MixPI: User Guide

Britta A. Johnson[1], Siyu Bu[2], Christopher J. Mundy[1], and Nandini Ananth[2]

[1]Physical Science Division, Pacific Northwest National Laboratory, Richland, Washington 99352, USA
[2] Department of Chemistry and Chemical Biology, Cornell University, Ithaca, New York 14850, USA

April 15, 2025

Contents

1 Downloading and Compiling CP2K and MixPI **2**
1.1 Download . . . 2
1.1.1 MixPI . . . 2
1.1.2 CP2K . . . 2
1.2 Compile . . . 3
1.2.1 CP2K . . . 3
1.2.2 MixPI . . . 4
1.2.3 Troubleshooting . . . 4
1.2.3.1 Cannot Locate Specific External Libraries . . . 4
1.2.3.1.1 ELPA . . . 4
1.2.3.1.2 Other External Libraries . . . 5

2 Running MixPI **5**
2.1 Input . . . 5
2.1.1 CP2K Input Files . . . 6
2.1.1.1 Force Information in FORCE_EVAL . . . 6
2.1.1.2 Coordinate Information: structure.xyz . . . 6
2.1.1.3 Topology Information: structure.psf; rp_molset.dat; and classical_molset.dat . . . 9
2.1.1.3.1 PSF File . . . 9
2.1.1.3.2 MOL_SET Information . . . 10
2.1.2 MixPI Input Files . . . 11
2.1.2.1 rp_molecule_info.inp . . . 11
2.1.2.2 md_run.inp . . . 11
2.1.2.3 file_info.inp . . . 12
2.1.3 Input Generation Script . . . 13
2.2 Output Files . . . 14
2.2.1 Energy Files . . . 15
2.2.2 Trajectory Files . . . 15

2.2.3 Final position and velocity files 16
2.2.4 RESTART Files 16
2.2.5 Other Files 16

3 Analysis Scripts **16**
3.1 rdf calculation 16

4 Theory **17**
4.1 All Replica Ring Polymer Hamiltonian 18
4.2 MixTS Treatment of 1/N 19
4.3 MTS-Expansion in Three-Dimensions 21

5 Funding Acknowledgment **24**

1 Downloading and Compiling CP2K and MixPI

1.1 Download

1.1.1 MixPI

The most recent version of MixPI is v2.1 and is available on the GitHub page: https://github.com/AnanthGroup/Path-Integral-Atomistic-Driver [1]. Subsequent versions will be developed and updated to this page. The following subdirectories are located within the MixPI download:

- src/ : Contains the source codes for MixPI.
- examples/ : Example input files for a MTS bulk water simulation and a $Co^{3+} + e^{-}$ simulation
- cp2k-updated-files/ : The cp2k source files that were updated to interface with MixPI

All files in the src/ folder should be downloaded. Files that contain the changes to CP2K are also located in the updated-cp2k-files/ and need to be downloaded as well.

1.1.2 CP2K

MixPI requires the installation of the molecular dynamics package CP2K v2023.2 [4] which can be found on the CP2K website. CP2K must be installed as both an executable and library. The instructions for each are also located on the CP2K website. A few changes to the CP2K source code were made in the development of MixPI. These changes were made to the following CP2K files:

- constraint.F
- constraint_vsite.F
- start/libcp2k.F
- subsys/molecule_kind_types.F
- topology_connectivity_util.F

- topology_coordinate_util.F
- topology_generate_util.F
- topology_psf.F
- topology_types.F
- topology_util.F

Unless indicated, these files are located in the src/ directory. The updated versions of these files are located in the MixPI directory cp2k-updated-files/. These changes do not interfere with running CP2K as normal.

1.2 Compile

1.2.1 CP2K

After replacing the updated files from cp2k-updated-files/ into the appropriate directory in the src/ folder in the CP2K build; the authors suggest using the toolchain build to generate arch files and compile CP2K. The instructions below will assume the toolchain was used. Instructions for the toolchain build are found in the CP2K documentation. To build the CP2K version used by the authors, the following steps were used.

- From the cp2k/ directory, access the toolchain.
  cd /tools/toolchain/

- Make sure you have updated paths to the gcc compiler and OpenMPI (we used gcc 10.3.0; OpenMPI 4.2.1). If not, it will go out and download/install it's own versions.

