/MM method.

- All files required for QM calculations
- *mm.dat*
- *structure.dat*
- *input.dat*

Prepare all files required for QM calculations. The data files required for QM calculations are:

- *abinit-mp.dat* for ABINIT-MP
- *dftb.dat* for DFTB
- *gamess.dat* for GAMESS
- *g98.dat* for GAUSSIAN 98
- *g03.dat* for GAUSSIAN 03
- *g09.dat* for GAUSSIAN 09
- *g16.dat* for GAUSSIAN 16
- *molpro.dat* for MOLPRO
- *mopac.dat* for MOPAC
- *orca.dat* for ORCA
- *turbo.dat*, *control.1* and *control.2* for TURBOMOLE

They should be placed in a subdirectory *./dat*, corresponding to the QM calculations of the subsystem (A) or (A+L). The atoms should be exactly in the same order as defined in *structure.dat*, i.e., the list of atoms in the primary region followed by the list of link atoms. Note that the files are not required for SMASH calculations since the information is already given by the keyword `<smash_input>` in the file *input.dat*.

Prepare mm.dat. The file *mm.dat* must be prepared in the same manner as in the case of the classical force field, `ipotential = MM`. Note that the double counting of QM-QM and QM-MM interactions is avoided automatically within the PIMD code. It is therefore not necessary to edit *mm.dat* to remove these interactions.

Prepare structure.dat. The file *structure.dat* must be in the following format for a water dimer:

```
6
BOHR
O  -1.13151  1.10684 -0.67760  1 A
H  -2.30412 -0.29737 -0.45427  2 A
H   0.45453  0.27505 -0.94620  2 A
O  -5.29474 -1.87202  0.09742  1 B
H  -5.90817 -0.13639  0.18675  2 B
H  -6.79303 -2.84927  0.37973  2 B
```

where “A” and “B” are the primary and secondary regions, respectively. Otherwise, the region “B” can be abbreviated as:

```
6
BOHR
O  -1.13151  1.10684 -0.67760  1 A
H  -2.30412 -0.29737 -0.45427  2 A
H   0.45453  0.27505 -0.94620  2 A
O  -5.29474 -1.87202  0.09742  1
H  -5.90817 -0.13639  0.18675  2
H  -6.79303 -2.84927  0.37973  2
```

The file *structure.dat* must be in the following format for ethane:

8

ANGSTROM

```
C      2.14200   1.39500  -8.93200   1 AL 5 0.70 H
H      1.60400   0.76000  -8.26000   2 A
H      1.74500   2.38800  -8.88000   2 A
H      2.04300   1.02400  -9.93000   2 A
C      3.63100   1.41600  -8.53700   1 B
H      4.16900   2.05100  -9.21000   2 B
H      3.73100   1.78800  -7.53900   2 B
H      4.20300  -0.01200  -8.61200   2 B
```

“AL” means the atom in the primary region (C_1) connected to a link atom (L). “5” means the atom in the secondary region (C_5) which is linked to C_1 . “0.70” is the equilibrium bond length ratio between C_1 -L and C_1 - C_5 . “H” is the atomic element of the link atom, L=H. The region “B” can be abbreviated in this case as well. The file *structure.dat* must be in the following format for butane:

14

ANGSTROM

```
C      2.14200   1.39500  -8.93200   1 B
H      1.60400   0.76000  -8.26000   2 B
H      1.74500   2.38800  -8.88000   2 B
H      2.04300   1.02400  -9.93000   2 B
C      3.63100   1.41600  -8.53700   1 AL2 1 0.70 H 8 0.70 H
H      4.16900   2.05100  -9.21000   2 A
H      3.73100   1.78800  -7.53900   2 A
C      4.20300  -0.01200  -8.61200   1 B
H      3.66500  -0.64700  -7.94000   2 B
H      4.10400  -0.38400  -9.61000   2 B
C      5.69100   0.00900  -8.21800   1 B
H      6.08800  -0.98300  -8.27000   2 B
H      5.79100   0.38100  -7.22000   2 B
H      6.23000   0.64400  -8.89000   2 B
```

“AL2” means the atom in the primary region (C_4) connected to two link atoms (L2).

Prepare input.dat. The details are explained in the next sections.

5.1.2 QM/MM with SMASH

The following keywords are important in the QM/MM method using an internal link with the SMASH code.

<ipotential>

QM/MM

This keyword specifies the use of the QM/MM method.

<qmmm_potential>

SMASH

This keyword specifies the use of the SMASH code.

<qmmm_embedding>

EE

This keyword specifies the embedding scheme, in the present case, electronic embedding. The use of mechanical embedding can be specified in the following way:

<qmmm_embedding>

ME

5.1.3 QM/MM with other codes

The following keywords are important in the QM/MM method using an external link with any code other than SMASH.

`<ipotential>`

QM/MM

This keyword specifies the use of the QM/MM method.

`<qmmm_potential>`

ORCA

This keyword specifies the use of an external code for QM calculations, in the present case, ORCA.

`<qmmm_embedding>`

ME

This keyword specifies the use of mechanical embedding.

`<qmmm_dat_dir>`

dat

This keyword specifies the input directory of QM calculations.

`<qmmm_scr_dir>`

scr

This keyword specifies the temporary scratch directory for QM calculations.

5.1.4 QM/MM with BEST method

The following keywords are important in the QM/MM method with the Boundary based on Exchange Symmetry Theory (BEST).

`<ioption_best>`

1

This keyword specifies the use of BEST.

`<iobest>`

1

This keyword specifies the central atom of BEST, in the present case, the first atom.

`<ikind_best>`

1

This keyword specifies the atomic kind of BEST, in the present case, the first kind.

`<fc_best>`

1.0

This keyword specifies the force constant of BEST, in the present case, 1.0 hartree/bohr².

`<eps_best>`

1.e-16

This keyword specifies the energy cutoff of BEST, in the present case, 10⁻¹⁶ hartree.

`<iprint_best>`

1

This keyword specifies the print interval of BEST in *best.out*.

5.1.5 Multiple time scale QM/MM method

The multiple time scale (MTS) molecular dynamics and path integral molecular dynamics methods are implemented in the PIMD code. The massive Nosé-Hoover chain thermostat is used to generate the NVT ensemble. The method can be used either with the ONIOM potential or the QM/MM potential. The method can be used in combination with the BEST method to deal with systems under open boundary conditions. The theoretical background of the MTS method is briefly described in Section 11.19.

To run the MTS simulations, the following keywords need to be specified:

`<method>`

MTS

This keyword sets the simulation method to the MTS method.

`<ensemble>`

NVT

This keyword sets the statistical ensemble to NVT.

`<bath_type>`

MNHC

This keyword sets the type of thermostat to massive Nosé-Hoover chain.

`<ipotential>`

QMMM

This keyword sets the potential, in the present case, QM/MM.

`<dt>`

0.25d0

This keyword sets the timestep, in the present case, 0.25 femtoseconds.

`<nstep>`

10

This keyword sets the number of steps, in the present case, ten steps.

`<nmulti>`

100

This keyword controls the time increment and the mass scaling factor for layer B. In the present case, the time increment for updating the forces with respect to layer B and the interaction between layers A and B is set to 1/100 of `<dt>`, which means 0.0025 fs. In addition, the masses of all atoms in layer B are set to 1/10000 of the physical masses.

5.2 ONIOM

The two-layer ONIOM method is implemented in the PIMD code. It can be used in combination with the link atom method. The three-layer ONIOM method is NOT implemented, although it may be a future target. The theoretical background of the ONIOM method is briefly explained in Section 11.18.

IMPORTANT: The user is solely responsible for all license issues related to external software used to compute hybrid potentials, e.g., SMASH, ABINIT-MP, DFTB, GAMESS, GAUSSIAN, MOLPRO, MOPAC, ORCA, TURBOMOLE. PIMD does not provide any of these licenses.

5.2.1 ONIOM setup

In the two-layer ONIOM method, the whole system is composed of two layers, A and B. Layer A is the primary region (e.g. the solute), while layer B is the secondary region (e.g. the solvent). If the boundary between layers A and B is chosen such that a bonded pair is separated into layers A and B, the link atom method is used. The link atom L caps the bonds broken in layer A when layer A is separated from layer B.

The following files are required for the ONIOM method:

- All files required for high- and low-level calculations
- *structure.dat*
- *input.dat*

Prepare all files required for high- and low-level calculations. The data files required for high- and low-level calculations are

- *abinit-mp.dat* for ABINIT-MP
- *dftb.dat* for DFTB
- *eam.dat* for EAM
- *gamess.dat* for GAMESS
- *g98.dat* for GAUSSIAN 98
- *g03.dat* for GAUSSIAN 03
- *g09.dat* for GAUSSIAN 09
- *g16.dat* for GAUSSIAN 16
- *mm.dat* for MM
- *molpro.dat* for MOLPRO
- *mopac.dat* for MOPAC
- *orca.dat* for ORCA
- *smash.dat* for SMASH (external link to PIMD)
- *turbo.dat*, *control.1*, and *control.2* for TURBOMOLE

They should be placed in three subdirectories, *./dat_1*, *./dat_2*, and *./dat_3*, corresponding to the calculations, respectively, as follows:

- The calculation of the subsystem (A) or (A+L) using a high-level potential. The atoms must be listed in exactly the same order as defined in *structure.dat*, i.e., the list of atoms in the primary region followed by the list of link atoms.
- The calculation of the subsystem (A) or (A+L) using a low-level potential. The atoms must be listed in exactly the same order as defined in *structure.dat*, i.e., the list of atoms in the primary region followed by the list of link atoms.
- The calculation of the system (A+B) using a low-level potential. The atoms must be listed in exactly the same order as defined in *structure.dat*, irrespective of the region.

Prepare structure.dat. The file *structure.dat* must be in the following format for a water dimer:

```
6
BOHR
O   -1.13151   1.10684  -0.67760   1 A
H   -2.30412  -0.29737  -0.45427   2 A
H    0.45453   0.27505  -0.94620   2 A
O   -5.29474  -1.87202   0.09742   1 B
H   -5.90817  -0.13639   0.18675   2 B
H   -6.79303  -2.84927   0.37973   2 B
```

where “A” and “B” denote the primary and secondary regions, respectively. Alternatively, region “B” can be abbreviated as follows:

```
6
BOHR
O   -1.13151   1.10684  -0.67760   1 A
H   -2.30412  -0.29737  -0.45427   2 A
H    0.45453   0.27505  -0.94620   2 A
O   -5.29474  -1.87202   0.09742   1
H   -5.90817  -0.13639   0.18675   2
H   -6.79303  -2.84927   0.37973   2
```

The file *structure.dat* must be in the following format for ethane:

```
8
ANGSTROM
C    2.14200   1.39500  -8.93200   1 AL 5 0.70 H
H    1.60400   0.76000  -8.26000   2 A
H    1.74500   2.38800  -8.88000   2 A
H    2.04300   1.02400  -9.93000   2 A
C    3.63100   1.41600  -8.53700   1 B
H    4.16900   2.05100  -9.21000   2 B
H    3.73100   1.78800  -7.53900   2 B
H    4.20300  -0.01200  -8.61200   2 B
```

“AL” indicates the atom in the primary region (C_1) connected to a link atom (L). “5” denotes the atom in the secondary region (C_5) that is linked to C_1 . “0.70” is the equilibrium bond-length ratio between C_1 -L and C_1 - C_5 . “H” is the atomic element of the link atom, L = H. Region “B” can also be abbreviated in this case. The file *structure.dat* must be in the following format for butane:

```
14
ANGSTROM
C    2.14200   1.39500  -8.93200   1 B
H    1.60400   0.76000  -8.26000   2 B
H    1.74500   2.38800  -8.88000   2 B
H    2.04300   1.02400  -9.93000   2 B
C    3.63100   1.41600  -8.53700   1 AL2 1 0.70 H 8 0.70 H
H    4.16900   2.05100  -9.21000   2 A
H    3.73100   1.78800  -7.53900   2 A
C    4.20300  -0.01200  -8.61200   1 B
H    3.66500  -0.64700  -7.94000   2 B
H    4.10400  -0.38400  -9.61000   2 B
C    5.69100   0.00900  -8.21800   1 B
H    6.08800  -0.98300  -8.27000   2 B
```

```
H      5.79100   0.38100  -7.22000   2 B
H      6.23000   0.64400  -8.89000   2 B
```

“AL2” indicates the atom in the primary region (C_4) connected to two link atoms (L2).

Prepare input.dat. Details are explained in the following sections.

5.2.2 ONIOM with any codes

The following keywords are important for the ONIOM method using external links with any codes other than SMASH.

```
<ipotential>
ONIOM
```

This keyword specifies the use of the ONIOM method.

```
<oniom_hi_potential>
ORCA
```

This keyword specifies the use of an external code for high-level calculations; in the present case, ORCA.

```
<oniom_hi_potential>
MM
```

This keyword specifies the use of an external code for low-level calculations; in the present case, MM.

```
<oniom_dat_dir>
dat_1 dat_2 dat_3
```

This keyword specifies the input directories for the high-level calculation of the subsystem, the low-level calculation of the subsystem, and the low-level calculation of the system.

```
<oniom_scr_dir>
scr_1 scr_2 scr_3
```

This keyword specifies the temporary scratch directories for the high-level calculation of the subsystem, the low-level calculation of the subsystem, and the low-level calculation of the system.

5.2.3 ONIOM with BEST method

The following keywords are important in the ONIOM method with the Boundary based on Exchange Symmetry Theory (BEST).

```
<ioption_best>
1
```

This keyword specifies the use of BEST.

```
<iobest>
1
```

This keyword specifies the central atom of BEST, in the present case, the first atom.

```
<ikind_best>
1
```

This keyword specifies the atomic kind of BEST, in the present case, the first kind.

```
<fc_best>
1.0
```

This keyword specifies the force constant of BEST, in the present case, 1.0 hartree/bohr².

```
<eps_best>
1.e-16
```

This keyword specifies the energy cutoff of BEST, in the present case, 10⁻¹⁶ hartree.

6 Data analysis

The PIMD code prints the trajectory file, *trj.out*. The code *calc.x* can perform the following analyses using the information from *trj.out*.

- Visualization of trajectory: Transform the data into the xyz format that can be read by visualization software, such as Visual Molecular Dynamics (VMD): <http://www.ks.uiuc.edu/Research/vmd>.
- Statistical analysis: Create lists and distributions of the geometrical variables specified by the user. The variables include bond lengths (interatomic distances), bond angles, dihedral angles, out-of-plane distances, etc. These variables can be specified either by a set of specific atoms or by a set of specific atomic species.

The code reads three files, which should be placed in the execution directory: the trajectory file, *trj.out*; the input file to be edited by the user, *calc.dat*; and the input file that contains the list of all default parameters, *calc.default.dat*. The file *calc.default.dat* can be copied from the directory *examples*.

6.1 Visualization of trajectory

The trajectory file *trj.out* can be transformed into xyz format that can be read by visualization software. By executing *calc.x*, the data is printed to a file named *calc.xyz*. NOTE: The data in *trj.out* are in bohrs, while those in *calc.xyz* are in angstroms.

As an example, a set of input files for a water molecule is available. Copy it to the directory *run*:

```
> cp -r examples/H2O/calc_xyz run
```

and enter that directory:

```
> cd run/calc_xyz
```

Let us examine the keywords present in *calc.dat* one by one.

```
<iboundary>
0
```

This keyword sets the boundary condition, as in *input.dat*. In the present case, “0” means a free boundary condition.

```
<nbead>
32
```

This keyword sets the number of beads, as in *input.dat*. In the present case, 32 beads are chosen.

Now come the keywords specific to *calc.dat*.

```
<iconf_ini_calc>
1
```

This keyword sets the first configuration to be used for the analysis. In this case, “1” means the first configuration of *trj.out*.

```
<iconf_fin_calc>
-1
```

This keyword sets the last configuration to be used for the analysis. In this case, “-1” means the last configuration of *trj.out*.

```
<iprint_std_calc>
10
```

This keyword sets the standard output. In this case, the standard output is printed every 10 configurations.

```
<jprint_xyz>
1
```

This keyword sets the print interval for xyz data. In this case, “1” means every step.

```
<jorigin_xyz>
2
```

This keyword sets the origin of the system. In this case, “2” means the origin is fixed in space.

```
<jformat_xyz>
2
```

This keyword sets the format of the xyz file. In this case, “2” means that all bead configurations are printed together.

To run, type

```
> calc.x
```

then the xyz trajectory will be printed to the file *calc.xyz*.

NOTE: When using the *calc.x* command, the file *calc.xyz* is always overwritten if it already exists.

6.2 Statistical analysis

For statistical analysis using *calc.x*, all the keywords in Sec. 6.1 are important. Additionally, the following keywords are important.

```
<iprint_calc>
1
```

This keyword sets the print interval of the statistical analysis. In the present case, “1” means every step.

```
<ncalc>
5
  lin.atom.dens   1  2           'calc.12a.dens'
  angl.atom.dens  2  1  3       'calc.213.dens'
  lin.atom.list   1  2           'calc.12a.list'
  lin.spec.list   1  2           'calc.12s.list'
  angl.atom.list  2  1  3       'calc.213.list'
```

This keyword sets the data to be analyzed. In this case, five types of analyses are specified.

- The density with respect to the distances between atoms #1 and #2 is printed to the file *calc.12a.dens*.

- The density with respect to the angles defined by atoms #1, #2, and #3 is printed to the file *calc.213.dens*.
- The list of interatomic distances of atoms #1 and #2 is printed to the file *calc.12a.list*.
- The list of interatomic distances with respect to atomic species #1 and #2 is printed to the file *calc.12s.list*.
- The list of bond angles with respect to atoms #1, #2, and #3 is printed to the file *calc.213.list*.

To run, type

```
> calc.x
```

After the run, the five output files *calc.12a.dens*, *calc.213.dens*, *calc.12a.list*, *calc.12s.list*, and *calc.213.list* will be created.

NOTE: When using the *calc.x* command, all output files specified in the keyword `<ncalc>` are always overwritten if they already exist.

7 Codes

7.1 Main codes

The main executable programs are:

- *pimd.x*: Serial version.
- *pimd.mpi.x*: Parallel (MPI) version.

The parallel version can be used for the following methods, which involve multiple replicas, beads, or images:

- Static calculations (STATIC) with multiple structures.
- Classical molecular dynamics (MD) for NVE and NVT ensembles with multiple independent trajectories.
- Normal mode analysis (NMA) with multiple geometric displacements.
- Phonon calculations (PHONON) with multiple geometric displacements.
- Path integral simulations (PIMD, CMD, RPMD, PIHMC) with multiple beads.
- String method (STRING) with multiple images.
- Metadynamics (MTD) with multiple walkers.
- Replica exchange hybrid Monte Carlo (REHMC) with multiple replicas.

A hierarchical parallelization scheme is implemented, in which force parallelization and bead parallelization can be used simultaneously. This scheme is available for first-principles potentials:

```
<ipotential> = ABINIT-MP, CP2K, NTCHEM, PHASE0, QE, SMASH, VASP
```

for semiempirical potentials:

```
<ipotential> = DFTB
```

and for machine-learning potentials:

```
<ipotential> = AENET, ASE, MACE, MTP, N2P2
```

Hierarchical parallelization is also applicable to all empirical potentials:

`<ipotential> = MM, EAM, POL, ADP, PAIR, TERSOFF`

By default, hierarchical parallelization follows this rule. Let P be the number of beads and C the number of processors.

- If $P < C$, all P beads are computed in parallel, each bead using C/P processors.
- If $P > C$, C beads are computed in parallel, and the calculation is repeated sequentially P/C times.

These behaviors can be controlled in *input.dat* using the keywords `<np_beads>` (number of beads computed simultaneously) and `<np_force>` (number of processors assigned to each bead).

Only bead parallelization is available for the following first-principles potentials:

`<ipotential> = GAMESS, G98, G03, G09, G16, MOLPRO, ORCA, TURBOMOLE`

and for the semiempirical potential:

`<ipotential> = MOPAC`

In these cases, the number of processors must not exceed the number of beads specified by `<nbead>`.

The MPI execution command depends on the MPI implementation. Typically, the calculation can be started as follows:

```
> mpirun -np 4 -machinefile host.list pimd.mpi.x >> monitor.out &
```

where the file *host.list* contains a list of host names:

```
machine_name_for_process_1  
machine_name_for_process_2  
machine_name_for_process_3  
machine_name_for_process_4
```

Please refer to the documentation of your MPI implementation for further details.

7.2 Codes for post-processing

The following code is provided for post-processing:

- *calc.x*: Statistical analysis of simulation results.

See Section 6.2 for details.

7.3 Subsidiary codes

The subsidiary codes are:

- *run_molpro.x*: Interface to run the MOLPRO code.
- *run_vasp.x*: Interface to run the VASP code.

These programs are called automatically from the main codes only when required.

7.4 Utility codes

The following utility programs are included:

- *convert_tinker.x*: Converts TINKER force-field files into the PIMD input format.
- *convert_charmm.x*: Converts CHARMM force-field files into the PIMD input format.
- *prep_liquid.x*: Generates random initial configurations for molecular liquids and creates TINKER xyz files.

See Section 8 for details.

7.5 External calls

The PIMD code can be invoked externally from other programs. For example, the `external` keyword in GAUSSIAN can be used to obtain partially optimized structures or transition-state geometries. In this case, the program *gau.x* acts as an interface between GAUSSIAN and *pimd.x* (the serial version only). A simple example for a water molecule is provided in the directory `examples/H2O/gau_static`.

8 Classical force field

As a convenient way to set up the classical force field, it is recommended to prepare an input file used for the TINKER code [70], which is referred to here as the TINKER xyz format (txyz). The TINKER xyz format (which is different from the usual xyz format used for visualization) provides a simple way to specify the atom types and bond topologies necessary to describe the classical force field. Once the TINKER xyz file is prepared, a utility code called *convert_tinker.x* is useful. Together with the parameter file (prm), the code can convert the file into the format used in the PIMD code, *mm.dat*, see Section 9.7. Currently, the available parameter files are

- OPLS-AA (oplsaa.prm) [71].
- AMBER 94 (amber94.prm) and AMBER 99 (amber99.prm) [72].
- CHARMM 19 (charmm19.prm) and CHARMM 22 (charmm22.prm) [73].
- CLAYFF (clayff.prm) [51].

These files can be found in the directory `./lib/tinker/`.

Currently, it is required that the force fields have the form of Eq.(10). Thus, the following force fields are NOT supported:

- Force fields including “ghost” atoms, such as TIP4P and TIP5P.
- United-atom force fields.
- Polarizable force fields.
- Allinger force fields.
- Force fields including multipoles.
- Any other force fields that are NOT in the form of Eq.(10).

In the next section, the instructions for this utility are described, together with some examples.

8.1 How to convert TINKER files

IMPORTANT: The user is solely responsible for all issues related to the TINKER license. PIMD does not provide any TINKER license.

As mentioned above, the utility code *convert_tinker.x* helps create an input file *mm.dat* from a TINKER input file. As an example, a set of input files for alanine dipeptide is available. Thus, let us copy it to the directory *run*:

```
> cp -r examples/dialanine/tinker run
```

and enter that directory:

```
> cd run/tinker
```

Two files should be present in the execution directory:

- An input file in TINKER xyz format (**.xyz*), in the present case, *dialanine.xyz*.
- A parameter file in TINKER format (**.prm*), in the present case, *charmm22.prm*.

The former is supposed to be edited by the user, while the latter is supposed to be left unchanged. Note that the format of the TINKER xyz file is not the standard xyz format, and here it is renamed as **.xyz* for convenience.

Let us take a look at the input file.

```
> cat dialanine.xyz
```

```
22  Alanine Dipeptide // CHARMM22 C5 Minimum
  1  CT3   -2.249880   -0.851680   -0.058940   27    2    4    5    6
  2  C     -0.786160   -1.071640   -0.041920   20    1    3    7
  3  O     -0.312190   -2.152570    0.279550   74    2
  4  HA     -2.764750   -1.780850    0.268810    1    1
  5  HA     -2.586730   -0.597290   -1.085860    1    1
  6  HA     -2.521030   -0.027170    0.633690    1    1
  7  NH1    -0.025660   -0.031640   -0.391820   63    2    8   11
  8  CT1     1.415230   -0.079650   -0.382260   23    7    9   12   13
  9  C       1.933490    1.322610   -0.124130   20    8   10   17
 10  O       1.170440    2.286690   -0.127990   74    9
 11  H      -0.424700    0.854010   -0.611480    3    7
 12  HB      1.746460   -0.702310    0.439450    4    8
 13  CT3     1.958160   -0.619200   -1.723660   27    8   14   15   16
 14  HA      1.627360    0.025240   -2.565870    1   13
 15  HA      3.066090   -0.674020   -1.723780    1   13
 16  HA      1.559770   -1.642940   -1.895670    1   13
 17  NH1     3.251530    1.467720    0.116150   63    9   18   19
 18  CT3     3.834060    2.763270    0.377670   61   17   20   21   22
 19  H       3.852350    0.675210    0.119160    3   17
 20  HA      4.929130    2.674280    0.551220    1   18
 21  HA      3.372750    3.226480    1.279840    1   18
 22  HA      3.670130    3.449260   -0.484700    1   18
```

In this file, the number of atoms n (= “22”, in the present case) and the title (Alanine Dipeptide) appear in the first line. The next n lines contain information on the respective atoms. These include the atom number (in the present case, “1”, in the second line), the atomic symbol (“CT3”), the Cartesian coordinates (x, y, z) = (“-2.249880”, “-0.851680”, “-0.058940”) in angstroms, the atom type (“27”), and the bonded atoms (“2,

3, 5, 6"). The atom types defined here should match those given in the parameter file. Therefore, the input file is specific to the chosen force field.

Meanwhile, all information on the force field is included in the parameter file, in the present case, *charmm22.prm*. Users are encouraged to look inside the file for details.

To run the program, type

```
> convert_tinker.x $1 $2 $3 $4 $5 $6 $7 $8
```

where the eight arguments (\$1–\$8) are explained as follows.

- \$1 is the name of the TINKER input file (INPUT), for example, "*dialanine.txyz*".
- \$2 is the name of the TINKER parameter file (INPUT), for example, "*charmm22.prm*".
- \$3 is the inner Lennard-Jones cutoff distance in bohr, for example, "20.0". This is where the LJ interaction starts to damp.
- \$4 is the outer Lennard-Jones cutoff distance in bohr, for example, "25.0". This is where the LJ interaction disappears.
- \$5 is the name of the PIMD force field file (OUTPUT), "*mm.dat*".
- \$6 is the name of the PIMD geometry file (OUTPUT), *centroid.dat*.
- \$7 is the name of the geometry printed in xyz format (OUTPUT), for example, "*structure.dat*".
- \$8 is the name of the PIMD input file (OUTPUT), "*input.dat*".

Upon execution, the following will appear on the screen:

...

Thus, the details of all individual interactions are displayed. At the end of the run, four files, *mm.dat*, *input.dat*, *centroid.dat*, and *structure.dat*, will be created. The files *mm.dat*, *structure.dat*, and *input.dat* are meant to be used as input for the PIMD code. The file *input.dat* is suitable for a static calculation, so the user needs to modify it appropriately. The file *structure.dat* is in xyz format, and it can be used to check the geometry with visualization software.

If the TINKER code is installed on your computer, the results can be cross-checked. The details of all individual interactions can be calculated individually with the "analyze" code implemented in TINKER [70]. This is done in the following way.

```
> analyze dialanine.txyz charmm22.prm D
```

One can see that the results are identical.

NOTE: In the *convert_tinker.x* code, the energy calculation is always performed under free boundary conditions. It may happen that the results for intermolecular interactions (i.e., LJ and charges) differ when the TINKER "analyze" code is used with periodic boundary conditions.

Now, by running the PIMD code, the following result will appear on the screen.

```
> pimd.x
```

```
=====
```

potential energy values			
bead	hartree	kcal/mol	kJ/mol
1	-0.02482693	-15.579146	-65.183148

```
=====
```

Again, one can check that the energy value is exactly the same.

8.2 Preparation of initial structure

The *prep_liquid.x* code is a utility code for the generation of random structures of molecular liquids and the TINKER xyz file.

This code creates the geometry files for the PIMD code in three different forms: *centroid.dat* (for initiations with `<input_style> = OLD`), *structure.dat* (for initiations with `<input_style> = NEW`), and *geometry.ini* (for restarts). It also creates the TINKER xyz file, *structure.txyz*, to prepare the classical force field.

This code reads the keywords from the file *input.dat*, which should be prepared by the user. The following is an example of the file *input.dat* to prepare the simulation of aqueous hexanediol.

```
<natom>
280
```

This keyword sets the number of atoms, in this case, “280” atoms, with 86 water molecules (3 atoms each) and one hexanediol molecule (22 atoms).

```
<nspec>
8
O      15.99900  86   63  2
H       1.00800 172   64  1
H       1.00800   2   97  1
O      15.99900   2   96  2
C      12.01100   2   81  4
C      12.01100   2  100  4
C      12.01100   2   80  4
H       1.00800  12   85  1
```

This keyword sets the number of species, followed by the atomic symbol, the atomic mass, the number of atoms of each species, the atomic type of the classical force field (in the present case *opls_aa.prm*), and the bond order. The atomic type and the bond order affect only the TINKER input file (txyz). In this case, there are “8” species.

```
<iboundary>
1
  26.589    0.000    0.000
    0.000  26.589    0.000
    0.000    0.000  26.589
```

This keyword sets the boundary condition and the simulation box. In this case, periodic boundary conditions are applied, and the simulation box is cubic with a side length of 26.589 bohr.

```
<components>
2
MOL      22      GENERAL
-0.93099 -4.50041  4.00409
 5.14070  5.24963  3.62029
-2.29140 -3.36393  3.96554
 3.33204  5.23867  3.69512
 0.36177  1.74551  2.75309
-0.00389 -0.45356  0.96768
-2.44787 -1.91602  1.71015
 2.74906  3.32740  1.96319
-3.38480 -3.57747 -0.43337
 2.37017  4.79893 -0.48344
 0.69625  5.79104 -0.37705
```

```

3.89386  6.10662 -0.98025
1.86864  3.71234 -2.06514
0.89071  1.18086  4.72407
-1.27549  2.94303  2.68714
-0.25390  0.24564 -0.88274
1.53510 -1.80849  1.13095
-3.80829 -0.46565  1.98775
4.20755  1.93864  1.77308
-5.47767 -3.92499 -0.29825
-3.16181 -2.45213 -2.13450
-2.49738 -5.43979 -0.67147
H2O      3      GENERAL
0.00000  0.00000  0.00000
1.48911  1.16342  0.00000
-1.48911  1.16342  0.00000

```

This keyword sets each molecular component. In this case, there are two components in the system: “MOL” and H2O. “MOL” is a non-linear molecule composed of 22 atoms, while “H2O” is a non-linear molecule composed of 3 atoms. The atomic coordinates in the molecular frame are given after “MOL” and “H2O”.

```

<molecules>
87
MOL  259  260  261  262  263  264  265  266  267  268
      269  270  271  272  273  274  275  276  277  278
      279  280
H2O   1   87  173
H2O   2   88  174
H2O   3   89  175
...   ..   ...   ...
...   ..   ...   ...
...   ..   ...   ...
H2O   84  170  256
H2O   85  171  257
H2O   86  172  258

```

This keyword gives the list of atoms in the molecules. The numbering should be consistent with that given by the keyword <nspec>.

To run, type

```
> prep_liquid.x
```

```

-----
atom 1  atom 2  bond order  dist/au  r1+r2/au
-----
      1      87      1      2  1.88971  3.45000
      1     173      2      2  1.88971  3.45000
      2      88      1      2  1.88971  3.45000
      .      ...      .      .  .....  .....
      .      ...      .      .  .....  .....
      .      ...      .      .  .....  .....

```

Created: structure.dat.

Created: centroid.dat.

Created: `geometry.ini`.

Created: `structure.dat`.

Normal termination of PREP_LIQUID.

Confirm that the files are created properly. The file *geometry.xyz* is in xyz format and can be visualized. The file *structure.txyz* can be used to create *mm.dat* as shown in the previous section.

NOTE: If one gets the following message, there are ways to fix it.

Warning - Number of bonds do not match: 4 and 3 for atom: 265.

This warns that the number of neighboring atoms of atom 265 (= 3) did not match the bond order of atom 265 (= 4). Therefore, one can

- check if the bond order is correct.
- tune the atomic radius in `<symbol>` in *input.default.dat*, which changes the values of the cutoff radii ($r1 + r2$) and thus affects the number of neighboring atoms.
- change manually the TINKER input file (txyz).

8.3 How to use CHARMM force field

IMPORTANT: Now that the *convert_tinker.x* code can convert CHARMM force fields as well, the prescription introduced in this section is becoming obsolete.

The *convert_charmm.x* code helps create the input files, especially *mm.dat*, using the CHARMM force field. In order to use this, three files should be prepared:

- CHARMM parameter file (*par_*.prm*)
- CHARMM protein structure file (**.psf*)
- Structure file in protein data bank format (**.pdb*)

All the parameters for intramolecular bonds and Lennard-Jones interactions, as well as atomic charges, are read from the *par* file. The residue name, the atom name, the atom type, the atomic masses, as well as the intramolecular bond topologies, are read from the *psf* file. The atomic coordinates are read from the *pdb* file.

With these data, the *convert_charmm.x* code generates

- *mm.dat*
- *input.dat*
- *centroid.dat*

which can be directly used as the input for the PIMD code. The files *mm.dat*, *centroid.dat*, and *input.dat* are meant to be used as input for the PIMD code. The file *input.dat* works for static calculations, so the user needs to modify it appropriately.

To run, type

```
> convert_charmm.x
```

Then, the user is asked to provide the following data interactively:

- The name of the *par* file.
- The name of the *psf* file.

- The name of the *pdb* file.
- Cutoff parameters for the Lennard-Jones interaction.

NOTES:

- The order of atoms in the *psf* and *pdb* files must be exactly the same, or else the result will be completely meaningless.
- In the CHARMM force field, a special parameter set for 1-4 Lennard-Jones interactions is prepared for some atoms, and 1-4 electrostatic interactions are assumed to be zero. This is properly considered in this code.
- Not only bonded interactions but also Urey-Bradley interactions are included below the `<linear_bonds>` keyword in *mm.dat*. Note that they have the same mathematical form (harmonic oscillator).
- The *psf* file can be created from CHARMM topology files (*top-*.prm*) using, for instance, the VMD code [4].

9 Keywords

In this section, all keywords used in the PIMD code are explained.

NOTE: Each keyword must start from the first column of the line. Do not place any spaces or characters before a keyword. If a keyword is indented or preceded by extra characters, it will not be recognized correctly by the code.

9.1 Necessary keywords in *input.dat*

9.1.1 `<ensemble>`

Used in ALL methods. This keyword specifies the type of thermodynamic ensemble. The data should be provided on the first line (character). The available options are as follows.

- `<ensemble>` = NONE: unspecified.
- `<ensemble>` = NVE: constant volume and energy.
- `<ensemble>` = NVT: constant volume and temperature.
- `<ensemble>` = NPT: constant pressure and temperature.
- `<ensemble>` = NTT: constant tension and temperature.
- `<ensemble>` = NPH: constant pressure and enthalpy.
- `<ensemble>` = NTH: constant tension and enthalpy.

The applicable conditions are as follows.

- For `<method>` = CMD, RPMD, one must choose NVE.
- For `<method>` = MTD, MTS, REHMC, one must choose NVT.
- For `<method>` = MD, ROTOR, NVE, one must choose NVT.
- For `<method>` = PIMD, one must choose among NVT, NPT, NPH, NTH.
- For `<method>` = PIHMC, one must choose among NVT, NPT.

The default value is “NONE”.

9.1.2 <ipotential>

Used in ALL methods. This keyword specifies the potential. The data should be provided on the first line (character). Choose one of the following standard potentials.

- <ipotential> = ABINIT-MP: ABINIT-MP.
- <ipotential> = ADP: angular dependent potential.
- <ipotential> = AENET: AENET.
- <ipotential> = ASE: ASE calculator interface.
- <ipotential> = CP2KLIB: CP2K (internal link).
- <ipotential> = DFTB: DFTB (external link).
- <ipotential> = DFTBLIB: DFTB (internal link).
- <ipotential> = EAM: embedded atom method.
- <ipotential> = G98: GAUSSIAN 98.
- <ipotential> = G03: GAUSSIAN 03.
- <ipotential> = G09: GAUSSIAN 09.
- <ipotential> = GAMESS: GAMESS.
- <ipotential> = METALWATER: GAL21.
- <ipotential> = MACE: MACE.
- <ipotential> = MM: molecular mechanics (non-polarizable).
- <ipotential> = MOLPRO: MOLPRO.
- <ipotential> = MTP: MTP.
- <ipotential> = N2P2: N2P2.
- <ipotential> = MOPAC: MOPAC.
- <ipotential> = N2P2: N2P2.
- <ipotential> = NTCHEM: NTCHEM.
- <ipotential> = ONIOM: ONIOM (two-layer).
- <ipotential> = ORCA: ORCA.
- <ipotential> = OSS: Ojamae–Shavitt–Singer (OSS2) water.
- <ipotential> = PFP: Matlantis PFP.
- <ipotential> = POL: molecular mechanics (polarizable).
- <ipotential> = PAIR: tabulated pair potential.
- <ipotential> = QE: QUANTUM ESPRESSO.
- <ipotential> = QMMM: QM/MM (mechanical embedding).

- `<ipotential>` = SMASH: SMASH.
- `<ipotential>` = TIP4P: water (test).
- `<ipotential>` = TERSOFF: Tersoff potential.
- `<ipotential>` = TURBOMOLE: TURBOMOLE.
- `<ipotential>` = USER: user-defined.
- `<ipotential>` = VASP: VASP (internal link).
- `<ipotential>` = VASP5: VASP (external link).
- `<ipotential>` = VASP6: VASP version 6.
- `<ipotential>` = WATER: SPC/F water (test).
- `<ipotential>` = XTB: (extended tight binding).

The default value is not defined.

9.1.3 `<method>`

Used in ALL methods. This keyword specifies the simulation method. The data should be provided on the first line (character). Choose one of the following options:

- `<method>` = STATIC: static calculation.
- `<method>` = TESTFORCE: test forces by numerical differentiation.
- `<method>` = TESTVIRIAL: test the virial by numerical differentiation.
- `<method>` = GEOOPT: geometry optimization using the limited-memory BFGS method.
- `<method>` = NMA: normal mode analysis.
- `<method>` = SD: steepest descent method.
- `<method>` = BOXOPT: box optimization using the limited-memory BFGS method.
- `<method>` = FULLOPT: simultaneous geometry and box optimization using the limited-memory BFGS method.
- `<method>` = ELASTIC: static elastic constants.
- `<method>` = MD: classical molecular dynamics.
- `<method>` = PIMD: path integral molecular dynamics.
- `<method>` = BCMD: Brownian chain molecular dynamics.
- `<method>` = CMD: centroid molecular dynamics.
- `<method>` = RPMD: ring polymer molecular dynamics.
- `<method>` = TRPMD: thermostatted ring polymer molecular dynamics.
- `<method>` = MTD: metadynamics.
- `<method>` = MTS: multiple time-scale path integral molecular dynamics.

- `<method>` = TASS: temperature accelerated sliced sampling.
- `<method>` = PIHMC: path integral hybrid Monte Carlo.
- `<method>` = REHMC: replica-exchange hybrid Monte Carlo.
- `<method>` = STRING: string method.
- `<method>` = PHONON: phonon calculation.
- `<method>` = TFS: nonadiabatic surface hopping dynamics (Tully ’ s fewest switches).
- `<method>` = MFE: nonadiabatic mean-field dynamics (Ehrenfest mean field).
- `<method>` = ROTOR: molecular dynamics of rigid-body molecules.
- `<method>` = SCAN: consecutive static calculations.
- `<method>` = GAD: gentlest ascent dynamics.
- `<method>` = AFED: adiabatic free energy dynamics.

The default value is “STATIC”.

9.1.4 `<natom>`

Used in ALL methods. This keyword specifies the number of atoms. The data should be provided on the first line (integer). The default value is 0 (not defined).

NOTE: This keyword is only used for `<input_style>` = OLD (default). For `<input_style>` = NEW, this keyword is obsolete. See Section 1 for details.

9.1.5 `<nspec>`

Used in ALL methods. This keyword specifies the atomic species present in the system. On the first line, the number of species should be given. On the following lines, the parameter set for each atomic species, namely the atomic symbol (character), the atomic mass in amu (real number), and the number of atoms (integer) for that species, should be provided. The default value is 0 (not defined). NOTE: The same atomic symbol may be used more than once.

NOTE: This keyword is only used for `<input_style>` = OLD. For `<input_style>` = NEW (default), this keyword is obsolete. See Section 1 for details.

9.2 Important keywords in *input.dat*

9.2.1 `<bath_type>`

Used in the cases of `<method>` = MD, PIMD, MTS, CMD, MTD. This keyword sets the type of Nosé-Hoover thermostats. The data should be provided in the first line (character).

- `<bath_type>` = NONE: no thermostat is used.
- `<bath_type>` = NHC: a Nosé-Hoover chain thermostat attached to the whole system.
- `<bath_type>` = NHCS: Nosé-Hoover chain thermostats attached independently to x , y , and z .
- `<bath_type>` = MNHC: massive Nosé-Hoover chain thermostats attached to each degree of freedom and to each bead.

The conditions are as follows.

- For `<method> = MD` with `<ensemble> = NVT`, one must choose MNHC.
- For `<method> = CMD, MTD, MTS`, one must choose MNHC.

The default value is “NONE” (no thermostat).

9.2.2 `<dt>`

Used in ALL cases except `<method> = STATIC, TESTFORCE, TESTVIRIAL, GEOOPT, NMA, PHONON, BOXOPT, FULLOPT`. This keyword sets the step size. The data should be provided in the first line (real number).

- For `<method> = SD, STRING, GAD`: the initial step size in hartree^{0.5} femtoseconds.
- For `<method> = MD, PIMD, PIHMC, BCMD, CMD, RPMD, TRPMD, REHMC, TFS, MFE, MTS, MTD, ROTOR`: the step size in femtoseconds.

The default value is “0.25” (either femtoseconds or hartree^{0.5} femtoseconds, depending on the method chosen).

NOTE: In `<method> = CMD`, it is recommended that the `<dt>` value be set smaller than usual by a factor of `<igamma>`, to ensure adiabatic separation between the centroid and the non-centroid modes.

9.2.3 `<iboundary>`

Used in ALL cases. This keyword sets the boundary condition. The data should be provided in the first line (integer). Choose one from below:

- `<iboundary> = 0`: free boundary (isolated system).
- `<iboundary> = 1`: periodic boundary; trajectory folded within the box; box matrix **h** in bohr.
- `<iboundary> = 2`: periodic boundary; trajectory not limited within the box; box matrix **h** in bohr.
- `<iboundary> = BOHR`: periodic boundary; trajectory folded within the box; box matrix **h** in bohr (same as 1).
- `<iboundary> = ANGSTROM`: periodic boundary; trajectory folded within the box; box matrix **h** in angstrom.

The default value is “0” (free boundary).

For `iboundary = 1`, the box information should be specified. By default, the last step and the box matrix, $\overleftrightarrow{\mathbf{h}}$, are read from the file *box.ini*. The file *box.ini* should include three lines

```
i, h(1,1), h(1,2), h(1,3)
i, h(2,1), h(2,2), h(2,3)
i, h(3,1), h(3,2), h(3,3)
```

where the first column is the last step (integer), and the second, third, and fourth columns (real numbers) correspond to the box-matrix elements in bohrs, $(h(1,1), h(2,1), h(3,1)) = (a_x, a_y, a_z)$, $(h(1,2), h(2,2), h(3,2)) = (b_x, b_y, b_z)$, and $(h(1,3), h(2,3), h(3,3)) = (c_x, c_y, c_z)$.

If the file *box.ini* is not found, the box matrix is read from the three lines after the keyword `<iboundary>` in the file *input.dat* as:

```
<iboundary>
1
h(1,1), h(1,2), h(1,3)
h(2,1), h(2,2), h(2,3)
h(3,1), h(3,2), h(3,3)
```

9.2.4 <iprint_rest>

This keyword sets the print interval for generating and overwriting the restart files, **.ini*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.2.5 <iprint_std>

This keyword sets the print interval of the standard output *standard.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off.

9.2.6 <iprint_trj>

This keyword sets the print interval of the standard trajectory file *trj.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.2.7 <iread_exit>

This keyword sets the interval for checking the existence of the *exit.dat* file for a soft exit. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off.

NOTE: Reading the file *exit.dat* from the disk will slightly slow down the computation, which can be noticeable when running test calculations with a small molecular system using the MM potential. In such a case, set <iread_exit> either to a larger value so that it is checked less frequently, or to “-1” so that it is not checked at all.

9.2.8 <nbead>

Used in ALL cases except <method> = STATIC, SD, GEOOPT. This keyword sets the number of beads.

- For <method> = PIMD, BCMD, CMD, RPMD, TRPMD, PIHMC, MTS: the number of beads.
- For <method> = TESTFORCE, TESTVIRIAL: the number of structures.
- For <method> = MTD: the number of walkers.
- For <method> = STRING: the number of images.
- For <method> = REHMC: the number of replicas.
- For <method> = NMA, PHONON: the number of shifted geometries calculated in parallel.
- For <method> = MD with <ncons> = 0, MFE, TFS: the number of independent trajectories.
- For <method> = MD with <ncons> = 1: the number of independent trajectories with different constraints.

The data should be provided in the first line (integer). The default value is “1”.

9.2.9 <nstep>

Used in ALL cases except <method> = STATIC, TESTFORCE, TESTVIRIAL, NMA. This keyword sets the number of steps.

- For <method> = GEOOPT, BOXOPT, FULLOPT: the maximum number of optimization steps.
- For <method> = STRING: the maximum number of string updates.
- For <method> = SD: the number of steps along the steepest-descent trajectory.
- For other methods: the number of time steps of the trajectory.

The data should be provided in the first line (integer). The default value is “4”.

9.2.10 <temperature>

Used in ALL cases except <method> = STATIC, TESTFORCE, TESTVIRIAL, GEOOPT, NMA, PHONON, BOXOPT, FULLOPT, ELASTIC. This keyword sets the temperature of the system in kelvin. The data should be provided in the first line (real number).

- For <method> = MD, PIMD, BCMD, CMD, RPMD, TRPMD, MTD, MTS, PIHMC, REHMC, TFS, MFE, ROTOR: the temperature of the Maxwell–Boltzmann distribution used to generate the initial random velocities.
- For <method> = MD, PIMD, BCMD, CMD, TRPMD, MTD, MTS, PIHMC, REHMC, ROTOR, with ensemble=NVT, NPT, NTT: the temperature of the Nosé–Hoover thermostats.

The default value is “300” kelvin.

9.3 Frequently used keywords in *input.dat*

9.3.1 <beadsread>

Used in the cases of <method> = PIMD, PIHMC, BCMD, CMD, RPMD, TRPMD, MD, MTS, REHMC, MTD.

- For <method> = PIMD, PIHMC, MTS, BCMD, CMD, RPMD and TRPMD, this keyword controls the generation of the initial set of bead configurations. After the centroid configuration is read from the file *centroid.dat*, the initial position of each bead is generated randomly around the centroid, with the distribution being that of a free quantum atom. This keyword sets the temperature, in kelvin, that characterizes the bead spread of a free quantum atom. The data should be provided in the first line (real number). As the temperature becomes higher, the bead spread shrinks, and all beads are generated closer to the centroid.
- The same idea is inherited for <method> = MD, REHMC, MTD. This keyword sets the temperature, in kelvin, that characterizes the spread used to generate the initial set of configurations for each trajectory in MD, each replica in REHMC, and each walker in MTD.

The default value is “500” kelvin.

NOTE: The temperature in the keyword <beadsread> has nothing to do with the physical temperature of the system designated by the keyword <temperature>. If one prefers the initial configurations to be more different from one another, a smaller value should be chosen.

9.3.2 <corrections>

Used in the cases of <method> = PIMD, PIHMC, MTS, BCMD, CMD, RPMD and TRPMD. This keyword sets the translational and rotational corrections applied to path-integral simulations. The data should be provided in the first line (two integers).

- The first integer: the translational-correction option.
 - If “0”, the correction is not applied.
 - If “1”, the correction is applied at the initial step only.
 - If “2”, the correction is applied at every step.
- The second integer: the rotational-correction option.
 - If “0”, the correction is not applied.
 - If “1”, the correction is applied at the initial step only.
 - If “2”, the correction is applied at every step.

The default values are “0, 0”, which mean that both translational and rotational corrections are not applied.

NOTE: In the PIMD code, there is a limitation in using the rotational correction for diatomic systems. In this specific case, the atoms must be aligned initially on the *x* axis.

9.3.3 <input_style>

Used in ALL cases. This keyword sets the input style for the initial molecular structure. The data should be provided in the first line (character).

- OLD: Old format using the file *centroid.dat*.
- NEW: New format using the file *structure.dat*.

The default value is “NEW”.

9.3.4 <iprint_dcd>

This keyword sets the print interval of several binary files, *.dcd. The data should be provided in the first line (integer). The default value is “-1” (do not print).

The formats of the files *box.dcd*, *dipoles.dcd*, *force.dcd*, *potential.dcd*, *trj.dcd*, *trj_unfolded.dcd*, *vel.dcd* are described in the output-files section.

The files *box.dcd* and *trj_unfolded.dcd* are only printed if periodic boundary conditions are used.

Note that in some cases the files *box.dcd*, *trj.dcd*, *trj_unfolded.dcd*, *vel.dcd* and *force.dcd* will be printed for each bead and will contain the bead number as part of their filename (e.g., *trj.001.dcd*).

In the case of dual potentials, the file *force.dcd* will be replaced by *force_low.dcd* and *force_high.dcd*, which contain the forces of the low- and high-accuracy potentials, respectively.

9.3.5 <iprint_xsf>

This keyword sets the print interval of the trajectory file in xsf format, *trj.xsf*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.3.6 <iprint_xyz>

This keyword sets the print interval of the trajectory file in xyz format, *trj.xyz*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.3.7 <ncolor>

Used in the cases of <method> = MD, PIMD, CMD, MTD. This keyword is only necessary when one wants to control the temperature very strongly using massive Nosé-Hoover chain thermostats. Nosé-Hoover chain thermostats are multiply attached with different masses; see Section 11.10.3 for details. This keyword sets the multiplicity *M*. The data should be provided in the first line (integer). The default value is “1” (single).

9.3.8 <nnhc>

Used in the cases of <method> = MD, PIMD, CMD, MTD. This keyword sets the chain length of the thermostat. The data should be provided in the first line (integer). The default value is “4”. This keyword is neglected if a thermostat is not used.

9.3.9 <np_beads>

Number of beads computed at the same time. The default is -1 (automatically defined). The product of *np_beads* and *np_force* should be the total number of cores used in the code *pimd.mpi.x*.

9.3.10 <np_force>

Number of processors assigned to a bead.

The default is -1 (automatically defined). The product of *np_beads* and *np_force* should be the total number of cores used in the code *pimd.mpi.x*.

9.3.11 <nref>

Used in the cases of <method> = MD, PIMD, CMD, MTS, MFE, TFS, AFED. This sets the number of updates for fast-varying forces per unit step <dt>.

- For <method> = MD with <ensemble> = NVE or NVT: the number of updates for the harmonic forces of constraints per step <dt>.
- For <method> = PIMD and CMD: the number of updates for the harmonic forces in path integrals, i.e., the forces between the beads, per step <dt>.
- For <method> = MTS: the number of updates for the harmonic forces in path integrals, i.e., between the beads, per step in the multiple time-scale method, <dt>/<nmulti>.
- For <method> = MFE and TFS: the number of updates of the electronic-state coefficients per step <dt>.
- For <method> = AFED with <afed_type> = TAMD or AFED: the number of updates for the harmonic forces of constraints per step <dt>.

In principle, the accuracy can be improved, but more computation is required as the <nref> value becomes larger. The data should be provided in the first line (integer). The default value is “1”.

9.3.12 <nsymbol>

Used in ALL methods when <input_style> = NEW. Usually, this keyword is not used in the file *input.dat* by keeping the file *input_default.dat* unchanged. This keyword sets the atomic species followed by the atomic number, atomic mass in amu, and atomic radius in au. The atomic radius is not used in the *pimd.x* and *pimd.mpi.x* codes; it is used specifically in the *prep_liquid.x* code to set the cutoff distance when identifying the bond topology of a classical force field.

This keyword was slightly modified after version 2.1.

9.3.13 <nys>

Used in the cases of <method> = MD, PIMD, CMD, MTD. This keyword sets the number of updates of the NHC, NHCS, and MNHC thermostats per unit step <dt>. The accuracy is usually improved by increasing the number of updates for the thermostats. The data should be provided in the first line (integer).

- <nys> = 3 or 5: the Suzuki–Yoshida time-step decomposition method is employed.
- Other values of <nys>: the step size <dt> is divided equally into <dt>/<nys>.

The default value is “5”.

9.3.14 <pimd_command>

Used in the cases of <ipotential> = ONIOM, QMMM, ALCHEM, DUAL. This is the execution command for the PIMD code (serial version). This information is only required when the PIMD code is called recursively via system calls. The default is “pimd.x”.

9.3.15 <pressure>

Used in the cases of <ensemble> = NPH, NPT. The external isotropic pressure is read from the first line (real value) in megapascals. The default value is “0.101325” megapascals, which is equal to 1 atm.

9.3.16 <time_bath>

Used in the cases of <method> = MD, PIMD, CMD, MTD. This keyword controls the masses of thermostats. The thermostat masses are related to the characteristic time scale of the system. This keyword sets this time scale in femtoseconds. The data should be provided in the first line (real number). The default value is “10” femtoseconds.

9.4 Specific keywords in *input.dat*

9.4.1 <abinit_mp_exe_command>

Used in the case of <ipotential> = ABINIT-MP5. This keyword sets the execution command of the ABINIT-MP5 code (the system call version). The first line gives the ABINIT-MP5 execution command. The default is “abinitmp”.

9.4.2 <abinit_mp_input>

Used in the case of <ipotential> = ABINIT-MP, ABINIT-MP5. This keyword sets the input file of the ABINIT-MP code. The name of the input file should be provided in the first line. The default is “input.ajf”.

9.4.3 <abinit_mp_output>

Used in the case of <ipotential> = ABINIT-MP, ABINIT-MP5. This keyword sets the log file of the ABINIT-MP code. Three data items should be provided in the first line.

- The name of the log file (character).
- The option, “0” or “1” (integer).
 - If “0”, only the information of the first bead is printed.
 - If “1”, the information of all beads is printed.
- The print interval (integer) of the ABINIT-MP log file.

The default is “output.abi 0 1”, which means that the information of the first bead at every step is printed to the ABINIT-MP log file, *output.abi*.

9.4.4 <abinit_mp_reuse_mo>

Used in the case of <ipotential> = ABINIT-MP. This keyword controls the initial guess of monomer, dimer, and trimer molecular orbitals calculated in the ABINIT-MP code. Three integers should be provided in the first line.

- The option, “0” or “1” (integer).
 - If “0”, the guess of monomer molecular orbitals is built from scratch at each step.
 - If “1”, the monomer molecular orbitals of the last step is reused as the guess for those of the next step.
- The option, “0” or “1” (integer).
 - If “0”, the guess of dimer molecular orbitals is built from scratch at each step.
 - If “1”, the dimer molecular orbitals of the last step is reused as the guess for those of the next step.
- The option, “0” or “1” (integer).

- If “0”, the guess of trimer molecular orbitals is built from scratch at each step.
- If “1”, the trimer molecular orbitals of the last step is reused as the guess for those of the next step.

The default values are “1, 1, 0”, which means that the monomer and dimer molecular orbitals are reused while the trimer molecular orbitals are built from scratch.

NOTE: When the orbital information is reused (with “1”), the memory space should be sufficient to preserve the molecular orbitals. The trimer molecular orbitals require the largest memory size.

9.4.5 <alchem.dat_dir>

Used in the case of <method> = PIHMC with <irem_type> = HX. This keyword sets the input directories of alchemical potential which include input files for the two potentials specified by the keyword <alchem_potential>. Two character strings should be provided in the first line. The default values are “dat_1” and “dat_2”.

9.4.6 <alchem_potential>

Used in the case of <method> = PIHMC with <irem_type> = HX. This keyword sets the two potentials of alchemistry. Two character strings should be provided in the first line. The default values are “MM” and “MM”.