- Run install command.
  ./install_cp2k_toolchain.sh
  ./install_cp2k_toolchain.sh –with-sirius=no
  For additional features to be linked to CP2K (like PLUMED), include those in the install line. (For example ./install_cp2k_toolchain.sh -with-plumed)

- After build, copy the arch files generated in install/arch to the cp2k arch directory
  cp install/arch/local* ../../arch/

- Source the locations of the libraries that are linked to CP2K
  source install/setup

- Return to the cp2k directory and compile.
  cd ../..
  make -j 12 ARCH=local VERSION='ssmp sdbg psmp pdbg'

- After successful compilation, compile cp2k as library.
  make -j 12 ARCH=local VERSION='ssmp sdbg psmp pdbg' libcp2k

- The executables will now be located in cp2k/exe/local/ folder.

1.2.2 MixPI

After compiling CP2K as an executable and library, the paths in the makefile for MixPI must be updated to find the correct libraries. This is done with three separate updates:

- Copy all libraries flags from the CP2K arch file (The arch file is arch/local.psmp and the -l*** flags are located following "LIBS=".) onto the "LIB =" line in the makefile located in the src/ directory of MixPI.

- For MixPI to find these libraries, update the -I and -L paths within the makefile or, if compiled with the toolchain build, source the same setup file as used when building cp2k. NOTE: WRITE OUT THE SOURCE LINE AND BETTER EXPLAIN THE LINKING NEEDED IN THE MAKE FILE!

- Update the locations of the cp2k/dbcsr libraries and object files. They are located at:
  CP2K_LIB = /**cp2k-directory**/lib/local/psmp
  DBSCR_LIB = /**cp2k-directory**/lib/local/psmp/exts/dbcsr
  CP2K_INC = /**cp2k-directory**/obj/local/psmp
  DBSCR_INC = /**cp2k-directory**/obj/local/psmp/exts/dbcsr

This should generate an executable (cp2k_pimd.out) which will run MixPI.

1.2.3 Troubleshooting

Some users have experienced issues a few specific issues with the instructions above. The most common issues (and their fixes) are outlined below.

1.2.3.1 Cannot Locate Specific External Libraries When using the toolchain build above, CP2K will automatically download, install, and link to various external libraries. Of these, the majority are optional and are only used for specific types of simulations in CP2K. A few of the prerequisites that are automatically generated in the toolchain have generated errors for some users; these include:

1.2.3.1.1 ELPA The ELPA libarary replaces the SCALAPACK matrix diagonalization in some CP2K subroutines. Some users report an error when compiling CP2K that it cannot locate the ELPA libaries. When this happens, the path that points towards the ELPA libararies is not generated correctly in the toolchain build. This can be fixed in two ways. The toolchain build can exclude the ELPA libarary by using the command ./install_cp2k_toolchain.sh –with-elpa=no
The user can also modify the install/setup file to point to location of the include

and library directories for elpa. In most cases, this involves changing the paths in the setup file from:
cp2k/tools/toolchain/install/elpa-2022.11/bin ← cp2k/tools/toolchain/install/elpa-2022.11/bin/cpu/
cp2k/tools/toolchain/install/elpa-2022.11/lib ← cp2k/tools/toolchain/install/elpa-2022.11/cpu/lib
cp2k/tools/toolchain/install/elpa-2022.11/include ← cp2k/tools/toolchain/install/elpa-2022.11/cpu/include

1.2.3.1.2 Other External Libraries Some users have also reported issues (usually when compiling MixPI) that it cannot locate certain external library functions (usually related to either libxc or libsxmm). If this error occurs, it is usually a result of a mismatch between specific versions of libxc/libxsmm and the versions of precompiled functions on the HPC. The user can track down these differences, but in this case, many of the external libraries are not needed to run the relevant MD simulations. Therefore, we suggest adding –with-LIBNAME=no to the toolchain build (similar to the ELPA example above). In our case, we always use the following install line:
./install_cp2k_toolchain.sh –with-elpa=no –with-libxc=no –with-libxsmm=no

2 Running MixPI

2.1 Input

In this section, we introduce the specific input files needed to run the MixPI software program. In this document, we have snapshots of input files; these input files are modified versions of the input files for these example files. In particular, we have kept the same molecules but have decreased the number of beads for ease of reading. For example, inputs from a simulation of two water molecules with 2 bead oxygen atoms and 4 bead hydrogen atoms are shown below; these mimic the input files needed for simulating a box of water with 32 bead H atoms and 4 bead O atoms. Similarly, we show example inputs for an aqueous Co^3+ ion with an explicit electron treated as a 8 bead ring polymer in a box of 2 classical water molecules. Two example simulations (a box of water with 32 bead H atoms and 4 bead O atoms; an aqueous Co^3+ ion with an explicit electron treated as 1024 bead ring polymer in a box of 91 classical water molecules) and the required input are located in the example directory on the MixPI GitHub page. The example directory contains the input files for each of these simulations as well as the output generated for a short MD run for each simulation.

The input files can be classified in three categories: CP2K input files related to the force/energy evaluations and the structure and connectivity of the system; MixPI specific input with PI specific information (such as number of PIs, size of PIs, etc.) and information on the MD simulations (step size, temperature, etc.); and system specific input files that are only required for particular simulation types. In this subsection below, we will discuss the input files for each category.

2.1.1 CP2K Input Files

A set of CP2K input files, similar to those needed for a traditional CP2K MD simulation, are required. While the inputs are structured like classical CP2K MD input, the MixPI CP2K input does need to be modified to reflect the accurate force and structure information for the path-integral set-ups. A simple program (called generate_input.F which is located within the examples/generate-input directory) will take the classical inputs and convert them into MixPI inputs. Each of the input files is outlined in a section below.

The CP2K input file (called cp2k.inp in our example folders) contains the input parameters dedicated to the force evaluations and system structure (in the CP2K input the section FORCE_EVAL which contains the subsections MM to describe the forces and SUBSYS which describes the coordinates and structure/topology of the system). Any parameters concerning the molecular dynamics steps (like temperature, time step, etc.) are read in from a different file. Below we will outline the information needed to accurately describe the force evaluations in the FORCE_EVAL section and the topology in the SUBSYS section of a CP2K input file.

The implementation describe below uses the internal CP2K format to describe the forces, along with the MOL_SET functionality combined with PSF files. This is one specific set-up for using MixPI. In the future, implementations using other formats in CP2K (like CHARMM and AMBER formats) will be described below.

2.1.1.1 Force Information in FORCE_EVAL The force definitions used in cp2k.inp must described the interactions between the classical atoms and RP beads, the RP beads on separate atoms, and the classical atom interactions. For example, a Lennard Jones interaction of the form

$$V_{\mathrm{LJ}} = 4\epsilon_{OO}\left[\left(\frac{\sigma_{OO}}{r_{OO}}\right)^{12} - \left(\frac{\sigma_{OO}}{r_{OO}}\right)^{6}\right] \tag{1}$$

which describes the interactions between the oxygen atoms of two water molecules for classical and PI descriptions has the form shown in Fig. 2a and b, respectively. As indicated, the overall strength of the interaction (ϵ_{OO}) is divided by the number of beads ($N = 2$) to account for the ring polymer bead interactions.

A similar force scaling is also done for the bonded interactions (shown in Fig.3) as well as the electrostatics (discussed in Sec. 2.1.1.3). Note that when determining the force scaling between RPs of different size (in this example, a 2 bead oxygen atom and a 4 bead hydrogen atom), the spring constant is scaled by the largest N value ($\max\{N_1, N_2\}$). Therefore, the classical spring constant (shown in Fig. 3a) is divided by 4 for the PI treatment in Fig. 3b.

2.1.1.2 Coordinate Information: structure.xyz A coordinate file that is readable by CP2K (file types include xyz, pdb, g96, crd, and cif) lists the positions if all the PI beads and classical atoms in the system. This file is specified by the keyword COORD_FILE_NAME in cp2k.inp. Note: The ordering in this file is important. The atoms should be listed by molecule and the molecules that contain atoms treated as PIs should be listed before any classical molecules. In

a)

```
        20

OW      1.52119189      3.59987467     -0.65267486
OW      1.41212811      3.58023486     -0.66121082
HW      1.40051269      4.48755586     -1.27161383
HW      1.51749487      4.36786417     -1.34831158
HW      1.56447467      4.37078067     -1.38028096
HW      1.54080963      4.34647665     -1.20164836
HW      2.49981521      3.62177425     -0.62069483
HW      2.42623010      3.74942244     -0.57205411
HW      2.30406005      3.89642216     -0.49243066
HW      2.39361826      3.94561109     -0.40747839
OW      7.14416691      5.94798807      1.97578711
OW      7.10550231      5.95219039      1.96504578
HW      7.99283069      6.24232509      2.35808672
HW      8.02492627      6.30709540      2.33746453
HW      8.06127738      6.18688167      2.26353234
HW      8.07629947      6.10077240      2.26389709
HW      7.31486588      5.63706251      1.03062343
HW      7.29932471      5.54765431      1.05641338
HW      7.28552433      5.47257206      1.03948812
HW      7.28614687      5.44442050      1.01137054
```