9.4.7 <alchem_scr_dir>

Used in the case of <method> = PIHMC with <irem_type> = HX. This keyword sets the temporary scratch directories of alchemical potential. Two character strings should be provided in the first line. The default values are “scr_1” and “scr_2”.

9.4.8 <auto_string>

Used in the case of <method> = STRING. This keyword sets the option to control the step size in string method. When <auto_string>=OFF, the step size is a constant value set by the keyword <dt>. When <auto_string>=ON, the step size, initially set by the keyword <dt>, is controlled during the run. In this case the step size decreases or increases automatically at a given step interval by monitoring the change in the highest energy value among the images. The default value is “ON”.

9.4.9 <box_anis>

This option is used for the cases of <npt_type>, <nph_type>, <ntt_type>, <nth_type> = PPHEX, when <method> = PIMD with <ensemble> = NPT, NTT, NPH, NTH; when <method> = PIHMC with <ensemble> = NPT, NTT; and when <method> = REHMC with <ensemble> = NPT. It controls whether anisotropic volume changes of the simulation box are allowed. The default value is <box_anis>=ON (fully flexible). When <box_anis>=OFF, all off-diagonal elements of the simulation box matrix are frozen.

9.4.10 <cells_phonon>

Used in the case of <method> = PHONON. This keyword sets the unit cell within the box. In the first line, three integers, n_i , n_j and n_k are read, which correspond to the number of unit cells replicated in the \vec{a} , \vec{b} and \vec{c} directions, respectively. In other words, the unit cell is composed of the Bravais lattice vectors $\vec{i} = \vec{a}/n_i$, $\vec{j} = \vec{b}/n_j$ and $\vec{k} = \vec{c}/n_k$. The default value is “1, 1, 1” (the box is the unit cell).

9.4.11 <cluster>

Used in the case of <mech_type> = CLUSTER. This keyword sets the parameters of the external force applied to prevent molecular evaporation under free boundary conditions. In the first line, two real numbers should be provided, corresponding to the radius r_c (bohr) and the force constant k_c (hartree/bohr²) of the trapping barrier. The default values are not available.

9.4.12 <cp2k.lib.calcforce>

Used in the case of <ipotential> = CP2KLIB. This keyword sets how often the CP2K forces are evaluated. In most cases, this should keep the default value of 1, as the forces are evaluated at every step. However for Hybrid Monte Carlo (HMC) methods the force evaluation is not needed, and can be skipped for steps not providing training data where forces are necessary. If this is set to -1, then the force is never evaluated.

9.4.13 <cp2k.lib.chargeout>

Used in the case of <ipotential> = CP2KLIB. This keyword collects the Mulliken and Hirshfeld charges from CP2K. To output the charges set this keyword to 1 and set the `iprint_dcd` keyword as well. This will cause the file *charges.dcd* to be output, which contains both sets of charges.

9.4.14 <cp2k.lib.output>

Used in the case of <ipotential> = CP2KLIB. This keyword sets the amount of calculations which will provide an output.cp2k file. If the serial code (pimd.x) is executed, setting this to 0 or 1 will result in a single file being created in the 001 directory of the calculation, this file keeps track of all the CP2K output. In the case of the parallel code (pimd.mpi.x), setting this to 0 will result in only the results of the first bead being printed to the directory 001. On the other hand, setting this to 1 will result in the DFTB+ output for each bead to be stored in a directory with a number corresponding to the bead index. In both cases, setting this to any other integer results in all contents of output.dftb from CP2K being written to /dev/null, i.e., will never be written. The default is “0”.

9.4.15 <cut_rec_3d>

Used in the case of <method> = MTD. This keyword sets the cutoff parameter, x (real number), for the hill reconstruction in three-dimensional metadynamics. The data should be provided in the first line.

The Gaussian hills are added only to the mesh points where the contributions are larger than $\exp(-x)$. A value larger than “10.0” is recommended to suppress errors. For a value more than “37.0”, the cutoff is simply neglected. The default value is “100.0” (no cutoff).

9.4.16 <delay_aenet>

Used in the case of Self-Learning Hybrid Monte Carlo and related methods using AENET. This sets a delay in seconds, that should be input as a real number between double quotation marks. This delay is applied before all disk read and writes related to the learning process. The reason for this delay is to ensure that distributed file systems are properly synced before reading files for creating training sets, etc. If you experience problems with reading files from disk that do not seem to be missing on further inspection, it might help to extend this delay beyond the default value of 0.25.

9.4.17 <dftb_exe_command>

Used in the case of <ipotential> = DFTB. This keyword sets the execution of DFTB. The DFTB execution command should be provided in the first line. The default is “dftb+”.

9.4.18 <dftb_lib_output>

Used in the case of <ipotential> = DFTBLIB. This keyword sets the amount of calculations which will provide an output.dftb file. If the serial code (pimd.x) is executed, setting this to 0 or 1 will result in a single file being created in the 001 directory of the calculation, this file keeps track of all the DFTB+ output. In the case of the parallel code (pimd.mpi.x), setting this to 0 will result in only the results of the first bead being printed to the directory 001. On the other hand, setting this to 1 will result in the DFTB+ output for each bead to be stored in a directory with a number corresponding to the bead index. In both cases, setting this to any other integer results in all contents of output.dftb from DFTB+ being written to /dev/null, i.e., will never be written. The default is “0”.

9.4.19 <dftb_version>

Used in the case of <ipotential> = DFTB. This keyword sets the version of DFTB. The DFTB version should be provided in the first line, either “18.2” or “19.1”. The default is “19.1”.

9.4.20 <dir_save_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the save directory for the trained networks which are saved periodically through the setting the <istep_save_aenet> keyword. The default value is “saved_networks”

9.4.21 <dir_save_n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the save directory for the trained networks which are saved periodically through the setting the <istep_save_n2p2> keyword. The default value is “saved_networks”

9.4.22 <dir_save_mace>

Used in the case of <ipotential> = MACE. This keyword sets the save directory for trained snapshot models written periodically through the <istep_save_mace> keyword. The default value is “saved_models”.

9.4.23 <dir_train_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the scratch directory of trained networks in AENET potential. The data should be provided in the first line (character). The default value is “trained_networks”.

9.4.24 <dir_train_mace>

Used in the case of <ipotential> = MACE. This keyword sets the working directory used by the MACE interface for generated runtime and training files. Typical files created below this directory are *structures*, *mace_training.xyz*, *mace.ini*, *mace_count_1.out*, *mace_count_2.out*, *checkpoints*, and *mace_results*. The directory is created automatically by PIMD when needed. The data should be provided in the first line (character). The default value is “mace-work”.

9.4.25 <dosrange_phonon>

Used in the case of <method> = PHONON. This keyword sets the frequency range of the phonon density of states to be printed in the output file *phonon_dos.out*. Three real values should be provided in the first line designating the lower bound, the upper bound, and the increment frequencies in cm^{-1} . The default values are “0.0” cm^{-1} , “5000.0” cm^{-1} and “5.0” cm^{-1} , respectively.

9.4.26 <dt_conv_gad>

Used in the case of <method> = GAD. This keyword sets the GAD converged step size in hartree^{0.5} femtoseconds. The data should be provided in the first line (real number) in mass femtoseconds.

9.4.27 <dt_om>

Used in the case of <method> = OMOPT, TESTOM. This keyword sets the time increment of Onsager-Machlup action. The data should be provided in the first line (real number) in femtoseconds.

9.4.28 <dt_poly>

Used in the POLYMER code. This keyword sets the displacement of centroid images per update cycle. The data should be provided in the first line (real number). The default value is “0.04d0”.

9.4.29 <dtemp_meta>

Used in the case of <method> = MTD. This keyword sets the temperature parameter of well tempered metadynamics. A real number should be provided in the first line in kelvin. If the value is equal to or less than zero, well tempered metadynamics is switched off, and conventional metadynamics is chosen. The default value is “0.0” (conventional metadynamics).

9.4.30 <dual_dat_dir>

Used in the case of <method> = PIHMC with <ipotential> = DUAL. This keyword sets the potential of the DUAL method. Two character strings, corresponding to the high-level potential and the low-level potential, should be provided. In the second line, two character strings, corresponding to the input data directory of high-level potential and that of the low-level potential, should be provided. The default values are “dat_hi” and “dat_lo”.

9.4.31 <dual_potential>

Used in the case of <method> = PIHMC with <ipotential> = DUAL. This keyword sets the potential of the DUAL method. Two character strings, corresponding to the high-level and low-level potentials, should be provided. The default values are “MM” and “MM”.

9.4.32 <dual_scr_dir>

Used in the case of <method> = PIHMC with <ipotential> = DUAL. This keyword sets the potential of the DUAL method. Two character strings, corresponding to the high-level potential and the low-level potential, should be provided. In the second line, two character strings, corresponding to the temporary execution directories of high-level potential and that of the low-level potential, should be provided. The default values are “scr_hi” and “scr_lo”.

9.4.33 <efei>

Used in the case of <mech_type> = EFEI. This keyword sets the parameters of the EFEI method. In the first line, two integers and one real number should be provided, corresponding to the two atoms in which the external force is applied, and the strength of the force between those atoms in hartree/bohr. The positive (negative) value of the force means repulsive (attractive). The default values are not available.

9.4.34 <ends_poly>

Used in the POLYMER code. This keyword sets the option for end centroid images, either “FIXED” or “FREE”. The data should be provided in the first line (character). The default value is “FIXED”.

9.4.35 <ends_string>

Used in the case of <method> = STRING. This keyword sets the boundary condition of the string. The data should be provided in the first line (character).

- For <method> = STRING:
 - <ends_string> = FIXED: Both ends of the string are fixed.
 - <ends_string> = FREE: Both ends of the string are optimized.
 - <ends_string> = FREEFIXED: The reactant end is optimized while the product end is fixed.
 - <ends_string> = FIXEDFREE: The reactant end is fixed while the product end is optimized.

The default is “FIXED”.

9.4.36 <eps_best>

Used in the case of <ipotential> = QMMM and <ioption_best> = 1. This keyword sets the cut off energy of the BEST method in hartree. The data should be provided in the first line (real number). The default is “1.0e-16” hartree.

9.4.37 <equation_om>

Used in the case of <method> = OMOPT, TESTOM. This keyword sets the equation of motion of Onsager-Machlup action. The data should be provided in the first line (character).

- OVERDAMPED+: Overdamped Langevin equation.
- UNDERDAMPED+: Underdamped Langevin equation.

The default is “OVERDAMPED”.

9.4.38 <fc_best>

Used in the case of <ipotential> = QMMM and <ioption_best> = 1. This keyword sets the force constant of the BEST method in hartree/bohr². The data should be provided in the first line (real number). The default is “25.0” hartree/bohr².

9.4.39 <fdiff>

Used in the case of <method> = NMA, PHONON, TESTFORCE, TESTVIRIAL, OMOPT, TESTOM. This keyword sets the finite shifts (in bohr) to compute those numerical derivatives. The data should be provided in the first line (real number).

- For <method> = NMA, PHONON: the Hessian matrix is evaluated as the numerical derivative of forces with the finite geometric shifts, <fdiff>.
- For <method> = TESTFORCE: the force is evaluated as the numerical derivative of potential with the finite geometric shifts, <fdiff>.
- For <method> = TESTVIRIAL: the virial is evaluated as the numerical derivative of potential with the finite box shifts, <fdiff>.
- For <method> = OMOPT and TESTOM: the gradient of OM action is evaluated as the numerical derivative of forces with the maximum geometric shifts, <fdiff>.

The default value is “0.0001” bohr.

9.4.40 <g03_command>

Used in the case of <ipotential> = G03. This keyword sets the execution of G03. The first line gives the G03 execution command, and the second line gives the *formchk* execution command. The default is “g03” and “formchk”.

9.4.41 <g03_dip>

Used in the case of <ipotential> = G03. This keyword sets the option for G03 calculation of dipole moment. The data should be provided in the first line (integer). The dipole calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.42 <g03_grad>

Used in the case of <ipotential> = G03. This keyword sets the option for G03 calculation of potential gradient. The data should be provided in the first line (integer). The gradient calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.43 <g09_command>

Used in the case of <ipotential> = G09. This keyword sets the execution of G09. The first line gives the G09 execution command, and the second line gives the *formchk* execution command. The default is “g09” and “formchk”.

9.4.44 <g09_dip>

Used in the case of <ipotential> = G09. This keyword sets the option for G09 calculation of dipole moment. The data should be provided in the first line (integer). The dipole calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.45 <g09_grad>

Used in the case of <ipotential> = G09. This keyword sets the option for G09 calculation of potential gradient. The data should be provided in the first line (integer). The gradient calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.46 <g09_oniom>

Used in the case of <ipotential> = G09. This keyword sets the option for the use of ONIOM implemented in G09. The data should be provided in the first line (integer). The ONIOM is switched on if “1”, switched off if “0”. The default value is “0” (off).

9.4.47 <g16_command>

Used in the case of <ipotential> = G16. This keyword sets the execution of G16. The first line gives the G16 execution command, and the second line gives the *formchk* execution command. The default is “g16” and “formchk”.

9.4.48 <g16_dip>

Used in the case of <ipotential> = G16. This keyword sets the option for G16 calculation of dipole moment. The data should be provided in the first line (integer). The dipole calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.49 <g16_grad>

Used in the case of <ipotential> = G16. This keyword sets the option for G16 calculation of potential gradient. The data should be provided in the first line (integer). The gradient calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.50 <g16_oniom>

Used in the case of <ipotential> = G16. This keyword sets the option for the use of ONIOM implemented in G16. The data should be provided in the first line (integer). The ONIOM is switched on if “1”, switched off if “0”. The default value is “0” (off).

9.4.51 <g98_command>

Used in the case of <ipotential> = G98. This keyword sets the execution of G98. The first line gives the G98 execution command, and the second line gives the *formchk* execution command. The default is “g98” and “formchk”.

9.4.52 <g98_dip>

Used in the case of <ipotential> = G98. This keyword sets the option for G98 calculation of dipole moment. The data should be provided in the first line (integer). The dipole calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.53 <g98_grad>

Used in the case of <ipotential> = G98. This keyword sets the option for G98 calculation of potential gradient. The data should be provided in the first line (integer). The gradient calculation is switched on if “1”, switched off if “0”. The default value is “1” (on).

9.4.54 <gamess_command>

Used in the case of <ipotential> = GAMESS. This keyword sets the execution of GAMESS. The first line gives the GAMESS execution command. The default is “rungms gamess 00 1”.

9.4.55 <gamma_gad>

Used in the case of <method> = GAD. This keyword sets the gamma value of GAD, in atomic unit. The data should be provided in the first line. The default is “0.1”.

9.4.56 <gamma_om>

Used in the case of <method> = OMOPT, TESTOM. This keyword sets the friction constant of Onsager-Machlup action. The data should be provided in the first line (real number) in femtosecond⁻¹. The default is “0.00625” femtosecond⁻¹.

9.4.57 <generate_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the generate inputs in AENET potential. Following this keyword, all lines of the AENET training input should be copied, except the AENET keyword “FILES”, the number of xsf files, and xsf file names. The default values are not available.

9.4.58 <gh_meta>

Used in the case of <method> = MTD. This keyword sets the height of Gaussian hills of metadynamics simulation, in kelvin (energy divided by Boltzmann constant). The data should be provided in the first line (real number). In principle, this value should not be higher than the value specified by the keyword <temperature>. The default value is “150” kelvin.

9.4.59 <guess_poly>

Used in the POLYMER code. This keyword sets the initial guess of centroid images from the file *structure.dat*, either “LINE” (line interpolation of two ends) or “SPLINE” (spline interpolation of all images). The data should be provided in the first line (character). The default value is “LINE”.

9.4.60 <gw_meta>

Used in the case of <method> = MTD. This keyword sets the Gaussian width for each type of the collective variable (CV). Each line gives the CV type (integer) followed by the Gaussian width (real number). The default values are as follows.

```
DIST    0.100d0    ! width of bond distance [bohr]
ANGL    1.000d0    ! width of bond angle [deg]
DIH     1.000d0    ! width of dihedral angle [deg]
DIFF    0.100d0    ! width of bond difference [bohr]
CN       0.025d0    ! width of coordination number [no unit]
DCN     0.025d0    ! width of difference in coordination number [no unit]
XYZ     0.100d0    ! width of center of mass
DXYZ    0.100d0    ! width of difference in center of masses
```

Note that the width values are specified by the CV type, not the CV number. These values are shared for all the CV’s specified after the keyword <nmeta>. Therefore, a full set of data with five lines should always be given even if some of them are not used.

9.4.61 <iatom_qtst>

This option is used when <method> = PIMD and <ensemble> = QTST. The data should be provided on the first line as an integer. The default value is “0”. When the value is nonzero (*i*), the centroid of atom *i* is fixed at its initial position along the *Z* direction.

9.4.62 <iformat_trj>

This keyword sets the print option of *trj.out*. It could be chosen from below.

- iformat_trj = 1: trajectory is printed with 16 digits.
- iformat_trj = 2: trajectory is printed with 8 digits.

The default value is “0”.

9.4.63 <iformat_xyz>

This keyword sets the print option of *trj.xyz*. It could be chosen from below:

- iformat_xyz = 1: prints each bead configuration separately.
- iformat_xyz = 2: prints all beads into one configuration.
- iformat_xyz = 3: prints only the centroid configuration.

The data should be provided in the first line (integer). The default value is “2”.

9.4.64 <igamma>

Used in the case of <method> = CMD. This keyword controls the adiabaticity in centroid molecular dynamics. The data should be provided in the first line (integer). The accuracy of CMD will increase as the value is larger. Note that the adiabaticity parameter, γ , is given by $\gamma = 1/\langle\text{igamma}\rangle^2$. The default value is “10”.

9.4.65 <ikind_best>

Used in the case of <ipotential> = QMMM and <ioption_best> = 1. This keyword designates the atomic kind in the QM/MM BEST method. The default value is “1”.

9.4.66 <ikind_xyz>

Used in the case of <iprint_xyz> > 0. This keyword controls the print range of atomic kinds in the output trajectory file, *trj.xyz*. The minimum and maximum values of atomic kinds should be provided in the first line (two integers). For the maximum value of atomic kind, the value “-1” could be used to specify the number of atomic kinds present in the system. The default values are “1, -1” (all atomic kinds).

9.4.67 <integrator_bcmd>

Time integrator for Brownian chain molecular dynamics. The options are “EULER”, “GILLESPIE”, “PLATEN”, and “RESPA”. The default is “RESPA”.

9.4.68 <integrator_trpmd>

Time integrator for thermostatted ring polymer molecular dynamics. The options are “LEIMKUHLER” and “VANDEN-EIJNDEN”. The default is “VANDEN-EIJNDEN”.

9.4.69 <iobest>

Used in the case of <ipotential> = QMMM and <ioption_best> = 1. This keyword designates the atomic kind in the QM/MM BEST method. The data should be provided in the first line (integer). The default value is “1”.

9.4.70 <ioption_best>

Used in the case of <method> = QMMM. This keyword designates the usage of the QM/MM BEST method. The data should be provided in the first line (integer), either 0 (off: BEST method not used) or 1 (on: BEST method used). The default value is “0” (off).

9.4.71 <ioption_meta>

Used in the case of <method> = MTD. This keyword sets the option for multiple walker metadynamics. It could be chosen from below:

- `ioption_meta = 0`: the hills are added for all walkers.
- `ioption_meta = 1`: the hills are for all walkers that are well separated. One hill is added for a walker close to another walker.

The data should be provided in the first line (integer). The default value is “1”.

9.4.72 <ioption_xsf_aenet>

Used in the case of <ipotential> = AENET with <method> = PIHMC or <method> = REHMC. This keyword sets the option of printing xsf files for AENET potential. It could be chosen from below:

- `ioption_xsf_aenet = 0`: the xsf files are printed only for the accepted structures; the last accepted structure is copied when a trial structure is rejected.
- `ioption_xsf_aenet = 1`: the xsf files are printed for all the trial structures, both for the accepted and rejected structures.

The data should be provided in the first line (integer). The default value is “1”.

9.4.73 <ioption_xsf_n2p2>

Used in the case of <ipotential> = N2P2 with <method> = PIHMC or <method> = REHMC. This keyword sets the option of printing xsf files for N2P2 potential. It could be chosen from below:

- `ioption_xsf_n2p2 = 0`: the xsf files are printed only for the accepted structures; the last accepted structure is copied when a trial structure is rejected.
- `ioption_xsf_n2p2 = 1`: the xsf files are printed for all the trial structures, both for the accepted and rejected structures.

The data should be provided in the first line (integer). The default value is “1”.

9.4.74 <ioption_xsf_mace>

Used in the case of <ipotential> = MACE with <method> = PIHMC or <method> = REHMC. This keyword sets the option of printing xsf files for the MACE potential. It can be chosen from below:

- `ioption_xsf_mace = 0`: the xsf files are printed only for accepted structures; the last accepted structure is copied when a trial structure is rejected.
- `ioption_xsf_mace = 1`: the xsf files are printed for all trial structures, both accepted and rejected.

The data should be provided in the first line (integer). The default value is “1”.

9.4.75 <iorder_hmc>

Used in the case of <method> = PIHMC. This keyword sets the order of Suzuki-Trotter expansion, “2” or “4”.

- `iorder_hmc = 2`: the second order Suzuki-Trotter expansion.
- `iorder_hmc = 4`: the fourth order Suzuki-Trotter expansion.

The data should be provided in the first line. The default value is “2” (the second order).

9.4.76 <iprint_akin>

This keyword sets the print interval of kinetic energy to the output file *akin.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.77 <iprint_alc>

Used in the case of <method> = PIHMC with <irem_type> = HX. This keyword sets the print interval of the alchemical analysis to the output file *alc.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.78 <iprint.best>

Used in the case of <ipotential> = QMMM and <ioption.best> = 1. This keyword sets the print interval to the output file *dipole.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.79 <iprint.bond>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. This keyword sets the print interval of the bond lengths (interatomic distances) to the output file *dipole.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

NOTE: This analysis is not recommended when the number of atomic species is large, since the amount of computation grows quadratically.

9.4.80 <iprint.box>

Used in the cases of <method> = PIMD, PIHMC with <ensemble> = NPT, NTT, NPH, NTH. This keyword sets the print interval of the following data to the output file *box.out*.

- For <ensemble> = NPT, NPH, the print interval of the box matrix and the pressure matrix.
- For <ensemble> = NPT, NPH, the print interval of the box matrix, the pressure matrix, the stress matrix and the strain matrix.

The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.81 <iprint.cavg>

This keyword sets the print interval of heat capacity to the output file *cavg.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.82 <iprint.cons>

Used in the case of <method> = MD. This keyword sets the print interval of the results of constrained molecular dynamics to the output file *cons.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.83 <iprint.cv.meta>

Used in the case of <method> = MTD. This keyword sets the print interval of the trajectory of the collective variables to the output file *cv.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.84 <iprint.cv.tass>

Used in the case of <method> = TASS. The same as *iprint_cv_meta* used in <method> = MTD.

9.4.85 <iprint.dip>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. This keyword sets the print interval of the dipole moments to the output file *dipole.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.86 <iprint.dual>

Used in the case of <method> = PIHMC with <ipotential> = DUAL. This keyword sets the print interval of high and low-level potentials to the output file *dual.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.87 <iprint.eavg>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. This keyword print interval of energy averages to the output file *eavg.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.88 <iprint.hfx.aenet>

Used in the case of <ipotential> = AENET. This keyword sets the print interval of heat flux to the output file *hfx.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.89 <iprint.hfx.n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the print interval of heat flux to the output file *hfx.out*. The data should be provided in the first line (integer). The default value is “-1” (no print).

9.4.90 <iprint.mech>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. This keyword sets the print interval of energy components in the file *mech.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.91 <iprint.meta>

Used in the case of <method> = MTD. This keyword sets the print interval of energy components to the output file *meta.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.92 <iprint.mfe>

Used in the case of <method> = MFE. This keyword sets the print interval of the data of nonadiabatic mean field dynamics to the output file *mfe.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.93 <iprint.minfo>

Used in <method> = NMA. This prints the normal mode coordinates in a format that is readable by the Sindo program. The default value is “-1” (do not print). Set to “1” to print the results of normal mode analysis in the .minfo format.

9.4.94 <iprint.mom>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. The keyword sets the print interval of linear momentum and angular momentum to the output file *momentum.out*. The data should be provided in the first line (integer). The momentum conservation can be checked for MD and RPMD methods without thermostats. For the free boundary condition, both linear and angular momenta are conserved, while for the periodic boundary condition only the linear momentum is conserved. The default value is “-1” (do not print).

9.4.95 <iprint_nac>

Used in the case of <method> = MFE and TFS. This keyword sets the print interval of the data of nonadiabatic dynamics to the output file *nac.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.96 <iprint_oniom>

Used in the case of <ipotential> = ONIOM. The keyword sets the print interval of the potential energy compositions to the output file *oniom.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.97 <iprint_poly>

Used in the POLYMER code. This keyword sets the print interval of centroid mean force during the PIMD runs. The data should be provided in the first line (integer). The default value is “10”.

9.4.98 <iprint_qmmm>

Used in the case of <ipotential> = QMMM. The keyword sets the print interval of the potential energy compositions to the output file *qmmm.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.99 <iprint_rdf>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS. This keyword sets the print interval of the distribution of atomic pairs to the output file *rdf.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

NOTE: The meshes is given by the keyword <params_rdf>.

NOTE: This analysis is not recommended when the number of atomic species is large, since the amount of computation grows quadratically.

9.4.100 <iprint_rdfbead>

This keyword sets the print interval of the bead-wise distribution of atomic pairs to the output file *rdfcent.out*. The data should be provided in the first line (integer). The default value is “-1” (not printed).

NOTE: The meshes is given by the keyword <params_rdf>.

NOTE: This analysis is not recommended when the number of atomic species is large, since the amount of computation grows quadratically.

9.4.101 <iprint_rdfcent>

This keyword sets the print interval of the centroid-wise distribution of atomic pairs to the output file *rdfbead.out*. The data should be provided in the first line (integer). The default value is “-1” (not printed).

NOTE: The meshes is given by the keyword <params_rdf>.

NOTE: This analysis is not recommended when the number of atomic species is large, since the amount of computation grows quadratically.

9.4.102 <iprint_rec_meta>

Used in the case of <method> = MTD. This keyword sets the print interval of free energy in metadynamics to the output file *rec.out*. The data should be provided in the first line (integer). The default value is “100” (every 100 steps). Set to “-1” to switch off.

NOTE: The accuracy of reconstructed hills in three dimensional metadynamics is controlled by the keyword <cut_rec_3d>.

9.4.103 <iprint_rgy>

Used in ALL cases except <method> = STATIC, NMA, PHONON, MTD, STRING, MFE, TFS, REHMC. This keyword sets the print interval of the radius of gyration to the output file *rgy.out*. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.104 <iprint_str>

Used in the cases of <method> = STRING, OMOPT. This keyword sets the print intervals of the output files *string.xyz* and *string_pot.out*. The data should be provided in the first line (integer). The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.105 <iprint_tfs>

Used in the case of <method> = TFS. This keyword sets the print interval of the data of nonadiabatic surface hopping dynamics to the output file *tfs.out*. The default value is “1” (every step). Set to “-1” to switch off printing.

9.4.106 <iprint_xsf_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the print interval of xsf files which are used to train and upgrade the AENET potential. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.107 <iprint_xsf_n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the print interval of xsf files which are used to train and upgrade the N2P2 potential. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.108 <iprint_xsf_mace>

Used in the case of <ipotential> = MACE. This keyword sets the print interval of xsf files used to train and upgrade the MACE potential. The data should be provided in the first line (integer). The default value is “-1” (do not print).

9.4.109 <irandom>

Used in the ALL the cases. Sometimes it is convenient to use time-dependent random numbers. This is useful for the cases in which the initial geometry, the initial velocity, etc., are randomly generated for each run. The data in the first line (integer) should be chosen from below:

- <irandom> = 0: random numbers are time-independent, giving the same series of random numbers.
- <irandom> = 1: random numbers are time-dependent by generating the seed from date command.

The default value is “0”.

9.4.110 <irem_type>

Used in the case of <method> = REHMC. This keyword specifies the type of replica exchange hybrid Monte Carlo method. It could be chosen from the following options.

- <irem_type> = T: parallel temperatures without the replica exchange.
- <irem_type> = TX: temperature replica exchange.

- `<iREM_type> = HX`: Hamiltonian replica exchange.

The data (character) should be specified in the first line. The default value is “TX”.

9.4.111 `<istate_init>`

Used in the cases of `<method> = MFE` and `<method> = TFS`. This keyword sets the initial electronic state for the nonadiabatic dynamics. The default value is “1” (the ground state).

9.4.112 `<istep_adjust_hmc>>`

Used in the cases of `<method> = PIHMC` and `<method> = REHMC`. This keyword sets the step interval of changing the MD steps per Metropolis step. When it is “-1” the MD steps per Metropolis step is set constant. The default value is “-1”.

9.4.113 `<istep_ax_hmc>`

Used in the case of `<method> = REHMC`. This keyword sets the interval of atom exchange. The data should be provided in the first line (integer). The default value is “-1” (no exchange).

9.4.114 `<istep_hmc>`

Used in the cases of `<method> = PIHMC` and `<method> = REHMC`. This keyword sets the number of MD steps per Metropolis step in hybrid Monte Carlo. When it is set to be varied by `istep_adjust_hmc > 0`, this keyword sets the smallest value. The default value is “2” (every two steps).

9.4.115 `<istep_max_hmc>>`

Used in the cases of `<method> = PIHMC` and `<method> = REHMC` with `istep_adjust_hmc > 0`. This keyword sets the largest value of the interval of changing the MD steps per Metropolis step. The default value is “128”.

9.4.116 `<istep_mul_hmc>>`

Used in the cases of `<method> = PIHMC` and `<method> = REHMC` with `istep_adjust_hmc > 0`. This keyword sets the common ratio of geometric progression, respectively, for the MD steps per Metropolis step. The default value is “2”.

9.4.117 `<iPrint_rec_tass>`

Used in the case of `<method> = TASS`. The same as `iPrint_rec_meta` used in `<method> = MTD`.

9.4.118 `<istep_save_aenet>`

Used in the case of `<ipotential> = AENET`. This keyword sets the save frequency for the trained networks. If set to a positive integer value the trained networks will be saved in the directory set by `<dir_save_aenet>` keyword. The trained networks will have the suffix “_iter_XXXXXX”, where XXXXXX will be a number indicating which iteration the saved network stems from. The default value is “-1” (do not save intermediary networks).

9.4.119 `<istep_save_n2p2>`

Used in the case of `<ipotential> = N2P2`. This keyword sets the save frequency for the trained networks. If set to a positive integer value the trained networks will be saved in the directory set by `<dir_save_n2p2>` keyword. The trained networks will have the suffix “_iter_XXXXXX”, where XXXXXX will be a number indicating which iteration the saved network stems from. The default value is “-1” (do not save intermediary networks).

9.4.120 <istep_save_mace>

Used in the case of <ipotential> = MACE. This keyword sets the save frequency for trained MACE models. If set to a positive integer value, snapshot models are saved in the directory specified by <dir_save_mace>. The default value is “-1” (do not save intermediary models).

9.4.121 <iprint_tass>

Used in the case of <method> = TASS. The same as iprint_meta used in <method> = MTD.

9.4.122 <istep_train_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the integer i to set the step interval to train and upgrade the AENET potential. The data should be provided in the first line (integer). The default value is “-1” (do not train).

9.4.123 <istep_train_n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the integer i to set the step interval to train and upgrade the N2P2 potential. The data should be provided in the first line (integer). The default value is “-1” (do not train).

9.4.124 <istep_train_mace>

Used in the case of <ipotential> = MACE. This keyword sets the integer i to specify the step interval for training and upgrading the MACE potential. The data should be provided in the first line (integer). The default value is “0” (do not train).

9.4.125 <ivar_qmmm>

Used in the case of <ipotential> = QMMM in the periodic boundary condition. However, this is currently not supported.

9.4.126 <joption_meta>

Used in the case of <method> = MTD. This keyword sets the option for metadynamics. It could be chosen from below:

- joption_meta = 0: a new hill is added every time the walker has displaced for more than 1.5 times the width of the hill. Otherwise a new hill is added when the time limit is reached given by the keyword <time_limit_meta>.
- joption_meta = 1: a new hill is added at constant time interval given by the keyword <time_limit_meta>.

The data should be provided in the first line (integer). The default value is “0”.

9.4.127 <kdisp_phonon>

Used in <method> = PHONON. This keyword sets the k-points for the calculation of phonon dispersion curve. In the first line, the number of k-points n should be provided. Then, in each of the following n lines, three real numbers should be provided specifying the k_x , k_y and k_z values. The default value is “0” (not calculated).

9.4.128 <kdos_phonon>

Used in <method> = PHONON. This keyword sets the k-point sampling for the calculation of the vibrational density of states. Three integers should be provided. The default values are “1, 1, 1” (Γ point only).

9.4.129 <kickstep_bcmd>

Used in <method> = BCMD. This keyword sets the step interval of the random kick used in the improved integrator for BCMD simulations. One integer should be provided. The default value is “1” (every step).

The kick is applied every `kickstep_bcmd` steps in a similar manner to Andersen thermostats. However, the magnitude of the kick is controlled so as to obey the form of the overdamped Langevin equation as long as the `kickstep_bcmd` value is small enough. As `kickstep_bcmd` becomes larger, the kick becomes softer, and energy conservation is generally improved. To find a reasonable range of values for `kickstep_bcmd`, simply use the code “test.bcnd.F” provided in the tools directory.

9.4.130 <lstep_train_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the integer *i* to set the last step to train and upgrade the AENET potential. The data should be provided in the first line (integer). The default value is “99999999”.

9.4.131 <lstep_train_n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the integer *i* to set the last step to train and upgrade the N2P2 potential. The data should be provided in the first line (integer). The default value is “99999999”.

9.4.132 <lstep_train_mace>

Used in the case of <ipotential> = MACE. This keyword sets the last step for training and upgrading the MACE potential. The data should be provided in the first line (integer). The default value is “99999999”.

9.4.133 <mace_generate_command>

Used in the case of <ipotential> = MACE. This keyword sets an optional shell command executed before the MACE training script starts. The default value is blank, meaning that no extra command is executed.

9.4.134 <ase_infer_options_file>

Used in the case of <ipotential> = ASE. This keyword sets the YAML file used for ASE calculator inference. The file is resolved relative to the current run directory unless an absolute path is given. Relative paths written inside that YAML file are resolved relative to the directory containing the YAML file itself. The default value is “ase_infer_options.yaml”.

9.4.135 <mace_infer_options_file>

Used in the case of <ipotential> = MACE. This keyword sets the YAML file used for MACE inference. The file is resolved relative to the current run directory unless an absolute path is given. The model path specified by the `model` entry inside the YAML file is resolved relative to the directory containing that YAML file. The default value is “mace_infer_options.yaml”.

9.4.136 <mace_train_options_file>

Used in the case of <ipotential> = MACE. This keyword sets the YAML file used for MACE training. The file is resolved relative to the current run directory unless an absolute path is given. The path given by `foundation_model` inside the YAML file is resolved relative to the directory containing that YAML file. The default value is “mace_train_options.yaml”.

9.4.137 <mech.type>

This keyword sets the use of external forces. The data in the first line (character) should be chosen from below:

- NONE: no external force.
- EFEI: external force explicitly included.
- CLUSTER: external force explicitly included.
- ZONE: confined to a zone with half harmonic potentials.

If “EFEI” is chosen, the EFEI term

$$V_{\text{mech}} = -Fr_{ij}, \quad (4)$$

is added to the potential. In this case, three parameters, atom i , atom j , and the force amplitude F in au (hartree/bohr), should be provided by the keyword <efeI>.

If “CLUSTER” is chosen, external force is applied to prevent from molecular evaporation under free boundary condition. This option could be used for molecular clusters. A trapping barrier

$$V_{\text{mech}} = \frac{1}{2} \sum_{i=1}^n k_c \{ \max(0, r_{i,g} - r_c) \}^2 \quad (5)$$

is added to the potential, where

$$r_{i,g} = |\mathbf{r}_i - \mathbf{r}_g| \quad (6)$$

is the distance of the i -th atom from the center of mass of the molecular system,

$$\frac{\sum_{j=1}^n m_j \mathbf{r}_j}{\sum_{j=1}^n m_j}. \quad (7)$$

In this case, two parameters, the radius r_c (bohr) and the force constant k_c (hartree/bohr²) of the trapping barrier should be provided by the keyword <cluster>.

If “ZONE” is chosen, external force is applied so as to confine the set of collective variables, $\{q\}$, within a given zone. We use the sum of half-harmonic potentials with respect to the outward shift from the zone of each collective variable, Δq_α , namely,

$$V_{\text{mech}} = \frac{1}{2} \sum_{\alpha} k_{\alpha} \Delta q_{\alpha}^2 \quad (8)$$

where k_{α} is the force constant and

$$\Delta q_{\alpha} = \min \left(\max(0, q_{\alpha} - q_{\alpha}^{\max}), q_{\alpha} - q_{\alpha}^{\min} \right), \quad (9)$$

and q_{α}^{\min} and q_{α}^{\max} are the minimum and maximum values, respectively, of the zone without external forces. In this case, the parameters are provided by the keyword <zone>. In the first line, number of collective variables, n , is specified. The following n lines should contain the parameters of collective variable, either “DIST” or “CN”, in the following way:

- DIST \$1 \$2 \$3 \$4 \$5
... distance of atoms \$1 and \$2
is confined in a zone with the lower bound \$3
and the upper bound \$4 (in bohrs). A half-harmonic
potential with force constant \$5 is applied outside
the zone.

- CN \$1 \$2 \$3 \$4 \$5 \$6 \$7 \$8
 ... coordination number of species \$1 and \$2 given
 as a rational function (nu, mu, req = \$3, \$4, \$5)
 is confined in a zone with the lower bound \$6
 and the upper bound \$7. A half-harmonic potential
 with force constant \$8 is applied outside the zone.

The default value is “NONE”.

9.4.138 <mg_meta>

Used in the case of <method> = MTD. This keyword sets maximum number of Gaussian hills in metadynamics. The data should be provided in the first line (integer). This is necessary in the code in order to control the memory size. The default value is “50000”.

9.4.139 <minxsf_train_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the minimum number of xsf structure files required to train the artificial neural network potential. The data should be provided in the first line (integer). The default value is 100.

9.4.140 <minxsf_train_n2p2>

Used in the case of <ipotential> = N2P2. This keyword sets the minimum number of xsf structure files required to train the artificial neural network potential. The data should be provided in the first line (integer). The default value is 300.

9.4.141 <minxsf_train_mace>

Used in the case of <ipotential> = MACE. This keyword sets the minimum number of xsf structure files required to start MACE training. The data should be provided in the first line (integer). The default value is 0.

9.4.142 <model_water>

Used in the case of <ipotential> = WATER. This keyword sets the water potential. The data should be provided in the first line (character). Choose one from below:

- <model_water> = 1: flexible SPC potential [43].
- <model_water> = 2: Reimers-Watts-Klein potential [44].

The default value is “1” (flexible SPC).

NOTE: When using these water potentials, the atomic order must be “O, H, H, O, H, H, ...”.

NOTE: Since the ewald sum has not been implemented for these water potentials, it could be used safely under the free boundary conditions only.

9.4.143 <molpro_command>

Used in the case of <ipotential> = MOLPRO. This keyword sets the execution command of MOLPRO. The data (character) should be provided in the first line. The default is “molpro”.

9.4.144 <mopac_command>

Used in the case of <ipotential> = MOPAC. This keyword sets the execution of MOPAC. The first line gives the MOPAC execution command. The default is “nohup /opt/mopac/mopac/MOPAC2007.out”

9.4.145 <n2p2_en_unit>

Used for <ipotential> = N2P2. This keyword sets the energy unit received from the N2P2 MLP. The options are: HARTREE (default), EV (electronic volts) or a floating point number indicating the conversion factor from N2P2 units to HARTREE.

9.4.146 <n2p2_len_unit>

Used for <ipotential> = N2P2. This keyword sets the length unit for the coordinates sent to the N2P2 MLP. The options are: BOHR (default), AA (Aangstroms) or a floating point number indicating the conversion factor from N2P2 units to BOHR.

9.4.147 <ncons>

Used in the case of <method> = MD, AFED. This keyword sets the number of constraints followed by the parameters of each constraint. In line 1, the number constraints, n (integer), is given. In lines 2- $(n+1)$, the format depends on the type of constraints. The format is as follows:

- type 1: bond distance of atoms #1-#2.
"1, atom #1, atom #2, minimum distance [bohr], maximum distance [bohr]".
- type 2: bond angle of atoms #1-#2-#3.
"2, atom #1, atom #2, atom #3, minimum angle [degrees], maximum angle [degrees]".
- type 3: dihedral angle of atoms #1-#2-#3-#4.
"3, atom #1, atom #2, atom #3, atom #4, minimum angle [degrees], maximum angle [degrees]".
- type 4: bond difference of atoms #1-#2 and atoms #2-#3.
"4, atom #1, atom #2, atom #3, minimum value [bohr], maximum value [bohr]".
- type 5: coordination number of species #1 surrounded by species #2.
"5, spec #1, spec #2, μ , ν , equilibrium distance [bohr], minimum coordination number, maximum coordination number".
- type 6: difference in coordination numbers for species #1 surrounded by species #2 and #3 surrounded by #4.
"6, spec #1, spec #2, μ , ν , equilibrium distance [bohr], #3, spec #4, μ , ν , equilibrium distance [bohr], minimum difference in coordination numbers, maximum difference in coordination numbers".
- type 7: center of mass of species #2.
"7, xyz component (1-3), spec #1, minimum value [bohr], maximum value [bohr]".
- type 8: difference in center of masses of species #2 and #3.
"8, xyz component (1-3), spec #1, spec #2, minimum value [bohr], maximum value [bohr]".

The constraint types, 1-8, can be replaced by "DIST", "ANGL", "DIH", "DIFF", "CN", "DCN", "XYZ" and "DXYZ", respectively. For the definition of coordination number, see Section 11.14.1.

9.4.148 <ncycle.poly>

Used in the POLYMER code. This keyword sets the number of update cycles of centroid images by the string method. The data should be provided in the first line (integer). The default value is "1".

9.4.149 <neighbor_list_skin>

Used for <ipotential> = N2P2. This sets the neighbor list skin distances for N2P2 calculations. The units used for this is BOHR, regardless of the length units used by the N2P2 MLP. The neighbor list radius is automatically deduced from the N2P2 input file, and this skin is added to avoid recalculating the neighbor list in each step.

9.4.150 <ngrid_poly>

Used in the POLYMER code. This keyword sets finer grids to compute free energy between the images by thermodynamic integration. The data should be provided in the first line (character). The default value is 10.

9.4.151 <ngrid_string>

Used in the case of <method> = STRING. This keyword sets the number of fine grids to interpolate between the images of the string. The default value is "11".

9.4.152 <nmeta>

Used in the case of <method> = MTD. This keyword sets the dimension of collective variables in metadynamics, and the parameters necessary for each collective variable.

- LINE 1: the dimension of collective variables n .
- LINE 2: the parameter set of 1st collective variable.
- LINE 3: the parameter set of 2nd collective variable.
- LINE ($n + 1$): the parameter set of n -th collective variable.

In lines 2-($n + 1$), the format depends on the type of collective variables. The format is as follows:

- type 1: bond distance of atoms #1-#2.
"1, atom #1, atom #2".
- type 2: bond angle of atoms #1-#2-#3.
"2, atom #1, atom #2, atom #3".
- type 3: dihedral angle of atoms #1-#2-#3-#4.
"3, atom #1, atom #2, atom #3, atom #4".
- type 4: bond difference of atoms #1-#2 and atoms #2-#3.
"4, atom #1, atom #2, atom #3"
- type 5: coordination number of species #1 surrounded by species #2.
"5, spec #1, spec #2, μ , ν , distance [bohr]"
- type 6: difference coordination numbers of species #1 surrounded by species #2 and #3 surrounded by species #4.
"6, spec #1, spec #2, μ , ν , distance [bohr], spec #3, spec #4, μ , ν , distance [bohr]"
- type 7: center of mass of species #2.
"7, xyz component (1-3), spec #1".
- type 8: difference in center of masses of species #2 and #3.
"8, xyz component (1-3), spec #1, spec #2".

The collective variable types, 1–8, can be replaced by “DIST”, “ANGL”, “DIH”, “DIFF”, “CN”, “DCN”, “XYZ” and “DXYZ”, respectively. For the definition of coordination number, see Section 11.14.1.

From version 2.1.1, it is possible to set the boundary to each collective variables. Add two numbers, i.e., the minimum value and the maximum value, at the end of each line. If these values are not set, the free boundary is assumed.

9.4.153 <nmulti>

Used in the case of <method> = MTS. This keyword controls the time increment in the multiple time scale molecular dynamics and path integral molecular dynamics using the ONIOM or QM/MM potentials. In the MTS method, the time step for updating the forces with respect to the layer A is given by <dt>. Meanwhile, the time step for updating the forces with respect to the layer B, and the interaction between the layers A and B, is set to be <dt>/<nmulti>. Thus, the parameter <nmulti> is the number of updates with respect to the changes in the layer B. In addition, the atomic masses in the layer B are set smaller than the reality, by the scaled factor of $1/\text{<nmulti>}^2$. The default value is “100”.

9.4.154 <np_poly>

Used in the POLYMER code. This keyword sets the parallel number with respect to centroid images. The data should be provided in the first line (integer). The default value is “1”.

9.4.155 <nph_type>

Used in the case of <ensemble> = NPH. This keyword sets the box shape of the NPH ensemble.

- For <ensemble> = NPH:
 - <nph_type> = CUBIC1: Andersen’s isobaric molecular dynamics for a cubic box.
 - <nph_type> = CUBIC2: Martyna’s isobaric molecular dynamics for a cubic box.
 - <nph_type> = PPHEX: Parrinello-Rahman’s isobaric molecular dynamics for a parallel-piped hexahedron box.

The default value is “CUBIC2”.

9.4.156 <npoly>

Used in the POLYMER code. This keyword sets the the number of centroid images. The data should be provided in the first line (integer). The default value is “16”.

9.4.157 <npt_type>

Used in the case of <ensemble> = NPT. This keyword sets the box shape of the NPT ensemble.

- For <ensemble> = NPT:
 - <npt_type> = CUBIC1: Andersen’s isobaric molecular dynamics for a cubic box.
 - <npt_type> = CUBIC2: Martyna’s isobaric molecular dynamics for a cubic box.
 - <npt_type> = PPHEX: Parrinello-Rahman’s isobaric molecular dynamics for a parallel-piped hexahedron box.

The default value is “CUBIC2”.

9.4.158 <nref_meta>

Used in the case of <method> = MTD. This keyword controls the reference step size of harmonic forces between collective variable and fictitious particle in metadynamics. The data should be provided in the first line (integer). The default value is “100”.

NOTE: The harmonic force is controlled by the keyword <time_fc_meta>.

9.4.159 <nref_tass>

Used in the case of <method> = TASS. The same as nref_meta used in <method> = MTD.

9.4.160 <nstep_modes>

Used in the case of <method> = NMA. This keyword sets the number of steps for the normal mode vibration printed in “modes/mode.*.xyz”. The default value is 0 (do not print).

9.4.161 <nstate>

Used in the cases of <method> = TFS and <method> = MFE. This keyword sets the number of electronic states in nonadiabatic dynamics. The default value is “1”.

NOTE: If <nstate> = 1, both TFS and MFE methods are equal to the standard (Born-Oppenheimer) MD of the NVE ensemble.

9.4.162 <ntass_rec>

Used in the case of <method> = TASS.

- line 1: CV dimension for free energy reconstruction (n)
- line 2: The n CVs for free energy reconstruction

9.4.163 <nth_type>

Used in the case of <ensemble> = NTH. This keyword sets the box type of NtH ensemble.

- For <ensemble> = NTH:
 - <nth_type> = CUBIC1: Andersen’s isobaric molecular dynamics for a cubic box.
 - <nth_type> = CUBIC2: Martyna’s isobaric molecular dynamics for a cubic box.
 - <nth_type> = PPHEX: Parrinello-Rahman’s isobaric molecular dynamics for a parallel-piped hexahedron box.

The default value is “PPHEX”.

9.4.164 <ntt_type>

Used in the case of <ensemble> = NTT. This keyword sets the box type of NtT ensemble.

- For <ensemble> = NTT:
 - <ntt_type> = CUBIC1: Andersen’s isobaric molecular dynamics for a cubic box.
 - <ntt_type> = CUBIC2: Martyna’s isobaric molecular dynamics for a cubic box.
 - <ntt_type> = PPHEX: Parrinello-Rahman’s isobaric molecular dynamics for a parallel-piped hexahedron box.

The default value is “PPHEX”.

9.4.165 <ntype.aenet>

Used in the case of <ipotential> = AENET. This keyword sets the atomic types used in the artificial neural network potential. The data in the first line (integer) should be the number of atomic types, n . The next n lines should be the atomic type, followed by the artificial neural network potential file of the atomic type. The default value is not available.

9.4.166 <oniom.dat_dir>

Used in the case of <ipotential> = ONIOM. This keyword sets the input directories of ONIOM method. Three data (character strings) should be provided in the first line, corresponding to the directory names which include the input files for the high-level calculations of subsystem A, for the low-level calculations of subsystem A, and for the low-level calculations of the system A+B. The default value is "dat_1, dat_2, dat_3".

9.4.167 <oniom.hi_potential>

Used in the case of <ipotential> = ONIOM. This keyword sets the high-level potential of ONIOM method. A data (character) should be provided in the first line, corresponding to the high-level potential. The default value is "SMASH".

9.4.168 <oniom.lo_potential>

Used in the case of <ipotential> = ONIOM. This keyword sets the low-level potential of ONIOM method. A data (character) should be provided in the first line, corresponding to the low-level potential. The default value is "MM".

9.4.169 <oniom.scr_dir>

Used in the case of <ipotential> = ONIOM. This keyword sets the temporary scratch directories of ONIOM method. Three data (character strings) should be provided in the first line, corresponding to the directory names which include the scratch files for the high-level calculations of subsystem A, for the low-level calculations of subsystem A, and for the low-level calculations of the system A+B. The default value is "scr_1, scr_2, scr_3".

9.4.170 <orca_command>

Used in the case of <ipotential> = ORCA. This keyword sets the execution command of ORCA. The data (character) should be provided in the first line. The default is "orca".

9.4.171 <params.cons>

Used in the cases of <method> = MD with ncons=1, and <method> = AFED. This keyword sets the force constants of the constraints. For each line, the type of constraint is followed by the force constant. The default values are as follows:

DIST	1.d0	! force constant of bond distance [hartree/bohr**2]
ANGL	4.d-4	! force constant of bond angle [hartree/degree**2]
DIH	1.d-4	! force constant of bond dihedral [hartree/degree**2]
DIFF	1.d0	! force constant of bond difference [hartree/bohr**2]
CN	1.d0	! force constant of coordination number [hartree]
DCN	1.d0	! force constant of difference in coordination numbers [hartree]
XYZ	1.d0	! force constant of center of mass [hartree/bohr**2]
DXYZ	1.d0	! force constant of difference in center of masses [hartree/bohr**2]

Note that the values are specified for the type of constraint, not the constraint itself. These values are shared for the constraints specified after the keyword `<ncons>`. Therefore, the full set of data with five lines should always be given even if only some of them are used.

9.4.172 `<params_dual>`

This keyword sets the mixing weight parameter for self-learning hybrid Monte Carlo (SLHMC-MIX). Two real numbers should be provided in the first line. The first one is the weight of high-level potential. If it is 1.0 (0.0), trial moves are accepted by purely high-level (low-level) potential. This can be tuned between 0.0 and 1.0 for higher efficiency. The second number is a parameter for safety. If it is larger than 0.0, the move is rejected when the correction is too large. The default is “1.0, 0.0” (high-level potential, with no correction).

9.4.173 `<params_lbfgs>`

Used in the cases of `<method>` = GEOOPT, BOXOPT, FULLOPT. This keyword sets the convergence criteria in the limited memory BFGS method. The data (four real numbers) should be provided in the first line. The four numbers correspond to the maximum displacement in au (bohr), the root-mean-square displacement in au (bohr), the maximum force in au (hartree/bohr), and the root-mean-square force in au (hartree/bohr), respectively.

9.4.174 `<params_rdf>`

Used in the cases when `<iprint_rdf>` is positive. This keyword sets the meshes of the atomic pair distribution. Three real numbers should be provided in the first line, i.e., the minimum value, the maximum value, and the mesh size in au (bohr). The default values are “1.0, 20.0, 0.01”.

9.4.175 `<params_rec_meta>`

Used in the case of `<method>` = MTD. This keyword sets the parameters necessary for hills reconstruction in metadynamics. For each line, the CV type is followed by three real numbers, i.e., minimum, maximum, and mesh size of the reconstructed hills. The default values are as follows:

DIST	2.0	5.0	0.1	!	min, max, mesh of bond distance [bohr]
ANGL	0.0	180.0	6.0	!	min, max, mesh of bond angle [deg]
DIH	0.0	360.0	12.0	!	min, max, mesh of dihedral angle [deg]
DIFF	-3.0	3.0	0.2	!	min, max, mesh of bond difference [bohr]
CN	0.0	3.0	0.1	!	min, max, mesh of coordination number [no unit]
DCN	0.0	3.0	0.1	!	min, max, mesh of difference in coordination numbers [no unit]
XYZ	2.0	5.0	0.1	!	min, max, mesh of center of mass [bohr]
DXYZ	2.0	5.0	0.1	!	min, max, mesh of difference in center of masses [bohr]

Note that the values are specified for the CV type, not the CV number. These values are shared for all the CV's specified after the keyword `<nmeta>`. Therefore, the full set of data with five lines should always be given even if only some of them are used.

9.4.176 `<phase0_proc>`

Used when `<ipotential>`=PHASE/0. This keyword sets the PHASE/0 settings for parallel computation. The three integers, `ne nk ng`, correspond to the number of band parallels, k point parallels, G point parallels, respectively. 1 should be specified as the number corresponding to `ng` when a two-dimensional version is used. Each value should be determined so that the MPI parallel number allocated to each replica by PIMD (`np_force`) is equal to the product of `ne nk ng`. The default is -1, 1, 1, in this case `ne` is adjusted to `np_force`, and `+nk=1+` and `+ng=1+`.

9.4.177 <projcmf_poly>

Used in the POLYMER code. This keyword sets the projection of mean forces applied to centroid images, either “ON” (projection applied) or “OFF” (not applied). The data should be provided in the first line (character). The default value is “OFF”.

9.4.178 <qe_input_file_name>

Used in the case of <ipotential> = QE. This keyword sets the input file name of the QE code. The data should be provided in the first line.

The default value is “qe.dat”.

9.4.179 <qe_output>

Used in the case of <ipotential> = QE. This keyword sets the log file of the QE code. The following two data should be provided in the first line.

- The option, “0” or “1”. If “0”, the log file of only the first bead is printed. If “1”, the log files of all the beads are printed.
- The print interval n (integer). The log file is printed every n steps.

The default values are “0, 1”, which means that the first bead is printed every step.

9.4.180 <qmmm_embedding>

Used in the case of <ipotential> = QMMM. This keyword sets the embedding type of electrostatic interaction in QM/MM method. A data (character) should be provided in the first line, either “ME” (mechanical embedding) or “EE” (electronic embedding). The default value is “EE”.

9.4.181 <qmmm_dat_dir>

Used in the case of <ipotential> = QMMM and <qmmm_embedding> = ME. This keyword sets the input directory of QMMM method. A data (character) should be provided in the first line, corresponding to the directory name which includes the input files for the QM calculations of subsystem A. The default value is “dat”.

9.4.182 <qmmm_potential>

Used in the case of <ipotential> = QMMM. This keyword sets the QM potential in QM/MM method. A data (character) should be provided in the first line. The default value is “SMASH”.

9.4.183 <qmmm_scr_dir>

Used in the case of <ipotential> = QMMM and <qmmm_embedding> = ME. This keyword sets the temporary scratch directory of QMMM method. A data (character) should be provided in the first line, corresponding to the directory name which includes the scratch files for the QM calculations of subsystem A. The default value is “scr”.