b)

```
       15

XX1   7.4998214332138531       7.0600360965994673       7.4657115878828666
XX2   7.3750054809032601       6.9603191914812088       7.2230723495121696
XX3   7.0379187699845369       6.7793013574326944       7.1562945443938117
XX4   7.1721635462039615       6.8855996059630673       7.0493341051416341
XX5   7.4513038953784907       6.9700067480225325       7.2054950448287283
XX6   7.3320077286154657       6.7846778930091105       7.3081813181867625
XX7   7.3242460509094567       6.9381173882641090       7.4106001015774696
XX8   7.2016850978494515       6.8800609074032097       7.4670236322431043
CO1   7.2514377950000002       7.0334793537999998       7.2374936876999998
O   8.3625721797000008       10.114931980100000       8.2990075364999996
H   7.5131330025000000       10.679729141599999       8.3130297377000009
H   9.1916837477000009       10.697869559000001       8.5837361454999996
O   3.6041146557000001       9.5224197500999992       10.496797300700001
H   2.9404470039000001       9.6895883376000000       11.342762965600000
H   3.0438899911999999       9.6005504217999995       9.6307834389000000
```

Figure 1: Two example configuration files (in the xyz format) for (a) a water simulation of two water molecules with 2 beads on the oxygen atoms and 4 beads on the hydrogen atoms and (b) a simulation of a classical Co^{3+} with an 8 bead PI electron solvated with two classical water molecules. These two configuration files are simpler examples of the configurations used in the example calculations. Note that the in the case of the Co^{3+} simulation, the molecules that contain PI are listed before the classical water molecules.

Fig. 1a, we show a few lines from a water.xyz file for a system with 2 beads on the oxygen atoms and 4 beads for the hydrogen atoms. In Fig. 1b, we show an XYZ file for a classical Co^{3+} ion with a PI electron (treated as 8 beads) and classical water. In this example, the Co^{3+} ion is bonded to the electron (since the electron should be centered on the Co^{3+} ion for this simulation) with a harmonic spring constant of 0. Because of this bond, the Co^{3+}-electron complex is treated as a molecule. The Co^{3+}-electron is listed first in the XYZ (since this molecule contains a PI) and the classical waters are listed last. Also note that

a)

```
&LENNARD-JONES
  ATOMS OW     OW
  EPSILON [K_e]                    78.26540000000000
  SIGMA [angstrom]                3.1654920000000000
  RCUT [angstrom]                 7.0000000000000000
&END LENNARD-JONES
```

b)

```
&LENNARD-JONES
  ATOMS 4OW     4OW
  EPSILON [K_e]                    39.13270000000000
  SIGMA [angstrom]                3.1654920000000000
  RCUT [angstrom]                 7.0000000000000000
&END LENNARD-JONES
```

Figure 2: The Lennard Jones parameters between two oxygen atoms in a (a) classical simulation and (b) PI simulation where the oxygen atoms are treated as 2 bead RPs using the qSPC/E force-field.

a)

```
&BOND
  ATOMS OW      HW
  K     [angstrom^-2kcalmol]   1059.162
  R0    [angstrom]                  1.0000000000000000
  KIND G87
&END BOND
```

b)

```
&BOND
  ATOMS 2OW      4HW
  K     [angstrom^-2kcalmol]   264.7905
  R0    [angstrom]                  1.0000000000000000
  KIND G87
&END BOND
```

Figure 3: The bonded parameters for an O-H bond in a (a) classical simulation and (b) PI simulation where the oxygen atoms are treated as 2 bead RPs and the hydrogen atoms are treated as 4 bead RPs using the qSPC/E force-field.

the atom label used in column 1 does not matter; the user can choose whichever atom labels are convenient.

2.1.1.3 Topology Information: structure.psf; rp_molset.dat; and classical_molset.dat Inside the TOPOLOGY section of the cp2k.inp file there are various methods for specifying how atoms are bonded to one another. Here we have chosen to use the method which uses PSF files and the MOL_SET section of CP2K. Below, we will outline how these files work together. Other methods for specifying bonding are allowed but the naming conventions outlined below must be followed with any of the methods.