9.4.184 <ratio_max_hmc>>

Used in the cases of <method> = PIHMC and <method> = REHMC with `istep_adjust_hmc` > 0. This keyword sets the lower bound of the target acceptance ratio. The MD steps per Metropolis step is increased if the acceptance ratio is higher than this value. The default value is “0.20”.

geometric progression, respectively, for the MD steps per Metropolis step.

9.4.185 <ratio_min_hmc>>

Used in the cases of <method> = PIHMC and <method> = REHMC with `istep_adjust_hmc` > 0. This keyword sets the lower bound of the target acceptance ratio. The MD steps per Metropolis step is decreased if the acceptance ratio is lower than this value. The default value is “0.05”.

9.4.186 <scale_bcmd>

A scaling factor for the friction constant in BCMD. The default is 1.0 (not scaled).

9.4.187 <scale_trpmd>

A scaling factor for the friction constant in TRPMD. The default is 1.0 (not scaled).

9.4.188 <scan_exe_shell>

Used in the case of <method> = SCAN. This keyword sets the shell command executed for each structure in the SCAN method. A data (character) should be provided in the first line, corresponding to the part of the file name of shell command (*name*). The file name is *name.integer.sh*. The *integer* is the bead number with three digits. The default value is “./scan”. By default (<nbead> = 1) the file “./scan.001.sh” is executed for each structure. Otherwise, for instance, <nbead> = 4, the files “./scan.001.sh” “./scan.002.sh” “./scan.003.sh” and “./scan.004.sh” are executed for bead 1, 2, 3 and 4, respectively.

9.4.189 <skin_aenet>

Used in the case of <ipotential> = AENET. This keyword sets the skin value of neighbor list in AENET. A real value should be provided in the first line in angstroms. The default value is 1.0 angstroms.

9.4.190 <smash_command>

Used in the case of <ipotential> = SMASH. This keyword sets the execution command of SMASH.

NOTE: This option has an effect only when SMASH is linked externally with PIMD.

9.4.191 <smash_dip>

Used in the case of <ipotential> = SMASH. This keyword sets the computation of dipole moment for the SMASH code, either “1” (on) or “0” (off). The default value is “1” (on).

NOTE: This option has an effect only when the SMASH code is linked externally with the PIMD code.

NOTE: This option is obsolete from version 2.2.0.

9.4.192 <smash_grad>

Used in the case of <ipotential> = SMASH. This keyword sets the computation of forces for the SMASH code, either “1” (on) or “0” (off). The default value is “1” (on).

NOTE: This option has an effect only when the SMASH code is linked externally with the PIMD code.

9.4.193 <smash_guess>

Used in the case of <ipotential> = SMASH. This keyword gives the option for the initial guess of molecular orbital coefficients.

- <smash_guess> = PREVIOUS: molecular orbital coefficients of the last step is used as the initial guess.
- <smash_guess> = PULAY: extrapolation of the molecular orbital coefficients of the last four steps are used as the initial guess using Pulay’s parameter set [18].

- `<smash_guess>` = KOLAFa, extrapolation of the molecular orbital coefficients of the last six steps are used as the initial guess using Kolafa's parameter set [19].

The default value is PREVIOUS.

9.4.194 `<smash_options>`

Used in the case of `<ipotential>` = SMASH. This keyword sets the options of SMASH standard input. The lines starting from 'job', 'control', etc. should be specified. The information of basis sets and electron core potentials could be included when necessary. Exceptionally, the lines 'geom' and the nuclear coordinates are not necessary.

9.4.195 `<smash_threads>`

Used in the case of `<ipotential>` = SMASH. This keyword sets the number of OPENMP threads used for the SMASH code. The default value is "1".

NOTE: This option has an effect only when the SMASH code is linked externally with the PIMD code.

9.4.196 `<temp_tamd_tass>`

Scaled temperature in TASS. If this is negative, it is set to the physical temperature set by the keyword `<temperature>`. The default is -1 (physical temperature).

9.4.197 `<temprange_phonon>`

Used in the case of `<method>` = PHONON. This keyword sets the temperature range of phonon energy calculation. Three real numbers should be provided in the first line, i.e., the minimum value, the maximum value, and the increment in kelvin. The default values are "0.0, 3000.0, 10.0".

9.4.198 `<temprange_rem>`

Used in the case of `<method>` = PHONON. This keyword sets the temperature range of replica exchange hybrid Monte Carlo. From `<nbead>` and `<temprange_rem>`, the temperature T_i of each replica is set automatically by the equation $T_i = (T_n/T_1)^{\frac{i-1}{n-1}}T_1$, where n is the number of replicas specified in the keyword `<nbead>`. The default values are "300.0, 1000.0".

9.4.199 `<tension>`

Used in the cases of `<ensemble>` = NTH, NTT. This keyword sets the tension matrix.

$$\begin{matrix} t_{xx}, & t_{xy}, & t_{xz} \\ t_{xy}, & t_{yy}, & t_{yz} \\ t_{xz}, & t_{yz}, & t_{zz} \end{matrix}$$

The default values are nine zeros (zero tension).

NOTE: the tension matrix must be real and symmetric.

9.4.200 `<time_baro>`

Used in the cases of `<ensemble>` = NPT, NTT, NPH, NTH. This keyword controls the mass of barostat. The mass of barostat is related to the characteristic time scale of the system. This keyword sets this time scale in femtoseconds. The data should be provided in the first line (real number). The default value is "1000.0" femtoseconds.

9.4.201 <time_cv_bath>

Used in the case of <method> = MTD. This keyword sets the characteristic time scale of thermostat attached to fictitious particle in femtoseconds. The data should be provided in the first line (real number). The default value is “1000.0” femtoseconds.

9.4.202 <time_cv_meta>

Used in the case of <method> = MTD. This keyword sets characteristic time scale, in femtoseconds, for the fictitious particle to travel 1.5 times of the Gaussian hill width, It is therefore the characteristic time interval of adding Gaussian hills. By controlling the mass of the fictitious particle, the adiabatic separation between the collective variables (slow) and fast all other degrees of freedom (fast) is controlled. A better adiabaticity and thus higher accuracy should be accomplished as the <time_cv_meta> value is larger. The data should be provided in the first line (real number). The default value is “75.0” femtoseconds.

9.4.203 <time_fc_meta>

Used in the case of <method> = MTD. This keyword sets time scale of oscillation, in femtoseconds, that controls the the strength of harmonic force between collective variable and fictitious particle. The value should be large enough so that harmonic interaction is strong, keeping the collective variable and the position of fictitious particle close to each other. The data should be provided in the first line (real number). The default value is “30.0” femtoseconds.

NOTE: The time increment of updating of the harmonic force is controlled by the keyword <nref_meta>.

9.4.204 <time_limit_meta>

Used in the case of <method> = MTD. This keyword sets the time limit of adding a new hill in femtoseconds. It is recommended to set this value at least more than three times as large as the value of <time_cv_meta>. The data should be provided in the first line (real number). The default value is “225.0” femtoseconds, three times the default value of <time_cv_meta>.

9.4.205 <time_mode>

The time scale of fictitious non-centroid mass in fs. If this is negative, it is set to the standard value. The default is -1 (standard value).

9.4.206 <train.aenet>

Used in the case of <ipotential> = AENET. This keyword sets the train inputs in AENET potential. Following this keyword, all the lines of AENET train input should be copied, with all blank lines removed. The default values are not available.

9.4.207 <turbo.command>

Used in the case of <ipotential> = TURBOMOLE. This keyword sets the execution of TURBOMOLE. Two execution commands should be given to compute energy, forces, and dipole moment. The first line gives the first TURBOMOLE execution command followed by the name of the first *control* file. The second line gives the second TURBOMOLE execution command followed by the name of the second *control* file.

9.4.208 <turbo.guess>

Used in the case of <ipotential> = TURBOMOLE. This keyword gives the option for the initial guess of molecular orbital coefficients.

- <turbo.guess> = PREVIOUS: Molecular orbital coefficients of the last step is used as the initial guess.

- **<turbo_guess> = PULAY:** Extrapolation of the molecular orbital coefficients of the last four steps are used as the initial guess using Pulay's parameter set [18].
- **<turbo_guess> = KOLAF:** Extrapolation of the molecular orbital coefficients of the last six steps are used as the initial guess using Kolafa's parameter set [19].

The default value is **PREVIOUS**.

NOTE: For safe execution of TURBOMOLE, it is recommended to set this value to **PREVIOUS**.

9.4.209 <user_command>

Used in the case of **<ipotential> = USER**. This keyword gives the user execution command to calculate potential, forces, and optionally, virial, for a given atomic positions. The default value is not available.

9.4.210 <user_input_file>

Used in the case of **<ipotential> = USER**. This keyword gives the name of the input file of user execution command. The input file should be in the following format. For a single bead case,

```
x1  y1  z1  (Cartesian coordinates of atom 1 in bohr)
x2  y2  z2  (Cartesian coordinates of atom 2 in bohr)
..  ..  ..
..  ..  ..
xn  yn  yz  (Cartesian coordinates of atom n in bohr)
```

For a multiple bead case,

```
x1  y1  z1  (Cartesian coordinates of atom 1 of bead 1 in bohr)
x2  y2  z2  (Cartesian coordinates of atom 2 of bead 1 in bohr)
..  ..  ..
xn  yn  yn  (Cartesian coordinates of atom n of bead 1 in bohr)
x1  y1  z1  (Cartesian coordinates of atom 1 of bead 2 in bohr)
x2  y2  z2  (Cartesian coordinates of atom 2 of bead 2 in bohr)
..  ..  ..
xn  yn  yn  (Cartesian coordinates of atom n of bead 2 in bohr)
..  ..  ..
..  ..  ..
x1  y1  z1  (Cartesian coordinates of atom 1 of bead m in bohr)
x2  y2  z2  (Cartesian coordinates of atom 2 of bead m in bohr)
..  ..  ..
xn  yn  yn  (Cartesian coordinates of atom n of bead m in bohr)
```

The default value is not available.

9.4.211 <user_output_file>

Used in the case of **<ipotential> = USER**. This keyword gives the name of the output file of user execution command. The output file should be in the following format. For a single bead case,

```
pot          (potential)
fx1 fy1 fz1  (force of atom 1 in bohr)
fx2 fy2 fz2  (force of atom 2 in bohr)
...  ...  ...
fxn fyn fyn  (forces of atom n in hartree/bohr)
v11 v12 v13  (virial matrix in au, optional)
v21 v22 v23  (virial matrix in au, optional)
v31 v32 v33  (virial matrix in au, optional)
```

For a multiple bead case,

```

pot                (potential of bead 1)
fx1  fy1  fz1      (force of atom 1 of bead 1 in bohr)
fx2  fy2  fz2      (force of atom 2 of bead 1 in bohr)
...   ...   ...
fxn  fyn  fyn      (force of atom n of bead 1 in hartree/bohr)
fx1  fy1  fz1      (force of atom 1 of bead 2 in hartree/bohr)
fx2  fy2  fz2      (force of atom 2 of bead 2 in hartree/bohr)
...   ...   ...
fxn  fyn  fyn      (force of atom n of bead 2 in hartree/bohr)
...   ...   ...
...   ...   ...
fx1  fy1  fz1      (force of atom 1 of bead m in hartree/bohr)
fx2  fy2  fz2      (force of atom 2 of bead m in hartree/bohr)
...   ...   ...
fxn  fyn  fyn      (force of atom n of bead m in hartree/bohr)
v11  v12  v13      (virial matrix in au, optional)
v21  v22  v23      (virial matrix in au, optional)
v31  v32  v33      (virial matrix in au, optional)

```

The default value is not available.

9.4.212 <vasp_command>

Used in the case of <ipotential> = VASP5. This keyword sets the execution command of the VASP code. The data should be provided in the first line. The default is “mpirun -np 4 vasp > output.vasp”.

9.4.213 <vasp_energy>

Used in the case of <ipotential> = VASP6. This keyword sets which energy is picked up from the VASP calculation. The default value is FREE_ENERGY, which picks up the energy corresponding to the line containing “free energy TOTEN” in the outcar file. It is also possible to set this to EXTRAPOLATED, in which case the energy on the line containing “energy(sigma->0)” is selected. All other strings are not accepted as a valid input to this keyword. In both cases the energy is transferred directly from the VASP code and not actually read from the OUTCAR file.

9.4.214 <vasp_keyword_energy>

Used in the case of <ipotential> = VASP5. This keyword sets the energy keyword in the output file of the VASP code. The data should be provided in the first line. The default is “FREE ENERGIE OF THE ION-ELECTRON SYSTEM (eV)”.

9.4.215 <vasp_keyword_gradient>

Used in the case of <ipotential> = VASP5. This keyword sets the gradient keyword in the output file of the VASP code. The data should be provided in the first line. The default is “POSITION TOTAL-FORCE (eV/Angst)”.

9.4.216 <vasp_keyword_stress>

Used in the case of <ipotential> = VASP5. This keyword sets the stress keyword in the output file of the VASP code. The data should be provided in the first line. The default is “FORCE on cell =-STRESS in cart. coord. units (eV):”.

9.4.217 <vasp_output>

Used in the case of <ipotential> = VASP. This keyword sets the log file of the VASP code. The following two data should be provided in the first line.

- The option, “0” or “1”. If “0”, the log file of only the first bead is printed. If “1”, the log files of all the beads are printed.
- The print interval n (integer). The log file is printed every n steps.

The default values are “0, 1”, which means that the first bead is printed every step.

9.4.218 <vasp_reuse_wavefunction>

Used in the case of <ipotential> = VASP. This keyword controls the reuse the guess of Kohn-Sham (KS) orbitals of VASP. An integer, “0” or “1”, should be provided in the first line. If “0”, the guess of the KS orbitals is build from scratch at each step. If “1”, the KS molecular orbitals of the last step is reused as the guess for those of the next step. The default value is “1” (reuse).

9.4.219 <xmpi>

Used in the case of <ipotential> = EAM, MM, N2P2. The following two data should be provided in the first line.

- The option is “OFF” or “ON”. If “ON”, extensively parallel MPI algorithm is used.

The default value is “OFF”.

9.4.220 <xtb_exe_command>

Used in the case of <ipotential> = XTB. This keyword sets the execution command of the XTB code. The data should be provided in the first line. The default is “xtb xtb.xyz -chrg 0 -uhf 0 -grad -vparam xtb.dat ; xtb.out”.

9.5 Optional keywords in *input.dat*

9.5.1 <atom_change>

This keyword resets the atomic masses of specified atoms.

- Line 1: Number of changes (n).
- Lines 2–($n + 1$): The type of change, the initial atom, and the final atom.

The type of change can be chosen from the following options.

- FREEZE or HEAVY: Reset the atomic mass (the fictitious centroid mass for path integrals) to a large value.
- HYDROGEN: Reset the atomic mass (the fictitious centroid mass) to the hydrogen mass.

9.5.2 <charge_libnnp>

Not explained.

9.5.3 <components>

Used when <method> = ROTOR. Also used by the *prep_liquid.x* code. This keyword defines the molecular components. The data should be provided in the following format.

- The first line: the number of molecular components, n (integer).
- The next n blocks: data for each molecular component.

Each of the latter blocks has the following format.

- The first line contains three entries: the name of the molecule (character), the number of atoms per molecule m (integer), and the molecular type (character). The molecular type must be chosen from the following options.
 - GENERAL: the molecule has a nonlinear structure with six degrees of freedom.
 - LINEAR: the molecule has a linear structure with five degrees of freedom.
 - MONOATOM: the molecule is monoatomic and has three degrees of freedom.
- Each of the next m lines contains three values, corresponding to the coordinates x , y , and z of each atom in the molecule, given in bohr.

9.5.4 <file_dms_libnp>

Not explained.

9.5.5 <file_ip_libnp>

Not explained.

9.5.6 <file_pes_libnp>

Not explained.

9.5.7 <ioption_dms_libnp>

Not explained.

9.5.8 <molecules>

Used when <method> = ROTOR. Also used by the *prep_liquid.x* code. This keyword specifies the atoms belonging to each molecule.

- The first line: the number of molecules, n (integer).
- The next n lines: the name of the molecule and the atoms that belong to that molecule. The molecule name must be chosen from those defined by the keyword <components>.

9.5.9 <natom_ip_libnp>

Not explained.

9.6 Keywords in *calc.dat*

9.6.1 <ncalc>

This keyword sets the statistical analysis.

- LINE 1: number of variables, n .
- LINES 2- $(n+1)$: information for each variable (analysis keywords, followed by integers specifying atoms or species, and the names of output files).

The default value is “0” (no variables). The analysis keywords should be chosen from the following:

- `lin.atom.list, #1, #2, file name`
Generates a list of bond lengths between atoms #1 and #2.
- `angl.atom.list, #1, #2, #3, file name`
Generates a list of bond angles for atoms #1-#2-#3.
- `dih.atom.list, #1, #2, #3, #4, file name`
Generates a list of dihedral angles for atoms #1-#2-#3-#4.
- `diff.atom.list, #1, #2, #3, file name`
Generates a list of bond length differences between atoms #1-#2 and #2-#3.
- `lin.spec.list, #1, #2, file name`
Generates a list of bond lengths between atomic species #1 and #2.
- `angl.spec.list, #1, #2, #3, file name`
Generates a list of bond angles for atomic species #1-#2-#3.
- `dih.spec.list, #1, #2, #3, #4, file name`
Generates a list of dihedral angles for atomic species #1-#2-#3-#4.
- `diff.spec.list, #1, #2, #3, file name`
Generates a list of bond length differences between atomic species #1-#2 and #2-#3.
- `sdiff.spec.list, #1, #2, #3, file name`
Generates a list of absolute bond length differences between atomic species #1-#2 and #2-#3.
- `lin.atom.dens, #1, #2, file name`
Generates a density profile of bond lengths between atoms #1 and #2. The mesh size of the density profile is set by the keyword <params_lin_dens>.
- `angl.atom.dens, #1, #2, #3, file name`
Generates a density profile of bond angles for atoms #1-#2-#3. The mesh size of the density profile is set by the keyword <params_angl_dens>.
- `dih.atom.dens, #1, #2, #3, #4, file name`
Generates a density profile of dihedral angles for atoms #1-#2-#3-#4. The mesh size of the density profile is set by the keyword <params_dih_dens>.
- `diff.atom.dens, #1, #2, #3, file name`
Generates a density profile of bond length differences between atoms #1-#2 and #2-#3. The mesh size of the density profile is set by the keyword <params_diff_dens>.
- `sdiff.atom.dens, #1, #2, #3, file name`
Generates a density profile of absolute bond length differences between atoms #1-#2 and #2-#3. The mesh size of the density profile is set by the keyword <params_sdiff_dens>.

- `lin.spec.dens, #1, #2, file name`
Generates a density profile of bond lengths of atomic species #1 and #2. The mesh size of the density profile is set by the keyword `<params_lin_dens>`.
- `angl.spec.dens, #1, #2, #3, file name`
Generates a density profile of bond angles of atomic species #1-#2-#3. The mesh size of the density profile is set by the keyword `<params_angl_dens>`.
- `dih.spec.dens, #1, #2, #3, #4, file name`
Generates a density profile of dihedral angles of atomic species #1-#2-#3-#4. The mesh size of the density profile is set by the keyword `<params_dih_dens>`.
- `diff.spec.dens, #1, #2, #3, file name`
Generates a density profile of bond length differences between atomic species #1-#2 and #2-#3. The mesh size of the density profile is set by the keyword `<params_diff_dens>`.
- `sdiff.spec.dens, #1, #2, #3, file name`
Generates a density profile of absolute bond length differences between atomic species #1-#2 and #2-#3. The mesh size of the density profile is set by the keyword `<params_sdiff_dens>`.
- `lin.atom.avg, #1, #2, file name`
Calculates the average and dispersion of bond lengths between atoms #1 and #2.
- `angl.atom.avg, #1, #2, #3, file name`
Calculates the average and dispersion of bond angles for atoms #1-#2-#3.
- `dih.atom.avg, #1, #2, #3, #4, file name`
Calculates the average and dispersion of dihedral angles for atoms #1-#2-#3-#4.
- `diff.atom.avg, #1, #2, #3, #4, file name`
Calculates the average and dispersion of bond length differences between atoms #1-#2 and #2-#3.
- `lin.spec.avg, #1, #2, file name`
Calculates the average and dispersion of bond lengths between atomic species #1 and #2.
- `angl.spec.avg, #1, #2, #3, file name`
Calculates the average and dispersion of bond angles of atomic species #1-#2-#3.
- `dih.spec.avg, #1, #2, #3, #4, file name`
Calculates the average and dispersion of dihedral angles of atomic species #1-#2-#3-#4.
- `diff.spec.avg, #1, #2, #3, #4, file name`
Calculates the average and dispersion of bond length differences between atomic species #1-#2 and #2-#3.
- `xla.spec.dens, #1, #2, #3, file name`
Generates a two-dimensional density profile between bond lengths of atomic species #1-#2 and bond angles of atomic species #1-#2-#3. The mesh sizes of the density profile are set by the keywords `<params_lin_dens>` and `<params_angl_dens>`.
- `3d.spec.cube, #1, #2, #3, #4, file name`
Generates a three-dimensional density profile of atomic species #1 using the frame defined by #2, #3, and #4. The mesh sizes of the density profile are set by the keyword `<3d_range_cube>`.

The species numbers correspond to those in the file *structure.dat*. NOTE: In the calculation of bond angles and dihedral angles, the Jacobian is not included.

9.6.2 <iprint_calc>

This keyword sets the print interval of statistical analysis. The data should be provided in the first line (integer). The default value is “1” (every step).

9.6.3 <iprint_std_calc>

This keyword sets the print interval of standard output. The data should be provided in the first line (integer). The default value is “100” (every 100 steps).

9.6.4 <iconf_fin_calc>

This keyword sets the final step number read from *trj.out*. The data should be provided in the first line (integer). The default value is “-1” (the last step in *trj.out*).

9.6.5 <iconf_ini_calc>

This keyword sets the initial step number read from *trj.out*. The data should be provided in the first line (integer). The default value is 1 (the first step in *trj.out*).

9.6.6 <jformat_xyz>

This keyword sets the print option of the xyz trajectory. One of the following options can be chosen:

- `jformat_xyz = 1`: print each bead configuration separately.
- `jformat_xyz = 2`: print all beads into a single configuration.
- `jformat_xyz = 3`: print only the centroid configuration.
- `jformat_xyz = 4`: print only the selected bead defined by the keyword `<jxyz_bead>`.

The data should be provided in the first line (integer). The default value is “2”.

9.6.7 <jorigin_xyz>

This keyword sets the origin option for the xyz trajectory. One of the following options can be chosen:

- `jorigin_xyz = 1`: origin is (0.0,0.0,0.0).
- `jorigin_xyz = 2`: Cartesian coordinates of the origin are defined by the keyword `<jxyz_origin>`.
- `jorigin_xyz = 3`: origin is fixed to an atom defined by the keyword `<jxyz_atom>`.

The data should be provided in the first line (integer). The default value is “1”.

9.6.8 <jprint_xyz>

This keyword sets the print interval of the xyz trajectory. The data should be provided in the first line (integer). The default value is -1 (do not print).

9.6.9 <jspec_xyz>

Used when `<jprint_xyz>` is positive. This keyword specifies the species to be printed in the output file *calc.xyz*. Two integers should be provided in the first line, indicating the first species to be printed and the last species to be printed. Exceptionally, “-1” refers to the last species in the system. The default values are “1” (first species in the system) and “-1” (last species in the system), meaning that all species are printed.

9.6.10 <jxyz_atom>

This keyword sets the central atom of the xyz trajectory. It is activated only when `jorigin_xyz = 3`. The data should be provided in the first line (integer). The default value is 1 (the first atom).

9.6.11 <jxyz_bead>

This keyword sets the bead number of the xyz trajectory. It is activated only when `jformat_xyz = 4`. The data should be provided in the first line (integer). The default value is “1” (the first bead).

9.6.12 <jxyz_origin>

This keyword sets the origin of the xyz trajectory. It is activated only when `jorigin_xyz = 2`. The data should be provided in the first line (three real numbers). The default values are “0.0, 0.0, 0.0”.

9.6.13 <params_lin_dens>

This keyword sets the mesh for bond length distributions. Three real numbers should be provided in the first line: the minimum value, the maximum value, and the mesh size in au (bohr).

9.6.14 <params_adiff_dens>

This keyword sets the mesh for absolute bond length difference distributions. Three real numbers should be provided in the first line: the minimum value, the maximum value, and the mesh size in bohrs.

9.6.15 <params_angl_dens>

This keyword sets the mesh for bond angle distributions. Three real numbers should be provided in the first line: the minimum value, the maximum value, and the mesh size in degrees.

9.6.16 <params_diff_dens>

This keyword sets the mesh for bond length difference distributions. Three real numbers should be provided in the first line: the minimum value, the maximum value, and the mesh size in bohrs.

9.6.17 <params_dih_dens>

This keyword sets the mesh for dihedral angle distributions. Three real numbers should be provided in the first line: the minimum value, the maximum value, and the mesh size in degrees.

9.6.18 <3d_range_cube>

This keyword sets the range of three-dimensional atomic distributions as follows:

```
minimum of x, maximum of x, mesh of x in bohr units
minimum of y, maximum of y, mesh of y in bohr units
minimum of z, maximum of z, mesh of z in bohr units
```

9.7 Keywords for MM potential in *mm.dat*

In the PIMD code, the molecular mechanics potential is defined as follows.

$$V_{\text{mm}} = V_{\text{lin}} + V_{\text{gen}} + V_{\text{angl}} + V_{\text{impr}} + V_{\text{dih}} + V_{\text{cmap}} + V_{\text{lj}} + V_{\text{es}} + V_{\text{mrs}} + V_{\text{buck}}. \quad (10)$$

Each term corresponds to a specific keyword in the input file, *mm.dat*.

- V_{lin} : <linear_bonds>.
- V_{gen} : <genlin_bonds>.
- V_{angl} : <angular_bonds>.
- V_{impr} : <improper_bonds>.
- V_{dih} : <dihedral_bonds>.
- V_{cmap} : <ncmap>, <nkind_cmap>.
- V_{lj} : <lennard-jones>.
- V_{es} : <charges>, <nbc>, <ewald> (optionally <ewald_type> and <pme_ewald> for the particle-mesh Ewald method, and <damping> and <ewpol> for polarizable force fields).
- V_{mrs} : <morse>.
- V_{buck} : <buckingham>.

If a keyword is not present in the file *mm.dat*, the corresponding contribution is assumed to be zero. For example, if <linear_bonds> is absent, V_{lin} is set to zero.

9.7.1 <linear_bonds>

This keyword defines the harmonic bond distance potential, which has the form

$$V_{\text{lin}} = \sum_{\alpha=1}^{N_{\text{lin}}} \frac{1}{2} k_{\text{lin}}^{(\alpha)} \left(r_{ij}^{(\alpha)} - r_{\text{lin}}^{(\alpha)} \right)^2 \quad (11)$$

where

$$r_{ij}^{(\alpha)} = \left| \mathbf{r}_{ij}^{(\alpha)} \right| \quad (12)$$

is the bond length of the α -th pair of atoms, $i^{(\alpha)}$ and $j^{(\alpha)}$, with

$$\mathbf{r}_{ij}^{(\alpha)} = \mathbf{r}_{i^{(\alpha)}} - \mathbf{r}_{j^{(\alpha)}}. \quad (13)$$

In the file *mm.dat*, the parameters are specified as follows.

<linear_bonds>

```

Nlin
i(1),    j(1),    rlin(1),    klin(1)
i(2),    j(2),    rlin(2),    klin(2)
...
```

The first line after the keyword <linear_bonds> specifies the number of terms N_{lin} (integer). In the following N_{lin} lines, the data for each bond is provided. Here, i and j are the interacting atoms (two integers), r_{lin} is the equilibrium bond distance in bohr (real number), and k_{lin} is the force constant in hartree/bohr² (real number).

NOTE: Urey-Bradley potentials (1-3 bonded harmonic potentials) may be included in this category.

9.7.2 <genlin_bonds>

This keyword defines the general bond distance potential, which has the form of a Taylor series.

$$V_{\text{gen}} = \sum_{\alpha=1}^{N_{\text{gen}}} k_{\text{gen}}^{(\alpha)} \left(r_{ij}^{(\alpha)} - r_{\text{gen}}^{(\alpha)} \right)^{m_{\text{gen}}^{(\alpha)}} \quad (14)$$

In the file *mm.dat*, the parameters are specified as follows.

<genlin_bonds>

N_{gen}

$i^{(1)}, \quad j^{(1)}, \quad m_{\text{gen}}^{(1)}, \quad r_{\text{gen}}^{(1)}, \quad k_{\text{gen}}^{(1)}$

$i^{(2)}, \quad j^{(2)}, \quad m_{\text{gen}}^{(2)}, \quad r_{\text{gen}}^{(2)}, \quad k_{\text{gen}}^{(2)}$

...

The first line after the keyword <genlin_bonds> specifies the number of terms N_{gen} (integer). In the following N_{gen} lines, the data for each bond is provided. Here, i and j are the interacting atoms (two integers), m_{gen} is the power (integer), $r_{\text{gen}}^{(\alpha)}$ is the equilibrium bond distance in bohr (real number), and k_{gen} is the force constant in hartree/bohr² (real number).

9.7.3 <angular_bonds>

This keyword defines the harmonic bond angle potential, which has the form

$$V_{\text{angl}} = \sum_{\alpha=1}^{N_{\text{angl}}} \frac{1}{2} k_{\text{angl}}^{(\alpha)} \left(\theta_{ijk}^{(\alpha)} - \theta_{\text{angl}}^{(\alpha)} \right)^2 \quad (15)$$

where the angle θ_{ijk} is defined as

$$\cos \theta_{ijk}^{(\alpha)} = \frac{\mathbf{r}_{ij}^{(\alpha)} \cdot \mathbf{r}_{kj}^{(\alpha)}}{\left| \mathbf{r}_{ij}^{(\alpha)} \right| \left| \mathbf{r}_{kj}^{(\alpha)} \right|}. \quad (16)$$

In the file *mm.dat*, the parameters are specified as follows.

<angular_bonds>

N_{angl}

$i^{(1)}, \quad j^{(1)}, \quad k^{(1)}, \quad \theta_{\text{angl}}^{(1)}, \quad k_{\text{angl}}^{(1)}$

$i^{(2)}, \quad j^{(2)}, \quad k^{(2)}, \quad \theta_{\text{angl}}^{(2)}, \quad k_{\text{angl}}^{(2)}$

...

The first line after the keyword <angular_bonds> specifies the number of terms N_{angl} (integer). In the following N_{angl} lines, the data for each angle is provided. Here, i , j , and k are the interacting atoms (three integers), θ_{angl} is the equilibrium bond angle in degrees (real number), and k_{angl} is the force constant in hartree/degrees² (real number).

9.7.4 <improper_bonds>

This keyword defines the harmonic improper angle potential, which has the form

$$V_{\text{impr}} = \sum_{\alpha=1}^{N_{\text{impr}}} \frac{1}{2} k_{\text{impr}}^{(\alpha)} \left(\chi_{ijkl}^{(\alpha)} - \chi_{\text{impr}}^{(\alpha)} \right)^2. \quad (17)$$

The improper angle, χ_{ijkl} , is defined as the angle between the vector normal to the i - j - k plane and the vector normal to the j - k - l plane,

$$\cos \chi_{ijkl}^{(\alpha)} = \frac{\mathbf{s}_{ijk}^{(\alpha)} \cdot \mathbf{s}_{ljk}^{(\alpha)}}{\left| \mathbf{s}_{ijk}^{(\alpha)} \right| \left| \mathbf{s}_{ljk}^{(\alpha)} \right|}, \quad (18)$$

where

$$\mathbf{s}_{ijk}^{(\alpha)} = \mathbf{r}_{ij}^{(\alpha)} \times \mathbf{r}_{kj}^{(\alpha)}, \quad \mathbf{s}_{ljk}^{(\alpha)} = \mathbf{r}_{lj}^{(\alpha)} \times \mathbf{r}_{kj}^{(\alpha)}. \quad (19)$$

Note that the definition of the improper angle is the same as that of the dihedral angle. In the file *mm.dat*, the parameters are specified as follows.

<improper_bonds>

N_{impr}

$i^{(1)}, \quad j^{(1)}, \quad k^{(1)}, \quad l^{(1)}, \quad \chi_{\text{impr}}^{(1)}, \quad k_{\text{impr}}^{(1)}$

$i^{(2)}, \quad j^{(2)}, \quad k^{(2)}, \quad l^{(2)}, \quad \chi_{\text{impr}}^{(2)}, \quad k_{\text{impr}}^{(2)}$

...

The first line after the keyword <improper_bonds> specifies the number of terms N_{impr} (integer). In the following N_{impr} lines, the data for each interaction is provided. Here, i , j , k , and l are the interacting atoms (four integers), χ_{impr} is the equilibrium bond angle in degrees (real number), and k_{impr} is the force constant in hartree/degrees² (real number).

9.7.5 <dihedral_bonds>

This keyword defines the dihedral angle potential, which has the form

$$V_{\text{dih}} = \sum_{\alpha=1}^{N_{\text{dih}}} \frac{1}{2} v_{\text{dih}}^{(\alpha)} \left\{ 1 + \mu_{\text{dih}}^{(\alpha)} \cos \left(\nu_{\text{dih}}^{(\alpha)} \chi_{ijkl}^{(\alpha)} \right) \right\}. \quad (20)$$

The dihedral angle χ_{ijkl} is defined as the angle between the vector normal to the i - j - k plane and the vector normal to the j - k - l plane,

$$\cos \chi_{ijkl}^{(\alpha)} = \frac{\mathbf{s}_{ijk}^{(\alpha)} \cdot \mathbf{s}_{ljk}^{(\alpha)}}{\left| \mathbf{s}_{ijk}^{(\alpha)} \right| \left| \mathbf{s}_{ljk}^{(\alpha)} \right|}. \quad (21)$$

Note that the definition of the dihedral angle is the same as that of the improper angle. However, the improper and dihedral angle potentials differ in their functional form. In the file *mm.dat*, the parameters are specified as follows.

<dihedral_bonds>

N_{dih}
 $i^{(1)}, \quad j^{(1)}, \quad k^{(1)}, \quad l^{(1)}, \quad v_{\text{dih}}^{(1)}, \quad \nu_{\text{dih}}^{(1)}, \quad \mu_{\text{dih}}^{(1)}$
 $i^{(2)}, \quad j^{(2)}, \quad k^{(2)}, \quad l^{(2)}, \quad v_{\text{dih}}^{(2)}, \quad \nu_{\text{dih}}^{(2)}, \quad \mu_{\text{dih}}^{(2)}$
 \dots

The first line after the keyword <dihedral_bonds> specifies the number of terms N_{dih} (integer). In the following N_{dih} lines, the data for each interaction is provided. Here, i, j, k , and l are the interacting atoms (four integers), χ_{ijkl} is the dihedral angle in degrees (real number), v_{dih} is the rotational barrier in hartree (real number), ν_{dih} is the multiplicity (integer), and μ_{dih} is the phase factor = ± 1 (integer).

9.7.6 <ncmap>, <nkind_cmap>

This keyword sets the CMAP correction. In the file *mm.dat*, the parameters are specified as follows.

<ncmap>

N_{cmap}
 $i^{(1)}, \quad j^{(1)}, \quad k^{(1)}, \quad l^{(1)}, \quad m^{(1)}, \quad \alpha^{(1)}$
 $i^{(2)}, \quad j^{(2)}, \quad k^{(2)}, \quad l^{(2)}, \quad m^{(2)}, \quad \alpha^{(2)}$
 \dots

In the first line after the keyword <ncmap>, the number of terms, N_{cmap} , is given (integer). In the following N_{cmap} lines, the data for each interaction are provided. Here, i, j, k, l , and m are the atoms comprising the two dihedral angles (i - j - k - l and j - k - l - m), and α is the CMAP kind label, each of which is specified as follows.

<nkind_cmap>

M_{cmap}
1,
 $V_{\text{cmap}}^{(1)}(0^\circ, 0^\circ), \quad V_{\text{cmap}}^{(1)}(15^\circ, 0^\circ), \quad \dots, \quad V_{\text{cmap}}^{(1)}(330^\circ, 345^\circ), \quad V_{\text{cmap}}^{(1)}(345^\circ, 345^\circ)$
2,
 $V_{\text{cmap}}^{(2)}(0^\circ, 0^\circ), \quad V_{\text{cmap}}^{(2)}(15^\circ, 0^\circ), \quad \dots, \quad V_{\text{cmap}}^{(2)}(330^\circ, 345^\circ), \quad V_{\text{cmap}}^{(2)}(345^\circ, 345^\circ)$
 \dots

In the first line after the keyword <nkind_cmap>, the number of CMAP kinds, M_{cmap} , is given (integer). The data for each CMAP kind are provided in the following M_{cmap} blocks. Each CMAP kind, α , is followed by $576 = 24 \times 24$ real numbers corresponding to the values of $V_{\text{cmap}}^{(\alpha)}(x, y)$ at evenly spaced grid points for the dihedral angles x and y ($0^\circ \leq x, y \leq 345^\circ$ with an interval of 15°). The grid data may be separated into lines containing 24 numbers each.

9.7.7 <lennard-jones>

This keyword sets the Lennard-Jones (LJ) potential, which has the form

$$V_{\text{lj}} = \sum_{\alpha=1}^{N_{\text{lj}}} V_{\text{lj}}^{(\alpha)}(r_{ij}^{(\alpha)}) = \sum_{\alpha=1}^{N_{\text{lj}}} 4\epsilon_{\text{lj}}^{(\alpha)} \left\{ \left(\frac{\sigma_{\text{lj}}^{(\alpha)}}{r_{ij}^{(\alpha)}} \right)^{12} - \left(\frac{\sigma_{\text{lj}}^{(\alpha)}}{r_{ij}^{(\alpha)}} \right)^6 \right\}. \quad (22)$$

In the file *mm.dat*, the parameters are specified as follows.

<lennard-jones>

N_{lj}

$r_{\text{in}}, r_{\text{out}}$

$i^{(1)}, j^{(1)}, \epsilon_{\text{lj}}^{(1)}, \sigma_{\text{lj}}^{(1)}$

$i^{(2)}, j^{(2)}, \epsilon_{\text{lj}}^{(2)}, \sigma_{\text{lj}}^{(2)}$

...

In the first line after the keyword <lennard-jones>, the number of terms, N_{lj} , is given (integer). In the second line, r_{in} and r_{out} are the inner and outer Lennard-Jones cutoff distances (two real numbers), respectively, which are shared by all atomic pairs. These values correspond to the range in which the LJ interaction is smoothly cut off. In the following N_{lj} lines, the data for each interaction are given. Here, i and j are the interacting atoms (two integers), ϵ_{lj} is the LJ well depth in hartree (real number), and σ_{lj} is the LJ radius in bohr (real number).

To smoothly cutoff the Lennard-Jones interaction, a switching function $f(r)$ is applied:

$$\tilde{V}_{\text{lj}}^{(\alpha)}(r) = V_{\text{lj}}^{(\alpha)}(r) \times f(r). \quad (23)$$

The same form of $f(r)$ is used for all atomic pairs: $f(r) = 1$ ($r \leq r_{\text{in}}$), $f(r) = 0$ ($r \geq r_{\text{out}}$) and

$$f(r) = \sum_{n=0}^3 c_n r^n, \quad (r_{\text{in}} < r < r_{\text{out}}) \quad (24)$$

with the coefficients

$$\begin{aligned} c_0 &= \frac{r_{\text{out}}^3 - 3r_{\text{in}}r_{\text{out}}^2}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_1 &= \frac{6r_{\text{in}}r_{\text{out}}}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_2 &= \frac{-3r_{\text{in}} - 3r_{\text{out}}}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_3 &= \frac{2}{(r_{\text{out}} - r_{\text{in}})^3}. \end{aligned} \quad (25)$$

This function decreases smoothly from $f(r_{\text{in}}) = 1$ to $f(r_{\text{out}}) = 0$. Within the inner cutoff distance, $r < r_{\text{in}}$, $f(r)$ is set to 1, so that the switching function has no effect. Beyond the outer cutoff distance, $r > r_{\text{out}}$, $f(r)$ is set to zero, and the Lennard-Jones interaction is zero.

9.7.8 <charges>, <nbc>, <ewald>

These keywords designate the electrostatic (Coulombic) potential. This potential has the form

$$V_{\text{es}}^{\text{non-pol}} = \sum_{i=1}^{N_{\text{chrg}}} \sum_{j>i}^{N_{\text{chrg}}} \frac{q_i q_j}{r_{ij}} - \sum_{\alpha=1}^{N_{\text{nbc}}} \left(1 - s^{(\alpha)}\right) \frac{q_{i_\alpha} q_{j_\alpha}}{r_{ij}^{(\alpha)}}. \quad (26)$$

The first term is the electrostatic interaction between all pairs of charged atoms. The second term subtracts the electrostatic interaction for specific pairs of charged atoms. This second term is necessary when the electrostatic interaction is scaled down or neglected for specific atomic pairs within the same molecule (such as 1-2, 1-3, or 1-4 bonded pairs of charged atoms).

In the file *mm.dat*, the parameters are specified as follows. For the first term:

<charges>

N_{chrg}

1, q_1

2, q_2

...

In the first line after the keyword **<charges>**, the number of charges, N_{chrg} , is given (integer). In the following N_{chrg} lines, the data for each charge are provided: i is the charged atom (integer), and q_i is the charge value (real number) in atomic units.

For the second term:

<nbcpr>

N_{nbcpr}

$i^{(1)}$, $j^{(1)}$, $s^{(1)}$

$i^{(2)}$, $j^{(2)}$, $s^{(2)}$

...

In the first line after the keyword **<nbcpr>**, the number of bonded charge pairs, N_{nbcpr} , is given (integer). In the following N_{nbcpr} lines, the bonded atomic pairs $i^{(\alpha)}$ and $j^{(\alpha)}$, along with the screening factor $s^{(\alpha)}$, are listed. If $s^{(\alpha)} = 0.0$, the electrostatic interaction is neglected for that pair; if $s^{(\alpha)} = 0.5$, the interaction is scaled down to 50%; and if $s^{(\alpha)} = 1.0$, the interaction is fully included with no effect.

When periodic boundary conditions are applied, the electrostatic interactions are calculated using the Ewald method. In this case, the parameters controlling the precision and efficiency of the Ewald sum are given using the keyword:

<ewald>

1.d-8, 4.d0 0

The first value (real number) is the precision of the energy. The second value (real number) is the ratio of computational time between the real-space and reciprocal-space Ewald sums. The third value (integer) indicates whether to include (=1) or not include (=0) the dipole contribution of the unit cell.

9.7.9 **<ewald_type>**, **<pme_ewald>**

For large systems, the particle mesh Ewald (PME) method can reduce computational time when computing electrostatic interactions under periodic boundary conditions. In this case, specify

<ewald_type>

PME

(The default value for this keyword is “STANDARD”, meaning PME is not used.) The PME mesh size can be controlled using the keyword

<pme_ewald>

1.09

The value (real number) specifies the number of meshes per bohr. This is converted to integers n_x , n_y , and n_z , which divide the x , y , and z components of the unit cell into 2^{n_x} , 2^{n_y} , and 2^{n_z} meshes, respectively.

9.7.10 <charges>, <damping>, <ewald>, <ewpol>

These keywords designate the electrostatic potential for a polarizable force field. This potential has the form

$$V_{\text{es}}^{\text{pol}} = + \sum_{i=1}^{N_{\text{chrg}}} \frac{\mu_i^2}{2\alpha_i} + \sum_{i>j}^{N_{\text{chrg}}} \frac{s_{\text{cc}} q_i q_j}{r_{ij}} + \sum_{i \neq j}^{N_{\text{chrg}}} \mu_i \mathbf{t}_{ij} q_j + \sum_{i>j}^{N_{\text{chrg}}} \mu_i \mathbf{T}_{ij} \mu_j - \sum_{\alpha=1}^{N_{\text{nbc}}} \left(1 - s^{(\alpha)}\right) \frac{q_{i\alpha} q_{j\alpha}}{r_{ij}^{(\alpha)}}. \quad (27)$$

The first term on the right-hand side is the same as in the non-polarizable force field. The second term represents the polarization energy, where α_i is the atomic polarizability. The third, fourth, and fifth terms correspond to the electrostatic interactions: charge-charge, charge-dipole, and dipole-dipole interactions, respectively, where μ_i is the induced dipole moment of atom i . These interactions are scaled by damping functions $s_{\text{cc}} = s_{\text{cc}}(r)$, $s_{\text{cd}} = s_{\text{cd}}(r)$, and $s_{\text{dd}} = s_{\text{dd}}(r)$, which depend on the interatomic distances $r = r_{ij}$ at short range and converge to 1 at long range. The last term subtracts the electrostatic interaction for specific pairs of charged atoms. This term is needed when the electrostatic interaction is scaled down or neglected for specific atomic pairs within the same molecule (such as 1-2, 1-3, and 1-4 bonded pairs of charged atoms).

In the file *mm.dat*, the parameters are given as follows. For the first term,

```
<charges>
N_chrg
1,   q1,   α1,   n1
2,   q2,   α2,   n2
...
```

In the first line after the keyword <charges>, the number of charges, N_{chrg} , is given (integer). In the following N_{chrg} lines, the data for each charge are listed: i is the charged atom (integer), q_i is the charge value (real number) in au, α_i is the polarizability (real number) in bohr³, and n_i is the atom type (integer) used to specify the damping function, as described below.

```
<damping>
N_damp
t(1), n(1), m(1), f(1), a(1), (b(1))
t(2), n(2), m(2), f(2), a(2), (b(2))
...
```

In the first line after the keyword <damping>, the number of damping functions, N_{damp} , is given. In the following N_{damp} lines, the data for each damping function are given: The interaction type, $t^{(i)}$, is either CC (charge-charge), CD (charge-dipole), or DD (dipole-dipole). The pair $n^{(i)}$ and $m^{(i)}$ are the atomic types specified in the keyword <charges>. The functional form, $f^{(i)}$, is either EXP (exponential Thole), LIN (linear Thole), GAU (Gaussian) [85], OSS (Ojamae-Shavitt-Singer) [86], or NONE (no damping). $a^{(i)}$ (and optionally $b^{(i)}$) are the parameters (real numbers) used in the damping functions. See Section 11.22 for details.

For the last term,

```
<nbc>
N_nbc
i(1), j(1), s(1)
i(2), j(2), s(2)
...
```

In the first line after the keyword `<nbc>`, the number of bonded charge pairs, N_{nbc} , is given (integer). In the following N_{nbc} lines, the bonded atomic pairs $i^{(\alpha)}$ and $j^{(\alpha)}$ as well as the screening factor $s^{(\alpha)}$ are listed. If $s^{(\alpha)} = 0.0$, the electrostatic interaction is neglected for the atomic pair; if $s^{(\alpha)} = 0.5$, the interaction is scaled down to 50%; and if $s^{(\alpha)} = 1.0$, the electrostatic interaction is fully included and thus unaffected.

When periodic boundary conditions are applied, the electrostatic interactions are calculated using the Ewald method. In this case, the parameters to control the precision and efficiency of the Ewald sum are given by the following keywords:

```
<ewald>
1.d-8, 4.0d0, 0
```

```
<ewpol>
1.d-8, 0.1d0
```

The keyword `<ewald>` controls the Ewald sum for the charge-charge interactions, while `<ewpol>` controls the Ewald sum for the charge-dipole and dipole-dipole interactions. The first value (real number) is the precision of the energy. The second value (real number) is the ratio of the computational time between the real-space Ewald sum and the reciprocal-space Ewald sum. The third value in the keyword `<ewald>` (integer) specifies whether to include (=1) or not to include (=0) the dipole contribution of the unit cell.

9.7.11 <morse>

This keyword sets the Morse potential. This potential has the form

$$V_{\text{mrs}} = \sum_{\alpha=1}^{N_{\text{mrs}}} d_{\text{mrs}}^{(\alpha)} \left(\left[1 - \exp \left\{ -a_{\text{mrs}}^{(\alpha)} \left(r_{ij}^{(\alpha)} - r_{\text{mrs}}^{(\alpha)} \right) \right\} \right]^2 - 1 \right). \quad (28)$$

In the file *mm.dat*, the parameters are given as follows.

```
<morse>
N_mrs
i(1), j(1), r_mrs(1), d_mrs(1), a_mrs(1)
i(2), j(2), r_mrs(2), d_mrs(2), a_mrs(2)
...
```

In the first line after the keyword `<morse>`, N_{mrs} is the number of interactions (integer). In the following N_{mrs} lines, the data for each interaction are given. Here, i and j are the interacting atoms (two integers), r_{mrs} is the equilibrium bond distance in bohr (real number), d_{mrs} is the energy depth in hartree (real number), and a_{mrs} is the exponent in bohr⁻¹ (real number).

When periodic boundary conditions are applied, the minimum image convention is used without a cutoff.

9.7.12 <buckingham>

This keyword sets the Buckingham potential, which has the form

$$V_{\text{buck}} = \sum_{\alpha=1}^{N_{\text{buck}}} V_{\text{buck}}^{(\alpha)} \left(r_{ij}^{(\alpha)} \right) = \sum_{\alpha=1}^{N_{\text{buck}}} A_{\text{buck}}^{(\alpha)} \exp \left(-B_{\text{buck}}^{(\alpha)} r_{ij}^{(\alpha)} \right) C r_{ij}^{(\alpha)-6}. \quad (29)$$

In the file *mm.dat*, the parameters are given as follows.

<buckingham>

N_{buck}

$r_{\text{in}}, r_{\text{out}}$

$i^{(1)}, j^{(1)}, a_{\text{buck}}^{(1)}, b_{\text{buck}}^{(1)}, c_{\text{buck}}^{(1)},$

$i^{(2)}, j^{(2)}, a_{\text{buck}}^{(2)}, b_{\text{buck}}^{(2)}, c_{\text{buck}}^{(2)},$

...

In the first line after the keyword <buckingham>, the number of terms, N_{buck} , is given (integer). In the second line, r_{in} and r_{out} are the inner and outer Buckingham cutoff distances (two real numbers), respectively, which are shared by all atomic pairs. As described below, these values correspond to the range over which the interaction is smoothly cut off. In the following N_{buck} lines, the data for each interaction are given. Here, i and j are the interacting atoms (two integers), and a_{buck} , b_{buck} , and c_{buck} are the parameters in atomic units (three real numbers).

To smoothly cutoff the Buckingham interaction, the switching function $f(r)$ is applied:

$$\tilde{V}_{\text{buck}}^{(\alpha)}(r) = V_{\text{buck}}^{(\alpha)}(r) \times f(r). \quad (30)$$

The same form of $f(r)$ is used for all atomic pairs: $f(r) = 1$ ($r \leq r_{\text{in}}$), $f(r) = 0$ ($r \geq r_{\text{out}}$), and

$$f(r) = \sum_{n=0}^3 c_n r^n \quad (r_{\text{in}} < r < r_{\text{out}}) \quad (31)$$

with the coefficients

$$\begin{aligned} c_0 &= \frac{r_{\text{out}}^3 - 3r_{\text{in}}r_{\text{out}}^2}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_1 &= \frac{6r_{\text{in}}r_{\text{out}}}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_2 &= \frac{-3r_{\text{in}} - 3r_{\text{out}}}{(r_{\text{out}} - r_{\text{in}})^3}, \\ c_3 &= \frac{2}{(r_{\text{out}} - r_{\text{in}})^3}. \end{aligned} \quad (32)$$

The function decreases smoothly from $f(r_{\text{in}}) = 1$ to $f(r_{\text{out}}) = 0$. Within the inner cutoff distance, $r < r_{\text{in}}$, $f(r)$ is set to 1, so that the switching function has no effect. Beyond the outer cutoff distance, $r > r_{\text{out}}$, $f(r)$ is set to 0, and thus the Buckingham interaction is zero.

9.8 Keywords for EAM potential in *eam.dat*

The EAM potential has the form [45, 46]

$$V_{\text{eam}} = \sum_{i=1}^{N_{\text{atom}}} f_{s_i}(\rho_i) + \sum_{i=1}^{N_{\text{atom}}} \sum_{j>i}^{N_{\text{atom}}} \phi_{s_i, s_j}(r_{ij}), \quad (33)$$

where s_i and s_j are the atomic species of atoms i and j , respectively. The electron density ρ_i is a function of the coordinates of all the surrounding atoms,

$$\rho_i = \sum_{j \neq i}^{N_{\text{atom}}} \varrho_{s_j}(r_{ij}). \quad (34)$$

The electron density $\varrho_s(r)$ for each species, the embedding energy $f_s(\rho)$ for each species, and the pair interaction $\phi_{s,s'}(r)$ for each pair of species are three functions that constitute the EAM potential. These functions are provided as interpolations of tables. In the file *eam.dat*, the tables for $\varrho_i(r)$, $f_s(\rho)$, and $\phi_{s,s'}(r)$ should be given following the keywords `<rhoeam>`, `<frhoeam>`, and `<phir_eam>`, respectively. The number of reference data points, N_{dp} , given by the keyword `<nref_eam>`, should be the same for $\varrho_s(r)$, $f_s(\rho)$, and $\phi_{s,s'}(r)$ for all s and s' . The cutoff distance, r_{cut} , given by the keyword `<rcut_eam>`, should be the same for $\varrho_s(r)$ and $\phi_{s,s'}(r)$ for all atomic species.

The ADP potential has the form [50]

$$V_{\text{adp}} = V_{\text{eam}} + \frac{1}{2} \sum_{i=1}^N \sum_{\alpha=1}^3 (\mu_{i,\alpha})^2 + \frac{1}{2} \sum_{i=1}^N \sum_{\alpha=1}^3 \sum_{\beta=1}^3 (\lambda_{i,\alpha\beta})^2 - \frac{1}{6} \sum_{i=1}^N \nu_i^2, \quad (35)$$

where

$$\mu_{i,\alpha} = \sum_{j(\neq i)}^N u_{ij}(r_{ij}) (\mathbf{r}_{ij})_{\alpha}, \quad (36)$$

$$\lambda_{i,\alpha\beta} = \sum_{j(\neq i)}^N w_{ij}(r_{ij}) (\mathbf{r}_{ij})_{\alpha} (\mathbf{r}_{ij})_{\beta}, \quad (37)$$

$$\nu_i = \sum_{\alpha=1}^3 \lambda_{i,\alpha\alpha} = \sum_{\alpha=1}^3 \sum_{j(\neq i)}^N w_{ij}(r_{ij}) (\mathbf{r}_{ij})_{\alpha} (\mathbf{r}_{ij})_{\alpha} = \sum_{j(\neq i)}^N w_{ij} r_{ij}^2, \quad (38)$$

and $r_{ij} = |\mathbf{r}_{ij}|$.

In addition to the functions of the EAM potential, the dipole function $u_{s,s'}(r)$ and the quadrupole function $w_{s,s'}(r)$ for each pair of species constitute the ADP potential. These functions are provided as interpolations of tables. In the file *eam.dat*, the tables for $u_{s,s'}(r)$ and $w_{s,s'}(r)$ should follow the keywords `<ur_adp>` and `<wr_adp>`, respectively. The number of reference data points, N_{dp} , given by the keyword `<nref_eam>`, should be the same for $u_{s,s'}(r)$ and $w_{s,s'}(r)$ for all s and s' . The cutoff distance, r_{cut} , given by the keyword `<rcut_eam>`, should be the same for $u_{s,s'}(r)$ and $w_{s,s'}(r)$ for all atomic species.

9.8.1 `<nref_eam>`

This keyword gives the number of data points for the tables of $\varrho_s(r)$, $f_s(\rho)$, and $\phi_{s,s'}(r)$. The data should be provided in the first line (integer).

`<nref_eam>`

N_{dp}

The number of data points, N_{dp} , is the same for all tables given by the keywords `<rhoeam>`, `<frhoeam>`, and `<phir>`.

9.8.2 `<rcut_eam>`

This keyword gives the cutoff distance, r_{cut} , in Å. The data should be provided in the first line (real number).

`<rcut_eam>`

r_{cut}

The cutoff distance is the same for all tables given by the keywords `<rhoeam>`, `<frhoeam>`, and `<phir>`.

9.8.3 <rhoeam>

This keyword gives the tables of $\varrho_s(r)$ as a function of interatomic distance r in Å, for all atomic species $1 \leq s \leq N_{\text{spec}}$, where the number of atomic species N_{spec} is given by the file *structure.dat*. The number of data points, N_{dp} , is given by the keyword <nref_eam>.

```
<rhoeam>
1
Table of   r   and    $\varrho_1(r)$        (This consists of  $N_{\text{dp}}$  lines)
2
Table of   r   and    $\varrho_2(r)$        (This consists of  $N_{\text{dp}}$  lines)
...
```

NOTE: The part with r values (in Å) larger than the cutoff distance r_{cut} (in Å) is not used.

9.8.4 <frhoeam>

This keyword gives the tables of $f_s(\rho)$ in eV, for all atomic species $1 \leq s \leq N_{\text{spec}}$, where the number of atomic species N_{spec} is given by the file *structure.dat*. The number of data points, N_{dp} , is given by the keyword <nref_eam>.

```
<frhoeam>
1
Table of    $\rho$    and    $f_1(\rho)$        (This consists of  $N_{\text{dp}}$  lines)
2
Table of    $\rho$    and    $f_2(\rho)$        (This consists of  $N_{\text{dp}}$  lines)
...
```

9.8.5 <phir_eam>

This keyword gives the tables of $\phi_{s,s'}(r)$ in eV as a function of interatomic distance in Å, for all combinations of atomic species, i.e., $N_{\text{spec}}(N_{\text{spec}} + 1)/2$ combinations ($1 \leq s \leq N_{\text{spec}}$ and $s' \geq s$), where the number of atomic species N_{spec} is given by the file *structure.dat*. The number of data points, N_{dp} , is given by the keyword <nref_eam>.

```
<phir_eam>
1   1
Table of   r   and    $\phi_{1,1}(r)$        (This consists of  $N_{\text{dp}}$  lines)
1   2
Table of   r   and    $\phi_{1,2}(r)$        (This consists of  $N_{\text{dp}}$  lines)
2   2
Table of   r   and    $\phi_{2,2}(r)$        (This consists of  $N_{\text{dp}}$  lines)
...
```

NOTE: The part with r values (in Å) larger than the cutoff distance r_{cut} (in Å) is not used.

9.8.6 <ur_adp>

This keyword gives the tables of $u_{s,s'}(r)$ in $\text{eV}^{1/2}\text{\AA}^{-1}$ as a function of interatomic distance in \AA , for all combinations of atomic species, i.e., $N_{\text{spec}}(N_{\text{spec}} + 1)/2$ combinations ($1 \leq s \leq N_{\text{spec}}$ and $s' \geq s$), where the number of atomic species N_{spec} is given by the file *structure.dat*. The number of data points, N_{dp} , is given by the keyword <nref_eam>.

```
<ur_adp>
1 1
Table of r and u1,1(r) (This consists of Ndp lines)
1 2
Table of r and u1,2(r) (This consists of Ndp lines)
2 2
Table of r and u2,2(r) (This consists of Ndp lines)
...
```

NOTE: The part with r values (in \AA) larger than the cutoff distance r_{cut} (in \AA) is not used.

9.8.7 <wr_adp>

This keyword gives the tables of $w_{s,s'}(r)$ in $\text{eV}^{1/2}\text{\AA}^{-1}$ as a function of interatomic distance in \AA , for all combinations of atomic species, i.e., $N_{\text{spec}}(N_{\text{spec}} + 1)/2$ combinations ($1 \leq s \leq N_{\text{spec}}$ and $s' \geq s$), where the number of atomic species N_{spec} is given by the file *structure.dat*. The number of data points, N_{dp} , is given by the keyword <nref_eam>.

```
<wr_adp>
1 1
Table of r and w1,1(r) (This consists of Ndp lines)
1 2
Table of r and w1,2(r) (This consists of Ndp lines)
2 2
Table of r and w2,2(r) (This consists of Ndp lines)
...
```

NOTE: The part with r values (in \AA) larger than the cutoff distance r_{cut} (in \AA) is not used.

9.9 Keywords for the Tersoff potential in *tersoff.dat*

The Tersoff potential has the following functional form [48]:

$$V_{\text{tersoff}} = \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N f_c(r_{ij}) [A_{ij} e^{-\lambda_{ij} r_{ij}} - b_{ij}(\{\mathbf{r}\}) B_{ij} e^{-\mu_{ij} r_{ij}}]. \quad (39)$$

The cutoff function $f_c(r)$ is defined as

$$f_c(r_{ij}) = \begin{cases} 1, & r_{ij} < R_{ij}, \\ \frac{1}{2} \left[1 + \cos \left(\pi \frac{r_{ij} - R_{ij}}{S_{ij} - R_{ij}} \right) \right], & R_{ij} < r_{ij} < S_{ij}, \\ 0, & r_{ij} > S_{ij}. \end{cases} \quad (40)$$

The bond-order term b_{ij} is given by

$$b_{ij}(\{\mathbf{r}\}) = \chi_{ij} \left(1 + \gamma_{ij} \zeta_{ij}^{n_{ij}}\right)^{-1/(2n_{ij})}, \quad (41)$$

where

$$\zeta_{ij} = \sum_{k \neq i,j}^N f_c(r_{ik}) g(\theta_{kij}) \omega_{ik} \exp\left[-\sigma_{ij} (r_{ij} - r_{ik})^3\right]. \quad (42)$$

The angular function $g(\theta)$ is defined as

$$g(\theta_{kij}) = 1 + \frac{c_{ij}^2}{d_{ij}^2} - \frac{c_{ij}^2}{d_{ij}^2 + (h_{ij} - \cos \theta_{kij})^2}, \quad (43)$$

with

$$\cos \theta_{kij} = \frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{ik}}{|\mathbf{r}_{ij}| |\mathbf{r}_{ik}|}. \quad (44)$$

For each pair of atomic species (i, j) , the Tersoff potential is characterized by 14 parameters:

$$\begin{aligned} \mathbf{A} &= A_{ij}, & \mathbf{B} &= B_{ij}, & \text{lambda} &= \lambda_{ij}, & \text{gamma} &= \gamma_{ij}, & \text{mu} &= \mu_{ij}, \\ \mathbf{n} &= n_{ij}, & \mathbf{c} &= c_{ij}, & \mathbf{d} &= d_{ij}, & \mathbf{h} &= h_{ij}, & \mathbf{R} &= R_{ij}, \\ \mathbf{S} &= S_{ij}, & \text{chi} &= \chi_{ij}, & \text{omega} &= \omega_{ij}, & \text{sigma} &= \sigma_{ij}. \end{aligned} \quad (45)$$

These parameters are specified in the input file *tersoff.dat*. An example for a SiC system (parameters collected by Liu Yuxin, Jilin University) is shown below.

```
<tersoff>
2
A      1393.60      1597.3111  1830.80      ! eV
B      346.740      395.126   471.180      ! eV
lambda 3.48790      2.9839   2.47990      ! angstrom**-1
mu      2.21190      1.97205   1.73220      ! angstrom**-1
gamma   1.57240e-7   0.00000   1.10000e-6   ! none
n       0.72751      0.00000   0.78734      ! none
c       38049.0      0.00000   100390.0     ! none
d       4.34840      0.00000   16.2170      ! none
h      -0.57058      0.00000  -0.59825     ! none
R       1.80000      2.20454   2.70000      ! angstrom
S       2.10000      2.50998   3.00000      ! angstrom
chi     1.00000      0.97760   1.00000      ! none
omega   1.00000      1.00000   1.00000      ! none
sigma   0.00000      0.00000   0.00000      ! angstrom**-3
```

The first line after the keyword `<tersoff>` specifies the number of atomic species, n . In this example, $n = 2$. The following 14 lines correspond to the Tersoff parameters listed above. Each line contains $n(n+1)/2$ values, which correspond to all pairs of atomic species.

The ordering of the parameter columns must be

$$1-1, 1-2, \dots, 1-n, 2-2, \dots, 2-n, \dots, n-n.$$

All parameters should be given in units of eV and Å.

If the parameter \mathbf{n} is set to zero for a given pair (i, j) (i.e., $n_{ij} = 0$), the bond-order term is assumed to be constant, $b_{ij}(\{\mathbf{r}\}) = \chi_{ij}$. In this case, the interaction between species i and j reduces to a purely two-body interaction.

9.10 Common keywords shared with constrained MD and metadynamics

9.10.1 <ncons>

See Section 9.4.147.

9.10.2 <nref>

See Section 9.3.11.

9.10.3 <params.cons>

See Section 9.4.171.

9.11 Common keywords shared in all types of AFED

9.11.1 <afed_type>

Used in the cases of <method> = AFED. This keyword sets the type of adiabatic free energy dynamics. The data should be provided in the first line (character). The options are as follows.

- GRAD: Calculation of the free energy gradient with a constant constraint.
- DESCENT: Descent search for free energy minimum points using the steepest descent method.
- ASCENT: Ascent search for free energy saddle points.
- AUTO: Automated search for free energy stationary points by combining DESCENT and ASCENT.
- TAMD: Temperature accelerated molecular dynamics.
- LOGMFD: Logarithmic mean force dynamics.

The default value is “GRAD”.

9.11.2 <fenergy_max.afed>

Used in the cases of <method> = AFED, with <afed_type> = DESCENT, ASCENT, AUTO, TAMD, LOGMFD. This keyword sets the upper bound of the free energy to be explored in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “0.16” hartree (about 100 kcal/mol).

9.11.3 <mdcycle_pro.afed>

Used in all cases of <method> = AFED. This keyword sets the ensemble of productive MD runs used to compute the average free energy gradients in adiabatic free energy dynamics. The data should be provided in the first line (character).

- NVT: NVT ensemble.
- NVE: NVE ensemble.

The default value is “NVT”.

9.11.4 <niter_afed>

Used in all cases of <method> = AFED and <afed_type> = DESCENT, ASCENT, AUTO, TAMD, LOGMFD.

- For <afed_type> = DESCENT, ASCENT, AUTO: This keyword sets the maximum number of AFED steps (iterations) in adiabatic free energy dynamics.
- For <afed_type> = TAMD, LOGMFD: This keyword sets the number of AFED steps (iterations) in temperature accelerated molecular dynamics and logarithmic mean force dynamics.

The data should be provided in the first line (real number). The default value is “200” steps.

9.11.5 <nstep_pre_afed>

Used in all cases of <method> = AFED. This keyword sets the number of preliminary MD steps in adiabatic free energy dynamics to equilibrate the system under constraint. The data should be provided in the first line (real number). The default value is “1000” steps.

9.11.6 <nstep_pro_afed>

Used in all cases of <method> = AFED. This keyword sets the number of productive MD steps used to compute the average free energy gradients in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “2000” steps.

9.11.7 <params_afed>

Used in all cases of <method> = AFED. This keyword sets the parameters for the collective variables in adiabatic free energy dynamics. For each line, the type of collective variable is followed by the four parameters: the reference shift $\Delta\mathcal{R}_d^{\text{ref}}$, the lower bound $\mathcal{R}_d^{\text{min}}$, and the upper bound $\mathcal{R}_d^{\text{max}}$. The default values are as follows:

DIST	0.08d0	1.0d0	10.0d0	! bond distance
ANGL	3.00d0	0.0d0	180.0d0	! bond angle
DIH	5.00d0	-180.0d0	180.0d0	! bond dihedral
DIFF	0.04d0	-3.0d0	3.0d0	! difference in distances
CN	0.02d0	0.0d0	4.0d0	! coordination number
DCN	0.02d0	0.0d0	4.0d0	! difference in coordination numbers
XYZ	0.08d0	1.0d0	10.0d0	! center of mass
DXYZ	0.08d0	1.0d0	10.0d0	! difference in center of masses

Note that the values are specified for the type of collective variable, not the collective variable itself. The type of the collective variable is the same as that specified after the keywords <ncons> and <params_cons>. The full set of five lines should always be given, even if only some of them are used.

In the case of TAMD and LOGMFD, the mass of the fictitious CV particle \mathcal{M}_d is determined by $\Delta\mathcal{R}_d^{\text{ref}}$ from Eq.(270) or (276) in Sec. 11.16; the resulting mass values are written in *logmfd.out*.

9.12 Additional keywords in DESCENT, ASCENT and AUTO

9.12.1 <algo_ascent_afed>

Used in the cases of <method> = AFED, with <afed_type> = ASCENT. This keyword sets the numerical algorithm for ascent search in adiabatic free energy dynamics. The data should be provided in the first line (character).

- GAD: Gentlest ascent dynamics.
- EVF: Eigenvector following.

The default value is “GAD”.

9.12.2 <ascent_sampling_afed>

Used in the cases of <method> = AFED, with <afed_type> = ASCENT. This keyword sets the sampling method of the Hessian term for the ascent search in adiabatic free energy dynamics. The data should be provided in the first line (character).

- ANALYTICAL: Analytical evaluation of the Hessian term.
- NUMERICAL: Numerical evaluation of the Hessian term using finite difference method.

The default value is “NUMERICAL”. For numerical evaluation, the finite difference value can be controlled by the keyword <fdiff_sampling_afed>.

9.12.3 <dt_ascent_afed>

Used in the cases of <method> = AFED with <afed_type> = ASCENT, AUTO. This keyword sets the initial step size for ascent search in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “1.0” femtoseconds.

9.12.4 <dt_conv_afed>

Used in the cases of <method> = AFED, with <afed_type> = DESCENT, ASCENT, AUTO. This keyword sets the AFED step size at convergence in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “0.05” (dimensionless).

9.12.5 <dt_damp_afed>

Used in the cases of <method> = AFED, with <afed_type> = DESCENT, ASCENT, AUTO. This keyword sets the damping factor, λ^{afed} , for the AFED step size when approaching a stationary point in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “0.7” (dimensionless).

9.12.6 <dt_descent_afed>

Used in the cases of <method> = AFED with <afed_type> = DESCENT, AUTO. This keyword sets the initial step size for descent search in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “0.5” femtoseconds.

9.12.7 <fdiff_sampling_afed>

Used in the cases of <method> = AFED, with <afed_type> = ASCENT and <ascent_sampling_afed> = NUMERICAL. This keyword sets the finite difference value used to numerically sample the Hessian term for the ascent search in adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “5.0”.

9.12.8 <gamma_ascent_afed>

Used in the cases of <method> = AFED, with <afed_type> = ASCENT, AUTO. This keyword sets the parameter γ^{gad} in the ascent search of adiabatic free energy dynamics, in hartree. The data should be provided in the first line (real number). The default value is “1.0” hartree.

9.12.9 <niter_gad2evf_afed>

Used in the cases of <method> = AFED with <afed_type> = ASCENT, AUTO and <algo_ascent_afed> = GAD. This keyword sets the iteration step to switch the algorithm from GAD to EVF when approaching the saddle point. The data should be provided in the first line (integer). The default value is “80”.

9.12.10 <niter_refresh_afed>

Used in the cases of <method> = AFED with <afed_type> = ASCENT, AUTO and <algo_ascent_afed> = GAD, EVF. This keyword sets the step interval for numerical Hessian calculations, refreshing the guiding vector to the lowest eigenvector of the Hessian. The data should be provided in the first line (integer). The default value is “20”.

9.12.11 <nmiss_auto_afed>

Used in the cases of <method> = AFED, with <afed_type> = AUTO. This keyword sets the number of consecutive missed shots from each minimum, N_{miss} . Exceeding this number, a new shooting starts from another minimum, which is listed in the file *auto.ini* with the label “EQ”. The data should be provided in the first line (integer). The default value is “5”.

9.12.12 <ioption_eigen_afed>

Used in the cases of <method> = AFED with <afed_type> = DESCENT, ASCENT, AUTO. This keyword sets the option for eigenvalue calculations at the free energy stationary points found in adiabatic free energy dynamics. The data should be provided in the first line (integer).

- 1: Eigenvalues are calculated.
- 0: Eigenvalues are not calculated.

The default value is “1”.

9.12.13 <nshot_auto_afed>

Used in the cases of <method> = AFED, with <afed_type> = AUTO. This keyword sets the maximum number of shots from each minimum, N_{shot} . Exceeding this number, the new shooting starts from another minimum, which is listed in the file *auto.ini* with the label “EQ”. The data should be provided in the first line (integer). The default value is “10”.

9.12.14 <radius_auto_afed>

Used in the cases of <method> = AFED, with <afed_type> = AUTO. This keyword sets the reference radius in the automated search of adiabatic free energy dynamics. The data should be provided in the first line (real number). The default value is “5.0”.

9.12.15 <iprint_xyz_afed>

Used in the cases of <method> = AFED, with <afed_type> = DESCENT, ASCENT, AUTO, TAMD, LOGMFD. This keyword sets the print interval of molecular snapshots written to the file *afed.xyz*. A snapshot is taken at the final step of the MD run per AFED step in adiabatic free energy dynamics, temperature accelerated molecular dynamics, and logarithmic mean force dynamics. The data should be provided in the first line (integer). The default value is “1” step (every AFED step). Set to “-1” to disable printing.

9.12.16 <iprint_test_afed>

Used in the cases of <method> = AFED, with <afed_type> = DESCENT, ASCENT, AUTO, TAMD, LOGMFD. This keyword sets the print interval of the mean force for the convergence test. The data should be provided in the first line (integer). The default value is “100” steps.

9.13 Additional keywords in TAMD and LogMFD

9.13.1 <alpha_logmfd>

Used in the case of <method> = AFED and <afed_type> = LOGMFD. This keyword sets the parameter α_{\log} ($= \alpha' / k_B T > 0$) for logarithmic mean force dynamics (α' is specified by this keyword). The data should be provided in the first line (dimensionless real number). Note that T is the physical temperature given by the keyword <temperature>.

9.13.2 <dt_logmfd>

Used in the case of <method> = AFED with <afed_type> = LOGMFD. This keyword sets the step size in logarithmic mean force dynamics, Δt_{cv} . The data should be provided in the first line (real number). The default value is “1.0”.

9.13.3 <dt_tamd>

Used in the case of <method> = AFED with <afed_type> = TAMD. This keyword sets the step size in temperature accelerated molecular dynamics, Δt_{cv} . The data should be provided in the first line (real number). The default value is “1.0”.

9.13.4 <fenergy_ini_logmfd>

Used in the case of <method> = AFED, with <afed_type> = LOGMFD. This keyword sets the initial value of the free energy estimated on-the-fly during the LogMFD run, which defines the origin of the energy scale. The default value is “0.005” hartree.

9.13.5 <fenergy_ini_tamd>

Used in the case of <method> = AFED, with <afed_type> = TAMD. This keyword sets the initial value of the free energy estimated on-the-fly during the TAMD run, which defines the origin of the energy scale. The default value is “0.0” hartree.

9.13.6 <gamma_logmfd>

Used in the case of <method> = AFED and <afed_type> = LOGMFD. This keyword sets the parameter γ_{\log} ($= \gamma' k_B T$) in logarithmic mean force dynamics (γ' is specified by this keyword). The data should be provided in the first line (dimensionless real number). The default setting is “ $\gamma' = 1/\alpha'$ ”, i.e., “ $\gamma_{\log} = 1/\alpha_{\log}$ ”. Note that T is the physical temperature given by the keyword <temperature>.

9.13.7 <ioption_afed>

Used in the cases of <method> = AFED, with <afed_type> = TAMD, LOGMFD. This keyword sets the weight W_l in parallel-dynamics discussed in Sec.11.16. With “1”, $W_l = 1$ for all the beads (replicas), while W_l is set by Eq.(237) (see Sec.11.16) with “2”. The default value is “1” ($W_l = 1$ for all the beads).

9.13.8 <logmfd_type>

Used in the case of <method> = AFED and <afed_type> = LOGMFD. This keyword sets the type of equation of motion for logarithmic mean force dynamics. The data should be provided in the first line. Choose one of the following:

- <logmfd_type> = NVE: no thermostat is used for fictitious particles.
- <logmfd_type> = NVT: single Nosé-Hoover thermostat is attached to fictitious particles.

- `<logmfd_type>` = VS: velocity scaling (Gaussian thermostat) applied to fictitious particles.

The default value is “NVE”.

9.13.9 `<mass_afed>`

Used in the case of `<method>` = AFED with `<afed_type>` = TAMD, LOGMFD. This keyword sets the parameters that re-adjust the mass of the fictitious CV particles. If “AUTOMATIC” is set, the mass is given by default using the keyword `<params_afed>` (see Sec.9.11.7). If “MANUAL” is set, the mass given by `<params_afed>` is multiplied by scaling factors. The scale factor of the mass of the i th particle should be provided in the i th line below “MANUAL”. The mass values used in the LogMFD/TAMD run can be checked in the output file *logmfd.out*.

9.13.10 `<niter_bath_afed>`

Used in the case of `<method>` = AFED with `<afed_type>` = TAMD, LOGMFD and `<tamd_type>` = NVT or `<logmfd_type>` = NVT. This keyword sets the number of AFED steps n_b , which determines the time scale, $\tau_{cv} = n_b \Delta t_{cv}$, of the Nosé-Hoover thermostat attached to fictitious particles in TAMD and LogMFD in the NVT ensemble. This sets the mass of the Nosé-Hoover thermostat, Q , from the relation $Q = D k_B T \tau_{cv}^2$, where D is the number of fictitious particles. The data should be provided in the first line (real number). The default value is “100” (dimensionless). Note that n_b should be appropriately adjusted according to the value of $\Delta \mathcal{R}_d^{\text{ref}}$; n_b should be increased when $\Delta \mathcal{R}_d^{\text{ref}}$ is decreased, and vice versa.

9.13.11 `<tamd_type>`

Used in the case of `<method>` = AFED and `<afed_type>` = TAMD. This keyword sets the type of equation of motion for temperature accelerated molecular dynamics. The data should be provided in the first line. Choose one of the following:

- `<tamd_type>` = NVE: no thermostat is used for fictitious particles.
- `<tamd_type>` = NVT: single Nosé-Hoover thermostat attached to fictitious particles.
- `<tamd_type>` = VS: velocity scaling (Gaussian thermostat) applied to fictitious particles.

The default value is “NVT”.

9.13.12 `<temperature_tamd>`

Used in the case of `<method>` = AFED with `<afed_type>` = TAMD. This keyword sets the temperature T_{tamd} for temperature accelerated molecular dynamics. The data should be provided in the first line (real number) in kelvin. The default value is “3000.0” kelvin.

9.13.13 `<start_afed>`

Used in the case of `<method>` = AFED with `<afed_type>` = TAMD, LOGMFD. This keyword sets the initial position and velocity of each fictitious CV particle (variable). If “MANUAL” is set in the first line, the initial position and velocity of each CV particle should be provided for each line (see Sec.3.13.6). If “AUTOMATIC” is set instead (default), the initial positions are automatically estimated from the initial configuration of the MD system, and the initial velocities are set randomly corresponding to the physical temperature given by the keyword `<temperature>`.

9.13.14 <start_therm_afed>

Used in the case of <method> = AFED with <afed_type> = TAMD, LOGMFD. This keyword sets the initial position and velocity of the Nosé-Hoover thermostat variable η . If “MANUAL” is specified, its initial position and velocity should be provided (default setting is “MANUAL” with $\eta(0) = 0$ and $v_\eta(0) = 0$). If “AUTOMATIC” is specified instead, η is set to “0” and v_η is automatically set according to the preset physical temperature.

10 Files

The files can be categorized as follows.

- Input files: named **.dat*.
- Output files: most are named **.out*. The exceptions are the trajectory files in xyz format, *trj.xyz* and *calc.xyz*.
- Restart files: named **.ini*. These files are created at the end of each run and include the data required to restart the next run. In principle, these files do not need to be modified by the user.

10.1 Input files

10.1.1 *input.dat*

Used in ALL calculations. This file contains the main input data that are necessary to set up the calculation. Keywords are represented by brackets, <...>. The line(s) following each keyword specify the data read by the code. All remaining parts are treated as comment lines. The order of the keywords is completely arbitrary.

This file is expected to be edited by the user. However, it does not need to contain the full set of keywords, since keywords that are not present in this file are complemented by those in *input_default.dat*. Details of each keyword are listed in Section 9.

10.1.2 *input_default.dat*

Used in ALL calculations. This file contains the default values of the keywords. Keywords that are not defined in *input_default.dat* must be explicitly specified in *input.dat*.

It is expected that this file is copied and kept unchanged. This file can be useful as a reference when the user edits *input.dat*.

10.1.3 *centroid.dat*

NOTE: This file is used only for <input_style> = OLD (default). For <input_style> = NEW, *structure.dat* is used instead.

Used in ALL calculations for an initial start, except for <method> = STRING. This file contains a list of Cartesian coordinates of atoms in atomic units (bohr), given as:

```
x  y  z  (of atom 1)
x  y  z  (of atom 2)
...
```

The number of lines must be equal to the number of atoms.

10.1.4 *eam.dat*

Used for EAM and ADP potentials, when <ipotential> = EAM, ADP, PAIR. This file contains all parameters necessary for the EAM and ADP potentials, as well as tabulated pair potentials. See Section 9.8.

10.1.5 *tersoff.dat*

Used for Tersoff potentials, when `<ipotential> = TERSOFF`. This file contains all parameters necessary for Tersoff potentials. See Section 9.9.

10.1.6 *gamess.dat*

Used for the GAMESS potential, when `<ipotential> = GAMESS`. This file must be a GAMESS input file. See Section 4.3.6.

10.1.7 *g03.dat*

Used for the GAUSSIAN 03 potential, when `<ipotential> = G03`. This file must be a G03 input file. See Section 4.3.7.

10.1.8 *g98.dat*

Used for the GAUSSIAN 98 potential, when `<ipotential> = G98`. This file must be a G98 input file. See Section 4.3.7.

10.1.9 *mm.dat*

Used for the MM potential, when `<ipotential> = MM`. This file contains all parameters necessary for the MM potential. See Section 9.7.

10.1.10 *molpro.dat*

Used for the MOLPRO potential, when `<ipotential> = MOLPRO`. This file must be a MOLPRO input file. See Section 4.3.8. See also Section 3.19.1 for performing nonadiabatic dynamics calculations, i.e., surface hopping dynamics and mean-field dynamics, using MOLPRO.

10.1.11 *mopac.dat*

Used for the MOPAC potential, when `<ipotential> = MOPAC`. This file must be a MOPAC input file. See Section 4.3.9.

10.1.12 *orca.dat*

Used for the ORCA potential, when `<ipotential> = ORCA`. This file must be an ORCA input file. See Section 4.3.11.

10.1.13 *structure.dat*

Used in ALL calculations for an initial start. This file contains a list of Cartesian coordinates of atoms in XYZ format, given as:

```
N
unit
symbol  x  y  z
symbol  x  y  z
.....
```

In the first line, `N` is the number of atoms (integer). In the second line, `unit` is an option (character), either `BOHR` or `ANGSTROM`. In lines 3–($N + 2$), `symbol` denotes the atomic species, and x , y , and z are the Cartesian coordinates of the corresponding atoms.

In the case of `<method> = STRING, OMOPT`, the list of Cartesian coordinates is duplicated, corresponding to the reactant and the product. An initial guess is generated as a linear path connecting the reactant and the product.

In general, this file must be prepared to start a new calculation. However, when the restart file *geometry.ini* is present, this file is no longer required. The PIMD code always searches for the restart file *geometry.ini* before looking for *centroid.dat*.

NOTE: This file is used only for `<input_style> = NEW`. For `<input_style> = OLD` (default), *centroid.dat* is used instead.

10.1.14 *turbo.dat*

Used for TURBOMOLE, when `<ipotential> = TURBOMOLE`. This file contains the set of TURBOMOLE execution commands used to perform energy and force calculations, along with the corresponding TURBOMOLE control files. If this file is not found in the execution directory, the code alternatively searches for the required information in *input.dat*, or otherwise in *input_default.dat*, following the keyword `<turbo_command>`. The following lines should be specified:

- Hartree-Fock calculation:

```
dscf    control.1
grad    control.2
```

- MP2 calculation:

```
dscf    control.1
mpgrad  control.2
```

- DFT calculation:

```
dscf    control.1
grad    control.2
```

- RI-MP2 calculation:

```
dscf    control.1
rimp2    control.2
```

- RI-DFT calculation:

```
ridft    control.1
rdgrad    control.2
```

- RI-CC2 calculation:

```
dscf    control.1
ricc2    control.2
```

10.1.15 TAMD

(For the initial run) The following files are required:

input.dat, *input_default.dat*, *mm.dat*, *centroid.dat*

(If `<input_style> = NEW`, *structure.dat* should be prepared instead of *centroid.dat*.)

(For restart) The following files are required:

input.dat, *input_default.dat*, *mm.dat*, *centroid.dat*, *geometry.ini*, *bath.ini*, *step.ini*, *afed.ini*

(If `<input_style> = NEW`, *structure.dat* should be prepared instead of *centroid.dat*.)

Optional files: *average.ini*

Note that all *.ini files are overwritten at the end of the run.

10.1.16 LogMFD

(For the initial run) The following files are required:

input.dat, *input_default.dat*, *mm.dat*, *centroid.dat*

(If `<input_style> = NEW`, *structure.dat* should be prepared instead of *centroid.dat*.)

(For restart) The following files are required:

input.dat, *input_default.dat*, *mm.dat*, *centroid.dat*, *geometry.ini*, *bath.ini*, *step.ini*, *afed.ini*

(If `<input_style> = NEW`, *structure.dat* should be prepared instead of *centroid.dat*.)

Optional files: *average.ini*

Note that all *.ini files are overwritten at the end of the run.

10.2 Output files

10.2.1 *afed.out*

Used when `<method> = AFED`. This file contains the trajectory of the collective variables. In n -dimensional dynamics, each of the n printed lines corresponds to one collective variable. Each line contains the following columns:

- `<afed_type> = GRAD, TEST`:
 - COLUMN 1 (**step**): AFED step number.
 - COLUMN 2 (**st**): AFED status.
 - * **GR**: mean-force calculation.
 - * **TE**: convergence-test calculation.
 - COLUMN 3 (**cv**): collective-variable index, d .
 - COLUMN 4 (**r-ideal**): reference position of the fictitious particle, $\mathcal{R}_d^{\text{ref}}$.
 - COLUMN 5 (**r-mean**): mean position of the collective variable, \mathcal{R}_d .
 - COLUMN 6 (**f-energy**): free energy in hartree.
 - COLUMN 7 (**-df/dr**): mean force on the fictitious particle, $-\partial A_{\text{cv}}/\partial \mathcal{R}_d^{\text{ref}}$.
 - COLUMN 8 (**dr/dt**): velocity of the fictitious particle, $d\mathcal{R}_d^{\text{ref}}/dt$.
 - COLUMN 9 (**dt**): AFED step size, Δt_{cv} .
- `<afed_type> = ASCENT, DESCENT, AUTO`:
 - COLUMN 1 (**step**): AFED step number.
 - COLUMN 2 (**st**): AFED status.
 - * **DE**: descent trajectory to a free-energy minimum.
 - * **D1**: descent trajectory started in the $+\mathbf{n}$ direction from a free-energy saddle point to a free-energy minimum.

- * **D2**: descent trajectory started in the $-\mathbf{n}$ direction from a free-energy saddle point to a free-energy minimum.
- * **AS**: ascent trajectory to search for a free-energy saddle point.
- * **EQ**: free-energy minimum point.
- * **TS**: free-energy saddle point.
- * **HI**: terminated because the free energy exceeds the limit defined by `<fenergy_max_afed>`.
- * **OB**: terminated because the CV is out of the bounds defined by `<params_afed>`.
- COLUMN 3 (**cv**): collective-variable index, d .
- COLUMN 4 (**r-ideal**): reference position of the fictitious particle, $\mathcal{R}_d^{\text{ref}}$.
- COLUMN 5 (**r-mean**): mean position of the collective variable, \mathcal{R}_d .
- COLUMN 6 (**f-energy**): free energy in hartree.
- COLUMN 7 (**-df/dr**): mean force on the fictitious particle, $-\partial A_{\text{cv}}/\partial \mathcal{R}_d^{\text{ref}}$.
- COLUMN 8 (**dr/dt**): velocity of the fictitious particle, $d\mathcal{R}_d^{\text{ref}}/dt$.
- COLUMN 9 (**dt**): AFED step size, Δt_{cv} .
- COLUMN 10 (**n**): unit vector, \mathbf{n} .
 - * In an ascent trajectory, \mathbf{n} is the guiding vector.
 - * In a descent trajectory, \mathbf{n} is parallel to the mean force $-\partial A_{\text{cv}}/\partial \mathcal{R}_d^{\text{ref}}$.
- COLUMN 11 (**xi**): angle ξ .
- `<afed_type>` = TAMD, LOGMFD:
 - COLUMN 1 (**step**): AFED step number.
 - COLUMN 2 (**st**): AFED status.
 - * **TA**: TAMD trajectory.
 - * **LO**: LogMFD trajectory.
 - * **HI**: terminated because the free energy exceeds the limit defined by `<fenergy_max_afed>`.
 - * **OB**: terminated because the CV is out of the bounds defined by `<params_afed>`.
 - COLUMN 3 (**cv**): collective-variable index, d .
 - COLUMN 4 (**r-ideal**): reference position of the fictitious particle, $\mathcal{R}_d^{\text{ref}}$.
 - COLUMN 5 (**r-mean**): mean position of the collective variable, \mathcal{R}_d .
 - COLUMN 6 (**f-energy**): free energy in hartree.
 - COLUMN 7 (**-df/dr**): mean force on the fictitious particle, $-\partial A_{\text{cv}}/\partial \mathcal{R}_d^{\text{ref}}$.
 - COLUMN 8 (**dr/dt**): velocity of the fictitious particle, $d\mathcal{R}_d^{\text{ref}}/dt$.
 - COLUMN 9 (**h-energy**): total Hamiltonian, H^{tamd} or H^{log} .
 - COLUMN 10 (**temp**): instantaneous temperature of the fictitious particle.

The file is appended when a run is restarted.

10.2.2 *afed.xyz*

This file contains molecular snapshots at the end of each AFED iteration in xyz format. For multiple beads, the data are printed `<nbead>` times. The file is appended when a run is restarted.

10.2.3 *afed.weight.out*

Used when `<method>` = AFED with `<afed_type>` = LOGMFD. The weight W_l in parallel dynamics (discussed in Sec. 11.16) is written at every AFED step to *afed.weight.out*. The file is appended when a run is restarted.

10.2.4 *alc.out*

Used when `<method> = PIHMC` with `<irem_type> = HX`. This file contains the alchemical analysis. The print interval can be controlled by the keyword `<iprint_alc>`.

- For `<method> = PIHMC` with `<irem_type> = HX`:
 - COLUMN 1: step number.
 - COLUMN 2: free-energy difference.
 - COLUMN 3: potential-energy difference.
 - COLUMN 4: entropy difference.
 - COLUMN 5: cumulative average of the free-energy difference.
 - COLUMN 6: cumulative average of the potential-energy difference.
 - COLUMN 7: cumulative average of the entropy difference.

10.2.5 *best.out*

Used when `<ipotential> = QMMM` and `<ioption_best> = 1`. This file contains data from the BEST method. The print interval can be controlled by the keyword `<iprint_best>`. Each line contains the following five columns:

- COLUMN 1: step number.
- COLUMN 2: number of boundary atoms.
- COLUMN 3: number of boundary atoms in the QM region.
- COLUMN 4: number of boundary atoms in the MM region.
- COLUMN 5: bias potential in BEST.

For `<nbead> = n` , the data are repeated n times.

10.2.6 *bond.out*

Used when `<method> = MD, PIMD, PIHMC, REHMC`. This file contains bond-length data. The print interval can be controlled by the keyword `<iprint_bond>`.

- For `<method> = MD, PIMD` and `<method> = PIHMC` with `iorder_hmc = 2`:
 - COLUMN 1: step number.
 - COLUMNS 2–3: atomic-species pair defined in *structure.dat*.
 - COLUMNS 4–6: bond lengths associated with the Kubo, centroid, and bead correlation functions, respectively.
 - COLUMNS 7–9: mean-square bond lengths associated with the Kubo, centroid, and bead correlation functions, respectively.
 - COLUMNS 10–12: cumulative averages of the bond lengths associated with the Kubo, centroid, and bead correlation functions, respectively.
 - COLUMNS 13–15: cumulative averages of the mean-square bond lengths associated with the Kubo, centroid, and bead correlation functions, respectively.
- For `<method> = PIHMC` with `iorder_hmc = 4`:
 - COLUMN 1: step number.

- COLUMNS 2–3: atomic-species pair defined in *structure.dat*.
 - COLUMN 4: bond length.
 - COLUMN 5: second-order contribution to the bond length.
 - COLUMN 6: fourth-order contribution to the bond length.
 - COLUMN 7: cumulative average of the bond length.
 - COLUMN 8: cumulative average of the second-order contribution.
 - COLUMN 9: cumulative average of the fourth-order contribution.
- For `<method> = REHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: bead number.
 - COLUMNS 3–4: atomic-species pair defined in *structure.dat*.
 - COLUMN 5: bond length.
 - COLUMN 6: mean-square bond length.
 - COLUMN 7: cumulative average of the bond length.
 - COLUMN 8: cumulative average of the mean-square bond length.

NOTE: This analysis is not recommended when the number of atomic species is large, since the computational cost grows quadratically.

10.2.7 *box.dcd*

Used when periodic boundary conditions are applied.

This binary file contains the periodic boundary box and the virial of the system. The print interval can be controlled by the keyword `<iprint_dcd>`.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the box information.
- BYTE 269–272: (int) number of beads.

After the header, the box data are organized as follows:

- BYTE 1–4: (int) size of the next read (144).
- BYTE 5–76: (doubles [8 bytes]) box information ordered as A(1), B(1), C(1), A(2), ...
- BYTE 77–148: (doubles [8 bytes]) virial ordered in the same way.

These data blocks are repeated for each bead when multiple beads are used.

10.2.8 *box.out*

Used when `<ensemble> = NPH, NPT, NTH, NTT`. This file contains the trajectory of the box matrix. The print interval can be controlled by the keyword `<iprint_box>`.

- For `<method> = NPH, NPT`:
 - LINE 1, COLUMN 1: step number.
 - LINE 1, COLUMN 2: volume.

- LINE 1, COLUMN 3: volume average.
 - LINES 2–4, COLUMNS 1–3: box matrix.
 - LINES 2–4, COLUMNS 4–6: average box matrix.
 - LINE 5, COLUMN 1: step number.
 - LINE 5, COLUMN 2: isotropic pressure.
 - LINE 5, COLUMN 3: average isotropic pressure.
 - LINES 6–8, COLUMNS 1–3: pressure matrix.
 - LINES 6–8, COLUMNS 4–6: average pressure matrix.
- For `<method> = NTH, NTT`:
 - LINE 1, COLUMN 1: step number.
 - LINE 1, COLUMN 2: volume.
 - LINE 1, COLUMN 3: volume average.
 - LINES 2–4, COLUMNS 1–3: box matrix.
 - LINES 2–4, COLUMNS 4–6: average box matrix.
 - LINE 5, COLUMN 1: step number.
 - LINE 5, COLUMN 2: isotropic pressure.
 - LINE 5, COLUMN 3: average isotropic pressure.
 - LINES 6–8, COLUMNS 1–3: pressure matrix.
 - LINES 6–8, COLUMNS 4–6: average pressure matrix.
 - LINE 9, COLUMN 1: step number.
 - LINE 9, COLUMN 2: isotropic stress.
 - LINE 9, COLUMN 3: average isotropic stress.
 - LINES 10–12, COLUMNS 1–3: stress matrix.
 - LINES 10–12, COLUMNS 4–6: average stress matrix.
 - LINE 13, COLUMN 1: step number.
 - LINE 13, COLUMN 2: isotropic strain.
 - LINE 13, COLUMN 3: average isotropic strain.
 - LINES 14–16, COLUMNS 1–3: strain matrix.
 - LINES 14–16, COLUMNS 4–6: average strain matrix.

10.2.9 *calc.xyz*

Used by the analysis code *calc.x*. This file contains the trajectory in xyz format. The print interval can be controlled by the keyword `<jprint_xyz>` in *calc.dat*. For `<nbead> = n`, the data are repeated *n* times. The output is controlled by the keywords `<jformat_xyz>` and `<jorigin_xyz>` in *calc.dat*.

10.2.10 *charges.dcd*

This binary file is currently only output for CP2K calculations. It contains Mulliken and Hirshfeld charges of each atom for the frames printed by PIMD, according to the `<iprint_dcd>` keyword.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the box information.
- BYTE 269–272: (int) number of beads.

After the header, the data are organized as follows, where N is four times the number of atoms:

- BYTE 1–4: (int) size of the next read ($8 \times \text{numatoms} \times \text{numbeads}$).
- NEXT N BYTES: (floats [4 bytes]) Mulliken charges.
- NEXT N BYTES: (floats [4 bytes]) Hirshfeld charges.

The two data blocks are repeated for each bead when multiple beads are used.

10.2.11 *cons.out*

Used when `<method> = MD` with `<cons> = 1`. This file contains constraint information. The print interval can be controlled by the keyword `<iprint_cons>`. When n constraints are applied, each of the n printed lines corresponds to one constraint. Each line contains the following ten columns:

- For `<method> = MD`:
 - COLUMN 1: step number.
 - COLUMN 2: equilibrium constraint value.
 - COLUMN 3: instantaneous constraint value.
 - COLUMN 4: constraint force.
 - COLUMN 5: average of column 3.
 - COLUMN 6: average of column 4.
 - COLUMN 7: potential of mean force with respect to the constraints.
 - COLUMN 8: potential of mean force with respect to the constraints, including the correction.
 - COLUMN 9: shortest distance between two groups restrained by the coordination number (zero if no coordination restraint is set).
 - COLUMN 10: average of column 9.

For `<nbead> = n` , the data are repeated n times.

10.2.12 *cv.out*

Used when `<method> = MTD`. This file contains the trajectory of the collective variables. The print interval can be controlled by the keyword `<iprint_cv>`. In n -dimensional metadynamics, each of the n printed lines corresponds to one collective variable. Each line contains the following three columns:

- For `<method> = MTD`:
 - COLUMN 1: step number.
 - COLUMN 2: collective-variable coordinate.
 - COLUMN 3: fictitious-particle coordinate.

For metadynamics with n walkers, the data are repeated n times.

10.2.13 *dip.dcd*

Used for all methods. This binary file contains the dipole moment. The print interval can be controlled by the keyword `<iprint_dcd>`. If the dipole moment is not computed in a given method, zeros are written using the same format described below.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the dipole information.
- BYTE 269–272: (int) number of beads.

After the header, the dipole data are organized as follows:

- BYTE 1–4: (int) size of the next read (24).
- BYTE 5–28: (doubles [8 bytes]) X , Y , and Z components of the molecular dipole moment.

This block is repeated for each bead.

10.2.14 *dipole.out*

Used for all methods. This file contains the dipole moment. The print interval can be controlled by the keyword `<iprint_dip>`.

- For all methods:
 - COLUMN 1: step number.
 - COLUMNS 2–4: x , y , and z components of the dipole moment, (μ_x, μ_y, μ_z) , in atomic units.

For `<nbead> = n` , the data are repeated n times.

10.2.15 *dipole_scan.out*

Used when `<method> = SCAN`. This file contains the dipole moment.

- The following data are repeated for each static calculation:
 - COLUMN 1: step number.
 - COLUMNS 2–4: x , y , and z components of the dipole moment, (μ_x, μ_y, μ_z) , in atomic units.

For multiple beads with `<nbead> = P` , the data are repeated P times.

10.2.16 *dual.out*

Used when `<method> = PIHMC` with `<ipotential> = DUAL`. This file contains energies from the high- and low-level methods. The print interval can be controlled by the keyword `<iprint_dual>`.

- For `<method> = PIHMC` with `<ipotential> = DUAL`:
 - COLUMN 1: step number.
 - COLUMN 2: high-level potential energy.
 - COLUMN 3: low-level potential energy.

For `<nbead> = n` , the data are repeated n times.

10.2.17 *eavg.out*

Used when `<method> = MD, PIMD, PIHMC, REHMC`. This file contains energy components. The print interval can be controlled by the keyword `<iprint_eavg>`.

- For `<method> = MD`:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy.
 - COLUMN 3: kinetic energy.
 - COLUMN 4: total energy (columns 2 + 3).
 - COLUMNS 5–7: cumulative averages of columns 2–4.
 - COLUMN 8: specific heat (NVT) in k_B units.
- For `<method> = PIMD, PIHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy.
 - COLUMN 3: kinetic energy (primitive estimator).
 - COLUMN 4: kinetic energy (virial estimator).
 - COLUMN 5: total energy (columns 2 + 4).
 - COLUMNS 6–9: cumulative averages of columns 2–5.
 - COLUMN 10: specific heat (NVT) in k_B units.
- For `<method> = REHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: bead number.
 - COLUMN 3: potential energy.
 - COLUMN 4: cumulative average of the potential energy.

10.2.18 *forces.out*

Used when `<method> = STATIC`. This file contains the potential energy and forces.

- For all methods:
 - LINE 1: potential energy in atomic units (hartree).
 - LINE 2: x , y , and z components of the force (negative of the potential gradient) on atom 1 in atomic units (hartree/bohr).
 - LINE 3: x , y , and z components of the force (negative of the potential gradient) on atom 2 in atomic units (hartree/bohr).
 - LINE n : x , y , and z components of the force (negative of the potential gradient) on atom $(n - 1)$ in atomic units (hartree/bohr).

Forces are listed in the same order as in the initial geometry file *structure.dat*. For `<nbead> = n` , the data are repeated n times.

10.2.19 *forces_scan.out*

Used when `<method> = SCAN`. This file contains forces (negative of the potential gradient).

- The following data are repeated for each static calculation:
 - COLUMN 1: step number.
 - COLUMNS 2–4: x , y , and z force components in atomic units (hartree/bohr). The data consist of a block of n lines, where n is the number of atoms. Forces are listed in the order specified in *structure.dat*.

For multiple beads with `<nbead> = P`, the data are repeated P times.

10.2.20 *force.dcd*

This binary file contains the calculated forces for the system (negative of the potential gradient). The print interval can be controlled by the keyword `<iprint_dcd>`.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the force information.
- BYTE 269–272: (int) number of beads times number of atoms.

After the header, the force data are organized as follows. If periodic boundary conditions are present, the box information is written first, as described for *trj.dcd*. The force components are then written as follows, where N is four times the number of atoms times the number of beads:

- NEXT 4 BYTES: (int) size of the next read (number of beads times number of atoms).
- NEXT N BYTES: (floats [4 bytes]) x components of the forces.
- NEXT 4 BYTES: size of the previous read.

This block is repeated for the y and z force components. The order of atoms is the same as in *template.xyz*. All forces are given in atomic units.

10.2.21 *force_high.dcd*

This binary file contains the calculated forces for the system computed by the high-level potential. Used when `<ipotential> = DUAL`. (negative of the potential gradient). The print interval can be controlled by the keyword `<iprint_dcd>`. This file follows the same layout as *force.dcd* described above.

10.2.22 *force_low.dcd*

This binary file contains the calculated forces for the system computed by the low-level potential. Used when `<ipotential> = DUAL`. (negative of the potential gradient). The print interval can be controlled by the keyword `<iprint_dcd>`. This file follows the same layout as *force.dcd* described above.

10.2.23 *hessian.out*

Used when `<method> = NMA, PHONON`. This file contains the Hessian matrix, i.e., the second derivative of the potential energy, $H_{ij} = \frac{\partial^2 V}{\partial r_i \partial r_j}$.

- For `<method> = NMA, PHONON`:
 - COLUMN 1: index i .

- COLUMN 2: index j .
- COLUMN 3: H_{ij} .

The data are printed for $1 \leq i, j \leq 3N$, where N is the number of atoms.

10.2.24 *logmfd.out*

Used when `<method> = AFED` with `<afed_type> = LOGMFD` or `TAMD`. This file contains the time series of the collective variables, fictitious CV particles, mean forces, thermostat variables, and free energy.

- For `<afed_type> = LOGMFD, TAMD`:
 - COLUMN 1: AFED step number.
 - COLUMN 2: free energy in hartree.
 - COLUMN 3: temperature of the CV particles in kelvin.
 - COLUMN 4: Nosé–Hoover variable η .
 - COLUMN 5: Nosé–Hoover variable v_η .
 - COLUMN $(4d+2)$: $R_d(\mathbf{r})$ for the d -th CV, averaged over N_m MD steps (`<nstep_pro_afed> = N_m`).
 - COLUMN $(4d+3)$: \mathcal{R}_d for the d -th fictitious CV particle.
 - COLUMN $(4d+4)$: mean force acting on \mathcal{R}_d for the d -th CV ($1 \leq d \leq n$).

Note that each line corresponds to one AFED step, in contrast to *afed.out*. Mass values for the CV particles and thermostats are printed at the beginning of the file. The file is appended when a run is restarted.

10.2.25 *mech.out*

Used when `<method> = MTD`. This file contains energy components when an external force is applied. The print interval can be controlled by the keyword `<iprint_mech>`.

- For `<method> = MTD`:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy of the system.
 - COLUMN 3: potential energy due to the external force.
 - COLUMN 4: total potential energy (columns 2 + 3).

10.2.26 *meta.out*

Used when `<method> = MTD`. This file contains energy components in metadynamics. The print interval can be controlled by the keyword `<iprint_meta>`.

- For `<method> = MTD`:
 - ROW 1: step number, $\sum_l V_l^{\text{sys}}$, V^{hills} , $\sum_l V_l^{\text{sys}} + V^{\text{hills}}$.
 - ROW 2: step number, $\sum_l V_l^{\text{cv}}$, V^{cor} , $\sum_l K_l^{\text{sys}}$.
 - ROW 3: step number, $\sum_l K_l^{\text{cv}}$, $H_{\text{bath}}^{\text{sys}}$, $H_{\text{bath}}^{\text{cv}}$.
 - ROW 4: step number, H , T^{sys} , T^{cv} .

See Section 11.13 for definitions.

10.2.27 *mfe.out*

Used when `<method> = MFE`. This file contains data from mean-field dynamics. The print interval can be controlled by the keyword `<iprint_mfe>`.

- For `<method> = MFE`:
 - COLUMN 1: step number.
 - COLUMN 2: bead number.
 - COLUMN 3: state number.
 - COLUMN 4: 0 (reserved).
 - COLUMN 5: population (state coefficient) of each state.
 - COLUMN 6: occupation ratio of each state.
 - COLUMN 7: potential energy of each state.
 - COLUMN 8: average potential energy of each bead.

10.2.28 *nac.out*

Used when `<method> = TFS, MFE`. This file contains nonadiabatic-dynamics data, including:

- For `<method> = TFS, MFE`:
 - BLOCK 1 (2 columns): real and imaginary parts of the state coefficients (m lines).
 - BLOCK 2 (1 column): adiabatic potentials (m^2 lines).
 - BLOCK 3 (3 columns): x , y , and z components of the forces ($n \times m^2$ lines).
 - BLOCK 4 (3 columns): x , y , and z components of the dipole moments (m lines).
 - BLOCK 5 (3 columns): x , y , and z components of the nonadiabatic coupling matrix ($n \times m^2$ lines).

Here, n is the number of atoms and m is the number of adiabatic states. The print interval can be controlled by the keyword `<iprint_nac>`. See the source files *standard_tfs.F* or *standard_mfe.F* for the detailed data format.

10.2.29 *oniom.out*

Used when `<ipotential> = ONIOM` or `<ipotential> = HYBRID` with `<hybrid> = ONIOM`. This file contains energy components from QM/MM calculations. The print interval can be controlled by the keyword `<iprint_oniom>`.

- For `<ipotential> = ONIOM`:
 - COLUMN 1: step number.
 - COLUMN 2: high-level potential energy of subsystem A (or A+L), in hartree.
 - COLUMN 3: low-level potential energy of subsystem A (or A+L), in hartree.
 - COLUMN 4: low-level potential energy of the whole system (A+B), in hartree.

10.2.30 *phonon_dos.out*

Used when `<method> = PHONON`. This file contains the vibrational density of states (VDOS).

- For `<method> = PHONON`:
 - COLUMN 1: phonon frequency in cm^{-1} .
 - COLUMN 2: density of states.

The frequency range can be controlled by the keyword `<dosrange_phonon>`.

10.2.31 *phonon_energy.out*

Used when `<method> = PHONON`. This file contains vibrational thermodynamic quantities.

- For `<method> = PHONON`:
 - COLUMN 1: temperature in kelvin.
 - COLUMN 2: internal energy (classical Boltzmann statistics).
 - COLUMN 3: Helmholtz free energy (classical Boltzmann statistics).
 - COLUMN 4: entropy (classical Boltzmann statistics).
 - COLUMN 5: internal energy (quantum Boltzmann statistics).
 - COLUMN 6: Helmholtz free energy (quantum Boltzmann statistics).
 - COLUMN 7: entropy (quantum Boltzmann statistics).

The temperature range can be controlled by the keyword `<temprange_phonon>`.

10.2.32 *phonon_kdisp.out*

Used when `<method> = PHONON`. This file contains phonon-dispersion data.

- For `<method> = PHONON`:
 - COLUMNS 1–3: components of the k vector in bohr^{-1} .
 - COLUMN 4: vibrational-branch index.
 - COLUMN 5: vibrational frequency in cm^{-1} .

The k -path (or k -points) is specified by the keyword `<kdisp_phonon>`.

10.2.33 *phonon_kdos.out*

Used when `<method> = PHONON`. This file contains phonon frequencies obtained for the VDOS calculation.

- For `<method> = PHONON`:
 - COLUMNS 1–3: components of the k vector in bohr^{-1} .
 - COLUMN 4: vibrational-branch index.
 - COLUMN 5: vibrational frequency in cm^{-1} .

The k -point sampling is specified by the keyword `<kdos_phonon>`.

10.2.34 *pot.dcd*

This binary file contains the potential energy of each bead, as well as the temperature and Hamiltonian of the whole system. The print interval can be controlled by the keyword `<iprint_dcd>`.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the potential information.
- BYTE 269–272: (int) number of beads.

After the header, the data are organized as follows:

- BYTE 1–4: (int) size of the next read ($8 \times (2 + \text{number of beads})$).
- BYTE 5–: (doubles [8 bytes]) potential energies of each bead, followed by the temperature and the Hamiltonian of the system.

Energies and the Hamiltonian are given in atomic units, and the temperature is given in kelvin.

10.2.35 *potential_scan.out*

Used when `<method> = SCAN`. This file contains the potential energy.

- The following data are repeated for each static calculation:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy in hartree.

For multiple beads with `<nbead> = P`, the data are repeated P times.

10.2.36 *qmmm.out*

Used when `<ipotential> = QMMM` or `<ipotential> = HYBRID` with `<hybrid> = QMMM`. This file contains energy components from QM/MM calculations. The print interval can be controlled by the keyword `<iprint_qmmm>`.

- For `<ipotential> = QMMM`:
 - COLUMN 1: step number.
 - COLUMN 2: high-level potential energy of subsystem A (or A+L), in hartree.
 - COLUMN 3: MM potential energy of subsystem A (or A+L), in hartree.
 - COLUMN 4: MM potential energy of the whole system (A+B), in hartree.

10.2.37 *rdf.out*

Used when `<method> = MD, PIMD, PIHMC, REHMC`. This file contains atomic pair-distribution data. The print interval can be controlled by the keyword `<iprint_rdf>`. Mesh parameters are specified by the keyword `<params_rdf>`.

- For `<method> = MD, PIMD, PIHMC, REHMC`:
 - COLUMN 1: step number.
 - COLUMNS 2–3: atomic-species pair defined in *structure.dat*.
 - COLUMN 4: mesh point of the interatomic distance.
 - COLUMN 5: pair distribution.
 - COLUMN 6: radial distribution (for `<iboundary> = 1`), or a copy of column 5 (for `<iboundary> = 0`).
 - COLUMN 7: coordination number (number of species 2 around species 1).
 - COLUMN 8: coordination number (number of species 1 around species 2).

For REHMC, the distribution of each bead i is printed in a separate file, *rdf.i.out*.

NOTE: This analysis is not recommended when the number of atomic species is large, since the computational cost grows quadratically.

10.2.38 *rec.out*

Used when `<method> = MTD`. This file contains the reconstructed free-energy surface from a metadynamics simulation. The print interval can be controlled by the keyword `<iprint_rec>`. In three-dimensional metadynamics, accuracy and efficiency can be controlled by the keyword `<cut_rec_3d>`. Before version 2.1.1:

- For `<method> = MTD`:

- COLUMN 1: mesh point of the collective variable.
- COLUMN 2: sum of history-dependent Gaussian hills (equal to the negative reconstructed free energy).

Version 2.1.1 and later:

- For `<method> = MTD`:
 - COLUMN 1: mesh point of the collective variable.
 - COLUMN 2: reconstructed free energy (positive sign).

10.2.39 *rgy.out*

Used when `<method> = PIMD, PIHMC`. This file contains the atomic radius of gyration. The print interval can be controlled by the keyword `<iprint_rgy>`.

- For `<method> = PIMD, PIHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: atomic species defined in *structure.dat*.
 - COLUMN 3: radius of gyration.

10.2.40 *standard.out*

Used in all cases. This file contains the standard output. The print interval can be controlled by the keyword `<iprint_std>`.

- For `<method> = MD`:
 - COLUMN 1: step number.
 - COLUMN 2: total energy.
 - COLUMN 3: potential energy.
 - COLUMN 4: instantaneous temperature.
 - COLUMN 5: wall-clock time.
- For `<method> = PIMD, CMD, RPMD, MTD`:
 - COLUMN 1: step number.
 - COLUMN 2: total energy.
 - COLUMN 3: bead-averaged potential energy.
 - COLUMN 4: instantaneous temperature.
 - COLUMN 5: wall-clock time.
- For `<method> = GEOOPT`:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy.
 - COLUMN 3: maximum displacement.
 - COLUMN 4: root-mean-square displacement.
 - COLUMN 5: maximum force.
 - COLUMN 6: root-mean-square force.

- For `<method> = SD`:
 - COLUMN 1: step number.
 - COLUMN 2: potential energy.
- For `<method> = REHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: step size.
 - COLUMN 3: acceptance ratio of the HMC trial move.
 - COLUMN 4: acceptance ratio of the REM trial move.
 - COLUMN 5: bead-averaged potential energy.
 - COLUMN 6: wall-clock time.
- For `<method> = PIHMC`:
 - COLUMN 1: step number.
 - COLUMN 2: step size.
 - COLUMN 3: acceptance ratio of the HMC trial move.
 - COLUMN 4: bead-averaged potential energy.
 - COLUMN 5: instantaneous temperature.
 - COLUMN 6: wall-clock time.
- For `<method> = STRING`:
 - COLUMN 1: step number.
 - COLUMN 2: image with the maximum potential energy.
 - COLUMN 3: maximum potential energy.
 - COLUMN 4: image with the minimum potential energy.
 - COLUMN 5: minimum potential energy.
 - COLUMN 6: wall-clock time.
- For `<method> = OMOPT`:
 - COLUMN 1: step number.
 - COLUMN 2: Onsager–Machlup action.
 - COLUMN 3: bead with the maximum potential energy.
 - COLUMN 4: maximum potential energy.
 - COLUMN 5: wall-clock time.

10.2.41 *string.out*

Used when `<method> = STRING`. This file contains the potential energies of the images along the string at each step.

- COLUMN 1: step number.
- COLUMN 2: image number.
- COLUMN 3: mass-weighted reaction coordinate [a.u.].
- COLUMN 4: potential energy [hartree].

The print interval can be controlled by the keyword `<iprint_str>`.

10.2.42 *string.xyz*

Used when `<method> = STRING`. This file contains the positions of the images along the string at the final step in xyz format. The print interval can be controlled by the keyword `<iprint_str>`.

10.2.43 *string_final.out*

Used when `<method> = STRING`. This file contains the final potential energies of the images along the string.

- COLUMN 1: step number.
- COLUMN 2: image number.
- COLUMN 3: mass-weighted reaction coordinate [a.u.].
- COLUMN 4: potential energy [hartree].

The print interval can be controlled by the keyword `<iprint_rest>`.

10.2.44 *template.xyz*

Used in all cases except `<method> = STATIC, NMA, PHONON`, when the trajectory is written in DCD format via the `<iprint_dcd>` keyword. This file contains copies of the initial coordinates repeated n times, where n is the number of beads/replicas/images. It is provided to simplify reading *trj.dcd* by supplying atomic information for the coordinates stored in that file. This file does *not* contain trajectory data. When using it to assist with reading DCD data, be sure to remove or ignore these duplicated initial-coordinate records before further analysis.

10.2.45 *tfs.out*

Used when `<method> = TFS`. This file contains data from the surface-hopping method. The print interval can be controlled by the keyword `<iprint_tfs>`.

- For `<method> = TFS`:
 - COLUMN 1: step number.
 - COLUMN 2: bead number.
 - COLUMN 3: state number.
 - COLUMN 4: occupied state.
 - COLUMN 5: population (state coefficient) of each state.
 - COLUMN 6: occupation ratio of each state.
 - COLUMN 7: potential energy of each state.
 - COLUMN 8: potential energy of the occupied state.

10.2.46 *trj.dcd*

Used in all cases except `<method> = STATIC, NMA, PHONON`. This file contains the trajectory in DCD format. For simulations with multiple beads/replicas/images (`<nbead> = n`), the data are repeated n times. The print interval can be controlled by the keyword `<iprint_dcd>`. To assist in reading the DCD file, *template.xyz* is also written when DCD output is enabled. This file contains the coordinates of the initial structure repeated n times, where n is the number of beads/replicas/images. Because the coordinates in *template.xyz* are *not* part of the simulation trajectory, you should delete or ignore them in any subsequent trajectory analysis. Coordinates are written unwrapped when periodic boundary conditions are used, so diffusion can be followed without unfolding the trajectory.

The format starts with a 276-byte header, described below. This header is also used in other **.dcd* outputs, with minor modifications described in the corresponding sections of this manual. The header contents are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 5–8: (chars [1 byte]) CORD.
- BYTE 9–12: (int) 0.
- BYTE 13–16: (int) starting step (usually 0).
- BYTE 17–20: (int) trajectory output interval.
- BYTE 21–44: (ints [4 bytes]) 0, 0, 0, 0, 0, 0.
- BYTE 45–48: (float) time step in AKMA units.
- BYTE 49–52: (int) 1 if PBC are used, 0 otherwise.
- BYTE 53–84: (ints [4 bytes]) 0, 0, 0, 0, 0, 0, 0, 0.
- BYTE 85–88: (int) 24.
- BYTE 89–92: (int) 84.
- BYTE 93–96: (int) 164.
- BYTE 97–100: (int) 2.
- BYTE 101–180: (chars [1 byte]) a string, 'PIMD - TRAJECTORY\0'.
- BYTE 181–260: (chars [1 byte]) a string containing the date of the calculation.
- BYTE 261–264: (int) 164.
- BYTE 265–268: (int) 4.
- BYTE 269–272: (int) number of beads times number of atoms.
- BYTE 273–276: (int) 4.

After the header, the trajectory data are organized as follows. If periodic boundary conditions are present, the box information is written first (same format as described above for other DCD outputs):

- BYTE 1–4: (int) size of the next read (48).
- BYTE 5–52: (doubles [8 bytes]) box information ordered as A , γ , B , β , α , C . Here, A , B , and C are side lengths in Å, and α , β , and γ are angles in radians.
- BYTE 53–56: (int) size of the previous read (48).

The coordinates are then written as follows, where N is four times the number of atoms times the number of beads:

- NEXT 4 BYTES: (int) size of the next read (number of beads times number of atoms).
- NEXT N BYTES: (floats [4 bytes]) x coordinates.
- NEXT 4 BYTES: size of the previous read.

This block is repeated for the y and z coordinates. The trajectory data are wrapped so that coordinates lie inside the periodic box (if present). All atomic coordinates are given in Å.

10.2.47 *trj.out*

Used in all cases except `<method> = STATIC, NMA, PHONON`. This file contains the standard trajectory output. The print interval can be controlled by the keyword `<iprint_trj>`.

- For all methods:
 - COLUMN 1: step number.
 - COLUMNS 2–4: atomic position (x, y, z) in atomic units (bohr).
 - COLUMNS 5–7: atomic velocity (v_x, v_y, v_z) in atomic units (bohr per atomic time), where atomic time = $\hbar/\text{hartree} \simeq 0.0241888$ fs.
 - COLUMNS 8–10: atomic force (f_x, f_y, f_z) in atomic units (hartree/bohr).
 - COLUMN 11: potential energy V in atomic units (hartree).

For simulations with multiple beads/replicas/images (`<nbead> = n`), the data are repeated n times. The block with the same step number corresponds to one configuration.

10.2.48 *trj.xyz*

Used in all cases except `<method> = STATIC, NMA, PHONON`. This file contains the trajectory in xyz format.

- For all methods:
 - LINE 1: number of atoms.
 - LINE 2: step number.
 - LINES 3– $(n + 2)$: atomic species followed by Cartesian coordinates (x, y, z) for each atom. For periodic boundary conditions, coordinates may be duplicated to display both folded and unfolded trajectories.

For simulations with multiple beads/replicas/images (`<nbead> = n`), the data are repeated n times. The print interval can be controlled by the keyword `<iprint_xyz>`. The output is controlled by the options `<iformat_xyz>` and `<jorigin_xyz>`.

10.2.49 *vel.dcd*

This binary file contains particle velocities. The print interval can be controlled by the keyword `<iprint_dcd>`.

The file contains a 276-byte header similar to that of *trj.dcd*. The important parts are as follows:

- BYTE 1–4: (int) 84, indicating the endianness of the file.
- BYTE 17–20: (int) output interval of the velocity information.
- BYTE 269–272: (int) number of beads times number of atoms.

After the header, the velocity data are organized as follows. If periodic boundary conditions are present, the box information is written first:

- BYTE 1–4: (int) size of the next read (48).
- BYTE 5–52: (doubles [8 bytes]) box information ordered as $A, \gamma, B, \beta, \alpha, C$. Here, A, B , and C are side lengths in Å, and α, β , and γ are angles in radians.
- BYTE 53–56: (int) size of the previous read (48).

The velocity data are then written as follows, where N is four times the number of atoms times the number of beads:

- NEXT 4 BYTES: (int) size of the next read (number of beads times number of atoms).
- NEXT N BYTES: (floats [4 bytes]) x velocities.
- NEXT 4 BYTES: size of the previous read.

This block is repeated for the y and z velocities. The atom order is the same as in *template.xyz*. All velocities are given in $\text{\AA}/\text{ps}$.

10.3 Restart files

10.3.1 *afed.ini*

Used in the case of `<method> = AFED` with `<afed_type> = DESCENT, ASCENT, AUTO, TAMD, LOGMFD`. This file contains the latest values of adiabatic free energy dynamics, temperature-accelerated molecular dynamics, and logarithmic mean force dynamics. Each line contains the following columns, where n is the dimension of the collective variables and m is the length of the Nosé-Hoover chain thermostats:

- For `<afed_type> = DESCENT, ASCENT, AUTO`:
 - COLUMN 1: AFED status.
 - * DE: descent trajectory to a free-energy minimum point.
 - * D1: descent trajectory started in the $+\mathbf{n}$ direction from a free-energy saddle point to a free-energy minimum point.
 - * D2: descent trajectory started in the $-\mathbf{n}$ direction from a free-energy saddle point to a free-energy minimum point.
 - * AS: ascent trajectory to search for a free-energy saddle point.
 - * EQ: free-energy minimum point.
 - * TS: free-energy saddle point.
 - * HI: point terminated where the free energy exceeds the limit defined by the keyword `<fenergy_max_afed>`.
 - * OB: point terminated where the CV is out of bounds defined by the keyword `<params_afed>`.
 - COLUMN 2: AFED step number (iteration number).
 - COLUMN 3: free energy in hartree.
 - COLUMN 4-(3+n): the position of the fictitious particle and the unit vector \mathbf{n} .
 - COLUMN (4+n): root point ID (starting point of the trajectory that leads to this point).
 - COLUMN (5+n): AFED step size.
- For `<afed_type> = TAMD`:
 - COLUMN 1: AFED status.
 - * TA: TAMD trajectory.
 - * HI: point terminated where the free energy exceeds the limit defined by the keyword `<fenergy_max_afed>`.
 - * OB: point terminated where the CV is out of bounds defined by the keyword `<params_afed>`.
 - COLUMN 2: AFED step number (iteration number).
 - COLUMN 3: free energy in hartree.
 - COLUMN 4-(3+n): the position and the mass-weighted velocity of the fictitious particle.
 - COLUMN (4+n+m): the position and the mass-weighted velocity of the Nosé-Hoover thermostats (only if they exist).

- For `<afed_type> = LOGMFD`:
 - COLUMN 1: AFED status.
 - * LO: LogMFD trajectory.
 - * HI: point terminated where the free energy exceeds the limit defined by the keyword `<fenergy_max_afed>`.
 - * OB: point terminated where the CV is out of bounds defined by the keyword `<params_afed>`.
 - COLUMN 2: AFED step number (iteration number).
 - COLUMN 3: free energy in hartree.
 - COLUMN 4-(3+n): the position and the mass-weighted velocity of the fictitious particle.
 - COLUMN (4+n): the origin of the free energy in hartree.
 - COLUMN (5+n+m): the position and the mass-weighted velocity of the Nosé-Hoover thermostats (only if they exist).

The file is overwritten for restarted runs.

10.3.2 *auto.ini*

Used in the case of `<method> = AFED` with `<afed_type> = AUTO`. This file contains the list of landmarks in the AFED automated search. Each line contains the following columns, where n is the dimension of the collective variables:

- COLUMN 1: landmark number.
- COLUMN 2: AFED status.
 - EQ: free-energy minimum point.
 - TS: free-energy saddle point.
 - HI: point terminated where the free energy exceeds the limit.
 - OB: point terminated where the CV is out of bounds.
- COLUMN 3: landmark label.
 - NEW: new point in the list.
 - OLD: old point in the list.
- COLUMN 4: free energy in hartree.
- COLUMN 5: landmark ID.
- COLUMN 6: name of restart tar file.
- COLUMN 7: root point ID (starting point of the trajectory that leads to this point).
- COLUMN 8-(7+n): position of each collective variable.
- COLUMN (8+n)-(7+2n): unit vector \mathbf{n} in the ascent trajectory, which should be parallel to the imaginary-frequency mode. These columns are present only when the AFED status is TS.

The file is appended for restarted runs.

10.3.3 *averages.ini*

Used in ALL methods. This file contains the data of statistical averages at the final step of the last run. If this file is not found at the beginning of the run, the data is initialized.

The data affect the results in the output files that include averages (*bond.out*, *eavg.out*, *rdf.out*, *rgy.out*). They also affect the acceptance ratio in PIHMC and REHMC methods (*standard.out*).

IMPORTANT: Do not change the keywords `<iprint_...>` in the file *input.dat* in restarted runs, or the average data may be corrupted.

10.3.4 *bath.ini*

Used in the cases of `<method> = MD, PIMD, CMD, MTD`. This file contains the thermostat positions and the thermostat velocities at the final step of the last run. If this file is not found at the beginning of the run, the data are initialized.

NOTE: The data depend on the bath type, the method, and the system. Exceptionally, the data are transferable when switching from `<method> = PIMD` to `<method> = CMD` when `<bath_type> = MNHC` is chosen.

10.3.5 *box.ini*

Used in ALL calculations with periodic boundary conditions, i.e., when `<iboundary> = 1`. This file contains the step number and the box matrix $h(i, j)$ at the final step of the last run.

step number, $h(1, 1)$, $h(1, 2)$, $h(1, 3)$
step number, $h(2, 1)$, $h(2, 2)$, $h(2, 3)$
step number, $h(3, 1)$, $h(3, 2)$, $h(3, 3)$

If this file is not found at the beginning of the run, the data are read from the file *input.dat* under the keyword `<iboundary>`.

10.3.6 *cstate.ini*

Used in the cases of `<method> = TFS, MFE`. This file contains the state coefficients (complex numbers) at the final step of the last run. If this file is not found at the beginning of the run, the data are read from the file *input.dat* under the keyword `<istate_init>`.

10.3.7 *cv.ini*

Used in the case of `<method> = MTD`. This file contains the positions and velocities of the collective variables at the final step of the last run. If this file is not found at the beginning of the run, the data are initialized.

10.3.8 *gad.ini*

Used in the case of `<method> = GAD`. This file contains the latest values of gentlest ascent dynamics. Each line contains the data of the guiding unit vector of each trajectory, followed by the convergence flag of each trajectory.

10.3.9 *geometry.ini*

Used in ALL methods except `<method> = STATIC, NMA, PHONON`. This file contains the latest values of the positions and velocities of atoms at the final step of the last run. If this file is not found at the beginning of the run, the data are read from the file *structure.dat*.

The data are transferable among `<method> = STATIC, GEOOPT, BOXOPT, FULLOPT, ELASTIC, SD, NMA, PHONON, MD`, with the following format for n atoms.

```
0,  x1,1,  y1,1,  z1,1,  vx1,1,  vy1,1,  vz1,1,  n1,1,1,  n2,1,1,  n3,1,1
...
0,  xn,1,  yn,1,  zn,1,  vxn,1,  vyn,1,  vzn,1,  n1,n,1,  n2,n,1,  n3,n,1
```

where indices $(n_{1,i,j}, n_{2,i,j}, n_{3,i,j})$ are important only for periodic boundary conditions. The indices indicate the displacement with respect to the box matrix in the folded trajectories.

The data are also transferable among the path-integral methods, `<method> = PIMD, CMD, RPMD, PIHMC`, where the normal-mode positions and velocities are given in mass-weighted coordinates.

10.3.10 *hills.ini*

Used in the case of `<method> = MTD`. This file contains the list of data for the history-dependent Gaussian hill potential.

- COLUMN 1: metadynamics step.
- COLUMN 2: Gaussian height in hartree.
- COLUMN 3: Gaussian width in bohr, degrees, etc.
- COLUMN 4: Gaussian center in bohr, degrees, etc.

The unit of the Gaussian width corresponds to the unit used under the keyword `<gw_meta>` in `input.dat`. One data unit consists of m lines, where m is the dimension of the collective variables, which is given by the keyword `<nmeta>` in `input.dat`. If this file is not found at the beginning of the run, the data are initialized with no Gaussian hills.

This file is read at the beginning of the metadynamics run. The user may simply add “wall potentials” (high Gaussian hills with small width) to the list, to limit the exploration of the collective variables. In this case, the step number can be any integer.

Note that in well-tempered metadynamics, the Gaussian hills are not the negative of the free energy. Thus, the deposited hills cannot be transferred between conventional and well-tempered metadynamics.

10.3.11 *step.ini*

Used in ALL methods except `<method> = STATIC, NMA, PHONON`. This file contains the final step number of the last run. If this file is not found at the beginning of the run, the data are initialized to zero.

10.3.12 *string.ini*

Used in `<method> = STRING, OMOPT`. This file contains the latest values of the positions and velocities of atoms at the final step of the last run. If this file is not found at the beginning of the run, the data are read from the file `structure.dat`.

This file can be edited by the user to start from a guessed path. For a system of n atoms with m images, the data should be provided in the following format.

```
0,  x1,1,  y1,1,  z1,1
...
0,  xn,1,  yN,1,  zn,1
0,  x1,2,  y1,2,  z1,2
...
```

$$\begin{array}{cccc}
0, & x_{n,2}, & y_{N,2}, & z_{n,2} \\
\dots & & & \\
\dots & & & \\
0, & x_{1,m}, & y_{1,m}, & z_{1,m} \\
\dots & & & \\
0, & x_{n,m}, & y_{n,m}, & z_{n,m}
\end{array}$$

where $(x_{i,j}, y_{i,j}, z_{i,j})$ are the Cartesian coordinates of the i -th atom in the j -th image in bohr. “0” in the first column is the step number.

11 Theoretical background

In this section, the theoretical background of the methods implemented in the PIMD code is briefly summarized. Throughout this section, the following notation is used.

- Scalars are denoted by regular font (e.g., x), vectors by bold font (e.g., \mathbf{x}), and matrices by bold font with arrows (e.g., $\overleftrightarrow{\mathbf{X}}$).
- The number of atoms is denoted by N .
- The mass of the i -th atom is denoted by m_i .
- The set of atomic positions (nuclear coordinates) is denoted by

$$\{\mathbf{r}\} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N\} = \{r_{1x}, r_{1y}, r_{1z}, \dots, r_{Nx}, r_{Ny}, r_{Nz}\}. \quad (46)$$

When P beads are used, the atomic positions of the s -th bead are

$$\{\mathbf{r}^{(s)}\} = \{\mathbf{r}_1^{(s)}, \mathbf{r}_2^{(s)}, \dots, \mathbf{r}_N^{(s)}\} = \{r_{1x}^{(s)}, r_{1y}^{(s)}, r_{1z}^{(s)}, \dots, r_{Nx}^{(s)}, r_{Ny}^{(s)}, r_{Nz}^{(s)}\}, \quad (47)$$

for $1 \leq s \leq P$.

- The potential energy is expressed as

$$V = V(\{\mathbf{r}\}). \quad (48)$$

- The set of atomic momenta is denoted by

$$\{\mathbf{p}\} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N\} = \{p_{1x}, p_{1y}, p_{1z}, \dots, p_{Nx}, p_{Ny}, p_{Nz}\}. \quad (49)$$

When P beads are used, the momenta of the s -th bead are

$$\{\mathbf{p}^{(s)}\} = \{\mathbf{p}_1^{(s)}, \mathbf{p}_2^{(s)}, \dots, \mathbf{p}_N^{(s)}\} = \{p_{1x}^{(s)}, p_{1y}^{(s)}, p_{1z}^{(s)}, \dots, p_{Nx}^{(s)}, p_{Ny}^{(s)}, p_{Nz}^{(s)}\}. \quad (50)$$

The phase-space coordinates of the s -th bead are then written as

$$\mathbf{\Gamma}^{(s)} = \{\mathbf{r}^{(s)}, \mathbf{p}^{(s)}\}. \quad (51)$$

- The physical atomic masses are denoted by m_i . When fictitious masses are introduced, they are denoted by μ_i .
- The simulation box is described by the box matrix

$$\overleftrightarrow{\mathbf{h}} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} = \begin{pmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{pmatrix}, \quad (52)$$

where $\mathbf{a} = (a_x, a_y, a_z)$, $\mathbf{b} = (b_x, b_y, b_z)$, and $\mathbf{c} = (c_x, c_y, c_z)$ are the box vectors. The volume of the simulation box is given by

$$\mathbb{V} = \det(\overleftrightarrow{\mathbf{h}}). \quad (53)$$

- The temperature is denoted by \mathbb{T} , and

$$\beta = \frac{1}{k_B \mathbb{T}}, \quad (54)$$

where k_B is the Boltzmann constant.

- The external pressure is denoted by \mathbb{P} . In the general case, the external stress (or tension) is represented by a real symmetric matrix

$$\overleftrightarrow{\mathbf{t}} = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{12} & t_{22} & t_{23} \\ t_{13} & t_{23} & t_{33} \end{pmatrix}. \quad (55)$$

11.1 Mechanochemistry

To deal with mechanochemistry, the external force explicitly included (EFEI) method is employed. A constant repulsive force, F , is applied to the specified pair of atoms i and j in the direction of the interatomic vector. The potential

$$V^{\text{efei}} = -F r_{ij} \quad (56)$$

and the force

$$F_{i\alpha}^{\text{efei}} = F \frac{r_{i\alpha} - r_{j\alpha}}{r_{ij}} \quad (57)$$

are added to the physical potential and the physical force, respectively, where

$$r_{ij} = \sqrt{\sum_{\alpha=x,y,z} (r_{i\alpha} - r_{j\alpha})^2} \quad (58)$$

is the distance between atoms i and j .

11.2 Steepest descent

The intrinsic reaction coordinate (IRC) is the steepest descent path starting from the saddle point and ending at the minimum point on the potential energy surface in mass-weighted coordinate space. The mass-weighted coordinate is defined as

$$\mathbf{q} = \{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_N\} = \{\sqrt{m_1} \mathbf{r}_1, \sqrt{m_2} \mathbf{r}_2, \dots, \sqrt{m_N} \mathbf{r}_N\}. \quad (59)$$

The steepest descent path is defined by the trajectory of the equation

$$\frac{d\mathbf{q}_i}{dt} = -\frac{\mathbf{g}_i}{|\mathbf{g}|}. \quad (60)$$

where the potential gradient with respect to \mathbf{q}_i is defined by

$$\mathbf{g}_i = \frac{dV}{d\mathbf{q}_i} = \frac{1}{\sqrt{m_i}} \frac{dV}{d\mathbf{r}_i} \quad (61)$$

and the norm is given by

$$|\mathbf{g}|^2 = \sum_{i=1}^N \frac{1}{m_i} \left\{ \left(\frac{dV}{dx_i} \right)^2 + \left(\frac{dV}{dy_i} \right)^2 + \left(\frac{dV}{dz_i} \right)^2 \right\}. \quad (62)$$

Eq. (60) is equivalent to

$$m_i \frac{d\mathbf{r}_i}{dt} = - \frac{1}{|\mathbf{g}|} \frac{dV}{d\mathbf{r}_i}. \quad (63)$$

In the PIMD code, Eq. (63) is solved using a fourth-order Runge-Kutta algorithm with step size dt . Note that dt has the dimension $(\sqrt{\text{mass}} \times \text{length})$ or $(\sqrt{\text{energy}} \times \text{time})$.

11.3 Box optimization

The box matrix, $h_{\alpha\beta}$, can be optimized using the limited-memory BFGS algorithm [3].

11.3.1 Zero pressure condition

At zero pressure, the bare potential V is minimized. The potential gradient with respect to the box

$$G_{\alpha\beta} = \frac{\partial V}{\partial h_{\alpha\beta}} \quad (64)$$

is related to the virial,

$$A_{\alpha\beta} = - \sum_{\gamma=1}^3 \frac{\partial V}{\partial h_{\alpha\gamma}} h_{\beta\gamma} = - \sum_{\gamma=1}^3 G_{\alpha\gamma} h_{\beta\gamma}, \quad (65)$$

and thus,

$$G_{\alpha\beta} = - \sum_{\gamma=1}^3 A_{\alpha\gamma} h_{\beta\gamma}^{-1}. \quad (66)$$

Note that the virial $A_{\alpha\beta}$ is associated with the internal pressure tensor at zero temperature,

$$P_{\alpha\beta}(0) = \mathbb{V}^{-1} A_{\alpha\beta}. \quad (67)$$

11.3.2 Finite pressure condition

In the presence of an external pressure, \mathbb{P} , the effective potential

$$\tilde{V} = V + \mathbb{P}\mathbb{V} \quad (68)$$

is minimized. Then, the gradient of the effective potential is given by

$$\tilde{G}_{\alpha\beta} = \frac{\partial \tilde{V}}{\partial h_{\alpha\beta}} = \frac{\partial V}{\partial h_{\alpha\beta}} + \mathbb{P} \mathbb{V} h_{\beta\alpha}^{-1}. \quad (69)$$

11.3.3 Finite tension condition

In the presence of an external thermodynamic tension, $\overleftrightarrow{\mathbf{t}}$, the effective potential

$$\tilde{V} = V - \bar{\mathbb{V}} \sum_{\alpha,\beta=1}^3 (t_{\alpha\beta} \epsilon_{\alpha\beta}) \quad (70)$$

is minimized.

$$\epsilon_{\alpha\beta} = \frac{1}{2} \left\{ \sum_{\mu,\nu,\lambda=1}^3 \left(\bar{h}_{\mu\alpha}^{-1} h_{\lambda\mu} h_{\lambda\nu} \bar{h}_{\nu\beta}^{-1} \right) - \delta_{\alpha\beta} \right\} \quad (71)$$

is the strain tensor, where $\bar{\mathbf{h}}$ and $\bar{\mathcal{V}} = \det(\bar{\mathbf{h}})$ are the box matrix and the volume, respectively, in the absence of external tension. Then, the gradient of the effective potential is given by

$$\tilde{G}_{\alpha\beta} = \frac{\partial \tilde{V}}{\partial h_{\alpha\beta}} = \frac{\partial V}{\partial h_{\alpha\beta}} - \bar{\mathcal{V}} \sum_{\mu,\nu,\lambda=1}^3 \left(h_{\alpha\lambda} \bar{h}_{\lambda\mu}^{-1} t_{\mu\nu} \bar{h}_{\beta\nu}^{-1} \right). \quad (72)$$

11.4 Static elastic constants

The static elastic constants are defined by

$$C_{\alpha\beta,\gamma\delta} = \frac{1}{\Omega} \frac{\partial^2 V}{\partial \epsilon_{\alpha\beta} \partial \epsilon_{\gamma\delta}}, \quad (73)$$

where the strain tensor is given by

$$\epsilon_{\alpha\beta} = \frac{1}{2} \left(\sum_{\nu,\gamma,\lambda=x,y,z} \bar{h}_{\gamma\alpha}^{-1} h_{\lambda\gamma} h_{\lambda\nu} \bar{h}_{\nu\beta}^{-1} - \delta_{\alpha\beta} \right). \quad (74)$$

The elastic constants are invariant under permutations of the indices between α and β , between γ and δ , and between (α, β) and (γ, δ) . In Voigt notation, the constants are abbreviated as C_{ij} , where $1 \leq i, j \leq 6$, and 1–6 correspond to (x, x) , (y, y) , (z, z) , (y, z) , (z, x) , and (x, y) , respectively.

11.5 Normal mode analysis

The mass-weighted Hessian matrix is defined as a $(3N \times 3N)$ real symmetric matrix:

$$\tilde{H}_{i\alpha,j\beta} = \frac{1}{\sqrt{m_i m_j}} \frac{\partial^2 V}{\partial r_{i\alpha} \partial r_{j\beta}}. \quad (75)$$

By diagonalizing \tilde{H} at a stationary point, one obtains the eigenvalues ε_k and the corresponding eigenvectors $U_{i\alpha,k}$ for each mode k ($1 \leq k \leq 3N$). The vibrational frequencies, converted from atomic units to wavenumbers (cm^{-1}), are given by

$$\nu_k = \frac{1}{100} \times \frac{E_h}{hc} \sqrt{\varepsilon_k}, \quad (76)$$

where c is the speed of light, h is Planck's constant, and $E_h = 4.35974417 \times 10^{-18}$ J is one hartree in SI units. The reduced mass of the k -th mode in atomic mass units is defined as

$$\mu_k = \frac{1}{1822.88853} \times \frac{1}{\sum_{i=1}^N \sum_{\alpha=x,y,z} m_i^{-1} U_{i\alpha,k}^2}, \quad (77)$$

and the normalized mass-weighted normal mode vector is

$$\tilde{U}_{i\alpha,k} = \frac{m_i^{-1/2} U_{i\alpha,k}}{\sqrt{\sum_{i=1}^N \sum_{\alpha=x,y,z} m_i^{-1} U_{i\alpha,k}^2}}. \quad (78)$$

For free boundary systems, there are six (five) zero-frequency modes, corresponding to three translations and three (two) rotations for nonlinear (linear) molecules. For periodic boundary systems, there are three zero-frequency modes corresponding to pure translations. In the PIMD code, numerical errors in the Hessian are corrected by applying a projection operator to remove contributions from zero-frequency translational and rotational modes.

11.6 Phonon calculations

Let the vector of three integers, $\mathbf{n} = (n_1, n_2, n_3)$, denote the numbering of the respective unit cells in a crystal. Let N atoms be contained in the unit cell defined by the matrix of three lattice vectors, $\vec{\mathbf{h}} = (\mathbf{a}, \mathbf{b}, \mathbf{c})$. Let $\mathbf{r}_{i,0}$ be the position of the i -th atom in the unit cell at the origin, $\mathbf{0} = (0, 0, 0)$. Then, the atomic positions in the unit cell labeled by \mathbf{n} are given by

$$\mathbf{r}_{i,\mathbf{n}} = \mathbf{r}_{i,0} + \mathbf{h}^{-1}\mathbf{n}. \quad (79)$$

For a given wave vector \mathbf{K} in reciprocal space, the $(3N \times 3N)$ dynamical matrix is defined as

$$D_{j\alpha,k\beta}(\mathbf{K}) = \frac{1}{\sqrt{m_j m_k}} \sum_{\mathbf{n}} \frac{\partial^2 V}{\partial r_{j\alpha,0} \partial r_{k\beta,\mathbf{n}}} \exp(i\mathbf{K} \cdot (\mathbf{r}_{j,0} - \mathbf{r}_{k,\mathbf{n}})). \quad (80)$$

By diagonalizing this matrix, one obtains the eigenvalues $\varepsilon_k(\mathbf{K})$, where k is the index of the vibrational mode. The phonon frequencies are obtained by converting from atomic units to cm^{-1} ,

$$\nu_k(\mathbf{K}) = \frac{1}{100} \times \frac{E_h}{hc} \sqrt{\varepsilon_k(\mathbf{K})}. \quad (81)$$

The dynamical matrix is evaluated and diagonalized for each wave vector \mathbf{K} within the first Brillouin zone,

$$(K_x, K_y, K_z)_{l_1, l_2, l_3} = \left(\sum_{\beta=1}^3 \frac{\pi}{L_\beta} l_\beta h_{\beta x}^{-1}, \sum_{\beta=1}^3 \frac{\pi}{L_\beta} l_\beta h_{\beta y}^{-1}, \sum_{\beta=1}^3 \frac{\pi}{L_\beta} l_\beta h_{\beta z}^{-1} \right). \quad (82)$$

The evenly spaced grid points with $-L_\beta \leq l_\beta \leq L_\beta$ ($1 \leq \beta \leq 3$) are used to obtain the phonon density of states,

$$\rho(\nu) = \sum_{\mathbf{K}} (\nu - \nu_k(\mathbf{K})). \quad (83)$$

The classical and quantum internal energies per unit cell are given by

$$U^{\text{cl}} = (3N - 3) k_B T, \quad (84)$$

and

$$U^{\text{qm}} = \frac{1}{N_{\mathbf{K}}} \sum_{\mathbf{K}} \sum_{m=4}^{3N} \frac{\hbar \omega_m(\mathbf{K})}{2} \coth \left(\frac{\beta \hbar \omega_m(\mathbf{K})}{2} \right), \quad (85)$$

respectively. The classical and quantum Helmholtz free energies per unit cell are given by

$$A^{\text{cl}} = \frac{1}{N_{\mathbf{K}}} \sum_{\mathbf{K}} \sum_{m=4}^{3N} k_B T \log(\beta \hbar \omega_m(\mathbf{K})), \quad (86)$$

and

$$A^{\text{qm}} = \frac{1}{N_{\mathbf{K}}} \sum_{\mathbf{K}} \sum_{m=4}^{3N} k_B T \log(1 - \exp(-\beta \hbar \omega_m(\mathbf{K}))), \quad (87)$$

respectively. The entropy per unit cell is defined as

$$S^{\text{cl}} = \frac{U^{\text{cl}} - A^{\text{cl}}}{T} \quad (88)$$

and

$$S^{\text{qm}} = \frac{U^{\text{qm}} - A^{\text{qm}}}{T}, \quad (89)$$

respectively. In the PIMD code, numerical errors in the dynamical matrix are removed by applying a projection operator with respect to the zero-frequency translational modes.

11.7 String method

The String Method is an efficient algorithm for obtaining the minimum energy path or intrinsic reaction coordinate that connects the reactant and the product.

The path is described in terms of P discrete images along the path in mass-weighted coordinates, denoted as \mathbf{q} in Eq. (59).

Starting from an initial guess, the images are optimized by iterating the following two-step procedure:

- Step 1: Move the positions of all images \mathbf{q} along the negative gradient direction with an increment parameter Δt as follows:

$$\mathbf{q}_i^{(l)} \leftarrow \mathbf{q}_i^{(l)} - \frac{\mathbf{g}_i^{(l)}}{|\mathbf{g}_i^{(l)}|} \Delta t \quad (90)$$

for all images l and for all atoms i , where $\mathbf{g}_i^{(l)}$ and $|\mathbf{g}_i^{(l)}|$ are defined in Eqs. (61) and (62), respectively, for the l -th image.

- Step 2: While keeping the positions of the terminal images $\mathbf{q}^{(1)}$ and $\mathbf{q}^{(P)}$ fixed, rearrange all other images along the path such that all neighboring images are equidistant:

$$\sum_{i=1}^N \sum_{\alpha=x,y,z} \left(q_{i\alpha}^{(l+1)} - q_{i\alpha}^{(l)} \right)^2 = \text{const.} \quad (91)$$

If the number of images P is sufficiently large, the iteration of Steps 1 and 2 will converge to a minimum energy path that is closest to the initial guess.

11.8 OM action

The path-integral formulation of overdamped Langevin dynamics was introduced by Onsager and Machlup (OM). The transition probability from the initial position, $\mathbf{r}(0) = \mathbf{r}^{(1)}$, to the final position, $\mathbf{r}(T) = \mathbf{r}^{(L)}$, over a time interval of T is given by

$$P \left(\mathbf{r}^{(L)} \middle| \mathbf{r}^{(1)} \right) \propto \int_{\mathbf{r}^{(1)}}^{\mathbf{r}^{(L)}} \mathcal{D}\mathbf{r} \exp(-\beta S[\mathbf{r}]) \quad (92)$$

where the OM action is defined by

$$S[\mathbf{r}(t)] = \int_0^T dt \left\{ \sum_{i=1}^N \frac{m_i \gamma}{4} \left(\dot{\mathbf{r}}_i(t) + \frac{\nabla_i V(\mathbf{r}(t))}{m_i \gamma} \right)^2 \right\}. \quad (93)$$

Here, γ is the friction constant in Langevin dynamics. For a smooth path, one obtains

$$S \simeq \frac{V(\mathbf{r}(T)) - V(\mathbf{r}(0))}{2} + \int_0^T dt \sum_{i=1}^N \frac{m_i \gamma}{4} \left\{ (\dot{\mathbf{r}}_i(t))^2 + \left(\frac{\nabla_i V(\mathbf{r}(t))}{m_i \gamma} \right)^2 \right\}. \quad (94)$$

A path $\mathbf{r}(t)$ that minimizes the action S is the most probable path. Such a path can be obtained by optimizing the value of S under the constraint that the path starts from a reactant minimum and ends at a product minimum. The path depends on the parameter T/γ . If T/γ is sufficiently small, the path becomes straight. If T/γ is sufficiently large, it becomes the minimum energy path, i.e., the intrinsic reaction coordinate.

In the PIMD code, the OM action with L discrete images,

$$S = \frac{V(\mathbf{r}^{(L)}) - V(\mathbf{r}^{(1)})}{2} + \sum_{l=1}^{L-1} \Delta t \sum_{i=1}^N \frac{m_i \gamma}{4} \left\{ \left(\frac{\mathbf{r}_i^{(l+1)} - \mathbf{r}_i^{(l)}}{\Delta t} \right)^2 + \left(\frac{\nabla_i V(\mathbf{r}^{(l)})}{m_i \gamma} \right)^2 \right\}, \quad (95)$$

is optimized using the limited-memory BFGS algorithm with respect to the normal-mode variables, where $\Delta t = T/(L - 1)$. Note that S can be expressed as a function of $\Delta\tau = \Delta t/\gamma$ and $\mathbf{q}_i = \sqrt{m_i}\mathbf{r}_i$, since

$$S = \frac{\tilde{V}(\mathbf{q}^{(L)}) - \tilde{V}(\mathbf{q}^{(1)})}{2} + \sum_{l=1}^{L-1} \sum_{i=1}^N \left\{ \frac{(\mathbf{q}_i^{(l+1)} - \mathbf{q}_i^{(l)})^2}{4\Delta\tau} + \frac{\Delta\tau}{4} \left(\nabla_i \tilde{V}(\mathbf{q}^{(l)}) \right)^2 \right\}. \quad (96)$$

Here, $\tilde{V}(\mathbf{q}) = V(\mathbf{r})$. In the PIMD code, the gradient of S is evaluated numerically using the finite-difference technique.

11.9 NVE molecular dynamics

In NVE molecular dynamics, the microcanonical ensemble is naturally generated by integrating Newton's equations of motion

$$\dot{r}_{i\alpha} = \frac{p_{i\alpha}}{m_i}, \quad \dot{p}_{i\alpha} = -\frac{\partial V}{\partial r_{i\alpha}} \quad (97)$$

for ergodic systems. The total energy is given by

$$H = \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_i^2}{2m_i} + V \quad (98)$$

and is conserved, since

$$\begin{aligned} \dot{H} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial H}{\partial p_{i\alpha}} \dot{p}_{i\alpha} + \frac{\partial H}{\partial r_{i\alpha}} \dot{r}_{i\alpha} \right) \\ &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{p_{i\alpha}}{m_i} \left(-\frac{\partial V}{\partial r_{i\alpha}} \right) + \frac{\partial V}{\partial r_{i\alpha}} \frac{p_{i\alpha}}{m_i} \right\} = 0. \end{aligned} \quad (99)$$

The Jacobian, J , satisfies

$$\dot{J} = -J \left\{ \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial \dot{r}_{i\alpha}}{\partial r_{i\alpha}} + \frac{\partial \dot{p}_{i\alpha}}{\partial p_{i\alpha}} \right) \right\} = 0, \quad (100)$$

and therefore J is constant. This result is known as Liouville's theorem. Since H is constant ($= E$), the phase-space volume of the trajectory can be written as

$$W \propto \int dr^{3N} \int dp^{3N} J \delta(H - E) \propto \int dr^{3N} \int dp^{3N} \delta(H - E), \quad (101)$$

assuming ergodicity. This corresponds to the number of microscopic states in the microcanonical ensemble. The probability density of finding the system at the point $(r_{1x}, \dots, r_{Nx}, p_{1x}, \dots, p_{Nx})$ is

$$P(r_{1x}, \dots, r_{Nx}, p_{1x}, \dots, p_{Nx}) = \frac{\delta(H - E)}{\int dr^{3N} \int dp^{3N} \delta(H - E)}. \quad (102)$$

11.9.1 Velocity-Verlet algorithm

The velocity-Verlet algorithm is used to integrate Newton's equations of motion,

$$\begin{aligned} p_{i\alpha} \left(t + \frac{\Delta t}{2} \right) &= p_{i\alpha}(t) - \frac{\partial V}{\partial r_{i\alpha}}(t) \frac{\Delta t}{2} \\ r_{i\alpha}(t + \Delta t) &= r_{i\alpha}(t) + \frac{p_{i\alpha}(t + \frac{\Delta t}{2})}{m_i} \Delta t \\ p_{i\alpha}(t + \Delta t) &= p_{i\alpha} \left(t + \frac{\Delta t}{2} \right) - \frac{\partial V}{\partial r_{i\alpha}}(t + \Delta t) \frac{\Delta t}{2}, \end{aligned} \quad (103)$$

or equivalently,

$$\begin{aligned} r_{i\alpha}(t + \Delta t) &= r_{i\alpha}(t) + \frac{p_{i\alpha}(t)}{m_i} \Delta t - \frac{1}{m_i} \frac{\partial V}{\partial r_{i\alpha}(t)} \frac{\Delta t^2}{2} \\ p_{i\alpha}(t + \Delta t) &= p_{i\alpha}(t) - \left(\frac{\partial V}{\partial r_{i\alpha}(t)} + \frac{\partial V}{\partial r_{i\alpha}(t + \Delta t)} \right) \frac{\Delta t}{2}. \end{aligned} \quad (104)$$

Note that this algorithm is time-reversible. Liouville's theorem is satisfied for any value of Δt .

11.10 NVT molecular dynamics

In NVT molecular dynamics, the equations of motion are designed to generate the canonical ensemble for ergodic systems. In the PIMD code, the temperature is controlled using a family of Nosé–Hoover thermostats. This technique is also used in path-integral molecular dynamics, centroid molecular dynamics, and metadynamics.

11.10.1 Nosé–Hoover method

In the Nosé–Hoover method [8, 10], N atoms interact with a thermostat. The equations of motion for the system of N atoms are

$$\begin{aligned} \dot{r}_{i\alpha} &= \frac{p_{i\alpha}}{m_i}, \\ \dot{p}_{i\alpha} &= -\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \frac{\mathcal{P}}{\mathcal{Q}}, \end{aligned} \quad (105)$$

which are coupled to the equations of motion for the thermostat,

$$\begin{aligned} \dot{\mathcal{R}} &= \frac{\mathcal{P}}{\mathcal{Q}}, \\ \dot{\mathcal{P}} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{m_i} - 3NkT, \end{aligned} \quad (106)$$

where \mathcal{R} and \mathcal{P} are the thermostat coordinate and momentum, respectively. The total energy,

$$H = \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \frac{\mathcal{P}^2}{2\mathcal{Q}} + 3NkT\mathcal{R}, \quad (107)$$

is conserved because

$$\begin{aligned} \dot{H} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial H}{\partial p_{i\alpha}} \dot{p}_{i\alpha} + \frac{\partial H}{\partial r_{i\alpha}} \dot{r}_{i\alpha} \right) + \frac{\partial H}{\partial \mathcal{P}} \dot{\mathcal{P}} + \frac{\partial H}{\partial \mathcal{R}} \dot{\mathcal{R}} \\ &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{p_{i\alpha}}{m_i} \left(-\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \frac{\mathcal{P}}{\mathcal{Q}} \right) + \frac{\partial V}{\partial r_{i\alpha}} \frac{p_{i\alpha}}{m_i} \right\} \\ &\quad + \frac{\mathcal{P}}{\mathcal{Q}} \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{m_i} - 3NkT \right) + 3NkT \frac{\mathcal{P}}{\mathcal{Q}} = 0. \end{aligned} \quad (108)$$

It can be shown that the canonical ensemble is generated assuming ergodicity. Since H is a constant ($= E$), the phase-space volume of the combined system (atoms plus thermostat) can be written as

$$W \propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R} \int d\mathcal{P} J \delta(H - E), \quad (109)$$

where the Jacobian J satisfies

$$\begin{aligned} \dot{J} &= -J \left\{ \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial \dot{r}_{i\alpha}}{\partial r_{i\alpha}} + \frac{\partial \dot{p}_{i\alpha}}{\partial p_{i\alpha}} \right) + \frac{\partial \dot{\mathcal{R}}}{\partial \mathcal{R}} + \frac{\partial \dot{\mathcal{P}}}{\partial \mathcal{P}} \right\} \\ &= J \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{\mathcal{P}}{\mathcal{Q}} = 3N \dot{\mathcal{R}} J, \end{aligned} \quad (110)$$

and thus

$$J = \exp(3N\mathcal{R}). \quad (111)$$

Now Eq. (109) can be rewritten as

$$\begin{aligned} W &\propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R} \int d\mathcal{P} \exp(3N\mathcal{R}) \\ &\times \delta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \frac{\mathcal{P}^2}{2\mathcal{Q}} + 3NkT\mathcal{R} - E \right). \end{aligned} \quad (112)$$

Using the identity

$$\delta[h(s)] = \frac{\delta(s - s_0)}{|h'(s_0)|} \quad \text{where} \quad h(s_0) = 0, \quad (113)$$

one obtains

$$\begin{aligned} W &\propto \frac{e^{\beta E}}{3NkT} \int dr^{3N} \int dp^{3N} \int d\mathcal{P} \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \frac{\mathcal{P}^2}{2\mathcal{Q}} \right) \right] \\ &\propto \int dr^{3N} \int dp^{3N} \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right] \propto Z, \end{aligned} \quad (114)$$

where $\beta = 1/kT$. This shows that W is proportional to the partition function Z of the canonical ensemble for the system of N atoms. In the same manner, it can be shown that the probability density of finding the system at the phase point $(r_{1x}, \dots, r_{Nz}, p_{1x}, \dots, p_{Nz})$ is identical to that of the canonical ensemble:

$$\begin{aligned} &P(r_{1x}, \dots, r_{Nz}, p_{1x}, \dots, p_{Nz}) \\ &= \frac{\exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right]}{\int dr^{3N} \int dp^{3N} \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right]}. \end{aligned} \quad (115)$$

11.10.2 Nosé–Hoover chain method

In the Nosé–Hoover chain method [11], the system interacts with a chain of Nosé–Hoover thermostats to control the temperature more strongly. The system interacts with a thermostat which, in turn, interacts with the second thermostat, and so on. The equations of motion for the system of N atoms and a Nosé–

Hoover thermostat chain of length L are expressed as follows:

$$\begin{aligned}
\dot{r}_{i\alpha} &= \frac{p_{i\alpha}}{m_i} \\
\dot{p}_{i\alpha} &= -\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \frac{\mathcal{P}_1}{\mathcal{Q}_1} \\
\dot{\mathcal{R}}_l &= \frac{\mathcal{P}_l}{\mathcal{Q}_l} \\
\dot{\mathcal{P}}_1 &= \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{m_i} - 3NkT \right) - \mathcal{P}_1 \frac{\mathcal{P}_2}{\mathcal{Q}_2} \\
\dot{\mathcal{P}}_l &= \left(\frac{\mathcal{P}_{l-1}^2}{\mathcal{Q}_{l-1}} - kT \right) - \mathcal{P}_l \frac{\mathcal{P}_{l+1}}{\mathcal{Q}_{l+1}} \quad (2 \leq l \leq L-1) \\
\dot{\mathcal{P}}_L &= \frac{\mathcal{P}_{L-1}^2}{\mathcal{Q}_{L-1}} - kT,
\end{aligned} \tag{116}$$

where \mathcal{R}_l and \mathcal{P}_l are the coordinates and momenta of the l th thermostat in the chain, respectively. The total energy,

$$H = \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \frac{\mathcal{P}_1^2}{2\mathcal{Q}_1} + 3NkT\mathcal{R}_1 + \sum_{l=2}^L \left(\frac{\mathcal{P}_l^2}{2\mathcal{Q}_l} + kT\mathcal{R}_l \right), \tag{117}$$

is conserved, since

$$\begin{aligned}
\dot{H} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial H}{\partial p_{i\alpha}} \dot{p}_{i\alpha} + \frac{\partial H}{\partial r_{i\alpha}} \dot{r}_{i\alpha} \right) + \sum_{l=1}^L \frac{\partial H}{\partial \mathcal{P}_l} \dot{\mathcal{P}}_l + \sum_{l=1}^L \frac{\partial H}{\partial \mathcal{R}_l} \dot{\mathcal{R}}_l \\
&= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{p_{i\alpha}}{m_i} \left(-\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \frac{\mathcal{P}_1}{\mathcal{Q}_1} \right) + \frac{\partial V}{\partial r_{i\alpha}} \frac{p_{i\alpha}}{m_i} \right\} \\
&+ \frac{\mathcal{P}_1}{\mathcal{Q}_1} \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{m_i} - 3NkT - \mathcal{P}_1 \frac{\mathcal{P}_2}{\mathcal{Q}_2} \right) \\
&+ \sum_{l=2}^{L-1} \left\{ \frac{\mathcal{P}_l}{\mathcal{Q}_l} \left(\frac{\mathcal{P}_{l-1}^2}{\mathcal{Q}_{l-1}} - kT - \mathcal{P}_l \frac{\mathcal{P}_{l+1}}{\mathcal{Q}_{l+1}} \right) \right\} \\
&+ \frac{\mathcal{P}_L}{\mathcal{Q}_L} \left(\frac{\mathcal{P}_{L-1}^2}{\mathcal{Q}_{L-1}} - kT \right) + 3NkT \frac{\mathcal{P}_1}{\mathcal{Q}_1} + \sum_{l=2}^L kT \frac{\mathcal{P}_l}{\mathcal{Q}_l} = 0.
\end{aligned} \tag{118}$$

It can be shown that the canonical ensemble is generated assuming ergodicity. Since H is a constant ($= E$), the phase-space volume of the combined system (atoms plus thermostat chain) can be written as

$$W \propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^L \int d\mathcal{P}^L J \delta(H - E), \tag{119}$$

where the Jacobian J satisfies

$$\begin{aligned}
J &= -J \left\{ \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial \dot{r}_{i\alpha}}{\partial r_{i\alpha}} + \frac{\partial \dot{p}_{i\alpha}}{\partial p_{i\alpha}} \right) + \sum_{l=1}^L \left(\frac{\partial \dot{\mathcal{R}}_l}{\partial \mathcal{R}_l} + \frac{\partial \dot{\mathcal{P}}_l}{\partial \mathcal{P}_l} \right) \right\} \\
&= J \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{\mathcal{P}_1}{\mathcal{Q}_1} + \sum_{l=2}^L \frac{\mathcal{P}_l}{\mathcal{Q}_l} \right) = \left(3N\dot{\mathcal{R}}_1 + \sum_{l=2}^L \dot{\mathcal{R}}_l \right) J,
\end{aligned} \tag{120}$$

and thus

$$J = \exp \left(3N\mathcal{R}_1 + \sum_{l=2}^L \mathcal{R}_l \right). \quad (121)$$

Now Eq. (119) can be rewritten as

$$\begin{aligned} W &\propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^L \int d\mathcal{P}^L \exp \left(3N\mathcal{R}_1 + \sum_{l=2}^L \mathcal{R}_l \right) \\ &\times \delta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \frac{\mathcal{P}_1^2}{2\mathcal{Q}_1} + 3NkT\mathcal{R}_1 + \sum_{l=2}^L \left(\frac{\mathcal{P}_l^2}{2\mathcal{Q}_l} + kT\mathcal{R}_l \right) - E \right). \end{aligned} \quad (122)$$

Using the identity in Eq. (113), one obtains

$$\begin{aligned} W &\propto \frac{e^{\beta E}}{3NkT} \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^{L-1} \int d\mathcal{P}^L \\ &\exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right] \exp \left(-\beta \sum_{l=1}^L \frac{\mathcal{P}_l^2}{2\mathcal{Q}_l} \right) \\ &\propto \int dr^{3N} \int dp^{3N} \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right] \propto Z, \end{aligned} \quad (123)$$

where $\beta = 1/kT$. This shows that W is proportional to the partition function Z of the canonical ensemble for the system of N atoms. In the same manner, it can be shown that the probability density of finding the system at the phase point $(r_{1x}, \dots, r_{Nz}, p_{1x}, \dots, p_{Nz})$ is identical to that of the canonical ensemble, Eq. (115).

11.10.3 Massive Nosé–Hoover chain method

In the massive Nosé–Hoover chain method, a Nosé–Hoover thermostat chain is attached to each of the $3N$ degrees of freedom. To control the temperature more strongly, multiple chains can be attached to each degree of freedom. In the latter case, the thermostat masses are chosen to control vibrational modes with different frequencies present in the system. In the PIMD code, the masses are chosen such that the characteristic frequencies of the thermostats form a Fourier series. The equations of motion for N nuclei coupled to a massive Nosé–Hoover chain thermostat, in which M chains of length L are attached to each degree of freedom, are

$$\begin{aligned} \dot{r}_{i\alpha} &= \frac{p_{i\alpha}}{m_i} \\ \dot{p}_{i\alpha} &= -\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,1,m}}{\mathcal{Q}_{i\alpha,1,m}} \\ \dot{\mathcal{R}}_{i\alpha,l,m} &= \frac{\mathcal{P}_{i\alpha,l,m}}{\mathcal{Q}_{i\alpha,l,m}} \\ \dot{\mathcal{P}}_{i\alpha,1,m} &= \left(\frac{p_{i\alpha}^2}{m_i} - kT \right) - \mathcal{P}_{i\alpha,1,m} \frac{\mathcal{P}_{i\alpha,2,m}}{\mathcal{Q}_{i\alpha,2,m}} \\ \dot{\mathcal{P}}_{i\alpha,l,m} &= \left(\frac{\mathcal{P}_{i\alpha,l-1,m}^2}{\mathcal{Q}_{i\alpha,l-1,m}} - kT \right) - \mathcal{P}_{i\alpha,l,m} \frac{\mathcal{P}_{i\alpha,l+1,m}}{\mathcal{Q}_{i\alpha,l+1,m}} \quad (2 \leq l \leq L-1) \\ \dot{\mathcal{P}}_{i\alpha,L,m} &= \frac{\mathcal{P}_{i\alpha,L-1,m}^2}{\mathcal{Q}_{i\alpha,L-1,m}} - kT, \end{aligned} \quad (124)$$

where $\mathcal{R}_{i\alpha,l,m}$ and $\mathcal{P}_{i\alpha,l,m}$ are the coordinates and momenta of the l th thermostat of the m th chain attached to the α coordinate of the i th atom, respectively. The total energy,

$$H = \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{i\alpha,l,m}^2}{2\mathcal{Q}_{i\alpha,l,m}} + kT\mathcal{R}_{i\alpha,l,m} \right), \quad (125)$$

is conserved, since

$$\begin{aligned} \dot{H} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left(\frac{\partial H}{\partial p_{i\alpha}} \dot{p}_{i\alpha} + \frac{\partial H}{\partial r_{i\alpha}} \dot{r}_{i\alpha} \right) \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \frac{\partial H}{\partial \mathcal{P}_{i\alpha,l,m}} \dot{\mathcal{P}}_{i\alpha,l,m} + \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \frac{\partial H}{\partial \mathcal{R}_{i\alpha,l,m}} \dot{\mathcal{R}}_{i\alpha,l,m} \\ &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{p_{i\alpha}}{m_i} \left(-\frac{\partial V}{\partial r_{i\alpha}} - p_{i\alpha} \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,1,m}}{\mathcal{Q}_{i\alpha,1,m}} \right) + \frac{\partial V}{\partial r_{i\alpha}} \frac{p_{i\alpha}}{m_i} \right\} \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,1,m}}{\mathcal{Q}_{i\alpha,1,m}} \left(\frac{p_{i\alpha}^2}{m_i} - kT - \mathcal{P}_{i\alpha,1,m} \frac{\mathcal{P}_{i\alpha,2,m}}{\mathcal{Q}_{i\alpha,2,m}} \right) \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=2}^{L-1} \sum_{m=1}^M \left\{ \frac{\mathcal{P}_{i\alpha,l,m}}{\mathcal{Q}_{i\alpha,l,m}} \left(\frac{\mathcal{P}_{i\alpha,l-1,m}^2}{\mathcal{Q}_{i\alpha,l-1,m}} - kT - \mathcal{P}_{i\alpha,l,m} \frac{\mathcal{P}_{i\alpha,l+1,m}}{\mathcal{Q}_{i\alpha,l+1,m}} \right) \right\} \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,L,m}}{\mathcal{Q}_{i\alpha,L,m}} \left(\frac{\mathcal{P}_{i\alpha,L-1,m}^2}{\mathcal{Q}_{i\alpha,L-1,m}} - kT \right) \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M kT \frac{\mathcal{P}_{i\alpha,l,m}}{\mathcal{Q}_{i\alpha,l,m}} = 0. \end{aligned} \quad (126)$$

It can be shown that the canonical ensemble is generated assuming ergodicity. Since H is a constant ($= E$), the phase-space volume of the combined system (atoms plus thermostats) can be written as

$$W \propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^{3NLM} \int d\mathcal{P}^{3NLM} J \delta(H - E), \quad (127)$$

where the Jacobian J satisfies

$$\begin{aligned} J &= -J \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \left(\frac{\partial \dot{r}_{i\alpha}}{\partial r_{i\alpha}} + \frac{\partial \dot{p}_{i\alpha}}{\partial p_{i\alpha}} \right) + \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\partial \dot{\mathcal{R}}_{i\alpha,l,m}}{\partial \mathcal{R}_{i\alpha,l,m}} + \frac{\partial \dot{\mathcal{P}}_{i\alpha,l,m}}{\partial \mathcal{P}_{i\alpha,l,m}} \right) \right\} \\ &= J \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,l,m}}{\mathcal{Q}_{i\alpha,l,m}} \right) = \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \dot{\mathcal{R}}_{i\alpha,l,m} \right) J, \end{aligned} \quad (128)$$

and thus

$$J = \exp \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \mathcal{R}_{i\alpha,l,m} \right). \quad (129)$$

Now Eq. (127) can be rewritten as

$$\begin{aligned} W &\propto \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^{3NLM} \int d\mathcal{P}^{3NLM} \exp \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \mathcal{R}_{i\alpha,l,m} \right) \\ &\times \delta \left[\sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{p_{i\alpha}^2}{2m_i} + \sum_{l=1}^L \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,l,m}^2}{2\mathcal{Q}_{i\alpha,l,m}} + kT\mathcal{R}_{i\alpha,l,m} \right\} + V - E \right]. \end{aligned} \quad (130)$$

Using the identity in Eq. (113), one obtains

$$\begin{aligned}
W &\propto \frac{e^{\beta E}}{kT} \int dr^{3N} \int dp^{3N} \int d\mathcal{R}^{3NLM-1} \int d\mathcal{P}^{3NLM} \\
&\times \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V + \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \frac{\mathcal{P}_{i\alpha,l,m}^2}{2Q_{i\alpha,l,m}} \right) \right] \\
&\propto \int dr^{3N} \int dp^{3N} \exp \left[-\beta \left(\sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^2}{2m_i} + V \right) \right] \propto Z,
\end{aligned} \tag{131}$$

where $\beta = 1/kT$. This shows that W is proportional to the partition function Z of the canonical ensemble for the system of N atoms. In the same manner, it can be shown that the probability density of finding the system at the phase point $(r_{1x}, \dots, r_{Nz}, p_{1x}, \dots, p_{Nz})$ is identical to that of the canonical ensemble, Eq. (115).

11.11 Path integral methods

In PI methods, each atom is represented by P replicas in imaginary time slices. The parameter P is known as the number of beads. The equation of motion of a $3NP$ -dimensional system is designed such that the distribution is proportional to $\exp(-\beta V^{\text{eff}})$, where the effective potential V^{eff} is given by:

$$V^{\text{eff}} = V^{\text{harm}} + V^{\text{phys}} + V^{\text{cor}}. \tag{132}$$

where the first and second terms are the contributions of the harmonic and physical potentials, respectively:

$$V^{\text{harm}} \equiv \sum_{l=1}^P \sum_{I=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{m_i \omega_P^2}{2} \left(r_{i\alpha}^{(l)} - r_{i\alpha}^{(l+1)} \right)^2 \right\} \tag{133}$$

$$V^{\text{phys}} \equiv \frac{1}{P} \sum_{l=1}^P V \left(r_{1x}^{(l)}, \dots, r_{Nz}^{(l)} \right), \tag{134}$$

The periodicity $r_{i\alpha}^{(P+1)} = r_{i\alpha}^{(1)}$ is assumed, and the constant $\omega_P = \sqrt{P}/\beta\hbar$ is defined. The third term is the correction for the fourth order Trotter expansion:

$$V^{\text{cor}} \equiv \frac{\beta^2 \hbar^2}{24P^3} \sum_{i=1}^N \frac{1}{m_i} \sum_{l=1}^P \left\{ \left(\frac{\partial V}{\partial r_{ix}^{(l)}} \right)^2 + \left(\frac{\partial V}{\partial r_{iy}^{(l)}} \right)^2 + \left(\frac{\partial V}{\partial r_{iz}^{(l)}} \right)^2 \right\}, \tag{135}$$

which is neglected in the second order Trotter expansion:

$$V^{\text{cor}} = 0 \tag{136}$$

Note that the first term on the rhs of Eq.(132) connects replicas into a cyclic chain by harmonic interaction. It is useful to introduce a linear transformation from Cartesian coordinates $r_{i\alpha}^{(l)}$ to so-called normal mode coordinates $q_{i\alpha}^{(k)}$:

$$q_{i\alpha}^{(k)} = \frac{1}{\sqrt{P}} \sum_{l=1}^P U_{kl} r_{i\alpha}^{(l)}. \tag{137}$$

The unitary matrix U_{kl} corresponds to the eigenvector of the matrix $\overleftrightarrow{\mathbf{A}}$, where $A_{kl} = P(2\delta_{kl} - \delta_{k,l+1} - \delta_{k,l-1})$ with periodicity $0 \rightarrow P$ and $P+1 \rightarrow 1$. The first mode with $k = 1$, known as the centroid variable, is given by

$$q_{i\alpha}^{(1)} = q_{i\alpha}^{(\text{cent})} = \frac{1}{P} \sum_{l=1}^P r_{i\alpha}^{(l)}. \quad (138)$$

After this transformation, we obtain

$$V^{\text{eff}} = \sum_{k=1}^P \sum_{i=1}^N \sum_{\alpha=x,y,z} \left\{ \frac{m_i \lambda^{(k)} \omega_P^2}{2} q_{i\alpha}^{(k)2} \right\} + \frac{1}{P} \sum_{l=1}^P V(r_{1x}^{(l)}, \dots, r_{Nz}^{(l)}), \quad (139)$$

where $\lambda^{(k)}$ is the k -th eigenvalue of the matrix $\overleftrightarrow{\mathbf{A}}$, and $r_{i\alpha}^{(l)}$ is given as a function of $q_{i\alpha}^{(k)}$ by the backward transformation

$$r_{i\alpha}^{(l)} = \sqrt{P} \sum_{k=1}^P U_{lk}^\dagger q_{i\alpha}^{(k)}. \quad (140)$$

The force is given by

$$F_{i\alpha}^{(k)} = -\frac{\partial V^{\text{eff}}}{\partial q_{i\alpha}^{(k)}} = -m_i \lambda^{(k)} \omega_P^2 q_{i\alpha}^{(k)} - \frac{1}{\sqrt{P}} \sum_{l=1}^P U_{lk}^\dagger \frac{\partial V}{\partial r_{i\alpha}^{(l)}}. \quad (141)$$

We can construct the effective Hamiltonian as

$$H^{\text{eff}} = \sum_{k=1}^P \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^{(k)2}}{2\mu_i^{(k)}} + V^{\text{eff}} + H^{\text{bath}} + H^{\text{baro}}, \quad (142)$$

where $p_{i\alpha}^{(k)}$ is the momentum conjugate to $q_{i\alpha}^{(k)}$, $\mu_i^{(k)}$ is the fictitious mass, H^{bath} and H^{baro} are the energy contributions of the thermostats and barostats, respectively.

In PIMD, BCMD, CMD and (T)RPMD methods, the equations of motion are integrated with a finite time step Δt . In PIHMC, after the equations of motion are integrated for a few steps, the last move is either accepted or rejected according to the Metropolis algorithm with the acceptance probability $\min(\exp(-\Delta H^{\text{eff}}), 1)$. When rejected, all the positions (atomic positions in the NVT ensemble, the atomic positions and the box vectors in NPT and NtT ensembles) are placed back, and all the velocities (the atomic velocities in the NVT ensemble, the atomic velocities and the box velocities in the NPT and NtT ensembles) are regenerated randomly according to the Maxwell-Boltzmann distribution. The step size Δt in PIHMC is usually taken to be larger than that of PIMD.

In BCMD and TRPMD, the deterministic equation of motion for the centroid variable is combined with the stochastic equation of motion for the non-centroid variables.

PIMD, PIHMC, BCMD, CMD and (T)RPMD methods are different in the settings of the fictitious mass and the way of using the thermostats and the barostats, which are mentioned below.

11.11.1 Fictitious mass

The fictitious mass is set differently between the centroid modes ($k = 1$) and the non-centroid modes ($k \neq 1$).

PIMD and PIHMC.

$$\mu_i^{(1)} = \mu_i^{(\text{cent})} = m_i, \quad \mu_i^{(k)} = \lambda^{(k)} m_i \quad (2 \leq k \leq P). \quad (143)$$

BCMD.

$$\mu_i^{(1)} = \mu_i^{(\text{cent})} = m_i, \quad \mu_i^{(k)} = \frac{P(\Delta t)}{2\tau} \lambda^{(k)} m_i \quad (2 \leq k \leq P). \quad (144)$$

where Δt is the step size and $\tau = \beta \hbar$ is the decoherence time.

CMD.

$$\mu_i^{(1)} = \mu_i^{(\text{cent})} = m_i, \quad \mu_i^{(k)} = \gamma^2 \lambda^{(k)} m_i \quad (2 \leq k \leq P). \quad (145)$$

Theoretically, the adiabaticity parameter γ should be taken sufficiently small, $0 < \gamma \ll 1$, in CMD. However, the step size should be set smaller as γ becomes smaller, so a good compromise in the choice of γ is needed.

RPMD and TRPMD.

$$\mu_i^{(k)} = m_i \quad (1 \leq k \leq P). \quad (146)$$

11.11.2 Thermostats

PIMD for NVT ensemble. In the present implementation of PIMD, the massive Nosé-Hoover chain thermostats are attached to both the centroid modes ($3N$) and the non-centroid modes ($3NP - 3N$). The energy of the thermostats is given by

$$\begin{aligned} H_{\text{bath}} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{i\alpha,l,m}^{(\text{cent})2}}{2\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})}} + kT\mathcal{R}_{i\alpha,l,m}^{(\text{cent})} \right) \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{k=2}^P \sum_{l=1}^L \left(\frac{\mathcal{P}_{i\alpha,l}^{(k)2}}{2\mathcal{Q}_{i\alpha,l}^{(k)}} + kT\mathcal{R}_{i\alpha,l}^{(k)} \right). \end{aligned} \quad (147)$$

The M thermostat chains with the length L are multiply attached to each centroid degree of freedom. A thermostat chain with the length L is attached to each non-centroid degree of freedom. The mass of the centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})} = kT \left(M^{m-1} \frac{\tau_{\text{bath}}}{2\pi} \right)^2, \quad (148)$$

where τ_{bath} is typically chosen to be the period of (the highest) vibrational frequency present in the system. The mass of the non-centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l}^{(k)} = \frac{kT}{\omega_P^2}. \quad (149)$$

PIMD for NPT and NtT ensembles with a cubic box. The energy of the thermostats is given by

$$\begin{aligned} H_{\text{bath}} &= \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{i\alpha,l,m}^{(\text{cent})2}}{2\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})}} + kT\mathcal{R}_{i\alpha,l,m}^{(\text{cent})} \right) \\ &+ \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{k=2}^P \sum_{l=1}^L \left(\frac{\mathcal{P}_{i\alpha,l}^{(k)2}}{2\mathcal{Q}_{i\alpha,l}^{(k)}} + kT\mathcal{R}_{i\alpha,l}^{(k)} \right) \\ &+ \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{l,m}^{\text{b}2}}{2\mathcal{Q}_{l,m}^{\text{b}}} + kT\mathcal{R}_{l,m}^{\text{b}} \right). \end{aligned} \quad (150)$$

The mass of the centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})} = kT \left(M^{m-1} \frac{\tau_{\text{bath}}}{2\pi} \right)^2, \quad (151)$$

where τ_{bath} is typically chosen to be the period of (the highest) vibrational frequency present in the system. The mass of the non-centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l}^{(k)} = \frac{kT}{\omega_P^2}. \quad (152)$$

The mass of the box thermostats is chosen to be

$$\mathcal{Q}_{l,m}^{\text{b}} = kT \left(M^{m-1} \frac{\tau_{\text{baro}}}{2\pi} \right)^2. \quad (153)$$

PIMD for NPT and NtT ensembles with a hexahedron box. The energy of the thermostats is given by

$$\begin{aligned} H_{\text{bath}} = & \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{i\alpha,l,m}^{(\text{cent})2}}{2\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})}} + kT \mathcal{R}_{i\alpha,l,m}^{(\text{cent})} \right) \\ & + \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{k=2}^P \sum_{l=1}^L \left(\frac{\mathcal{P}_{i\alpha,l}^{(k)2}}{2\mathcal{Q}_{i\alpha,l}^{(k)}} + kT \mathcal{R}_{i\alpha,l}^{(k)} \right) \\ & + \sum_{\alpha=x,y,z} \sum_{\beta=x,y,z} \sum_{l=1}^L \sum_{m=1}^M \left(\frac{\mathcal{P}_{\alpha\beta,l,m}^{\text{b}2}}{2\mathcal{Q}_{\alpha\beta,l,m}^{\text{b}}} + kT \mathcal{R}_{\alpha\beta,l,m}^{\text{b}} \right). \end{aligned} \quad (154)$$

The mass of the centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l,m}^{(\text{cent})} = kT \left(M^{m-1} \frac{\tau_{\text{bath}}}{2\pi} \right)^2, \quad (155)$$

where τ_{bath} is typically chosen to be the period of (the highest) vibrational frequency present in the system. The mass of the non-centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l}^{(k)} = \frac{kT}{\omega_P^2}. \quad (156)$$

The mass of the box thermostats is chosen to be

$$\mathcal{Q}_{\alpha\beta,1,m}^{\text{b}} = kT \left(6M^{m-1} \frac{\tau_{\text{baro}}}{2\pi} \right)^2 \quad (157)$$

and

$$\mathcal{Q}_{\alpha\beta,l,m}^{\text{b}} = kT \left(M^{m-1} \frac{\tau_{\text{baro}}}{2\pi} \right)^2 \quad (158)$$

for $l \neq 1$.

BCMD. Non-centroid velocities are randomized at each step (i.e., an Andersen thermostat is attached to the non-centroid modes). The energy is conserved by keeping track of the jumps due to the randomization.

CMD. In the present implementation of CMD, the massive Nosé-Hoover chain thermostats are attached only to the non-centroid modes ($3NP - 3N$). The energy of the thermostats is given by

$$H_{\text{bath}} = \sum_{i=1}^N \sum_{\alpha=x,y,z} \sum_{k=2}^P \sum_{l=1}^L \left(\frac{\mathcal{P}_{i\alpha,l}^{(k)2}}{2\mathcal{Q}_{i\alpha,l}^{(k)}} + kT\mathcal{R}_{i\alpha,l}^{(k)} \right). \quad (159)$$

A thermostat chain with length L is attached to each non-centroid degree of freedom. The mass of the non-centroid thermostats is chosen to be

$$\mathcal{Q}_{i\alpha,l}^{(k)} = \frac{kT}{\omega_P^2} \gamma^2. \quad (160)$$

RPMD and TRPMD. A thermostat is not used in the RPMD method. Thus, the energy of the thermostats is zero,

$$H_{\text{bath}} = 0. \quad (161)$$

In TRPMD, a Langevin thermostat is attached to the non-centroid modes, and thus the energy is not conserved.

11.11.3 Barostats

PIMD and PIHMC for NVT ensemble. The barostat is not used in the PIMD and PIHMC methods for the NVT ensemble since the volume is fixed. Thus, the energy of the barostats is zero,

$$H_{\text{baro}} = 0. \quad (162)$$

PIMD and PIHMC for NPT ensemble with a cubic box. The energy of the barostat is given by

$$H_{\text{baro}} = \frac{P^{\text{b}2}}{2Q^{\text{b}}} + \mathbb{P}\mathbb{V}, \quad (163)$$

where P^{b} is the box velocity conjugate to the volume \mathbb{V} . The box mass is chosen to be

$$Q^{\text{b}} = 3(N+1)kT \left(\frac{\tau_{\text{baro}}}{2\pi} \right)^2. \quad (164)$$

PIMD and PIHMC for NPT ensemble with a hexahedron box. The energy of the barostats is given by

$$H_{\text{baro}} = \sum_{\alpha,\beta=x,y,z} \frac{P_{\alpha\beta}^{\text{b}2}}{2Q_{\alpha\beta}^{\text{b}}} + \mathbb{P}\mathbb{V}. \quad (165)$$

where $P_{\alpha\beta}^{\text{b}}$ is the box velocity conjugate to the box matrix $h_{\alpha\beta}$. The box masses are chosen to be

$$Q_{\alpha\beta}^{\text{b}} = (N+1)kT \left(\frac{\tau_{\text{baro}}}{2\pi} \right)^2. \quad (166)$$

PIMD and PIHMC for NtT ensemble with a hexahedron box. The energy of the barostats is given by

$$H_{\text{baro}} = \sum_{\alpha,\beta=x,y,z} \left(\frac{P_{\alpha\beta}^{\text{b}2}}{2Q_{\alpha\beta}^{\text{b}}} - t_{\alpha\beta} \epsilon_{\alpha\beta} \bar{\mathbb{V}} \right), \quad (167)$$

The strain tensor is defined by

$$\epsilon_{\alpha\beta} = \frac{1}{2} \left(\sum_{\nu,\gamma,\lambda=x,y,z} \bar{h}_{\gamma\alpha}^{-1} h_{\lambda\gamma} h_{\lambda\nu} \bar{h}_{\nu\beta}^{-1} - \delta_{\alpha\beta} \right) \quad (168)$$

where $\overleftrightarrow{\mathbf{h}}$ is the box matrix under zero tension. The box masses are chosen to be

$$Q_{\alpha\beta}^b = (N+1)kT \left(\frac{\tau_{\text{baro}}}{2\pi} \right)^2. \quad (169)$$

Meanwhile, the stress tensor is given by

$$\sigma_{\alpha\beta} = \frac{\bar{\mathbb{V}}}{\mathbb{V}} P_{\alpha\beta} \quad (170)$$

where the pressure tensor is defined as

$$P_{\alpha\beta} = - \sum_{\gamma,\nu,\mu,\lambda=x,y,z} h_{\alpha\gamma} \bar{h}_{\gamma\mu}^{-1} t_{\mu\lambda} \bar{h}_{\nu\lambda}^{-1} h_{\beta\nu}. \quad (171)$$

CMD and RPMD. A barostat is not used in the CMD and RPMD methods since the volume is fixed. Thus, the energy of the barostat is zero,

$$H_{\text{baro}} = 0. \quad (172)$$

11.11.4 How to set up RPMD and CMD

In the RPMD and CMD calculations, the initial positions and the initial velocities could be taken from the previous PIMD calculation [31]. In such cases, since the fictitious masses of RPMD and CMD are different from those of PIMD, the initial velocities must be scaled appropriately such that the temperature is kept the same. This can be done by simply scaling the normal mode velocities for all the modes k ,

$$\dot{q}_k^X = \sqrt{\frac{\mu_k^{\text{PIMD}}}{\mu_k^X}} \dot{q}_k^{\text{PIMD}}. \quad (173)$$

where X is either RPMD or CMD. In CMD, the initial positions and the initial velocities of the massive thermostats attached to the non-centroid modes can also be taken from the previous PIMD calculation. As the masses of these thermostats in CMD are different from those of PIMD by the factor $1/\gamma^2$, all the initial velocities of the thermostats must be scaled by $1/\gamma$ in CMD. All the above velocity scaling is automatically done in the code.

11.11.5 Dual-level hybrid Monte Carlo

In the dual-level HMC method, the low-level force, $-\nabla V^{\text{lo}}$, is used for the trial move. Meanwhile the high-level potential, V^{hi} , is used for the Metropolis acceptance probability, i.e., $P = \min[1, \exp(-\beta H^{\text{hi}})]$. In this way, the computation of the high-level force, $-\nabla V^{\text{hi}}$, can be avoided. This method works well if the difference between ΔV^{hi} and ΔV^{lo} in each trial move is marginal compared to the thermal energy, $k_{\text{B}}\mathbb{T}$.

11.11.6 Fourth order path integral hybrid Monte Carlo

In the fourth order PIHMC method, the force without the correction, $-\nabla(V^{\text{harm}} + V^{\text{phys}})$, is used for the trial move. Meanwhile, the full effective potential, V^{eff} , is used for the Metropolis acceptance probability, i.e., $P = \min[1, \exp(-\beta H^{\text{eff}})]$. In this way, the computation of the Hessian matrix, $\nabla^2 V$, can be avoided. This method works well if the change of the fourth order correction, ΔV^{cor} , in each trial move is marginal compared to the thermal energy, $k_{\text{B}}\mathbb{T}$.

11.11.7 Translational and rotational corrections

In order to obtain the vibrational spectra of isolated molecules from RPMD and CMD calculations, it would be better to eliminate the overall translation and the overall rotation of the system [31]. To do this, the correction to the centroid velocity should be applied at the initial step. In addition, the correction to the centroid force should be applied at every step during the run. Note, however, that as a trade-off of applying these translational and rotational corrections, the energy of the extended system will not be strictly conserved. For convenience, the origin of coordinates is chosen to be the center of mass of the whole system.

correction of centroid velocity. First, the centroid velocity vector of i -th atom is

$$\dot{\mathbf{q}}_i^{(\text{cent})} = \frac{1}{P} \sum_{k=1}^P \dot{\mathbf{r}}_i^{(k)}. \quad (174)$$

The translational correction is applied using

$$\dot{\mathbf{q}}_i^{(\text{cent})} \longleftarrow \dot{\mathbf{q}}_i^{(\text{cent})} - \Delta \dot{\mathbf{q}}'. \quad (175)$$

The correction term is calculated as the center-of-mass velocity of the whole system,

$$\Delta \dot{\mathbf{q}}' = \frac{\sum_{j=1}^{3N} \mu_j^{(\text{cent})} \dot{\mathbf{q}}_j^{(\text{cent})}}{\sum_{j=1}^{3N} \mu_j^{(\text{cent})}}. \quad (176)$$

Next, the rotational correction is applied using

$$\dot{\mathbf{q}}_i^{(\text{cent})} \longleftarrow \dot{\mathbf{q}}_i^{(\text{cent})} - \Delta \dot{\mathbf{q}}_i''. \quad (177)$$

The correction term is calculated as the contribution of velocity to the angular momentum of the whole system,

$$\Delta \dot{\mathbf{q}}_i'' = \boldsymbol{\theta} \times \mathbf{q}_i^{(\text{cent})}, \quad (178)$$

where the centroid angular velocity vector is defined as

$$\boldsymbol{\theta} = \overleftrightarrow{\mathbf{I}}^{(\text{cent})^{-1}} \mathbf{L}^{(\text{cent})}, \quad (179)$$

the centroid angular momentum vector as

$$\mathbf{L}^{(\text{cent})} = \sum_{i=1}^N \left(\mathbf{q}_i^{(\text{cent})} \times \mu_i^{(\text{cent})} \dot{\mathbf{q}}_i^{(\text{cent})} \right), \quad (180)$$

and the centroid moment of inertia tensor as

$$I_{xx}^{(\text{cent})} = \sum_{i=1}^N \mu_i^{(\text{cent})} (q_{iy}^{(\text{cent})} q_{iy}^{(\text{cent})} + q_{iz}^{(\text{cent})} q_{iz}^{(\text{cent})}), \quad I_{xy}^{(\text{cent})} = - \sum_{i=1}^N \mu_i^{(\text{cent})} q_{ix}^{(\text{cent})} q_{iy}^{(\text{cent})}, \quad \text{etc.} \quad (181)$$

correction of centroid force. The centroid force of i -th atom is

$$\dot{\mathbf{F}}_i^{(\text{cent})} = \frac{1}{P} \sum_{k=1}^P \dot{\mathbf{F}}_i^{(k)}. \quad (182)$$

The translational correction is applied using

$$\mathbf{F}_i^{(\text{cent})} \longleftarrow \mathbf{F}_i^{(\text{cent})} - \Delta \mathbf{F}_i'. \quad (183)$$

The correction term is calculated as the force applied to the whole system,

$$\Delta \mathbf{F}'_i = \frac{1}{N} \sum_{i=1}^N \mathbf{F}_i^{(\text{cent})}. \quad (184)$$

Next, the rotational correction is applied using

$$\mathbf{F}_i^{(\text{cent})} \leftarrow \mathbf{F}_i^{(\text{cent})} - \Delta \mathbf{F}'_i. \quad (185)$$

The correction term is calculated as the contribution of force to torque applied to the whole system,

$$\Delta \mathbf{F}''_i = \mu_i^{(\text{cent})} \dot{\boldsymbol{\omega}} \times \mathbf{r}_i^{(\text{cent})}. \quad (186)$$

Here the time derivative of centroid angular momentum,

$$\dot{\boldsymbol{\omega}} = \overleftrightarrow{\mathbf{I}}^{(\text{cent})^{-1}} \mathbf{N}^{(\text{cent})}, \quad (187)$$

has been introduced using the centroid torque

$$\mathbf{N}^{(\text{cent})} = \sum_{i=1}^N \mathbf{r}_i^{(\text{cent})} \times \mathbf{F}_i^{(\text{cent})}. \quad (188)$$

For the special case of diatomic molecules, the component parallel to the bond in the $\dot{\boldsymbol{\omega}}$ vector is eliminated to avoid the redundant rotation.

11.12 Replica exchange hybrid Monte Carlo

11.12.1 Temperature replica exchange

Consider P replicas of a system of N atoms with different temperatures. If the replicas do not interact with each other, the s -th replica constitutes the canonical ensemble at temperature $T^{(s)}$. The whole system of all replicas combined together will then constitute the ensemble corresponding to the probability density

$$\rho(\boldsymbol{\Gamma}^{(1)}, \dots, \boldsymbol{\Gamma}^{(P)}) = \rho(\boldsymbol{\Gamma}^{(1)}) \times \dots \times \rho(\boldsymbol{\Gamma}^{(P)}) \propto \exp \left\{ - \sum_{s=1}^P \beta^{(s)} H(\boldsymbol{\Gamma}^{(s)}) \right\} \quad (189)$$

where $\beta^{(s)} = \frac{1}{k_B T^{(s)}}$, and the temperature of the s -th replica is set to

$$T^{(s)} = T^{(1)} \left(\frac{T^{(P)}}{T^{(1)}} \right)^{\frac{s-1}{P-1}}. \quad (190)$$

This probability density has the following symmetry under the exchange of replicas s and t :

$$\rho(\dots, \boldsymbol{\Gamma}^{(t)}, \dots, \boldsymbol{\Gamma}^{(s)}, \dots) = \rho(\dots, \boldsymbol{\Gamma}^{(s)}, \dots, \boldsymbol{\Gamma}^{(t)}, \dots) \exp \left\{ \left(\beta^{(s)} - \beta^{(t)} \right) \left(H(\boldsymbol{\Gamma}^{(s)}) - H(\boldsymbol{\Gamma}^{(t)}) \right) \right\}. \quad (191)$$

Therefore, according to the detailed balance condition, this ensemble is not changed if the configurations are exchanged at random, with the Metropolis acceptance probability given by

$$P(\dots, \boldsymbol{\Gamma}^{(s)}, \dots, \boldsymbol{\Gamma}^{(t)}, \dots \rightarrow \dots, \boldsymbol{\Gamma}^{(t)}, \dots, \boldsymbol{\Gamma}^{(s)}, \dots) = \min(1, \exp(-\Delta)) \quad (192)$$

where

$$\Delta = - \left(\beta^{(s)} - \beta^{(t)} \right) \left(H(\boldsymbol{\Gamma}^{(s)}) - H(\boldsymbol{\Gamma}^{(t)}) \right). \quad (193)$$

In the REHMC method, this exchange attempt is performed at the Monte Carlo stage, which occurs every few steps of the integration of the equations of motion.

11.12.2 Hamiltonian replica exchange

Consider P replicas of a system of N atoms with different potentials, each describing an intermediate alchemical stage between the physical systems A and B, with energies H_A and H_B , respectively. In this case, the energy of the s -th intermediate alchemical stage for the t -th replica is given by

$$\tilde{H}^{(s)}(\mathbf{\Gamma}^{(t)}) = \left(\frac{P-s}{P-1}\right) H_A(\mathbf{\Gamma}^{(t)}) + \left(\frac{s-1}{P-1}\right) H_B(\mathbf{\Gamma}^{(t)}). \quad (194)$$

If the replicas do not interact with each other, the s -th replica constitutes the canonical ensemble of the corresponding system. The whole system of all replicas combined together will then constitute the ensemble corresponding to the probability density

$$\rho(\mathbf{\Gamma}^{(1)}, \dots, \mathbf{\Gamma}^{(P)}) = \rho(\mathbf{\Gamma}^{(1)}) \times \dots \times \rho(\mathbf{\Gamma}^{(P)}) \propto \exp \left\{ -\beta \sum_{s=1}^P \tilde{H}_s(\mathbf{\Gamma}_s) \right\} \quad (195)$$

where $\beta = \frac{1}{k_B T}$. Upon exchanging replicas s and t , the Metropolis acceptance probability that preserves this distribution is

$$P(\dots, \mathbf{\Gamma}^{(s)}, \dots, \mathbf{\Gamma}^{(t)}, \dots \rightarrow \dots, \mathbf{\Gamma}^{(t)}, \dots, \mathbf{\Gamma}^{(s)}, \dots) = \min(1, \exp(-\Delta)) \quad (196)$$

where

$$\Delta = \beta \left\{ \tilde{H}^{(t)}(\mathbf{\Gamma}^{(s)}) + \tilde{H}^{(s)}(\mathbf{\Gamma}^{(t)}) - \tilde{H}^{(s)}(\mathbf{\Gamma}^{(s)}) - \tilde{H}^{(t)}(\mathbf{\Gamma}^{(t)}) \right\}. \quad (197)$$

In the REHMC method, this exchange attempt is performed at the Monte Carlo stage, which occurs every few steps of the integration of the equations of motion.

11.13 Metadynamics

The metadynamics method was proposed by Laio and Parrinello [36]. In the PIMD code, the implementation follows closely the version described in reference [38]. The total energy is expressed as

$$\begin{aligned} H &= \sum_{l=1}^P \left[H_{\text{sys}}^{(l)}(\{\mathbf{r}^{(l)}, \mathbf{p}^{(l)}\}) + T_{\text{cv}}^{(l)}(\{\mathcal{P}^{(l)}\}) + V_{\text{cv}}^{(l)}(\{\mathcal{R}^{(l)}, \mathbf{r}^{(l)}\}) \right] \\ &+ V^{\text{hills}}(\{\mathcal{R}^{(1)}, \dots, \mathcal{R}^{(P)}, \mathbf{r}^{(1)}, \dots, \mathbf{r}^{(P)}\}) + V^{\text{cor}} + H^{\text{bath}}. \end{aligned} \quad (198)$$

Here, $\{\mathbf{r}^{(l)}\}$ and $\{\mathbf{p}^{(l)}\}$ denote the sets of $3N$ -dimensional Cartesian coordinates and momenta of the l -th walker, while $\{\mathcal{R}^{(l)}\} = (\mathcal{R}_1^{(l)}, \dots, \mathcal{R}_D^{(l)})$ and $\{\mathcal{P}^{(l)}\} = (\mathcal{P}_1^{(l)}, \dots, \mathcal{P}_D^{(l)})$ are the sets of D -dimensional coordinates and momenta of the fictitious particle for the l -th walker. The energy of the l -th walker is given by

$$H_{\text{sys}}^{(l)} = \sum_{i=1}^N \sum_{\alpha=x,y,z} \frac{p_{i\alpha}^{(l)2}}{2m_i} + V(\mathbf{r}^{(l)}). \quad (199)$$

The kinetic energy of the fictitious particle is

$$T_{\text{cv}}^{(l)} = \sum_{d=1}^D \frac{\mathcal{P}_d^{(l)2}}{2\mathcal{M}_d}, \quad (200)$$

where \mathcal{M}_d is the mass of the d -th component of the fictitious particle. The harmonic potential between the fictitious particle $\mathcal{R}_d^{(l)}$ and the collective variable $R_d^{(l)}$ is

$$V_{\text{cv}}^{(l)} = \sum_{d=1}^D \frac{\mathcal{K}_d}{2} \left\{ \mathcal{R}_d^{(l)} - R_d^{(l)}(\mathbf{r}^{(l)}) \right\}^2, \quad (201)$$

where \mathcal{K}_d is the force constant and $R_d^{(l)}$ is the d -th collective variable of the l -th walker. The sum of Gaussian hill potentials is

$$V^{\text{hills}}(t) = \sum_{l=1}^P \sum_{a=1}^{N_{\text{hills}}} \mathcal{H}(t_a) \Theta(t - t_a) \prod_{d=1}^D \exp \left\{ -\frac{\left(\mathcal{R}_d^{(l)}(t) - \mathcal{R}_d^{(l)}(t_a) \right)^2}{2\mathcal{W}_d(t_a)} \right\}, \quad (202)$$

where $\mathcal{H}(t_a)$ and $\mathcal{W}_d(t_a)$ are the height and width of each Gaussian hill added at time t_a , and $\Theta(t)$ is the Heaviside step function. Since the hills are added where the fictitious particle has passed, this term is history-dependent. The correction term,

$$V^{\text{cor}}(t) = - \sum_{l=1}^P \sum_{a=1}^{N_{\text{hills}}} \mathcal{H}(t_a) \Theta(t - t_a), \quad (203)$$

is required to ensure total energy conservation. Note that $V^{\text{hills}}(t)$ cancels with $V^{\text{cor}}(t)$ at $t = t_a$, making the potential continuous at the position of the evolving fictitious particle. This correction term was not included in the original metadynamics paper. However, it does not affect the metadynamics trajectory, since the derivative of $V^{\text{cor}}(t)$ with respect to $\mathcal{R}_d^{(l)}(t)$ is always zero. Thus, this term is needed only when monitoring the energy conservation of the trajectory. Finally, H^{bath} represents the energy contribution from massive Nosé–Hoover chain thermostats. Thermostat chains are attached to each degree of freedom of the system and to the fictitious particle. See below for the procedure to set thermostat masses.

The RESPA integrator is useful when the forces can be separated into slow and fast components. In the PIMD code, this technique is applied to metadynamics. The forces arising from V_{cv} and V^{hills} are updated with a smaller timestep than those from the physical potential V .

Several parameters control metadynamics. The dimension of the collective variables D , the hill height \mathcal{H} , and the hill width \mathcal{W}_d ($1 \leq d \leq D$) must be determined first. During the simulation, a new hill is added whenever the collective variables move by more than 1.5 times the hill width from the last hill, or when a specified time limit has passed since the previous hill. The latter time is defined by the parameter τ_{lim} . The motion of the fictitious particle is related to the parameter τ_{cv} , which represents the characteristic time for the particle to move 1.5 times the hill width via thermal diffusion at temperature T . This determines the mass of the collective variables by

$$\mathcal{M}_d = kT \left(\frac{\tau_{\text{cv}}}{1.5\mathcal{W}_d} \right)^2. \quad (204)$$

The parameter τ_{fc} is the oscillation period for V^{cv} . This determines the force constant by

$$\mathcal{K}_d = \mathcal{M}_d \left(\frac{2\pi}{\tau_{\text{fc}}} \right)^2. \quad (205)$$

The following is a recipe to start metadynamics simulations.

1. Decide the dimension of collective variables, D . For each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. Currently, six types are implemented:

(1) bond distance of atoms A–B,

- (2) bond angle of atoms A–B–C,
- (3) dihedral angle of atoms A–B–C–D,
- (4) difference between A–B and B–C distances,
- (5) coordination number between A and B species,
- (6) difference in coordination numbers between A and B species and C and D species.

Set the collective variables using the keyword `<nmeta>`.

2. Set the hill height \mathcal{H} using the keyword `<gh_meta>`.
3. Set the hill width \mathcal{W}_d for each collective variable using the keyword `<gw_meta>`.
4. Set the characteristic time τ_{cv} for the fictitious particle to move 1.5 times the hill width, using the keyword `<time_cv_meta>`.
5. Set the time limit for adding hills, τ_{lim} , using the keyword `<time_limit_meta>`.
6. Set the oscillation period of the harmonic potential, τ_{fc} , using the keyword `<time_fc_meta>`.
7. Set the timestep dt for updating the physical potential V using `<dt>`. Also set the number of updates of the rapidly varying potentials V^{hills} and V^{cv} per step dt using `<nref_meta>`.
8. (Optional) Set the maximum number of Gaussian hills using `<mg_meta>`.
9. (Optional) Set the print interval for energy components using `<iprint_meta>`.
10. (Optional) Set the print interval for hill reconstruction using `<iprint_rec_meta>`. Set the parameters for hill reconstruction using `<params_rec_meta>` and `<ndim_rec_meta>`.
11. (Optional) Set the print interval of the collective variable trajectory using `<iprint_cv_meta>`.
12. Set the characteristic timescale for determining the mass of the thermostat attached to the fictitious particle using `<time_cv_bath>`.

In some cases, it may be convenient to change the Gaussian hill height and width during a simulation. To do so, first interrupt the run with a soft exit, and then restart it with new settings of `<gh_meta>` and `<gw_meta>`. During the restarted run, Gaussian hill functions specified in *input.dat* are simply appended to the list *hills.ini*.

11.14 Conventional and well-tempered metadynamics

In conventional metadynamics, the bias potential V^{hills} is equal to the deposited hills U^{hills} , i.e.,

$$V^{hills} = U^{hills}, \quad (206)$$

and the free energy is estimated as

$$A^{hills} = -U^{hills}. \quad (207)$$

Meanwhile, in well-tempered metadynamics, the bias potential is

$$V^{hills} = k_B \Delta T \log \left(1 + \frac{U^{hills}}{k_B \Delta T} \right), \quad (208)$$

and the free energy landscape is estimated as

$$A^{hills} = -k_B (T + \Delta T) \log \left(1 + \frac{U^{hills}}{k_B \Delta T} \right), \quad (209)$$

and therefore

$$A^{\text{hills}} = -\gamma V^{\text{hills}}, \quad (210)$$

where $\gamma = \frac{T+\Delta T}{\Delta T}$. Well-tempered metadynamics with $\Delta T = \infty$ corresponds to conventional metadynamics. The deposited hills U^{hills} are given in the file *hills.ini*. The free energy landscape A^{hills} is given in the file *rec.out*.

The relation between the unbiased probability

$$P_{\text{unbiased}}(\mathcal{R}) = \frac{e^{-\beta A^{\text{hills}}(\mathcal{R})}}{\int d\mathcal{R} e^{-\beta A^{\text{hills}}(\mathcal{R})}} \quad (211)$$

and the biased probability

$$P_{\text{biased}}(\mathcal{R}, t) = \frac{e^{-\beta(A^{\text{hills}}(\mathcal{R}) + V^{\text{hills}}(\mathcal{R}, t))}}{\int d\mathcal{R} e^{-\beta(A^{\text{hills}}(\mathcal{R}) + V^{\text{hills}}(\mathcal{R}, t))}} \quad (212)$$

is

$$P_{\text{unbiased}}(\mathcal{R}) = P_{\text{biased}}(\mathcal{R}, t) e^{\beta(V^{\text{hills}}(\mathcal{R}, t) - c(t))}, \quad (213)$$

where

$$c(t) = \beta^{-1} \log \frac{\int d\mathcal{R} e^{-\beta A^{\text{hills}}(\mathcal{R})}}{\int d\mathcal{R} e^{-\beta(A^{\text{hills}}(\mathcal{R}) + V^{\text{hills}}(\mathcal{R}, t))}} = \beta^{-1} \log \frac{\int d\mathcal{R} e^{\beta \gamma V^{\text{hills}}(\mathcal{R}, t)}}{\int d\mathcal{R} e^{\beta(\gamma-1)V^{\text{hills}}(\mathcal{R}, t)}}. \quad (214)$$

Assuming the local equilibrium approximation and ergodicity, the unbiased average of A is given by

$$\langle A \rangle_{\text{unbiased}} \approx \frac{\left\langle A e^{\beta(V^{\text{hills}}(\mathcal{R}, t) - c(t))} \right\rangle_{\text{biased}}}{\left\langle e^{\beta(V^{\text{hills}}(\mathcal{R}, t) - c(t))} \right\rangle_{\text{biased}}} = \frac{\int_0^T A(t) e^{\beta(V^{\text{hills}}(\mathcal{R}, t) - c(t))} dt}{\int_0^T e^{\beta(V^{\text{hills}}(\mathcal{R}, t) - c(t))} dt}, \quad (215)$$

where the last expression describes the time average of the biased metadynamics trajectory.

11.14.1 Coordination number

The coordination number between species A and species B, using the rational function, is given by

$$R_d = \sum_{i \in A} \sum_{j \in B} \frac{1 - \left(\frac{r_{ij}}{d_{ij}}\right)^{\nu_{ij}}}{1 - \left(\frac{r_{ij}}{d_{ij}}\right)^{\mu_{ij}}}. \quad (216)$$

where μ_{ij} , ν_{ij} , and d_{ij} are three parameters that determine the functional form. The coordination number within species A is given by

$$R_d = \sum_{i \in A} \sum_{j > i \in A} \frac{1 - \left(\frac{r_{ij}}{d_{ij}}\right)^{\nu_{ij}}}{1 - \left(\frac{r_{ij}}{d_{ij}}\right)^{\mu_{ij}}}. \quad (217)$$

11.15 Well-sliced metadynamics

The total Hamiltonian is [81, 82]

$$H = H^{\text{cons}} + H^{\text{meta}} + H^{\text{afed}} + H^{\text{bath}}, \quad (218)$$

where

$$H^{\text{cons}} = \sum_{l=1}^P \left[H_{\text{sys}}^{(l)} \left(\left\{ \mathbf{r}^{(l)}, \mathbf{p}^{(l)} \right\} \right) + V_{\text{cv}}^{(l)} \left(\left\{ \mathcal{R}_c^{(l)}, \mathbf{r}^{(l)} \right\} \right) \right] \quad (219)$$

is the contribution from constrained MD,

$$\begin{aligned} H^{\text{meta}} &= \sum_{l=1}^P \left[H_{\text{sys}}^{(l)} \left(\left\{ \mathbf{r}^{(l)}, \mathbf{p}^{(l)} \right\} \right) + T_{\text{cv}}^{(l)} \left(\left\{ \mathcal{P}_m^{(l)} \right\} \right) + V_{\text{cv}}^{(l)} \left(\left\{ \mathcal{R}_m^{(l)}, \mathbf{r}^{(l)} \right\} \right) \right] \\ &+ V^{\text{hills}} \left(\left\{ \mathcal{R}_m^{(1)}, \dots, \mathcal{R}_m^{(P)}, \mathbf{r}^{(1)}, \dots, \mathbf{r}^{(P)} \right\} \right) + V^{\text{cor}} \end{aligned} \quad (220)$$

is the contribution from metadynamics,

$$H^{\text{afed}} = \sum_{l=1}^P \left[H_{\text{sys}}^{(l)} \left(\left\{ \mathbf{r}^{(l)}, \mathbf{p}^{(l)} \right\} \right) + T_{\text{cv}}^{(l)} \left(\left\{ \mathcal{P}_a^{(l)} \right\} \right) + V_{\text{cv}}^{(l)} \left(\left\{ \mathcal{R}_a^{(l)}, \mathbf{r}^{(l)} \right\} \right) \right] \quad (221)$$

is the contribution from adiabatic free energy dynamics, and H^{bath} is the contribution from thermostats. Note that, in the implementation in the PIMD code, the temperature is always set equal to the physical temperature. The free energy with respect to the constrained MD collective variable, \mathcal{R}_c , and the metadynamics collective variable, \mathcal{R}_m , is defined as

$$A(\mathcal{R}_c, \mathcal{R}_m) = -\beta^{-1} \log \int d\mathbf{r} \exp(-\beta V_{\text{sys}}(\mathbf{r})). \quad (222)$$

The reweighting scheme employed in the PIMD code is different from that of the original method. We use the equivalence for the free energy:

$$A(\mathcal{R}_c, \mathcal{R}_m) = A'_c(\mathcal{R}_c) + \Delta A_{\mathcal{R}_c}(\mathcal{R}_m), \quad (223)$$

where we have defined the free energy in the reduced space

$$A'_c(\mathcal{R}_c) = -\beta^{-1} \log \int d\mathcal{R}_m \exp(-\beta A(\mathcal{R}_c, \mathcal{R}_m)), \quad (224)$$

and the shift from it

$$\Delta A_{\mathcal{R}_c}(\mathcal{R}_m) = -\beta^{-1} \log \frac{\exp(-\beta A(\mathcal{R}_c, \mathcal{R}_m))}{\int d\mathcal{R}_m \exp(-\beta A(\mathcal{R}_c, \mathcal{R}_m))}. \quad (225)$$

The first term is evaluated by thermodynamic integration along the set of different constraints $(\mathcal{R}_c^{(1)}, \dots, \mathcal{R}_c^{(P)})$,

$$A'_c(\mathcal{R}_c) = \int d\mathcal{R}_c \nabla A'_c(\mathcal{R}_c) = \sum_l \left(\mathcal{R}_c^{(l+1)} - \mathcal{R}_c^{(l)} \right) \left(\frac{\nabla A'_c(\mathcal{R}_c^{(l+1)}) + \nabla A'_c(\mathcal{R}_c^{(l)})}{2} \right), \quad (226)$$

where

$$\begin{aligned} \nabla A'_c(\mathcal{R}_c) &= \frac{\nabla \int d\mathcal{R}_m \exp(-\beta A(\mathcal{R}_c, \mathcal{R}_m))}{\int d\mathcal{R}_m \exp(-\beta A(\mathcal{R}_c, \mathcal{R}_m))} = \frac{\int d\mathbf{r} \exp(-\beta V_{\text{sys}}(\mathbf{r})) \nabla \delta(\mathcal{R}_c - R_c(\mathbf{r}))}{\int d\mathbf{r} \exp(-\beta V_{\text{sys}}(\mathbf{r})) \delta(\mathcal{R}_c - R_c(\mathbf{r}))} \\ &\approx \frac{\int d\mathcal{R}_c \int d\mathbf{r} \exp(-\beta (V_{\text{sys}}(\mathbf{r}) + V_{\text{cv}}(\mathbf{r}, \mathcal{R}_c))) [-\beta \mathcal{K}_c(\mathcal{R}_c - R_c(\mathbf{r}))]}{\int d\mathcal{R}_c \int d\mathbf{r} \exp(-\beta (V_{\text{sys}}(\mathbf{r}) + V_{\text{cv}}(\mathbf{r}, \mathcal{R}_c)))} \\ &= \frac{\int d\mathcal{R}_m \int d\mathcal{R}_c \int d\mathbf{r} \exp(-\beta V) [-\beta \mathcal{K}_c(\mathcal{R}_c - R_c(\mathbf{r}))] \exp(\beta (V^{\text{hills}}(\mathcal{R}_m) + V^{\text{cor}}))}{\int d\mathcal{R}_m \int d\mathcal{R}_c \int d\mathbf{r} \exp(-\beta V) \exp(\beta (V^{\text{hills}}(\mathcal{R}_m) + V^{\text{cor}}))} \\ &= \frac{\langle [-\beta \mathcal{K}_c(\mathcal{R}_c - R_c(\mathbf{r}))] \exp(\beta (V^{\text{hills}}(\mathcal{R}_m) + V^{\text{cor}})) \rangle}{\langle \exp(\beta (V^{\text{hills}}(\mathcal{R}_m) + V^{\text{cor}})) \rangle}, \end{aligned} \quad (227)$$

with

$$V = V_{\text{sys}}(\mathbf{r}) + V_{\text{cv}}(\mathbf{r}, \mathcal{R}_c) + V^{\text{hills}}(\mathcal{R}_m) + V^{\text{cor}}. \quad (228)$$

Note that the second term does not change upon the addition of any constant function of \mathcal{R}_c . Explicitly,

$$\Delta A_{\mathcal{R}_c}(\mathcal{R}_m) = -\beta^{-1} \log \frac{\exp(-\beta A''_{\mathcal{R}_c}(\mathcal{R}_m))}{\int d\mathcal{R}_m \exp(-\beta A''_{\mathcal{R}_c}(\mathcal{R}_m))}, \quad (229)$$

for

$$A''_{\mathcal{R}_c}(\mathcal{R}_m) = A(\mathcal{R}_c, \mathcal{R}_m) + c(\mathcal{R}_c). \quad (230)$$

Thus, $A''_{\mathcal{R}_c}(\mathcal{R}_m)$ is taken as the metadynamics free energy surface at each constraint, \mathcal{R}_c .

11.16 Mean force dynamics (AFED)

The Helmholtz free energy of the l -th bead (replica) is defined as

$$A_{\text{cv}}^{(l)}(\mathcal{R}_1, \dots, \mathcal{R}_D) = -\beta^{-1} \log \int dr_{1x}^{(l)} \dots dr_{Nz}^{(l)} e^{-\beta V^{(l)}(r_{1x}^{(l)}, \dots, r_{Nz}^{(l)})} \prod_{d=1}^D \delta(\mathcal{R}_d^{(l)} - R_d^{(l)}(\mathbf{r}^{(l)})) \quad (231)$$

where $\{\mathbf{r}^{(l)}\}$ and $\{\mathbf{p}^{(l)}\}$ denote the sets of $3N$ -dimensional Cartesian coordinates and momenta of the system for the l -th bead, while $\{\mathcal{R}^{(l)}\} = (\mathcal{R}_1^{(l)}, \dots, \mathcal{R}_D^{(l)})$ and $\{\mathcal{P}^{(l)}\} = (\mathcal{P}_1^{(l)}, \dots, \mathcal{P}_D^{(l)})$ denote the sets of D -dimensional coordinates and momenta of the fictitious CV particles, respectively, for the l -th bead. $R_d^{(l)}$ is the d -th collective variable of the l -th bead.

In driven AFED [77], the free energy can be approximated as

$$A_{\text{cv}}^{(l)} \approx -\beta^{-1} \log \int dr_{1x} \dots dr_{Nz} e^{-\beta(V^{(l)} + V_{\text{cv}}^{(l)})} + \text{const}. \quad (232)$$

where

$$V_{\text{cv}}^{(l)} = \sum_{d=1}^D \frac{\mathcal{K}_d}{2} \left\{ \mathcal{R}_d^{(l)} - R_d^{(l)}(\mathbf{r}^{(l)}) \right\}^2 \quad (233)$$

is the sum of harmonic potentials acting between the fictitious particle $\mathcal{R}_d^{(l)}$ and the collective variable $R_d^{(l)}$ with the force constant \mathcal{K}_d . The free energy gradient is expressed as

$$\frac{\partial A_{\text{cv}}^{(l)}}{\partial \mathcal{R}_d^{(l)}} = \frac{\int dr_{1x} \dots dr_{Nz} \mathcal{K}_d \left\{ \mathcal{R}_d^{(l)} - R_d^{(l)}(\mathbf{r}^{(l)}) \right\} e^{-\beta(V^{(l)} + V_{\text{cv}}^{(l)})}}{\int dr_{1x} \dots dr_{Nz} e^{-\beta(V^{(l)} + V_{\text{cv}}^{(l)})}} = \left\langle \mathcal{K}_d \left\{ \mathcal{R}_d^{(l)} - R_d^{(l)}(\mathbf{r}^{(l)}) \right\} \right\rangle_{\text{md}}. \quad (234)$$

Except for the finite-difference version of gentlest ascent dynamics, the coordinates and momenta are identical for all beads $1 \leq l \leq P$, i.e.,

$$\mathcal{R}_d = \mathcal{R}_d^{(l)}, \quad \mathcal{P}_d = \mathcal{P}_d^{(l)} \quad (235)$$

In this case, the mean force in Eq.(234) can be estimated using parallel dynamics [75] as

$$\frac{\partial A_{\text{cv}}}{\partial \mathcal{R}_d} = \sum_{l=1}^P W_l \left\langle \mathcal{K}_d \left\{ \mathcal{R}_d - R_d^{(l)}(\mathbf{r}^{(l)}) \right\} \right\rangle_{\text{md}}, \quad (236)$$

where

$$W_l = \frac{e^{-\beta w^l}}{\sum_l e^{-\beta w^l}}, \quad (237)$$

and

$$w^l = \sum_d \int_{\mathcal{R}_d(0)}^{\mathcal{R}_d(t)} \left\langle \frac{\partial V_{\text{cv}}^{(l)}}{\partial \mathcal{R}_d} \right\rangle_{\text{md}} d\mathcal{R}_d. \quad (238)$$

W_l in Eq.(237) is used when `<ioption_afed> = 2`, while W_l is set to “1” for any replica with `<ioption_afed> = 1` (default setting). See ref.[75] for details of the parallel dynamics.

11.16.1 Steepest descent

Theory. The minimum of the free energy is searched using the steepest descent method. The mass-weighted coordinate of a collective variable is defined as

$$\mathcal{Q}_d = \sqrt{\mathcal{M}_d} \mathcal{R}_d. \quad (239)$$

The steepest descent path is defined by the trajectory of the equation

$$\frac{d\mathcal{Q}_d}{dt} = -\frac{\nabla_d A_{\text{cv}}}{|\nabla A_{\text{cv}}|} \quad (240)$$

where the free energy gradient with respect to \mathcal{Q}_d is defined as

$$\nabla_d A_{\text{cv}} = \frac{\partial A_{\text{cv}}}{\partial \mathcal{Q}_d} = \frac{1}{\sqrt{\mathcal{M}_d}} \frac{\partial A_{\text{cv}}}{\partial \mathcal{R}_d}, \quad (241)$$

and the norm is given by

$$|\nabla A_{\text{cv}}|^2 = \sum_{d=1}^D \frac{1}{\mathcal{M}_d} \left(\frac{\partial A_{\text{cv}}}{\partial \mathcal{R}_d} \right)^2. \quad (242)$$

Then Eq.(240) is equivalent to

$$\mathcal{M}_d \frac{d\mathcal{R}_d}{dt} = -\frac{1}{|\nabla A_{\text{cv}}|} \frac{\partial A_{\text{cv}}}{\partial \mathcal{R}_d}. \quad (243)$$

In the PIMD code, the mass of the fictitious particle is set as

$$\mathcal{M}_d = \frac{1}{\Delta \mathcal{R}_d^{\text{ref}2}} \quad (244)$$

where the reference shift $\Delta \mathcal{R}_d^{\text{ref}}$ is a parameter with the same dimension as the corresponding collective variable. Then Δt_{cv} becomes dimensionless.

Implementation.

- The Euler method

$$\mathcal{Q}_d(t + dt) = \mathcal{Q}_d(t) - \frac{\nabla_d A_{\text{cv}}}{|\nabla A_{\text{cv}}|} dt \quad (245)$$

is employed.

- The PIMD code checks whether the position of the fictitious particle is inside the designated range

$$\mathcal{R}_d^{\min} < \mathcal{R}_d < \mathcal{R}_d^{\max} \quad (246)$$

and the free energy satisfies

$$A_{\text{cv}} < A_{\text{cv}}^{\max}. \quad (247)$$

The calculation stops if these conditions are violated. The parameters \mathcal{R}_d^{\min} and \mathcal{R}_d^{\max} are specified by the keyword `<params_afed>`, while the parameter A_{cv}^{\max} is specified by the keyword `<fenergy_max_afed>`.

- The AFED step size is initially set by $\Delta t_{\text{cv}} =$ (the value given by the keyword `<dt_descent_afed>`), but it is reduced when the system is judged to be close to a minimum. This judgement is based on the angle between the free energy gradients of the current AFED step (t) and the previous AFED step ($t - \Delta t_{\text{cv}}$),

$$\xi = \arccos \left(\frac{\nabla A_{\text{cv}}(t) \cdot \nabla A_{\text{cv}}(t - \Delta t_{\text{cv}})}{|\nabla A_{\text{cv}}(t)| |\nabla A_{\text{cv}}(t - \Delta t_{\text{cv}})|} \right). \quad (248)$$

If $\xi > 90^\circ$, the system is judged to be converging toward a stationary point (a minimum in this case). Then Δt_{cv} is decreased by multiplying by a damping factor λ^{afed} , i.e., $\Delta t_{\text{cv}} \rightarrow \lambda^{\text{afed}} \Delta t_{\text{cv}}$, where λ^{afed} is specified by the keyword `<dt_damp_afed>`. On the other hand, if $\xi \leq 90^\circ$ persists for more than five consecutive AFED steps, the system is judged to be far from a stationary point. Then Δt_{cv} is increased again by dividing by the damping factor, $\Delta t_{\text{cv}} \rightarrow \Delta t_{\text{cv}} / \lambda^{\text{afed}}$. This adjustment occurs only if Δt_{cv} is smaller than the initial value specified by the keyword `<dt_descent_afed>`. When Δt_{cv} becomes smaller than the converged AFED step size, $\Delta t_{\text{cv}} <$ (the value given by the keyword `<dt_conv_afed>`), the calculation stops.

Simulation setup. Here is a recipe for starting steepest descent simulations.

1. Decide the dimension of the collective variables, D . Then, for each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. Currently, five types are implemented, i.e.,
 - (1) bond distance between atoms A–B,
 - (2) bond angle associated with atoms A–B–C,
 - (3) dihedral angle associated with atoms A–B–C–D,
 - (4) difference between A–B and B–C distances,
 - (5) coordination number between A and B species.

Set the collective variables using the keyword `<ncons>`.

2. Set the force constant of the harmonic potential, \mathcal{K}_d , using the keyword `<params_cons>`.
3. Set the initial AFED step size, the damping factor, and the converged AFED step size using the keywords `<dt_descent_afed>`, `<dt_damp_afed>`, and `<dt_conv_afed>`, respectively.
4. Set the reference shift, $\Delta \mathcal{R}_d^{\text{ref}}$, as well as the lower bound \mathcal{R}_d^{\min} and the upper bound \mathcal{R}_d^{\max} , for each type of collective variable using the keyword `<params_afed>`.
5. Set the upper bound of the free energy to be explored, A_{cv}^{\max} , using the keyword `<fenergy_max_afed>`.
6. Set the number of preliminary and production MD steps using the keywords `<nstep_pre_afed>` and `<nstep_pro_afed>`, respectively.

11.16.2 Gentlest ascent dynamics

Theory. The saddle point of the free energy is searched using gentlest ascent dynamics. The position of the fictitious particle is described in terms of the mass-weighted coordinates \mathcal{Q}_d . Gentlest ascent dynamics is defined by the trajectory of the equation

$$\frac{d\mathcal{Q}_d}{dt} = \frac{\tilde{F}_d}{\sqrt{\sum_{d=1}^D \tilde{F}_d^2}} \quad (249)$$

with

$$\tilde{F}_d = -\nabla_d A_{\text{cv}} + \left(2 \sum_{d'=1}^D \nabla_{d'} A_{\text{cv}} \cdot n_{d'} \right) n_d, \quad (250)$$

where n_d is a vector that evolves according to

$$\frac{dn_d}{dt} = \frac{1}{\gamma^{\text{gad}}} \left[-\sum_{d'=1}^D \nabla_{dd'}^2 A_{\text{cv}} \cdot n_{d'} + \left(\sum_{d'=1}^D n_{d'} \sum_{d''=1}^D \nabla_{d'd''}^2 A_{\text{cv}} \cdot n_{d''} \right) n_d \right] \quad (251)$$

with the Hessian

$$\nabla_{dd'}^2 A_{\text{cv}} = \frac{\partial^2 A_{\text{cv}}}{\partial \mathcal{Q}_d \partial \mathcal{Q}_{d'}} \quad (252)$$

and the parameter γ^{gad} .

Implementation.

- The Euler method is used to update the position of the fictitious particle.

$$\mathcal{Q}_d(t+dt) = \mathcal{Q}_d + \frac{\tilde{F}_d}{\sqrt{\sum_{d=1}^D \tilde{F}_d^2}} dt \quad (253)$$

with

$$\tilde{F}_d = -a_d + \left(2 \sum_{d'=1}^D a_{d'} \cdot n_{d'} \right) n_d. \quad (254)$$

The vector a_d is computed either analytically or numerically (see below).

- The following equation is used to update the unit vector.

$$n_d(t+dt) = \frac{n_d(t) - b_d(t)dt/\gamma^{\text{gad}}}{\sqrt{\sum_{d'=1}^D [n_{d'}(t) - b_{d'}(t)dt/\gamma^{\text{gad}}]^2}} \quad (255)$$

where

$$b_d(t) = \sum_{d'=1}^D \nabla_{dd'}^2 A_{\text{cv}}(t) \cdot n_{d'}(t). \quad (256)$$

The vector b_d is computed either analytically or numerically (see below).

- The PIMD code ensures that the position of the fictitious particle is inside the designated range

$$\mathcal{R}_d^{\min} < \mathcal{R}_d < \mathcal{R}_d^{\max}. \quad (257)$$

The calculation stops if this range is exceeded. The parameters \mathcal{R}_d^{\min} and \mathcal{R}_d^{\max} are given by the keyword `<params_afed>`, while the parameter A_{cv}^{\max} is given by the keyword `<fenergy_max_afed>`.

- The AFED step size is initially set by $\Delta t_{\text{cv}} =$ (the value given by the keyword `<dt_ascent_afed>`), but it is reduced when the system is judged to be close to the saddle point. This judgement is based on the angle between the free energy gradients of the current AFED step (t) and the previous AFED step ($t - \Delta t_{\text{cv}}$),

$$\xi = \arccos \left(\frac{\nabla A_{\text{cv}}(t) \cdot \nabla A_{\text{cv}}(t - \Delta t_{\text{cv}})}{|\nabla A_{\text{cv}}(t)| |\nabla A_{\text{cv}}(t - \Delta t_{\text{cv}})|} \right). \quad (258)$$

If $\xi > 90^\circ$, the system is judged to be converging to a stationary point (a saddle point, in this case). Then Δt_{cv} is decreased by multiplying by a damping factor λ^{afed} , i.e., $\Delta t_{\text{cv}} \rightarrow \lambda^{\text{afed}} \Delta t_{\text{cv}}$, where λ^{afed} is given by the keyword `<dt_damp_afed>`. On the other hand, if $\xi \leq 90^\circ$ persists for more than five consecutive AFED steps, the system is judged to be far from a stationary point. Then Δt_{cv} is increased again by dividing by the damping factor, $\Delta t_{\text{cv}} \rightarrow \Delta t_{\text{cv}} / \lambda^{\text{afed}}$. This adjustment occurs only if Δt_{cv} is smaller than the initial value given by the keyword `<dt_descent_afed>`. When Δt_{cv} becomes smaller than the converged AFED step size, $\Delta t_{\text{cv}} <$ (the value given by the keyword `<dt_conv_afed>`), the calculation stops.

- In the case of `<hessian_ascent_afed> = ANALYTICAL`, the Hessian is calculated analytically.

$$\frac{\partial^2 A_{\text{cv}}^{(l)}}{\partial \mathcal{R}_d^{(l)} \partial \mathcal{R}_{d'}^{(l)}} = \delta_{dd'} \mathcal{K}_d - \beta \left[\left\langle \mathcal{F}_d^{(l)} \mathcal{F}_{d'}^{(l)} \right\rangle_{\text{md}} - \left\langle \mathcal{F}_d^{(l)} \right\rangle_{\text{md}} \left\langle \mathcal{F}_{d'}^{(l)} \right\rangle_{\text{md}} \right] \quad (259)$$

where

$$\mathcal{F}_d^{(l)} = -\mathcal{K}_d \left\{ \mathcal{R}_d^{(l)} - R_d^{(l)} \left(\mathbf{r}^{(l)} \right) \right\}. \quad (260)$$

- In the case of `<hessian_ascent_afed> = NUMERICAL`, the finite difference method is employed. This is done as follows. The free energy gradient, $\nabla_{d'} A_{\text{cv}}$, is evaluated at each bead s , at the position $\mathcal{Q}^{(s)} = \left\{ \mathcal{Q}_1^{(s)}, \dots, \mathcal{Q}_D^{(s)} \right\}$ with $1 \leq s \leq P$. The P beads are centered at $\mathcal{Q} = \{ \mathcal{Q}_1, \dots, \mathcal{Q}_D \}$ and are distributed along the unit vector $\mathbf{n} = \{ n_1, \dots, n_D \}$ with equal spacing from $-\epsilon^{\text{gad}}$ to $+\epsilon^{\text{gad}}$, i.e.,

$$\mathcal{Q}_d^{(s)} = \mathcal{Q}_d + n_d \epsilon_s^{\text{gad}} \quad (261)$$

where

$$\epsilon_s^{\text{gad}} = \epsilon^{\text{gad}} \left(\frac{2s - P - 1}{P - 1} \right) \quad (262)$$

and ϵ^{gad} is a constant value given by the keyword `<fdiff_sampling_afed>`. From the first-order Taylor expansion of the gradient of each bead $\nabla_{d'} A_{\text{cv}} \left(\mathcal{Q}^{(s)} \right)$ around the center \mathcal{Q} , one obtains

$$\nabla_{d'} A_{\text{cv}} \left(\mathcal{Q}^{(s)} \right) = a_{d'} + b_{d'} \epsilon_s^{\text{gad}} + c_{d'} \epsilon_s^{\text{gad}^2} + \dots \quad (263)$$

where

$$a_{d'} = \nabla_{d'} A_{\text{cv}} (\mathcal{Q}) = \nabla_{d'} A_{\text{cv}} \quad (264)$$

and

$$b_{d'} = \sum_{d=1}^D \nabla_{dd'}^2 A_{\text{cv}}(\mathcal{Q}) \cdot n_d = \sum_{d=1}^D \nabla_{dd'}^2 A_{\text{cv}} \cdot n_d. \quad (265)$$

Thus, $a_{d'}$ and $b_{d'}$ can be obtained by a least-squares fitting of the function $\nabla_{d'} A_{\text{cv}}(\mathcal{Q}^{(s)})$ with respect to ϵ_s^{gad} . In the PIMD code, the least-squares fitting is performed using a quadratic function.

The implementation has been modified from version 2.3 as follows with respect to the guiding vector \mathbf{n} .

- When restarted with the status “EQ”, a random displacement is applied in the first AFED step, and \mathbf{n} is generated along the displacement vector.
- When restarted with the status “AS”, a calculation of the full Hessian \mathbf{H} (which requires computational time equivalent to D AFED steps, where D is the CV dimension) is performed at the first AFED step, and \mathbf{n} is set as the lowest eigenvector.
- A calculation of \mathbf{H} is performed every 20 AFED steps, and \mathbf{n} is reset as the lowest eigenvector.
- \mathbf{n} is not updated at any AFED step while the value of $\omega^2 = \mathbf{n}^\dagger \cdot \mathbf{H} \mathbf{n}$ is positive (which means that the curvature is downward along the direction \mathbf{n}).

The AFED step size is controlled as follows.

- The AFED step size is adjusted after the fifth AFED step.
- The AFED step is accepted as long as the angle between the last two moves does not exceed 90 degrees.
- If the last AFED step is rejected, the AFED step size is reduced by 70 percent.
- However, if the last five moves are accepted, the AFED step size is increased by 30 percent.
- The run is terminated once the AFED step size falls below a threshold value.

Simulation setup. Here is a recipe to start gentlest ascent dynamics simulations.

1. Decide the dimension of the collective variables, D . Then, for each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. So far, eight types are implemented, i.e.,
 - (1) bond distance of atoms A–B,
 - (2) bond angle associated with atoms A–B–C,
 - (3) dihedral angle associated with atoms A–B–C–D,
 - (4) difference between A–B and B–C distances,
 - (5) coordination number between A and B species,
 - (6) difference in coordination numbers between A–B and C–D species,
 - (7) center of mass of A and B species,
 - (8) difference of centers of mass of A and B species.

Set the collective variables using the keyword `<ncons>`.

2. Set the force constant of the harmonic potential, \mathcal{K}_d , using the keyword `<params_cons>`.
3. Set the parameter of gentlest ascent dynamics, γ^{gad} , using the keyword `<gamma_ascent_afed>`.

4. Set the initial AFED step sizes for descent and ascent trajectories, the damping factor, and the converged AFED step size using the keywords `<dt_descent_afed>`, `<dt_ascent_afed>`, `<dt_damp_afed>`, and `<dt_conv_afed>`, respectively.
5. Set the reference shift, $\Delta\mathcal{R}_d^{\text{ref}}$, as well as the lower bound $\mathcal{R}_d^{\text{min}}$ and the upper bound $\mathcal{R}_d^{\text{max}}$, for each collective variable using the keyword `<params_afed>`.
6. Set the keyword `<hessian_ascent_afed>`, either “NUMERICAL” or “ANALYTICAL”. For the numerical case, set the finite difference parameter, ϵ^{gad} , using the keyword `<fdiff_sampling_afed>`.
7. Set the upper bound of the free energy to be explored, $A_{\text{cv}}^{\text{max}}$, using the keyword `<fenergy_max_afed>`.
8. Set the number of preliminary and productive MD steps using the keywords `<nstep_pre_afed>` and `<nstep_pro_afed>`, respectively.

11.16.3 Automated search

Implementation. The automated search calculation proceeds as follows.

1. The calculation starts with an initial descent search (DE) to locate a free energy minimum point (EQ). The data is written to the list file *auto.ini*.
2. An ascent search (AS) is launched from this EQ in a random direction.
3. If a free energy saddle point (TS) is found, check the list file to determine whether this is a new one. If it is an old one (previously found TS), return to step 2.
4. Once a new TS is found, the data is added to the list file. Then start two descent searches in the forward and backward directions from the TS. These trajectories, referred to as D1 and D2, should lead to two minima. The minima are checked, and if they are new, the data is added to the list file.
5. The ascent search fails if it either finds an old TS, enters regions with free energy that is too high, or goes out of bounds.
6. If the ascent search is performed N_{shot} times from the same EQ, or if it does not find a new TS for N_{miss} consecutive attempts from the same EQ, the ascent search from that EQ is abandoned. In this case, another EQ in the list file is selected, and the process returns to step 2.
7. The calculation stops when all EQs have undergone the ascent search.

The output files are:

- *auto.ini*: The list of EQ and TS points that have been explored.
- *afed.ini*: The AFED restart file.
- *EQ.*.tar* and *TS.*.tar*: The tape archive containing the geometry file *geometry.ini* and the restart file *afed.ini* of EQ and TS, respectively, where * is the ID number.

A point in the mass-weighted CV space, \mathcal{Q}_d , is identified as a new point if it is distant from all other points in the list, $\mathcal{Q}_d^{\text{list}}$, i.e.,

$$\sqrt{\sum_{d=1}^D (\mathcal{Q}_d - \mathcal{Q}_d^{\text{list}})^2} > \mathcal{Q}^{\text{rad}}. \quad (266)$$

Otherwise, \mathcal{Q}_d is identified as an old point.

Simulation setup. Here is a recipe to start gentlest ascent dynamics simulations.

1. Decide the dimension of the collective variables, D . Then, for each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. So far, five types are implemented, i.e.,
 - (1) bond distance of atoms A–B,
 - (2) bond angle associated with atoms A–B–C,
 - (3) dihedral angle associated with atoms A–B–C–D,
 - (4) difference between A–B and B–C distances,
 - (5) coordination number between A and B species.

Set the collective variables using the keyword `<ncons>`.

2. Set the force constant of the harmonic potential, \mathcal{K}_d , using the keyword `<params_cons>`.
3. Set the parameter of gentlest ascent dynamics, γ^{gad} , using the keyword `<gamma_ascent_afed>`.
4. Set the initial AFED step size, the damping factor, and the converged AFED step size using the keywords `<dt_ascent_afed>`, `<dt_damp_afed>`, and `<dt_conv_afed>`, respectively.
5. Set the reference shift, $\Delta \mathcal{R}_d^{\text{ref}}$, as well as the lower bound $\mathcal{R}_d^{\text{min}}$ and the upper bound $\mathcal{R}_d^{\text{max}}$, for each type of collective variable using the keyword `<params_afed>`.
6. Set the keyword `<hessian_ascent_afed>`, either “NUMERICAL” or “ANALYTICAL”. For the numerical case, set the finite difference parameter, ϵ^{gad} , using the keyword `<fdiff_sampling_afed>`.
7. Set the upper bound of the free energy to be explored, $A_{\text{cv}}^{\text{max}}$, using the keyword `<fenergy_max_afed>`.
8. Set the number of preliminary and productive MD steps using the keywords `<nstep_pre_afed>` and `<nstep_pro_afed>`, respectively.
9. Set the reference radius, \mathcal{Q}^{rad} , using the keyword `<radius_auto_afed>`.
10. Set the maximum number of shots from a minimum, N_{shot} , using the keyword `<nshot_auto_afed>`.
11. Set the number of consecutive missed shots from a minimum, N_{miss} , using the keyword `<nmiss_auto_afed>`.

11.16.4 Temperature accelerated molecular dynamics

Theory. In temperature accelerated molecular dynamics (TAMD) with the Nosé-Hoover thermostat [77], the pseudo Hamiltonian is defined as

$$H^{\text{tamd}} = T_{\text{cv}} + V_{\text{cv}}^{\text{tamd}} + H_{\text{cv}}^{\text{bath}} \quad (267)$$

where

$$T_{\text{cv}} = \sum_{d=1}^D \frac{\mathcal{P}_d^2}{2\mathcal{M}_d}, \quad (268)$$

is the kinetic energy of the fictitious particles, and $H_{\text{cv}}^{\text{bath}}$ is the energy of the thermostat attached to the fictitious particles. The effective potential is defined as

$$V_{\text{cv}}^{\text{tamd}} = \left(\frac{T}{T_{\text{tamd}}} \right) A_{\text{cv}} \quad (269)$$

where the temperature T_{tamd} is an input parameter.

The mass of the collective variables is determined by

$$\mathcal{M}_d = \frac{k_B T \Delta t_{\text{cv}}^2}{\Delta \mathcal{R}_d^{\text{ref}^2}} \quad (270)$$

where the parameter $\Delta \mathcal{R}_d^{\text{ref}}$ is the reference displacement of the fictitious particle given by `<params_afed>` (see Sec.3.13.1).

When the AFED time step Δt_{cv} is set to be dimensionless, \mathcal{M}_d has the dimension of (energy)/(collective variable) (default is “ $\Delta t_{\text{cv}} = 1.0$ ”).

Implementation. For NVE-type TAMd, the thermostats are not applied, and thus $H_{\text{cv}}^{\text{bath}} = 0$. For NVT- and VS-type TAMd, the temperature is maintained at the physical temperature, T , by the Nosé-Hoover thermostat or the velocity scaling (Gaussian thermostat), respectively, which are attached to the whole set of fictitious variables (global thermostatting). The energy of the Nosé-Hoover thermostat is given by

$$H_{\text{cv}}^{\text{bath}} = \frac{\mathcal{P}_1^2}{2\mathcal{Q}_1} + Dk_B T \mathcal{R}_1. \quad (271)$$

where the thermostat mass is chosen to be $\mathcal{Q}_1 = Dk_B T \tau_{\text{cv}}^2$, and τ_{cv} is the typical time scale of the fictitious variables.

The parameters $\mathcal{R}_d^{\text{min}}$ and $\mathcal{R}_d^{\text{max}}$ are the lower and upper bounds, respectively, in the collective variable space to be explored. If the component d of the collective variables goes out of bounds, $\mathcal{R}_d > \mathcal{R}_d^{\text{max}}$ or $\mathcal{R}_d < \mathcal{R}_d^{\text{min}}$, then the velocity of the component d is reflected, i.e., $\mathcal{V}_d \rightarrow -\mathcal{V}_d$.

The parameter $A_{\text{cv}}^{\text{max}}$ sets the upper limit of the free energy. If A_{cv} exceeds $A_{\text{cv}}^{\text{max}}$, then the velocity is reversed, i.e., $\mathcal{V}_d \rightarrow -\mathcal{V}_d$ for all d .

Simulation setup. Here is a recipe to start TAMd simulations.

1. Decide the dimension of the collective variables, D . Then, for each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. So far, five types are implemented, i.e.,

- (1) bond distance of atoms A–B,
- (2) bond angle associated with atoms A–B–C,
- (3) dihedral angle associated with atoms A–B–C–D,
- (4) difference between A–B and B–C distances,
- (5) coordination number between A and B species.

Set the collective variables using the keyword `<ncons>`.

2. Set the TAMd type, `<tamd_type> = NVE, NVT, VS`.

3. Set the TAMd parameter T_{tamd} , using the keyword `<temperature_tamd>`.

4. Set the reference displacement, $\Delta \mathcal{R}_d^{\text{ref}}$, as well as the lower bound $\mathcal{R}_d^{\text{min}}$ and the upper bound $\mathcal{R}_d^{\text{max}}$, for each type of collective variable using the keyword `<params_afed>`. Note that $\Delta \mathcal{R}_d^{\text{ref}}$ is used to determine \mathcal{M}_d using Eq.(270). Additionally, one can use `<mass_afed>` to further tune \mathcal{M}_d .

5. Set the force constant of the harmonic potential, \mathcal{K}_d , using the keyword `<params_cons>`.

6. Optionally, the maximum free energy value ($A_{\text{cv}}^{\text{max}}$) to be sampled and the AFED step size Δt_{cv} can be manually set using `<fenergy_max_afed>` and `<dt_tamd>`, respectively. Also, the mass of the thermostat should be set using `<niter_bath_afed>` for `<tamd_type> = NVT`.

11.16.5 Logarithmic mean force dynamics

Theory. In logarithmic mean force dynamics (LogMFD) with the Nosé-Hoover thermostat, the pseudo Hamiltonian is defined as

$$H^{\log} = T_{\text{cv}} + V_{\text{cv}}^{\log} + H_{\text{cv}}^{\text{bath}} \quad (272)$$

where

$$T_{\text{cv}} = \sum_{d=1}^D \frac{\mathcal{P}_d^2}{2\mathcal{M}_d}, \quad (273)$$

is the kinetic energy of the fictitious particles, and $H_{\text{cv}}^{\text{bath}}$ is the energy of the thermostat attached to the fictitious particles. The effective potential is defined as

$$V_{\text{cv}}^{\log} = \text{sgn}(A_{\text{cv}}) \gamma_{\log} \log \{ \alpha_{\log} |A_{\text{cv}}| + 1 \} \quad (274)$$

where α_{\log} and γ_{\log} are positive input parameters. In the case that the largest free-energy barrier in the CV space, ΔF , can be roughly estimated, α_{\log} is given as $\alpha_{\log} \sim 1.5 \log \left(\frac{\Delta F}{k_B T} \right) / k_B T$ (see ref. [75]). Note that α_{\log} can take any positive value as long as the fictitious variables sufficiently explore the CV space. Also, γ_{\log} is normally set as $\gamma_{\log} = 1/\alpha_{\log}$ (default setting).

The free energy is calculated on-the-fly as

$$A_{\text{cv}} = \frac{\text{sgn}(H - T_{\text{cv}} - H_{\text{cv}}^{\text{bath}})}{\alpha_{\log}} \left\{ \exp \left(\frac{|H^{\log} - T_{\text{cv}} - H_{\text{cv}}^{\text{bath}}|}{\gamma_{\log}} \right) - 1 \right\} \quad (275)$$

from the conservation of H^{\log} . The origin of the free energy is set by specifying the value of $A_{\text{cv}}(0)$ (default is “0.003”). When restarted, $A_{\text{cv}}(0)$ is automatically set by *afed.ini*, which contains the final value of A_{cv} in the previous LogMFD run. In the same way as Eq. (275), A_{cv} is calculated at each AFED step in NVE and VS LogMFD calculations (for the latter, more details will be discussed in a forthcoming paper by Morishita *et al.*).

The mass of the collective variables is determined by

$$\mathcal{M}_d = \frac{k_B T \Delta t_{\text{cv}}^2}{\Delta \mathcal{R}_d^{\text{ref}2}} \quad (276)$$

where the parameter $\Delta \mathcal{R}_d^{\text{ref}}$ is the reference displacement of the fictitious particle given by `<params_afed>` (see Sec.3.13.1). When the AFED time step Δt_{cv} is set to be dimensionless, \mathcal{M}_d has the dimension of (energy)/(collective variable) (default is “ $\Delta t_{\text{cv}} = 1.0$ ”).

Implementation. For NVE-type LogMFD, the thermostat is not applied, and thus $H_{\text{cv}}^{\text{bath}} = 0$. For NVT- and VS-type LogMFD, the temperature is maintained at the physical temperature, T , by the Nosé-Hoover thermostat or the velocity scaling (Gaussian thermostat), respectively, which are attached to the whole set of fictitious variables (global thermostatting). The energy of the Nosé-Hoover thermostat is given by

$$H_{\text{cv}}^{\text{bath}} = \frac{\mathcal{P}_1^2}{2\mathcal{Q}_1} + Dk_B T \mathcal{R}_1. \quad (277)$$

where the thermostat mass is chosen to be $\mathcal{Q}_1 = Dk_B T \tau_{\text{cv}}^2$, and τ_{cv} is the typical time scale of the fictitious variables.

The parameters $\mathcal{R}_d^{\text{min}}$ and $\mathcal{R}_d^{\text{max}}$ are the lower and upper bounds, respectively, in the collective variable space to be explored. If the component d of the collective variables goes out of bounds, $\mathcal{R}_d > \mathcal{R}_d^{\text{max}}$ or $\mathcal{R}_d < \mathcal{R}_d^{\text{min}}$, then the velocity of the component d is reflected, i.e., $\mathcal{V}_d \rightarrow -\mathcal{V}_d$.

The parameter $A_{\text{cv}}^{\text{max}}$ sets the upper limit of the free energy. If A_{cv} exceeds $A_{\text{cv}}^{\text{max}}$, then the velocity is reversed, i.e., $\mathcal{V}_d \rightarrow -\mathcal{V}_d$ for all d .

Simulation setup. Here is a recipe to start LogMFD simulations.

1. Decide the dimension of the collective variables, D . Then, for each collective variable $1 \leq d \leq D$, choose the type that properly describes the chemical reaction. So far, five types are implemented, i.e.,
 - (1) bond distance of atoms A-B,
 - (2) bond angle associated with atoms A-B-C,
 - (3) dihedral angle associated with atoms A-B-C-D,
 - (4) difference between A-B and B-C distances,
 - (5) coordination number between A and B species.

Set the collective variables using the keyword `<ncons>`.

2. Set the LogMFD type, `<logmfd_type>` = NVE, NVT, VS.
3. Set the LogMFD parameter α' ($\alpha_{\log} = \alpha'/k_B T$) using the keyword `<alpha_logmfd>` (γ_{\log} can be independently set using `<gamma_logmfd>`, but normally the default setting, $\gamma_{\log} = 1/\alpha_{\log}$, is sufficient).
4. Set the reference displacement, $\Delta \mathcal{R}_d^{\text{ref}}$, as well as the lower bound $\mathcal{R}_d^{\text{min}}$ and the upper bound $\mathcal{R}_d^{\text{max}}$, for each type of collective variable using the keyword `<params_afed>`. Note that $\Delta \mathcal{R}_d^{\text{ref}}$ is used to determine \mathcal{M}_d using Eq.(276). Additionally, one can use `<mass_afed>` to further tune \mathcal{M}_d .
5. Set the force constant of the harmonic potential, \mathcal{K}_d , using the keyword `<params_cons>`.
6. Optionally, the origin of the free energy [$A_{\text{cv}}(0)$], the maximum free energy value ($A_{\text{cv}}^{\text{max}}$) to be sampled, and the AFED step size Δt_{cv} can be manually set using `<fenergy_ini_logmfd>`, `<fenergy_max_afed>`, and `<dt_logmfd>`, respectively. Also, the mass of the thermostat should be set using `<niter_bath_afed>` for `<logmfd_type>` = NVT.

11.17 Nonadiabatic dynamics

In nonadiabatic dynamics, the time-dependent coefficients of the adiabatic electronic states, $C_k(t)$, are introduced.

$$|\Psi(t)\rangle = \sum_k^{\text{state}} C_k(t) |\Phi_k(\mathbf{r}(t))\rangle \quad (278)$$

where $\Psi(t)$ is the total electronic wave function and $\Phi_k(\mathbf{r}(t))$ is the adiabatic wave function at time t , which depends on t implicitly through the instantaneous nuclear configuration $\mathbf{r}(t)$. The adiabatic states are defined as the eigenvectors of the Born-Oppenheimer Hamiltonian:

$$\hat{H}_{\text{BO}} |\Phi_k(\mathbf{r}(t))\rangle = V_k(\mathbf{r}(t)) |\Phi_k(\mathbf{r}(t))\rangle, \quad (279)$$

where the eigenvalue

$$V_k(\mathbf{r}(t)) = \langle \Phi_k(\mathbf{r}(t)) | \hat{H}_{\text{BO}}(\mathbf{r}(t)) | \Phi_k(\mathbf{r}(t)) \rangle \quad (280)$$

corresponds to the adiabatic potential energy of the k -th state. The density matrix is defined as

$$\rho_{kl}(t) = C_k^*(t) C_l(t), \quad (281)$$

and its diagonal element, $\rho_{kk}(t)$, represents the population of the k -th state.

The electronic equations of motion are given by

$$i\hbar \dot{C}_l(t) = C_l(t) V_l(\mathbf{r}(t)) - i\hbar \sum_{k \neq l}^{\text{state}} C_k(t) \sum_{j=1}^N \sum_{\alpha=x,y,z} \dot{r}_{j\alpha}(t) \cdot d_{j\alpha,kl}(\mathbf{r}(t)), \quad (282)$$

where

$$d_{j\alpha,kl}(\mathbf{r}(t)) = \langle \Phi_k(\mathbf{r}(t)) | \frac{\partial}{\partial r_{j\alpha}} | \Phi_l(\mathbf{r}(t)) \rangle \quad (283)$$

is the nonadiabatic coupling vector.

Mean field dynamics In Ehrenfest mean-field dynamics, the nuclei evolve under the average force weighted by the state populations. The nuclear equations of motion are expressed as

$$\dot{p}_{j\alpha}(t) = - \sum_k^{\text{state}} |C_k(t)|^2 \frac{\partial V_k(\mathbf{r}(t))}{\partial r_{j\alpha}} - \sum_k^{\text{state}} \sum_l^{\text{state}} C_l^*(t) C_k(t) [V_l(\mathbf{r}(t)) - V_k(\mathbf{r}(t))] \cdot d_{lk,j\alpha}(\mathbf{r}(t)). \quad (284)$$

Surface hopping dynamics In Tully's surface hopping dynamics, the nuclei propagate on the potential energy surface of the currently occupied state, $k(t)$. The nuclear equations of motion are expressed as

$$\dot{p}_{j\alpha}(t) = - \frac{\partial V_{k(t)}(\mathbf{r}(t))}{\partial r_{j\alpha}}. \quad (285)$$

Here, $k(t)$ is a stochastic variable that evolves in time. According to the fewest switches algorithm, the probability for a transition (hop) from state l to state k during a time step Δt is given by

$$P^{l \rightarrow k}(t) = \max \left[\frac{f_{kl}(t)}{\rho_{ll}(t)}, 0 \right] \Delta t, \quad (286)$$

where

$$f_{kl}(t) = -2 \operatorname{Re} \left[\rho_{kl}(t) \left(\sum_{j=1}^N \sum_{\alpha=x,y,z} \dot{r}_{j\alpha}(t) \cdot d_{j\alpha,kl}(\mathbf{r}(t)) \right) \right]. \quad (287)$$

11.18 Hybrid potentials: ONIOM and QM/MM

In the ONIOM method, the total potential is composed of three parts

$$V^{\text{ONIOM}}(\mathbf{r}_A, \mathbf{r}_B) = V_{\text{AL}}^{\text{hi}}(\mathbf{r}_A, \mathbf{r}_L) - V_{\text{AL}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_L) + V_{\text{AB}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_B). \quad (288)$$

The first two terms describe the energy difference of the subsystem consisting of layer A and the link atoms L, calculated at high- and low-level methods. The last term is the energy of the system composed of layers A and B (the whole system), calculated at a low-level method. As shown below, the positions of the link atoms are determined once the positions of the bridging atoms are fixed. The ONIOM energy has the correct limits when the high- and low-level methods are identical,

$$\lim_{\text{hi} \rightarrow \text{lo}} V^{\text{ONIOM}}(\mathbf{r}_A, \mathbf{r}_B) = V_{\text{AB}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_B) \quad (289)$$

and

$$\lim_{\text{lo} \rightarrow \text{hi}} V^{\text{ONIOM}}(\mathbf{r}_A, \mathbf{r}_B) = V_{\text{AB}}^{\text{hi}}(\mathbf{r}_A, \mathbf{r}_B). \quad (290)$$

In the absence of link atoms, it can be shown that the interaction between layer A and layer B is given by the low-level energy. Formally, the A-B interaction can be expressed as

$$V_{\text{int}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_B) = V_{\text{AB}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_B) - V_{\text{A}}^{\text{lo}}(\mathbf{r}_A) - V_{\text{B}}^{\text{lo}}(\mathbf{r}_B), \quad (291)$$

and thus, the total potential energy can be rewritten as

$$V^{\text{ONIOM}}(\mathbf{r}_A, \mathbf{r}_B) = V_A^{\text{hi}}(\mathbf{r}_A) + V_B^{\text{lo}}(\mathbf{r}_B) + V_{\text{int}}^{\text{lo}}(\mathbf{r}_A, \mathbf{r}_B). \quad (292)$$

This clearly indicates that the ONIOM energy is composed of the high-level energy of layer A, the low-level energy of layer B, and the low-level energy of the interaction between A and B.

The mechanical embedding version of the QM/MM energy is a special case of the ONIOM energy, where the high-level method corresponds to QM and the low-level calculation corresponds to MM:

$$V^{\text{QM/MM}}(\mathbf{r}_A, \mathbf{r}_B) = V_{\text{AL}}^{\text{QM}}(\mathbf{r}_A, \mathbf{r}_L) - V_{\text{AL}}^{\text{MM}}(\mathbf{r}_A, \mathbf{r}_L) + V_{\text{AB}}^{\text{MM}}(\mathbf{r}_A, \mathbf{r}_B). \quad (293)$$

Let the link atom, l , be placed along the bond i - j , between atom i in layer A and atom j in layer B. Let the ratio of i - l and i - j interatomic distances be a constant,

$$\alpha = \frac{|\mathbf{r}_i - \mathbf{r}_l|}{|\mathbf{r}_i - \mathbf{r}_j|}. \quad (294)$$

Then, the position of the link atom is determined as

$$\mathbf{r}_l = \mathbf{r}_i + \alpha(\mathbf{r}_j - \mathbf{r}_i) \quad (295)$$

once the positions of atoms i and j , \mathbf{r}_i and \mathbf{r}_j , are fixed. In this case, the forces acting on atoms i and j are

$$\mathbf{F}_i = -\frac{\partial V^{\text{ONIOM}}}{\partial \mathbf{r}_i} - (1 - \alpha) \frac{\partial (V_{\text{AL}}^{\text{hi}} - V_{\text{AL}}^{\text{lo}})}{\partial \mathbf{r}_l} \quad (296)$$

and

$$\mathbf{F}_j = -\frac{\partial V^{\text{ONIOM}}}{\partial \mathbf{r}_j} - \alpha \frac{\partial (V_{\text{AL}}^{\text{hi}} - V_{\text{AL}}^{\text{lo}})}{\partial \mathbf{r}_l}, \quad (297)$$

respectively. Note that the sum of forces

$$-\sum_{i=1}^N \left(\frac{\partial V^{\text{ONIOM}}}{\partial \mathbf{r}_i} \right) = \mathbf{0} \quad (298)$$

and the total torque

$$-\sum_{i=1}^N \left(\mathbf{r}_i \times \frac{\partial V^{\text{ONIOM}}}{\partial \mathbf{r}_i} \right) = \mathbf{0} \quad (299)$$

are preserved.

11.19 Multiple time scale method

The multiple time scale (MTS) method is implemented in the PIMD code. This technique allows for efficient sampling in molecular dynamics (MD) and path integral molecular dynamics (PIMD) when a hybrid potential, such as QM/MM or ONIOM, is employed.

In general, the MTS method is an algorithm to integrate the equations of motion using different time steps for forces that vary on different time scales. For two-layer systems, the MTS method is applicable only under the following conditions:

- Only properties at thermal equilibrium are of interest. Nonequilibrium properties, such as time-correlation functions, are not considered.

- Layer A is much smaller than layer B, so that the phase space of layer A is much smaller than that of layer B.
- Force calculations for layer A are significantly more expensive than those for layer B.

Under these conditions, the MTS algorithm is suitable.

- **Mass scaling:** Phase space sampling of layer B can be accelerated by scaling the masses of its atoms. In classical MD, all masses in layer B are scaled by a factor n_{ref}^{-2} . Although this mass scaling alters dynamical information, it does not affect thermodynamic properties, because the classical NVT ensemble distribution, $\propto \exp(-\beta V)$, remains unchanged. For PIMD, the fictitious masses of layer B can also be scaled without changing the quantum NVT ensemble distribution, $\propto \exp(-\beta V^{\text{eff}})$.
- **Force decomposition:** In the two-layer ONIOM method and mechanical embedding QM/MM, the total force can be decomposed as

$$-\nabla V_{\text{ONIOM}}(\mathbf{r}_A, \mathbf{r}_B) = -\nabla V_{\text{high}}(\mathbf{r}_A, \mathbf{r}_B) + \nabla V_{\text{low}}(\mathbf{r}_A, \mathbf{r}_B) - \nabla V_{\text{low}}(\mathbf{r}_A, \mathbf{r}_B). \quad (300)$$

To utilize the MTS algorithm, the first term on the right-hand side is updated with the larger time step, Δt , while the second and third terms are updated more frequently, using a smaller time step, $\Delta t/n_{\text{ref}}$. Rapidly varying forces in layer B require more frequent updates than those in layer A.

The basic MD cycle is then constructed as follows:

```
[1] force of system A at high level
[2] force of system A at low level
[3] force of system B at low level

do i = 1, nstep

  update velocity due to forces [2] by -dt/2
  update velocity due to forces [3] by +dt/2

  do j = 1, nref

    update velocity due to force [1] by dt/nref/2

    update position by dt/nref

    [1] force of system A at high level

    update velocity due to force [1] by dt/nref/2

  end do

  [2] force of system A at low level
  [3] force of system B at low level

  update velocity due to forces [2] by -dt/2
  update velocity due to forces [3] by +dt/2

end do
```

Here, updates of thermostats, harmonic forces in PIMD, and other details have been omitted for simplicity. The parameter `nref` can be increased until the computational costs of the inner and outer cycles are balanced. **Note:** The MTS method can be combined with the BEST method.

11.20 BEST method

The BEST (Boundary based on Exchange Symmetry Theory) method is a treatment of hybrid potentials for open boundary systems [52]. A bias potential is imposed to enforce a natural phase separation of the layers introduced in hybrid potentials. The form of the bias potential has been determined according to the exchange symmetry between the atoms in different layers. The bias potential is designed to have no influence on the thermodynamics of the system if the atoms were identical. (Of course, the atoms in different layers are not identical in any hybrid potentials, but at least the bias potential preserves the correct limit as if they were identical.) The details will be described elsewhere [52].

11.21 Ewald sum

Let us consider the electrostatic interaction of N particles with charges q_i at positions \mathbf{r}_i for $1 \leq i \leq N$, confined in a parallelepiped box under periodic boundary conditions. For a vector of three integers, $\mathbf{n} = (n_1, n_2, n_3)$, the lattice vectors are defined by

$$(\mathbf{R}_{\mathbf{n}})_{\alpha} = \sum_{\beta=1}^3 h_{\alpha\beta} n_{\beta} \quad (301)$$

where $\alpha = (x, y, z)$, $\beta = (1, 2, 3)$, and \mathbf{h} is the box matrix. Then, the electrostatic interaction is

$$V^{\text{es}} = \sum_{\mathbf{n}} \sum_{j=1}^N \sum_{k=1}^N {}' \frac{q_j q_k}{|\mathbf{r}_j - \mathbf{r}_k + \mathbf{R}_{\mathbf{n}}|}. \quad (302)$$

Here, \sum' indicates that the contribution of $k = j$ for the special case of $\mathbf{n} = \mathbf{0}$ has been excluded from the sum to avoid self-interactions. Meanwhile, for a vector of three integers, $\mathbf{k} = (k_1, k_2, k_3)$, the reciprocal lattice vector is defined by

$$(\mathbf{G}_{\mathbf{k}})_{\beta} = 2\pi \sum_{\alpha=1}^3 k_{\alpha} h_{\alpha\beta}^{-1}. \quad (303)$$

Then, the Ewald sum can be expressed as

$$V^{\text{es}} = V^{\text{short}} + V^{\text{long}} + V^{\text{self}} + V^{\text{charge}} (+V^{\text{dipole}}) \quad (304)$$

where the short-range, real-space contribution is

$$V^{\text{short}} = \frac{1}{2} \sum_{\mathbf{n}} \sum_{j=1}^N \sum_{k \neq j}^N {}' \frac{q_j q_k \text{erfc}(\eta |\mathbf{r}_j - \mathbf{r}_k + \mathbf{R}_{\mathbf{n}}|)}{|\mathbf{r}_j - \mathbf{r}_k + \mathbf{R}_{\mathbf{n}}|}, \quad (305)$$

the long-range, reciprocal-space contribution is

$$V^{\text{long}} = \frac{2\pi}{\mathbb{V}} \sum_{\mathbf{k} \neq \mathbf{0}} \frac{\exp\left(-\frac{|\mathbf{G}_{\mathbf{k}}|^2}{4\eta^2}\right)}{|\mathbf{G}_{\mathbf{k}}|^2} \left| \sum_{j=1}^N q_j e^{i(\mathbf{G}_{\mathbf{k}} \cdot \mathbf{r}_j)} \right|^2, \quad (306)$$

the correction of self-interaction in the reciprocal-space contribution is

$$V^{\text{self}} = -\frac{\eta}{\sqrt{\pi}} \sum_{i=1}^N q_i^2, \quad (307)$$

and the correction for a charged system is

$$V^{\text{charge}} = -\frac{\pi}{2\eta^2\mathbb{V}} \left(\sum_{i=1}^N q_i \right)^2, \quad (308)$$

The parameter η can be determined so that the calculation is both accurate and efficient by balancing the computational cost between the real-space and reciprocal-space contributions. The elimination of the $\mathbf{k} = \mathbf{0}$ contribution from the long-range term corresponds to the introduction of a uniform background charge that maintains the charge neutrality of the system. The dipole correction is

$$V^{\text{dipole}} = \frac{2\pi}{3\mathbb{V}} \left| \sum_{i=1}^N q_i \mathbf{r}_i \right|^2. \quad (309)$$

In the PIMD code, this term can be either included or neglected by option.

11.22 Polarizable force field

The polarizable force field is described by

$$V_{\text{pol}} = V_{\text{mm}} + V_{\text{ind}} + V_{\text{cd}} + V_{\text{dd}}, \quad (310)$$

where V_{mm} is the non-polarizable molecular mechanics force field given by Equation (10). With the dipole moment $\boldsymbol{\mu}_i$ and the polarizability α_i of each atom i , the polarization energy is given by

$$V_{\text{ind}} = \sum_{i=1}^N \frac{\boldsymbol{\mu}_i^2}{2\alpha_i}. \quad (311)$$

Under free boundary conditions, the charge-dipole interaction is described by

$$V_{\text{cd}} = \sum_{i=1}^N \sum_{j \neq i}^N q_i \mathbf{t}_{ij} \boldsymbol{\mu}_j = \sum_{i=1}^N \sum_{j > i}^N (q_i \mathbf{t}_{ij} \boldsymbol{\mu}_j - q_j \mathbf{t}_{ji} \boldsymbol{\mu}_i), \quad (312)$$

and the dipole-dipole interaction by

$$V_{\text{dd}} = -\frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N \boldsymbol{\mu}_i \mathbf{T}_{ij} \boldsymbol{\mu}_j = -\sum_{i=1}^N \sum_{j > i}^N \boldsymbol{\mu}_i \mathbf{T}_{ij} \boldsymbol{\mu}_j. \quad (313)$$

Here, we have defined a vector

$$(\mathbf{t}_{ij})_{\alpha} = \frac{s_3^{\text{cd}}(r_{ij})}{r_{ij}^3} (\mathbf{r}_{ij})_{\alpha}, \quad (314)$$

and a matrix

$$(\mathbf{T}_{ij})_{\alpha\beta} = -\frac{s_3^{\text{dd}}(r_{ij})}{r_{ij}^3} \delta_{\alpha\beta} + \frac{3s_5^{\text{dd}}(r_{ij})}{r_{ij}^5} (\mathbf{r}_{ij})_{\alpha} (\mathbf{r}_{ij})_{\beta}. \quad (315)$$

These quantities are functions of the interatomic separation $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and the interatomic distance $r_{ij} = |\mathbf{r}_{ij}|$. The damping functions $s_3^{\text{cd}}(r)$, $s_3^{\text{dd}}(r)$, and $s_5^{\text{dd}}(r)$ monotonically increase from 0 to 1 with increasing r . The dipole moments are determined by the variational condition

$$\frac{\partial V^{\text{pol}}}{\partial \boldsymbol{\mu}_i} = 0. \quad (316)$$

Thus,

$$\boldsymbol{\mu}_i = \alpha_i \sum_{j \neq i}^N (-\mathbf{t}_{ij} q_j + \mathbf{T}_{ij} \boldsymbol{\mu}_j). \quad (317)$$

Under periodic boundary conditions, the polarizable force field is described by

$$V_{\text{pol}} = V_{\text{mm}} + V_{\text{ind}} + V_{\text{cd}}^{\text{short}} + V_{\text{cd}}^{\text{long}} + V_{\text{dd}}^{\text{short}} + V_{\text{dd}}^{\text{long}} + V_{\text{dd}}^{\text{self}} + V_{\text{dd}}^{\text{net}}. \quad (318)$$

The real-space contributions of the charge–dipole and dipole–dipole interactions are given by

$$V_{\text{cd}}^{\text{short}} = \sum_n \sum_{i=1}^N \sum_{j=1}^N q_i \mathbf{t}_{ij}^{\text{short}} \boldsymbol{\mu}_j, \quad (319)$$

and

$$V_{\text{dd}}^{\text{short}} = \frac{1}{2} \sum_n \sum_{i=1}^N \sum_{j=1}^N \boldsymbol{\mu}_i \mathbf{T}_{ij}^{\text{short}} \boldsymbol{\mu}_j, \quad (320)$$

respectively, where we have defined a vector

$$(\mathbf{t}_{ij}^{\text{short}})_{\alpha} = \left(\frac{s_3^{\text{cd}}(r_{ij}) - \text{erf}(\eta r_{ij}) + \frac{2\eta r_{ij}}{\sqrt{\pi}} e^{-\eta^2 r_{ij}^2}}{r_{ij}^3} \right) (\mathbf{r}_{ij})_{\alpha}, \quad (321)$$

and a matrix

$$\begin{aligned} (\mathbf{T}_{ij}^{\text{short}})_{\alpha\beta} = & - \left(\frac{s_3^{\text{dd}}(r_{ij}) - \text{erf}(\eta r_{ij}) + \frac{2\eta r_{ij}}{\sqrt{\pi}} e^{-\eta^2 r_{ij}^2}}{r_{ij}^3} \right) \delta_{\alpha\beta} \\ & + \left(\frac{3(s_5^{\text{dd}}(r_{ij}) - \text{erf}(\eta r_{ij})) + (3 + 2\eta^2 r_{ij}^2) \frac{2\eta r_{ij}}{\sqrt{\pi}} e^{-\eta^2 r_{ij}^2}}{r_{ij}^5} \right) (\mathbf{r}_{ij})_{\alpha} (\mathbf{r}_{ij})_{\beta}. \end{aligned} \quad (322)$$

The reciprocal-space contributions of the charge–dipole and dipole–dipole interactions are given by

$$V_{\text{cd}}^{\text{long}} = \frac{4\pi}{\mathbb{V}} \sum_{\mathbf{k} \neq \mathbf{0}} \frac{\exp\left(-\frac{|\mathbf{G}_{\mathbf{k}}|^2}{4\eta^2}\right)}{|\mathbf{G}_{\mathbf{k}}|^2} \sum_{i=1}^N q_i \sum_{j=1}^N (\boldsymbol{\mu}_j \cdot \mathbf{G}_{\mathbf{k}}) \sin(\mathbf{G}_{\mathbf{k}} \cdot \mathbf{r}_{ij}), \quad (323)$$

and

$$V_{\text{dd}}^{\text{long}} = \frac{2\pi}{\mathbb{V}} \sum_{\mathbf{k} \neq \mathbf{0}} \frac{\exp\left(-\frac{|\mathbf{G}_{\mathbf{k}}|^2}{4\eta^2}\right)}{|\mathbf{G}_{\mathbf{k}}|^2} \left| \sum_{j=1}^N (\boldsymbol{\mu}_j \cdot \mathbf{G}_{\mathbf{k}}) e^{i\mathbf{G}_{\mathbf{k}} \cdot \mathbf{r}_j} \right|^2, \quad (324)$$

respectively. The dipole self-interaction correction in the reciprocal-space contribution is

$$V_{\text{dd}}^{\text{self}} = -\frac{2\eta^3}{3\sqrt{\pi}} \sum_{i=1}^N \boldsymbol{\mu}_i^2. \quad (325)$$

The correction with respect to the net dipole moment is

$$V_{\text{dd}}^{\text{net}} = \frac{2\pi}{3\mathbb{V}} \left| \sum_{i=1}^N \boldsymbol{\mu}_i \right|^2. \quad (326)$$

The dipole moments $\boldsymbol{\mu}_i$ are obtained from the variational condition

$$\frac{\partial V^{\text{pol}}}{\partial \boldsymbol{\mu}_i} = 0, \quad (327)$$

and thus

$$\boldsymbol{\mu}_i = \alpha_i \left(- \sum_{j \neq i}^N (\mathbf{t}_{ij}^{\text{short}} q_j - \mathbf{T}_{ij}^{\text{short}} \boldsymbol{\mu}_j) - \frac{\partial V_{\text{cd}}^{\text{long}}}{\partial \boldsymbol{\mu}_i} - \frac{\partial V_{\text{dd}}^{\text{long}}}{\partial \boldsymbol{\mu}_i} + \frac{4\eta^3}{3\sqrt{\pi}} \boldsymbol{\mu}_i - \frac{4\pi}{3\mathbb{V}} \sum_{j=1}^N \boldsymbol{\mu}_j \right), \quad (328)$$

which leads to a set of $3N$ -dimensional linear equations of the form $\mathbf{A}\boldsymbol{\mu} = \mathbf{b}$. This system can be solved using the LAPACK `dgesv` routine.

The damping functions can be chosen from the linear Thole (LIN), exponential Thole (EXP), Gaussian (GAU) [85], and Ojamae–Shavitt–Singer (OSS) types [86]. The OSS-type damping function is defined with two parameters, denoted by a_x^y and b_x^y , as

$$s_1^{\text{cc}} = \frac{r^2}{r^2 + a_1^{\text{cc}} e^{-b_1^{\text{cc}} r}}, \quad s_3^{\text{cd/dd}} = \frac{r^2}{r^2 + a_3^{\text{cd/dd}} e^{-b_3^{\text{cd/dd}} r}}, \quad s_3^{\text{dd}} = \frac{r^2}{r^2 + a_3^{\text{dd}} e^{-b_3^{\text{dd}} r}}. \quad (329)$$

Other types use a single parameter a , and the functions are described in terms of $z = r/a$. The linear Thole-type damping functions are given by

$$s_1^{\text{cc}} = \begin{cases} 2z - 2z^3 + z^4 & (0 \leq z \leq 1) \\ 1 & (z > 1) \end{cases} \quad s_3^{\text{cd/dd}} = \begin{cases} 4z^3 - 3z^4 & (0 \leq z \leq 1) \\ 1 & (z > 1) \end{cases} \quad s_5^{\text{dd}} = \begin{cases} z^4 & (0 \leq z \leq 1) \\ 1 & (z > 1) \end{cases} \quad (330)$$

The exponential Thole-type damping functions are given by

$$s_1^{\text{cc}} = 1 - e^{-z^3} + z\Gamma\left(\frac{2}{3}, z^3\right), \quad s_3^{\text{cd/dd}} = 1 - e^{-z^3}, \quad s_5^{\text{dd}} = 1 - (1 + z^3)e^{-z^3}, \quad (331)$$

using the incomplete gamma function

$$\Gamma(a, x) = \int_x^\infty e^{-t} t^{a-1} dt. \quad (332)$$

The Gaussian-type damping functions are given by

$$s_1^{\text{cc}} = \text{erf}(z), \quad s_3^{\text{cc/dd}} = \text{erf}(z) - \frac{2z}{\sqrt{\pi}} e^{-z^2}, \quad s_5^{\text{cc}} = \text{erf}(z) - \left(1 + \frac{2z^2}{3}\right) \frac{2z}{\sqrt{\pi}} e^{-z^2}. \quad (333)$$

11.23 Dihedral bond potential

As mentioned in Section 9.7.5, the dihedral angle potential takes the form

$$V = \frac{v}{2} \{1 + \mu \cos(\nu\chi)\} \quad (334)$$

where v is the rotational barrier height, $\mu = \pm 1$ is the phase, and χ is the dihedral angle with respect to atoms i , j , k , and l , which can be expressed as

$$\chi = \text{sgn}[(\mathbf{r}_{ijk} \times \mathbf{r}_{jkl}) \cdot \mathbf{r}_{jk}] \cos^{-1} \left(\frac{|\mathbf{r}_{ijk} \cdot \mathbf{r}_{jkl}|}{|\mathbf{r}_{ijk}| |\mathbf{r}_{jkl}|} \right) \quad (335)$$

with the vectors normal to the $i-j-k$ and $j-k-l$ planes given by

$$\mathbf{r}_{ijk} = \mathbf{r}_{ij} \times \mathbf{r}_{kj}, \quad \mathbf{r}_{jkl} = \mathbf{r}_{lj} \times \mathbf{r}_{kj}. \quad (336)$$

The gradients of V with respect to each atom can be expressed as

$$\frac{\partial V}{\partial \mathbf{r}_i} = \frac{v\mu\nu s_\nu}{2} \left[\frac{\mathbf{r}_{kj} \times \mathbf{r}_{jkl}}{|\mathbf{r}_{ijk}| |\mathbf{r}_{jkl}|} - \frac{\mathbf{r}_{kj} \times \mathbf{r}_{ijk}}{|\mathbf{r}_{ijk}|^2} \cos \chi \right], \quad (337)$$

$$\frac{\partial V}{\partial \mathbf{r}_j} = \frac{v\mu\nu s_\nu}{2} \left[\left(\frac{\mathbf{r}_{kl} \times \mathbf{r}_{jkl}}{|\mathbf{r}_{jkl}|^2} + \frac{\mathbf{r}_{ki} \times \mathbf{r}_{ijk}}{|\mathbf{r}_{ijk}|^2} \right) \cos \chi - \frac{\mathbf{r}_{kl} \times \mathbf{r}_{ijk} + \mathbf{r}_{ik} \times \mathbf{r}_{jkl}}{|\mathbf{r}_{ijk}| |\mathbf{r}_{jkl}|} \right], \quad (338)$$

$$\frac{\partial V}{\partial \mathbf{r}_k} = \frac{v\mu\nu s_\nu}{2} \left[\left(\frac{\mathbf{r}_{ij} \times \mathbf{r}_{ijk}}{|\mathbf{r}_{ijk}|^2} + \frac{\mathbf{r}_{lj} \times \mathbf{r}_{jkl}}{|\mathbf{r}_{jkl}|^2} \right) \cos \chi - \frac{\mathbf{r}_{ij} \times \mathbf{r}_{jkl} + \mathbf{r}_{lj} \times \mathbf{r}_{ijk}}{|\mathbf{r}_{ijk}| |\mathbf{r}_{jkl}|} \right], \quad (339)$$

and

$$\frac{\partial V}{\partial \mathbf{r}_l} = \frac{v\mu\nu s_\nu}{2} \left[\frac{\mathbf{r}_{kj} \times \mathbf{r}_{ijk}}{|\mathbf{r}_{ijk}| |\mathbf{r}_{jkl}|} - \frac{\mathbf{r}_{kj} \times \mathbf{r}_{jkl}}{|\mathbf{r}_{jkl}|^2} \cos \chi \right], \quad (340)$$

where the function

$$s_\nu = \frac{1}{\nu} \frac{d \cos(\nu\chi)}{d \cos \chi} = \frac{1}{\nu} \frac{d \cos(\nu\chi)}{d \chi} \left(\frac{d \cos \chi}{d \chi} \right)^{-1} = \frac{\sin(\nu\chi)}{\sin \chi} \quad (341)$$

can be calculated using the recursive relation

$$s_\nu = s_{\nu-2} - \sin((\nu-2)\chi) \sin \chi + \cos((\nu-2)\chi) \cos \chi + \cos((\nu-1)\chi) \quad (342)$$

with $s_1 = 1$ and $s_2 = 2 \cos \chi$.

12 Publications

The following is the list of publications in which the PIMD code or its primitive version has been used.

- K. Nishioka, S. Dekura, T. Fujino, M. Mizuno, R. Kumai, B. Thomsen, M. Shiga, Y. Hori, Y. Shigeta, H. Mori, *Angew. Chem. Int. Ed.*, 65, e20785 (2026). "Proton Tautomerism for Anhydrous Superprotonic Conduction in 1,2,3-Triazolium Dihydrogen Phosphate Crystal"
- Y. Liu, J. Lu, X. Yuan, X. Liu, F. Tian, X. Li, Y. Ma, P. Hou, M. Pang, *Physica B: Condens. Matter* 734, 418596 (2026). "Prediction of isotope influenced negative thermal expansion in 14C diamond via PIMD simulation" [i/li](#)
- M. Shiga, J. Elsner, J. Behler, B. Thomsen, *J. Chem. Phys.*, 163, 134119 (2025). "Computation of the Heat Capacity of Water from First Principles"
- H. Kimizuka, S. Ogata, B. Thomsen, M. Shiga, *J. Phys. Condens. Matter*, 37, 193001 (2025). "Topical Review: Ab initio path-integral simulations of hydrogen-isotope diffusion in face-centered cubic metals"
- H. Matsubuchi, D. Hayashi, D. Okamoto, A. Noguchi, S. Nakagawa, T. Takayanagi, T. Murakami, *ChemPhysChem* 26 e202500494 (2025). "Solvent Effects on the Selectivity of Ambimodal Dipolar/Diels–Alder Cycloadditions: A Study Using Explicit Solvation Models"

- T. Murakami, H. Ota, S. Nakagawa, K. Okada, M. Tachikawa, T. Takayanagi, J. Phys. Chem. A, 29 2308 (2025). "Attractive Force-Induced Isotope Effects through Ring-Polymer Molecular Dynamics Simulations for the Barrierless Reaction between HNCO and H_3^+ Isotopologues: H_3^+ , H_2D^+ , HD_2^+ , and D_3^+ "
- T. Murakami, H. Ueno, Y. Kikuma, T. Takayanagi, Molecules 30, 442 (2025). "Nuclear Quantum Effects in the Ionic Dissociation Dynamics of HCl on the Water Ice Cluster"
- B. Thomsen, Y. Nagai, K. Kobayashi, I. Hamada, M. Shiga, J. Chem. Phys., 161, 204109 (2024). "Self-Learning Path Integral Hybrid Monte Carlo with Mixed Ab Initio and Machine Learning Potentials for Modelling Nuclear Quantum Effects in Water"
- H. Li, Y. Kataoka, S. Tanaka, J. Haruyama, O. Sugino, J. Yoshinobu, J. Phys. Chem. C, 128, 15393 (2024). "Elucidation of CO Oxidation and CO_2 Desorption Dynamics on Pt(111) by van der Waals DFT Calculations: Hyperthermal Kinetic Energy, Sharp Desorption Angle, and Excited Vibrational States"
- Y. Kataoka, J. Haruyama, O. Sugino, M. Shiga, Phys. Rev. Res., 6, 043224 (2024). "Predictive evaluation of hydrogen diffusion coefficient on Pd(111) surface by path integral simulations using neural network potential"
- J. Tsuchiya, M. Shiga, S. Tsuneyuki, E. C. Thompson, Phys. Rev. Res., 6, 023302 (2024). "Nuclear quantum effect on the elasticity of ice VII under pressure: A path-integral molecular dynamics study"
- Y. Nagai, Y. Iwasaki, K. Kitahara, Y. Takagiwa, K. Kimura, M. Shiga, Phys. Rev. Lett., 132, 196301 (2024). "High-temperature atomic diffusion and specific heat in quasicrystals"
- Y. Nakata, T. Sasaki, B. Thomsen, M. Shiga, Chem. Phys. Lett., 845, 141285 (2024). "Theoretical Study of Cellobiose Conversion by Supported Metal Catalysts"
- M. Shiga, B. Thomsen, H. Kimizuka, Phys. Rev. B. 109, 054303 (2024). "Inelastic Neutron Scattering of Hydrogen in Palladium Studied by Semiclassical Dynamics"
- T. Murakami, Y. Kikuma, D. Hayashi, S. Ibuki, S. Nakagawa, H. Ueno, T. Takayanagi, J. Phys. Org. Chem., 37, e4611 (2024). "Molecular dynamics simulation study of post-transition state bifurcation: A case study on the ambimodal transition state of dipolar/Diels-Alder cycloaddition"
- T. Murakami, S. Takahashi, Y. Kikuma, T. Takayanagi, Molecules, 29, 2789 (2024). "Theoretical Study of the Thermal Rate Coefficients of the $\text{H}_3^+ + \text{C}_2\text{H}_4$ Reaction: Dynamics Study on a Full-Dimensional Potential Energy Surface"
- T. Murakami, T. Takayanagi, Phys. Chem. Chem. Phys., 26, 19195 (2024). "Computational study of the post-transition state dynamics for the $\text{OH} + \text{CH}_3\text{OH}$ reaction probed by photodetachment of the $\text{CH}_3\text{O}^-(\text{H}_2\text{O})$ anion"
- T. Murakami, D. Hayashi, Y. Kikuma, K. Yamaki, T. Takayanagi, J. Comput. Chem., 45, 2778 (2024). "Temperature effects on the branching dynamics in the model ambimodal $(6 + 4)/(4 + 2)$ intramolecular cycloaddition reaction"
- T. Murakami, N. Matsumoto, T. Takayanagi, Comput. Theor. Chem., 1240, 114835 (2024). "Computational study on the bifurcation mechanism in the $\text{H}_2\text{CO}^- + \text{CH}_3\text{Cl} \rightarrow \text{CH}_3\text{CH}_2\text{O} + \text{Cl}^- / \text{H}_2\text{CO} + \text{CH}_3 + \text{Cl}^-$ reaction: The importance of intramolecular vibrational redistributions"
- T. Murakami, S. Nakagawa, H. Ota, K. Okada, T. Takayanagi, ACS Earth Space Chem., 8(11), 2294 (2024). "Kinetics and Ring-Polymer Molecular Dynamics Studies of the $\text{H}_3^+ + \text{HNCO} \rightarrow \text{H}_2 + \text{HNCOH}^+ / \text{H}_2\text{NCO}^+$ Branching Reaction on a Developed Full-Dimensional Potential Energy Surface"

- Y. Hashimoto, T. Takayanagi, T. Murakami, ACS Earth Space Chem., 7, 623 (2023). "Theoretical Calculations of the Thermal Rate Coefficients for the Interstellar $\text{NH}_3^+ + \text{H}_2 \rightarrow \text{NH}_4^+ + \text{H}$ Reaction on a New Δ -Machine Learning Potential Energy Surface"
- T. Murakami, S. Ibuki, Y. Hashimoto, Y. Kikuma, T. Takayanagi, Phys. Chem. Chem. Phys., 25, 14016(2023). "Dynamics study of the post-transition-state-bifurcation process of the $(\text{HCOOH})\text{H}^+ \rightarrow \text{CO} + \text{H}_3\text{O}^+ / \text{HCO}^+ + \text{H}_2\text{O}$ dissociation: Application of machine-learning techniques"
- T. Murakami, S. Ibuki, T. Takayanagi, Comput. Theor. Chem., 1227, 114239 (2023). "Molecular dynamics simulations and machine-learning assisted study of the reaction path bifurcation: Application to the intramolecular Diels-Alder cycloaddition between cyclobutadiene and butadiene"
- T. Murakami, Y. Kikuma, S. Ibuki, N. Matsumoto, K. Ogino, Y. Hashimoto, T. Takayanagi, J. Phys. Org. Chem., 36, e4561 (2023). "Machine learning-assisted study of correlation between post-transition-state bifurcation and initial phase information at the ambimodal transition state"
- H. Wang, P. T. Salzbrenner, I. Errea, F. Peng, Z. Lu, H. Liu, L. Zhu, C. J. Pickard, Y. Yao, Nature Commun. 14, 1674 (2023). "Quantum structural fluxion in superconducting lanthanum polyhydride"
- H. Kwon, M. Shiga, H. Kimizuka, T. Oda, Acta Mater, 247, 118739 (2023). "Accurate Description of Hydrogen Diffusivity in bcc Metals Using Machine-Learning Moment Tensor Potentials and Path-Integral Methods"
- K. Ogino, Y. Hashimoto, T. Takayanagi, ChemPhysChem, in press (2023). Ring-polymer Molecular Dynamics Simulation for the Adsorption of H_2 on Ice Clusters $(\text{H}_2\text{O})_n$ ($n = 8, 10$, and 12)"
- T. Murakami, R. Iida, Y. Hashimoto, Y. Takahashi, S. Takahashi, T. Takayanagi, J. Phys. Chem. A 126, 9244 (2022). "Ring-Polymer Molecular Dynamics and Kinetics for the $\text{H}^- + \text{C}_2\text{H}_2 \rightarrow \text{H}_2 + \text{C}_2\text{H}^-$ Reaction Using the Full-Dimensional Potential Energy Surface"
- H. Suzuki, T. Otomo, K. Ogino, Y. Hashimoto, T. Takayanagi, ACS Earth Space Chem. 6, 1390 (2022). "Nuclear Quantum Effects in H_2 Adsorption Dynamics on a Small Water Cluster Studied with Ring-Polymer Molecular Dynamics Simulations"
- M. Shiga, J. Comput. Chem., 43, 1864 (2022). "Path Integral Brownian Chain Molecular Dynamics: A Simple Approximation of Quantum Vibrational Dynamics"
- H. Kimizuka, B. Thomsen, M. Shiga, J. Phys. Energy 4, 034334 (2022) (13 pages). "Artificial neural network-based path integral simulations of hydrogen isotope diffusion in palladium"
- B. Thomsen, M. Shiga, Phys. Chem. Chem. Phys., 24, 10851-10859 (2022). "Structure of Liquid and Aqueous Water Isotopologues at Ambient Temperature from ab initio Path Integral Simulations"
- D. Akazawa, T. Sasaki, M. Nagasaka, M. Shiga, J. Chem. Phys., 56, 044202 (2022) (7 pages). "X-ray Absorption Spectra of Aqueous Cellobiose: Experiment and Theory"
- Y. Watanabe, T. Nomoto, R. Arita, Phys. Rev. B, 105, 174111 (2022). "Quantum and temperature effects on the crystal structure of superhydride LaH_{10} : A path integral molecular dynamics study"
- K. Yeo, K. Park, S. Jeong, Curr. Appl. Phys., 39, 62-69 (2022). "Neural network approach to diffusion of B and N adatoms on the Pt (111) surface"
- K. Saito, Y. Sugiura, T. Miyazaki, Y. Takahashi, T. Takayanagi, Phys. Chem. Chem. Phys., 23, 6950-6958 (2021). "Quantum calculations of the photoelectron spectra of the OH^-NH_3 anion: implications for $\text{OH} + \text{NH}_3 \rightarrow \text{H}_2\text{O} + \text{NH}_2$ reaction dynamics"

- Y. Hashimoto, K. Saito, T. Takayanagi, H. Tachikawa, *Phys. Chem. Chem. Phys.*, 23, 16958 (2021). "Theoretical study of the dissociative photodetachment dynamics of the hydrated superoxide anion cluster"
- K. Saito, Y. Hashimoto, T. Takayanagi, *J. Phys. Chem. A*, 125, 10750-10756 (2021). "Ring-Polymer Molecular Dynamics Calculations of Thermal Rate Coefficients and Branching Ratios for the Interstellar $\text{H}_3^+ + \text{CO} \rightarrow \text{H}_2^+ + \text{HCO}^+/\text{HOC}^+$ Reaction and Its Deuterated Analogue."
- Y. Takahashi, Y. Hashimoto, K. Saito, T. Takayanagi, *Molecules*, 26, 7250 (2021). "On-the-Fly Ring-Polymer Molecular Dynamics Calculations of the Dissociative Photodetachment Process of the Oxalate Anion."
- B. Thomsen, M. Shiga, *J. Chem. Phys.*, 155, 194107 (2021) (11 pages). "Ab Initio Study of Nuclear Quantum Effects on Sub- and Supercritical Water"
- A. Pal, S. Pal, S. Verma, M. Shiga, N. N. Nair, *J. Comput. Chem.*, 42, 1996-2003 (2021). "Mean Force Based Temperature Accelerated Sliced Sampling: Efficient Reconstruction of High Dimensional Free Energy Landscapes"
- K. Kobayashi, Y. Nagai, M. Itakura, M. Shiga, *J. Chem. Phys.*, 155, 034106 (2021) (9 pages). "Self-learning hybrid Monte Carlo for isothermal-isobaric ensemble: Application to liquid silica"
- T. Kondo, T. Sasaki, M. Shiga, *J. Comput. Chem.*, 42, 1783-1791 (2021). "The Mechanism of Sorbitol Dehydration in Hot Acidic Solutions"
- H. Wang, Y. Yao, F. Peng, H. Liu, R. J. Hemley, *Phys. Rev. Lett.*, 126, 117002 (2021). "Quantum and Classical Proton Diffusion in Superconducting Clathrate Hydrides"
- B. Thomsen, M. Shiga, *J. Chem. Phys.*, 154, 084117 (2021). "Nuclear Quantum Effects on Autoionization of Water Isotopologues Studied by Ab Initio Path Integral Molecular Dynamics"
- T. Kondo, T. Sasaki, S. Ruiz Barragan, J. Ribas, M. Shiga, *J. Comput. Chem.*, 42, 156-165 (2021). "Refined metadynamics through canonical sampling using time - invariant bias potential: A study of polyalcohol dehydration in hot acidic solutions"
- L. Yan, Y. Yamamoto, M. Shiga, O. Sugino, *Phys. Rev. B*, 101, 165414 (2020). "Nuclear quantum effect for hydrogen adsorption on Pt(111)"
- Y. Nagai, M. Okumura, K. Kobayashi, M. Shiga, *Phys. Rev. B (Rapid Commun.)*, 102, 041124 (2020). "Self-learning hybrid Monte Carlo: A first principles approach"
- T. Ishiyama, A. Morita, *J. Phys. Chem. Lett.* 10, 5070-5075 (2019). "Nuclear Quantum Effect on the $\chi(2)$ Band Shape of Vibrational Sum Frequency Generation Spectra of Normal and Deuterated Water Surfaces"
- Y. Kawashima, K. Ishimura, M. Shiga, *J. Chem. Phys.*, 150, 124103 (2019). "Ab initio Quantum Mechanics/Molecular Mechanics Method with Periodic Boundaries Employing Ewald Summation Technique to Electron-Charge Interaction: Treatment of the Surface-Dipole Term"
- Y. L. Chang, T. Sasaki, J. Ribas Arino, M. Machida, M. Shiga, *J. Phys. Chem. B*, 23, 1662-1671 (2019). "Understanding Competition of Polyalcohol Dehydration Reactions in Hot Water"
- M. Shiga, M. E. Tuckerman, *J. Phys. Chem. Lett.*, 9, 6207-6214 (2018). "Finding Free Energy Landmarks of Chemical Reactions"
- Y. Watabe, T. Miyazaki, E. Ozama, T. Takayanagi, Y. Suzuki, *Comp. Theo. Chem.*, 1140, 56-62 (2018). "Theoretical interpretation of photoelectron spectrum of $(\text{AuCO}_2)^-$ anion"

- E. Ozama, S. Adachi, T. Takayanagi, M. Shiga, Chem. Eur. J., 24, 12716-12721 (2018). "Quantum simulation verifies the stability of an 18 coordinated actinium-helium complex"
- M. Machida, K. Kato, M. Shiga. J. Chem. Phys., 148, 102324 (2018). "Nuclear quantum effects of light and heavy water studied by all-electron first principles path integral simulations."
- Y. Seki, T. Takayanagi, M. Shiga, Phys. Chem. Chem. Phys., 19, 13798-13806, (2017). "Photoexcited Ag ejection from a low-temperature He cluster: A simulation study by nonadiabatic Ehrenfest ring-polymer molecular dynamics"
- S. Ruiz-Barragan, J. Ribas-Arino, M. Shiga, Phys. Chem. Chem. Phys., 18, 32438-32447 (2016). "The reaction mechanism of polyalcohol dehydration in hot pressurized water"
- Y. Minoshima, Y. Seki, T. Takayanagi, M. Shiga, Chem. Phys., 472, 1-8 (2016). "Effects of temperature and isotopic substitution on electron attachment dynamics of guanine-cytosine base pair: Ring-polymer and classical molecular dynamics simulations"
- S. Ruiz-Barragan, K. Ishimura, M. Shiga, Chem. Phys. Lett., 646, 130-135 (2016). "On the hierarchical parallelization of ab initio simulations"
- T. Honda, Y. Minoshima, Y. Yokoi, T. Takayanagi and M. Shiga, Chem. Phys. Lett., 625, 174-178 (2015). "Semiclassical dynamics of electron attachment to guanine-cytosine base pair"
- M. Shiga, M. Masia, J. Chem. Phys., 139, 144103 (2013) (14 pages). "Boundary based on Exchange Symmetry Theory for Multilevel Simulations II. Multiple Time Scale Approach"
- M. Shiga, M. Masia, J. Chem. Phys., 139, 044120 (2013) (8 pages); (E) J. Chem. Phys., 139, 119901 (2013). "Boundary based on Exchange Symmetry Theory for Multilevel Simulations I. Basic Theory"
- K. Suzuki, M. Tachikawa, M. Shiga, J. Chem. Phys., 138, 184307 (2013) (7 pages). "Temperature dependence on the structure of Zundel cation and its isotopomers"
- A. Koizumi, M. Tachikawa, M. Shiga, Chem. Phys., 419, 44-49 (2013). "Quantum fluctuation and vibrational dynamics of aqueous Cu^+ and Ag^+ clusters"
- T. Yoshikawa, T. Takayanagi, H. Kimizuka, M. Shiga, J. Phys. Chem. C, 116, 23113-23119 (2012). "Quantum-Thermal Crossover of Hydrogen and Tritium Diffusion in α -Iron"
- M. Shiga, H. Fujisaki, J. Chem. Phys., 136, 184103 (2012) (11 pages). "A quantum generalization of intrinsic reaction coordinate using path integral centroid coordinates"
- K. Suzuki, H. Ishibashi, K. Yagi, M. Shiga, M. Tachikawa, Progress in Theoretical Chemistry and Physics B 26: Quantum Systems in Chemistry and Physics: Progress in Methods and Applications edited by K. Nishikawa et al., Chapter 10, 207-216 (2012). "Ab initio path integral molecular dynamics simulations of F_2H^- and F_2H_3^+ "
- T. Yoshikawa, S. Sugawara, T. Takayanagi, M. Shiga, M. Tachikawa, Chem. Phys., 394, 46-51 (2012). "Quantum tautomerization in porphycene and its isotopomers: Path-integral molecular dynamics simulations"
- A. Koizumi, K. Suzuki, M. Shiga, M. Tachikawa, Int. J. Quant. Chem., 112, 136-139 (2012). "Ab initio path integral simulation of $\text{AgOH}(\text{H}_2\text{O})$ "
- S. Sugawara, T. Yoshikawa, T. Takayanagi, M. Shiga, M. Tachikawa, J. Phys. Chem. A, 115, 11486-11494 (2011). "Quantum Proton Transfer in Hydrated Sulfuric Acid Clusters: A Perspective from Semiempirical Path Integral Simulations"

- M. Daido, A. Koizumi, M. Shiga, M. Tachikawa, *Theoret. Chem. Acc.*, 130, 385-391 (2011). “Nuclear quantum effect on the structure of guanine cytosine pair”
- P. Dopieralski, P. Anjukandi, M. Rueckert, M. Shiga, J. Ribas-Arino, D. Marx, *J. Mater. Chem.*, 21, 8309-8316 (2011). “On the role of polymer chains in transducing external mechanical forces to benzocyclobutene mechanophores”
- A. Koizumi, K. Suzuki, M. Shiga, M. Tachikawa, *J. Chem. Phys.*, (communication), 134, 031101 (2011); (E) *J. Chem. Phys.* 134, 169901 (2011). “A concerted mechanism between proton transfer of Zundel anion and displacement of counter cation”
- M. Sugimoto, M. Shiga, M. Tachikawa, *Comp. Theor. Chem.*, 975, 31-37 (2011). “Nuclear quantum effect on the dissociation energies of cationic hydrogen clusters”
- J. Ribas-Arino, M. Shiga, D. Marx, *J. Am. Chem. Soc.*, 132, 10609-10614 (2010). “Mechanochemical Transduction of Externally Applied Forces to Mechanophores”
- T. Yoshikawa, S. Sugawara, T. Takayanagi, M. Shiga, M. Tachikawa, *Chem. Phys. Lett.*, 496, 14-19 (2010). “Theoretical study on the mechanism of double proton transfer in porphycene by path-integral molecular dynamics simulations”
- M. Shiga, K. Suzuki, M. Tachikawa, *J. Chem. Phys.*, 132, 114104 (2010) (7 pages). “The chemical shift of deprotonated water dimer: Ab initio path integral simulation”
- K. Suzuki, M. Tachikawa, M. Shiga, *J. Chem. Phys.*, 132, 144108 (2010) (7 pages). “Efficient ab initio path integral hybrid Monte Carlo based on the fourth-order Trotter expansion – Application to fluoride ion-water cluster”
- S. D. Ivanov, A. Witt, M. Shiga, D. Marx, *J. Chem. Phys.*, (communication), 132, 031101 (2010) (4 pages). “On Artificial Frequency Shifts in Infrared Spectra obtained from Centroid Molecular Dynamics: Quantum Liquid Water”
- A. Kakizaki, H. Motegi, T. Yoshikawa, T. Takayanagi, M. Shiga, M. Tachikawa, *J. Molec. Struct. (THEOCHEM)*, 901, 1-8 (2009). “Path-integral molecular dynamics simulations of small hydrated sulfuric acid clusters $\text{H}_2\text{SO}_4(\text{H}_2\text{O})_n$ ($n = 1-6$) on semiempirical PM6 potential surfaces”
- T. Takayanagi, K. Takahashi, A. Kakizaki, M. Shiga, M. Tachikawa, *Chem. Phys.*, 358, 196-202 (2009). “Path-integral molecular dynamics simulations of hydrated hydrogen chloride cluster $\text{HCl}(\text{H}_2\text{O})_4$ on a semiempirical potential energy surface”
- A. Kaczmarek, M. Shiga, D. Marx, *J. Phys. Chem. A*, 113, 1985-1994 (2009). “Quantum effects on vibrational and electronic spectra of hydrazine studied by on the fly ab initio ring polymer molecular dynamics”
- J. Ribas-Arino, M. Shiga, D. Marx, *Angew. Chem. Int. Ed.* (communication), 48, 4190-4193 (2009). “Understanding Covalent Mechanochemistry”
- A. Witt, S. D. Ivanov, M. Shiga, H. Forbert, D. Marx, *J. Chem. Phys.*, 130, 194510 (2009) (15 pages). “On the applicability of centroid and ring polymer path integral molecular dynamics for vibrational spectroscopy”
- A. Nakayama, T. Taketsugu, M. Shiga, *Chem. Lett.*, 38, 976-977 (2009). “Speed-up of ab initio hybrid Monte Carlo and ab initio path integral hybrid Monte Carlo simulations using an auxiliary potential energy surface”
- J. Ribas-Arino, M. Shiga, D. Marx, *Chem. Eur. J.* (communication), 15, 13331-13335 (2009). “Unravelling the mechanism of force-induced ring-opening of benzocyclobutenes”

- T. Takayanagi, T. Yoshikawa, H. Motegi, M. Shiga, Chem. Phys. Lett., 482, 195-200 (2009). "Path-integral molecular dynamics simulations for water anion clusters $(\text{H}_2\text{O})_5^-$ and $(\text{D}_2\text{O})_5^-$ "
- T. Yoshikawa, H. Motegi, A. Kakizaki, T. Takayanagi, M. Shiga, M. Tachikawa, Chem. Phys., 365, 60-68 (2009). "Path-integral molecular dynamics simulations of glycine- $(\text{H}_2\text{O})_n$ ($n = 1-7$) clusters on semiempirical PM6 potential energy surfaces"
- T. Takayanagi, T. Yoshikawa, A. Kakizaki, M. Shiga, M. Tachikawa, J. Molec. Struct. (THEOCHEM), 869, 29-36 (2008). "Molecular dynamics simulations of small glycine- $(\text{H}_2\text{O})_n$ ($n = 2-7$) clusters on semiempirical PM6 potential energy surfaces"
- H. Ishibashi, A. Hayashi, M. Shiga, M. Tachikawa, ChemPhysChem (communication), 9, 383-387 (2008). "Geometric isotope effect on N_2H_7^+ cation and N_2H_5^- anion by ab initio path integral molecular dynamics simulation"
- M. Shiga, A. Nakayama, Chem. Phys. Lett., 451, 175-181 (2008). "ab initio path integral ring polymer molecular dynamics: vibrational spectra of molecules"
- H. Motegi, A. Kakizaki, T. Takayanagi, Y. Taketsugu, T. Takatsugu, M. Shiga, Chem. Phys., 454, 1-6 (2008). "Path-integral molecular dynamics simulations of BeO embedded in helium clusters: Formation of the stable HeBeO complex"
- K. Suzuki, M. Shiga, M. Tachikawa, J. Chem. Phys., 129, 144310 (8 pages) (2008); (E) J. Chem. Phys., 131, 039903 (2009). "Temperature and isotope effects on water cluster ions with path integral molecular dynamics based on 4th order Trotter expansion"
- A. Hayashi, M. Shiga, M. Tachikawa, Mol. Simul., 33, 185-188 (2007). "H/D isotope effect on the dihydrogen bond by ab initio path integral molecular dynamics simulation"
- M. Shiga, M. Tachikawa, Mol. Simul., 33, 171-184 (2007). "Ab initio quantum mechanical/molecular mechanical molecular dynamics using multiple-time-scale approach and perturbation theory"
- T. Kobayashi, M. Shiga, A. Murakami, S. Nakamura, J. Am. Chem. Soc., 129, 6405-6424 (2007). "Ab initio Study of Ultrafast Photochemical Reaction Dynamics of Phenol Blue"
- A. Kakizaki, T. Takayanagi, M. Shiga, Chem. Phys. Lett., 449, 28-32 (2007). "Path integral molecular dynamics calculations of the H_6^+ and D_6^+ clusters on an ab initio potential energy surface"
- A. Hayashi, M. Shiga, M. Tachikawa, J. Chem. Phys., 125, 204310 (7 pages) (2006). "H/D isotope effect on the dihydrogen bond of $\text{NH}_4^+\text{BeH}_2$ by ab initio path integral molecular dynamics simulation"
- M. Shiga, W. Shinoda, J. Chem. Phys., 123, 134502 (8 pages) (2005). "Calculation of heat capacities of light and heavy water by path integral molecular dynamics"
- M. Tachikawa, M. Shiga, J. Am. Chem. Soc., (communication) 127, 11906-11909 (2005). "Geometrical Isotope Effect on Hydrogen Bond in Charged Water Clusters"
- M. Tachikawa, M. Shiga, J. Theor. Comput. Chem., 4, 175-181 (2005). "Ab initio path integral study on isotope effect of ammonia molecule"
- T. Takayanagi, M. Shiga, T. Taketsugu, J. Theor. Comput. Chem., 4, 197-207 (2005). "Development of a three-dimensional ab initio potential energy surface for the He-Cl_2 (X) system and its application to solvation structures in the He_nCl_2 cluster"
- W. Shinoda, M. Shiga, Phys. Rev. E, 71, 041204 (4 pages) (2005). "Quantum simulation of the heat capacity of water"

- M. Tachikawa, M. Shiga, Chem. Phys. Lett., 407 135-138 (2005). “Ab initio path integral simulation study on $^{16}\text{O}/^{18}\text{O}$ isotope effect in water and hydronium ion”
- A. Hayashi, M. Shiga, M. Tachikawa, Chem. Phys. Lett., 407 135-138 (2005). “Ab initio path integral simulation study on the dihydrogen bond of $\text{NH}_4^+\text{BH}_2^-$ ”
- M. Tachikawa, M. Shiga, J. Chem. Phys. 121, 5985-5991 (2004). “Theoretical study on isotope and temperature effect in hydronium ion using ab initio path integral simulation”
- T. Takayanagi, M. Shiga, Phys. Chem. Chem. Phys., 6, 3241-3247 (2004). “Theoretical study on photoabsorption dynamics of the K atom attached to helium clusters and the solvation structures of K^*He_n exciplexes”
- M. Shiga, M. Tachikawa, Chem. Phys. Lett., 374, 229-234 (2003). “Ab initio path integral study of hydronium ion”
- A. Wada, T. Takayanagi, M. Shiga, J. Chem. Phys., 119, 5478-5486 (2003). “Theoretical simulations on photoexcitation dynamics of the silver atoms embedded in helium clusters”
- T. Takayanagi, M. Shiga, Chem. Phys. Lett., 372, 90-96 (2003). “Photodissociation of Cl_2 in helium clusters: an application of hybrid method of quantum wavepacket dynamics and path integral centroid molecular dynamics”
- M. Shiga, T. Takayanagi, Chem. Phys. Lett., 378, 539-547 (2003). “Quantum path-integral molecular dynamics calculations of the dipole-bound state of the water dimer anion”
- M. Shiga, M. Yamaguchi, H. Kaburaki, Phys. Rev. B, 68, 245402 (8 pages) (2003). “Structure and energetics of clean and hydrogenated Ni surfaces and symmetric tilt grain boundaries using embedded-atom method”
- T. Takayanagi, M. Shiga, Chem. Phys. Lett., 362, 504-510 (2002). “Path integral molecular dynamics combined with discrete-variable-representation approach: the effect of solvation structures on vibrational spectra of Cl_2 in helium clusters”
- M. Shiga, M. Tachikawa, S. Miura, J. Chem. Phys., 115, 9149-9159 (2001). “A unified scheme for ab initio molecular orbital theory and path integral molecular dynamics”
- M. Shiga, M. Tachikawa, S. Miura, Chem. Phys. Lett., 332, 396-402 (2000). “Ab initio molecular orbital calculation considering the quantum mechanical effect of nuclei by path integral molecular dynamics”

References

- [1] PIMD code, version 2.0, M. Shiga, 2016; PIMD code, version 2.1, M. Shiga, 2017; PIMD code, version 2.2, M. Shiga, 2018; PIMD code, version 2.3, M. Shiga, 2019; PIMD code, version 2.4, M. Shiga, 2020; PIMD code, version 2.5, M. Shiga, 2022; PIMD code, version 2.6, M. Shiga, 2023. PIMD code, version 2.7, M. Shiga, 2025.
- [2] C. G. Broyden, Journal of the Institute of Mathematics and Its Applications 6, 76 (1970); R. Fletcher, Computer Journal 13, 317 (1970); D. Goldfarb, Mathematics of Computation 24, 23 (1970); D. F. Shanno, Mathematics of Computation 24, 647 (1970).
- [3] Limited memory BFGS: J. Nocedal, “Updating Quasi-Newton Matrices with Limited Storage”, Mathematics of Computation, 35, 773-782 (1980); D. C. Liu and J. Nocedal, “On the Limited Memory Method for Large Scale Optimization”, Mathematical Programming B, 45, 3, 503-528 (1989), available from website.

- [4] W. Humphrey, A. Dalke and K. Schulten, “VMD - Visual Molecular Dynamics”, J. Molec. Graphics, 14, 33-38 (1996); available from website.
- [5] K. Fukui, Acc. Chem. Res., 14, 363 (1981).
- [6] M. P. Allen and D. J. Tildesley, Computer Simulation of Liquids, Oxford University Press, Oxford, 1987.
- [7] D. Frenkel and B. Smit, Understanding molecular simulation Academic Press, San Diego, 2001.
- [8] S. Nosé, J. Chem. Phys. 81, 511 (1984); Mol. Phys. 52, 255-268 (1984).
- [9] S. Nosé and M. L. Klein, Mol. Phys. 50, 1055 (1983).
- [10] W. Hoover, Phys. Rev. A 31, 1695 (1985).
- [11] G. J. Martyna, M. L. Klein and M. E. Tuckerman, J. Chem. Phys. 97, 2635 (1992).
- [12] M. E. Tuckerman, B. J. Berne and G. J. Martyna, J. Chem. Phys. 97, 1990 (1992).
- [13] G. J. Martyna, M. E. Tuckerman, D. J. Tobias and M. L. Klein, Mol. Phys. 87, 1117 (1996).
- [14] G. J. Martyna, A. Hughes and M. E. Tuckerman, J. Chem. Phys. 110, 3275 (1999).
- [15] W. Shinoda, M. Shiga and M. Mikami, Phys. Rev. B 69, 134103 (2004).
- [16] H. C. Andersen, J. Chem. Phys. 72, 2384 (1980).
- [17] M. Parrinello and A. Rahman, Phys. Rev. Lett. 45, 1196 (1980); J. Appl. Phys. 52, 7182 (1981); J. Chem. Phys. 76, 2662 (1982).
- [18] P. Pulay and G. Fogarasi, Chem. Phys. Lett., 386, 272 (2004).
- [19] J. Kolafa, J Comput Chem. 25:335-42 (2004).
- [20] J. C. Tully, J. Chem. Phys. 93, 1061 (1990).
- [21] D. Marx and J. Hutter, Ab Initio Molecular Dynamics, Basic Theory and Advanced Methods, Cambridge, 2009.
- [22] M. Tuckerman, Statistical Mechanics: Theory and Molecular Simulation, Oxford, 2010.
- [23] M. E. Tuckerman, B. J. Berne, G. J. Martyna and M. L. Klein, J. Chem. Phys. 99, 2796 (1993).
- [24] I. R. Craig and D. E. Manolopoulos, J. Chem. Phys. 121, 3368 (2004); J. Chem. Phys. 122, 084106 (2005); J. Chem. Phys. 123, 034102 (2005).
- [25] B. J. Braams and D. E. Manolopoulos, J. Chem. Phys. 125, 124105 (2006).
- [26] J. Cao and G. A. Voth, J. Chem. Phys. 99, 10070 (1993); J. Chem. Phys. 100, 5093 (1994); J. Chem. Phys. 100, 5106 (1994); J. Chem. Phys. 101, 6157 (1994); J. Chem. Phys. 101, 6168 (1994); J. Chem. Phys. 101, 6184 (1994).
- [27] S. Jang and G. A. Voth, J. Chem. Phys. 111, 2357 (1999); J. Chem. Phys. 111, 2371 (1999).
- [28] M. Shiga, M. Tachikawa and S. Miura, Chem. Phys. Lett. 332, 396 (2000); J. Chem. Phys. 115, 9149 (2001).
- [29] M. Shiga and A. Nakayama, Chem. Phys. Lett. 451, 175 (2008).
- [30] A. Kaczmarek, M. Shiga and D. Marx, J. Phys. Chem. A 113, 1985 (2009).

- [31] A. Witt, S. Ivanov, M. Shiga, H. Forbert and D. Marx, *J. Chem. Phys.* 130, 194510 (2009).
- [32] M. Shiga and H. Fujisaki, *J. Chem. Phys.* 136, 184103 (2012).
- [33] M. Shiga, *J. Comput. Chem.*, 43, 1864 (2022).
- [34] M. Rossi, M. Ceriotti, and D. E. Manolopoulos, *J. Chem. Phys.* 140, 234116 (2014).
- [35] K. Suzuki, M. Tachikawa, M. Shiga, *J. Chem. Phys.* 132, 144108 (2010)
- [36] A. Laio and M. Parrinello, *Proc. Natl. Acad. Sci. USA*, 99, 12562 (2002).
- [37] A. Raiteri, A. Laio, F. L. Gervasio, C. Micheletti and M. Parrinello, *J. Phys. Chem. B*, 110, 3533 (2006).
- [38] B. Ensing, A. Laio, M. Parrinello, and M. Klein, *J. Phys. Chem. B*, 109, 6676 (2004);
- [39] W. E, W. Ren, and E. Vanden-Eijnden, *Phys. Rev. B* 66, 052301 (2002); *J. Chem. Phys.* 126, 164103 (2007).
- [40] J. Ribas-Ariño, M. Shiga and D. Marx, *Angew. Chem. Int. Ed.*, 48, 4190 (2009); *J. Am. Chem. Soc.* 132, 10609-10614 (2010).
- [41] W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey and M. L. Klein, *J. Chem. Phys.* 79, 926 (1983).
- [42] E. Neria, S. Fischer and M. Karplus, *J. Chem. Phys.* 105, 1902 (1996).
- [43] J. Lobaugh and G. A. Voth, *J. Chem. Phys.* 106, 2400 (1997).
- [44] J. R. Reimers, R. O. Watts and M. L. Klein, *Chem. Phys.* 64, 95 (1982).
- [45] M. S. Daw and M. I. Baskes, *Phys. Rev. Lett.* 50, 1285 (1983); *Phys. Rev. B* 29, 6443 (1984).
- [46] S. M. Foiles, M. I. Baskes and M. S. Daw, *Phys. Rev. B* 33, 7983 (1986).
- [47] J. E. Angelo, N. R. Moody, and M. I. Baskes, *Modelling Simul. Mater. Sci. Engr.* 3, 289-307 (1995); M. I. Baskes, X. W. Sha, J. E. Angelo, and N. R. Moody, *Modelling Simul. Mater. Sci. Eng.* 5, 651-652 (1997).
- [48] J. Tersoff, *Phys. Rev. Lett* 61, 2879 (1988); J. Tersoff, *Phys. Rev. B* 37, 6991 (1988); J. Tersoff, *Phys. Rev. B* 39, 5566 (1989); J. Tersoff, *Phys. Rev. B* 41, 3248 (1990).
- [49] P. Clabaut, M. Beisert, C. Michel, S. N. Steinmann, *J. Chem. Phys.* 157, 194705 (2022).
- [50] Y. Mishin, M. J. Mehl, D. A. Papaconstantopoulos, *Acta Mater.* 53, 4029 (2005).
- [51] R. T. Cygan, J.-J. Liang, A. G. Kalinichev, *J. Phys. Chem. B* 108, 1255 (2004).
- [52] M. Shiga, M. Masia, *J. Chem. Phys.* 139, 144103 (2013); *J. Chem. Phys.* 139, 044120 (2013); *J. Chem. Phys.* 139, 119901 (2013).
- [53] J. J. P. Stewart, "MOPAC: A General Molecular Orbital Package." *Quant. Chem. Prog. Exch.*, 10, 86 (1990); MOPAC2009, MOPAC2012, James J. P. Stewart, Stewart Computational Chemistry, Colorado Springs, CO, USA.

Available from website <http://openmopac.net>.

- [54] Density Functional based Tight Binding (DFTB): D. Porezag, T. Frauenheim, T. Köhler, G. Seifert, and R. Kaschner, “Construction of tight-binding-like potentials on the basis of density-functional theory: Application to carbon”, *Phys. Rev. B* 51, 12947 (1995); G. Seifert, D. Porezag, and T. Frauenheim, “Calculations of molecules, clusters, and solids with a simplified LCAO-DFT-LDA scheme”, *Int. J. Quantum Chemistry* 58, 185 (1996); M. Elstner, D. Porezag, G. Jungnickel, J. Elsner, M. Haugk, T. Frauenheim, S. Suhai, and G. Seifert, “Self-consistent-charge density-functional tight-binding method for simulations of complex materials properties”, *Phys. Rev. B* 58, 7260 (1998).
Available from website <http://www.dftb.org/home>.
- [55] The CP2K developers group, 2020. CP2K is freely available from website <https://www.cp2k.org>. Please see the end of your PIMD output for the list of citations recommended for your particular type of CP2K calculation.
J. Hutter, M. Iannuzzi, F. Schiffmann, J. VandeVondele, “CP2K: Atomistic simulations of condensed matter systems”, *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4, 15 (2014).
T. D. Kühne, M. Iannuzzi, M. Del Ben, V. V. Rybkin, P. Seewald, F. Stein, T. Laino, R. Z. Khaliullin, O. Schütt, F. Schiffmann, D. Golze, J. Wilhelm, S. Chulkov, M. H. Bani-Hashemian, V. Weber, U. Borštnik, M. Taillefumier, A. S. Jakobovits, A. Lazzaro, H. Pabst, T. Müller, R. Schade, M. Guidon, S. Andermatt, N. Holmberg, G. K. Schenter, A. Hehn, A. Bussy, F. Belleflamme, G. Tabacchi, A. Glöß, M. Lass, I. Bethune, C. J. Mundy, C. Plessl, M. Watkins, J. VandeVondele, M. Krack, and J. Hutter, “CP2K: An electronic structure and molecular dynamics software package-Quickstep: Efficient and accurate electronic structure calculations”, *J. Chem. Phys.* 152, 194103 (2020).
- [56] T. Yamasaki, A. Kuroda, T. Kato, J. Nara, J. Koga, T. Uda, K. Minami, and T. Ohno, “Multi-axis Decomposition of Density Functional Program for Strong Scaling up to 82,944 Nodes on the K Computer: Compactly Folded 3D-FFT Communicators in the 6D Torus Network”, *Comp. Phys. Commun.* 244, 264 (2019).
T. Hamada and T. Ohno, “First-Principles Broadband Dielectric Spectroscopy”, *J. Aust. Ceram. Soc.*, 47, 61 (2011).
PHASE/0 is available from website <https://azuma.nims.go.jp>.
- [57] GAUSSIAN 98, Revision A.6, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, V. G. Zakrzewski, J. A. Montgomery, Jr., R. E. Stratmann, J. C. Burant, S. Dapprich, J. M. Millam, A. D. Daniels, K. N. Kudin, M. C. Strain, Ö. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G. A. Petersson, P. Y. Ayala, Q. Cui, K. Morokuma, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. Cioslowski, J. V. Ortiz, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. Gomperts, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, C. Gonzalez, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, J. L. Andres, C. Gonzalez, M. Head-Gordon, E. S. Replogle, and J. A. Pople, GAUSSIAN, Inc., Pittsburgh PA, 1998.
available from website <http://www.gaussian.com/index.htm>.
- [58] GAUSSIAN 03, Revision C.02, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, J. A. Montgomery, Jr., T. Vreven, K. N. Kudin, J. C. Burant, J. M. Millam, S. S. Iyengar, J. Tomasi, V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega, G. A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, M. Klene, X. Li, J. E. Knox, H. P. Hratchian, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, P. Y. Ayala, K. Morokuma, G. A. Voth, P. Salvador, J. J. Dannenberg, V. G. Zakrzewski, S. Dapprich, A. D. Daniels, M. C. Strain, Ö. Farkas, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. V. Ortiz, Q. Cui, A. G. Baboul, S. Clifford, J. Cioslowski, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng,

A. Nanayakkara, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, C. Gonzalez, and J. A. Pople, GAUSSIAN, Inc., Wallingford CT, 2004.

Available from website <http://www.gaussian.com/index.htm>.

- [59] GAUSSIAN 09, Revision D.01, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, B. Mennucci, G. A. Petersson, H. Nakatsuji, M. Caricato, X. Li, H. P. Hratchian, A. F. Izmaylov, J. Bloino, G. Zheng, J. L. Sonnenberg, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. Bearpark, J. J. Heyd, E. Brothers, K. N. Kudin, V. N. Staroverov, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, N. Rega, J. M. Millam, M. Klene, J. E. Knox, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, R. L. Martin, K. Morokuma, V. G. Zakrzewski, G. A. Voth, P. Salvador, J. J. Dannenberg, S. Dapprich, A. D. Daniels, Ö. Farkas, J. B. Foresman, J. V. Ortiz, J. Cioslowski, and D. J. Fox, Gaussian, Inc., Wallingford CT, 2009.

Available from website <http://www.gaussian.com/index.htm>.

- [60] GAUSSIAN 16, Revision B.01, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox, Gaussian, Inc., Wallingford CT, 2016.

Available from website <http://www.gaussian.com/index.htm>.

- [61] GAMESS: General atomic and molecular electronic structure system, M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. J. Su, T. L. Windus, M. Dupuis, J. A. Montgomery, *J. Comput. Chem.*, **14**, 1347 (1993); “Advances in electronic structure theory: GAMESS a decade later”, M. S. Gordon, M. W. Schmidt pp. 1167-1189, in “Theory and Applications of Computational Chemistry: the first forty years”, C. E. Dykstra, G. Frenking, K. S. Kim, G. E. Scuseria (editors), Elsevier, Amsterdam, 2005.

Available from website <http://www.msg.chem.iastate.edu/gamess>.

- [62] TURBOMOLE V6.2 2010, a developmet of University of Karlsruhe and Forschungszentrum Karlsruhe GmbH, 1989-2007, TURBOMOLE GmbH, since 2007; R. Ahlrichs, M. B̈, M. Häser, H. Horn, C. Kölmel, “Electronic structure calculations on workstation computers: The program system turbomole.” *Chem. Phys. Lett.* 162, 165–169 (1989).

Available from website <http://www.turbomole.com>.

- [63] MOLPRO quantum chemistry package: H.-J. Werner, P. J. Knowles, G. Knizia, F. R. Manby, M. Schütz, “Molpro: a general purpose quantum chemistry program package”, *WIREs Comput. Mol. Sci.*, **2**, 242-253 (2012); MOLPRO, version 2012.1, H.-J. Werner, P. J. Knowles, G. Knizia, F. R. Manby, M. Schütz, P. Celani, T. Korona, R. Lindh, A. Mitrushenkov, G. Rauhut, K. R. Shamasundar, T. B. Adler, R. D. Amos, A. Bernhardsson, A. Berning, D. L. Cooper, M. J. O. Deegan, A. J. Dobbyn, F. Eckert, E. Goll, C. Hampel, A. Hesselmann, G. Hetzer, T. Hrenar, G. Jansen, C. Köppl, Y. Liu, A. W. Lloyd, R. A. Mata, A. J. May, S. J. McNicholas, W. Meyer, M. E. Mura, A. Nicklass, D. P. O’Neill, P.

- Palmieri, D. Peng, K. Pflüger, R. Pitzer, M. Reiher, T. Shiozaki, H. Stoll, A. J. Stone, R. Tarroni, T. Thorsteinsson, M. Wang.
Available from website <http://www.molpro.net>.
- [64] NTChem: T. Nakajima, M. Katouda, M. Kamiya, and Y. Nakatsuka, *Int. J. Quantum Chem.* 115, 349–359 (2015).
Available from website http://labs.aics.riken.jp/nakajimat_top/ntchem_j.html.
- [65] ORCA - An ab initio, DFT and semiempirical SCF-MO package - version 3.0: F. Neese, F. Wennmohs, U. Becker, D. Bykov, D. Ganyushin, A. Hansen, R. Izsák, D. G. Liakos, C. Kollmar, S. Kossmann, D. A. Pantazis, T. Petrenko, C. Reimann, C. Riplinger, M. Roemelt, B. Sandhoefer, I. Schapiro, K. Sivalingham, B. Wezislá, M. Kállay, S. Grimme, E. Valeev, G. Chan.
Available from website <https://orcaforum.cec.mpg.de>.
- [66] Quantum Espresso: P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, A. Dal Corso, S. Fabris, G. Fratesi, S. de Gironcoli, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A. P. Seitsonen, A. Smogunov, P. Umari, R. M. Wentzcovitch, *J. Phys.: Condens. Matter* 21, 395502 (2009).
Available from website <https://www.quantum-espresso.org>.
- [67] The Vienna Ab initio Simulation Package (VASP): G. Kresse and J. Hafner. “Ab initio molecular dynamics for liquid metals”, *Phys. Rev. B*, 47, 558 (1993); G. Kresse and J. Hafner, “Ab initio molecular-dynamics simulation of the liquid-metal-amorphous-semiconductor transition in germanium”, *Phys. Rev. B*, 49, 14251 (1994). G. Kresse and J. Furthmüller, “Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set”, *Comput. Mat. Sci.*, 6, 15 (1996). G. Kresse and J. Furthmüller, “Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set”, *Phys. Rev. B*, 54, 11169 (1996).
Available from website <http://www.vasp.at>.
- [68] Scalable Molecular Analysis Solver for High-performance computing systems (SMASH), K. Ishimura.
Available from website <http://smash-qc.sourceforge.net>.
- [69] ABINIT-MP.
Available from website <http://moldb.nihs.go.jp/abinitmp>.
- [70] J. W. Ponder, TINKER: Software tools for molecular design, Washington University School of Medicine, Saint Louis, MO.
Available from website <http://dasher.wustl.edu/tinker>.
- [71] W. L. Jorgensen, “BOSS - Biochemical and Organic Simulation System”, *The Encyclopedia of Computational Chemistry*, P. v. R. Schleyer (editor-in-chief), John Wiley & Sons Ltd, Athens, USA, 5, 3281-3285 (1998); W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives, *J. Am. Chem. Soc.* 118, 11225-11236 (1996); W. Damm, A. Frontera, J. Tirado-Rives, and W. L. Jorgensen, *J. Comput. Chem.* 18, 1955-1970 (1997); W. L. Jorgensen and N. A. McDonald, *Theochem* 424, 145-155 (1998); W. L. Jorgensen and N. A. McDonald, *J. Phys. Chem. B* 102, 8049-8059 (1998); R. C. Rizzo and W. L. Jorgensen, *J. Am. Chem. Soc.* 121, 4827-4836 (1999); E. K. Watkins and W. L. Jorgensen, *J. Phys. Chem. A* 105, 4118-4125 (2001); M. L. P. Price, D. Ostrovsky, and W. L. Jorgensen, *J. Comput. Chem.*, 22, 1340-1352 (2001); W. L. Jorgensen, J. P. Ulmschneider, J. Tirado-Rives, *J. Phys. Chem. B* 108, 16264-70 (2004); W. L. Jorgensen, J. Tirado-Rives, *J. Comput. Chem.* 26, 1689-1700 (2005); K. P. Jensen and W. L. Jorgensen, *J. Chem. Theory Comput.* 2, 1499-1509 (2006).

- [72] D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, S. DeBolt, D. Ferguson, G. Seibel and P. Kollman, AMBER, a Package of Computer Programs for Applying Molecular Mechanics, Normal Mode Analysis, Molecular Dynamics and Free Energy Calculations to Simulate the Structural and Energetic Properties of Molecules, *Comp. Phys. Commun.* 91, 1-41 (1995); J. Wang, P. Cieplak and P. A. Kollman, *J. Comput. Chem.*, 21, 1049-1074 (2000) W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, Jr., D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell and P. A. Kollman, *J. Am. Chem. Soc.*, 117, 5179-5197 (1995); G. Moyna, H. J. Williams, R. J. Nachman and A. I. Scott, *Biopolymers*, 49, 403-413 (1999); W. S. Ross and C. C. Hardin, *J. Am. Chem. Soc.*, 116, 6070-6080 (1994); J. Aqvist, *J. Phys. Chem.*, 94, 8021-8024 (1990).
- [73] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan and M. Karplus, CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics Calculations, *J. Comput. Chem.* 4, 187-217 (1983). W. E. Reiher III, "Theoretical Studies of Hydrogen Bonding", Ph.D. Thesis, Department of Chemistry, Harvard University, Cambridge, MA, 1985. L. Nilsson and M. Karplus, "Empirical Energy Functions for Energy Minimizations and Dynamics of Nucleic Acids", *J. Comput. Chem.*, 7, 591-616 (1986). E. Neria, S. Fischer and M. Karplus, "Simulation of Activation Free Energies in Molecular Systems", *J. Chem. Phys.*, 105, 1902-1921 (1996). A. D. MacKerrell, Jr., et al., "All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins", *J. Phys. Chem. B*, 102, 3586-3616 (1998); N. Foloppe and A. D. MacKerell, Jr., "All-Atom Empirical Force Field for Nucleic Acids: I. Parameter Optimization Based on Small Molecule and Condensed Phase Macromolecular Target Data", *J. Comput. Chem.*, 21, 86-104 (2000); Current parameter values are available from the CHARMM website, http://mackerell.umaryland.edu/CHARMM_ff_params.html.
- [74] T. Morishita, S. G. Itoh, H. Okumura, M. Mikami, *Phys. Rev. E* 85, 066702 (2012); *J. Comp. Chem.* 34, 1375 (2013).
- [75] T. Morishita, Y. Yonezawa, A. M. Ito, *J. Chem. Theory Comput.* 13, 3106 (2017).
- [76] L. Rosso, P. Mináry, Z. Zhu, M. E. Tuckerman, *J. Chem. Phys.* 116, 4389 (2002).
- [77] J. B. Abrams, M. E. Tuckerman, *J. Phys. Chem. B* 112, 15742 (2008).
- [78] L. Maragliano, E. Vanden-Eijnden, *Chem. Phys. Lett.* 426, 168 (2006).
- [79] W. E, X. Zhou, *Nonlinearity* 24, 1831 (2011); A. Samanta, W. E, *J. Chem. Phys.* 136, 124104 (2012).
- [80] C. Ming, T.-Q. Yu, M. E. Tuckerman, *Proc. Nat. Acad. Sci.* 112, 3235 (2015).
- [81] S. Awasthi, V. Kapil, N. N. Nair, *J. Comput. Chem.* 37, 1413-1424 (2016).
- [82] S. Awasthi, N. N. Nair, *J. Chem. Phys.* 146, 094108 (2016).
- [83] FFTPACK: P. N. Swarztrauber, Vectorizing the FFT's, in *Parallel Computations*, edited by G. Rodrigue, Academic Press, 1982, pages 51-83.
- [84] N. Artrith, The Atomic Energy Neural Network (AENET) version 2.0.0: N. Artrith and A. Urban, *Comput. Mater. Sci.* 114, 135-150 (2016); N. Artrith, A. Urban, and G. Ceder, *Phys. Rev. B* 96, 014112 (2017); available from AENET website.
- [85] J. Sala, E. Guàrdia, M. Masia, *J. Chem. Phys.* 133, 234101 (2010).
- [86] L. Ojamäe, I. Shavitt, S. J. Singer, *J. Chem. Phys.* 109, 5547 (1998).
- [87] I. S. Novikov, K. Gubaev, E. V. Podryabinkin, A. V. Shapeev, *Machine Learning: Science and Technology*, 2 025002 (2020).
- [88] A. V. Shapeev, *Multiscale Modeling and Simulation*, 14, 1153 (2016).

- [89] I. Batatia, D. P. Kovács, G. Simm, C. Ortner and G. Csányi, “MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields”, arXiv:2206.07697 (2022).
- [90] n2p2 - A neural network potential package: A. Singraber, et al. Available from website <https://github.com/CompPhysVienna/n2p2>.
- [91] PFP - PreFerred Potential. Website <https://matlantis.com/en/product/about-pfp/>.