2.1.1.3.1 PSF File A PSF file (protein structure file) is used to indicate the atom types and connectivity information for the system. A psf file for "classical" qSPC/E water and PI qSPC/E water is shown in Fig. 4a and b, respectively. The first block contains the atom information. The order of the columns is: atom ID, segment name, residue ID, residue name, atom name, atom type, charge, mass, and an unused 0. As we will see below, the important columns are segment name, atom name, atom type (which, when you compare to Fig. 2, matches the force-field information), charge, and mass. The format required for the PI variables in the atom definition block is as follows:

- Column 1: atom number. All particles (atoms or PI beads) are numbered started at 1. There is not change from the classical treatment.
- Column 2: molecule name. This name must start with "RP" if the molecule contains any ring polymers. In this example, we used RP10 as a molecule name (since it does contain ring polymers, and has 10 total beads).
- Column 3: molecule number. This can be any number. No change from the classical treatment.
- Column 4: atom type. The atom type starts with a two character atom identifier (in this case, an atomic kinds descriptor like "HW" but can be whatever the user chooses) and the bead number of that atom within its ring polymer.
- Column 5: force type. This is the force-field descriptor for that atom type. Since the force is dependent on atom type and the number of beads for the RP, the convention is the number of beads on that RP (N) and a two character atom type descriptor (which does not need to match the one used in Column 4, we just did in the this case).
- Column 6: charge. This is the charge of the atom divided by the number of beads.
- Column 7: mass. This is the mass of the atom divided by the number of beads.
- Column 8: a dummy variable.

a)

```
PSF EXT

       1 !SPCFw
  CP2K generated DUMP of connectivity

       3 !NATOM
       1 MOL1     1        SOL      OW       OW     -0.840000        15.999          0
       2 MOL1     1        SOL      HW       HW     0.420000         1.008           0
       3 MOL1     1        Sol      HW       HW     0.420000         1.008           0


       2 !NBOND
       1         2         1         3


       1 !NTHETA
       2         1         3


       0 !NPHI


       0 !NIMPHI
```

b)

```
PSF EXT

         1 !NTITLE


        10 !NATOMS
       1 RP10            1      RP  OW1     2OW    -0.420000        7.99950000       1
       2 RP10            1      RP  OW2     2OW    -0.420000        7.99950000       1
       3 RP10            1      RP  HW1     4HW     0.105000        0.25200000       1
       4 RP10            1      RP  HW2     4HW     0.105000        0.25200000       1
       5 RP10            1      RP  HW3     4HW     0.105000        0.25200000       1
       6 RP10            1      RP  HW4     4HW     0.105000        0.25200000       1
       7 RP10            1      RP  HW1     4HW     0.105000        0.25200000       1
       8 RP10            1      RP  HW2     4HW     0.105000        0.25200000       1
       9 RP10            1      RP  HW3     4HW     0.105000        0.25200000       1
      10 RP10            1      RP  HW4     4HW     0.105000        0.25200000       1


         8 !NBONDS
       1         3         1         4         2         5         2         6
       1         7         1         8         2         9         2        10


         4 !NTHETA
       3         1         7         4         1         8         5         2         9
       6         2        10


         0 !NPHI


         0 !NIMPHI
```

Figure 4: The PSF file for (a) classical qSPC/E water and (b) PI water.

The second, third, and fourth sections of the PSF file defines the bonds, bends, and dihedral angles relative to the atom numbers specified in column 1 above. These definitions are consistent with the classical descriptions when combined with the bead matching between ring-polymers defined by MixPI.

2.1.1.3.2 MOL_SET Information In Fig. 5, we show an example of the input "TOPOLOGY" section of cp2k.inp for the Co+PI electron with 2 water molecules (the configuration file is shown in Fig. 1b). The MOL_SET section allows a molecular PSF file to be repeated a certain number of times in a system. In this example, we have one PSF file that contains the information about the PI electron and the Co^{3+} ion. We have a second PSF file, that describes the connectivity of a single water molecule, that is repeated twice (NMOL 2) in order to describe the two water molecules in the system. This method means we only need to create PSf files for each unique molecule type in our system.

```
&TOPOLOGY
  COORD_FILE_NAME RP_structure.xyz
  COORD_FILE_FORMAT XYZ
  CONN_FILE_FORMAT MOL_SET
  &MOL_SET
    &MOLECULE
      NMOL             1
      CONNECTIVITY PSF
      CONN_FILE_NAME rp-electron-co.psf
    &END MOLECULE
    &MOLECULE
      NMOL             2
      CONNECTIVITY PSF
      CONN_FILE_NAME water.psf
    &END MOLECULE
  &END MOL_SET
&END TOPOLOGY
```

Figure 5: The TOPOLOGY section of the cp2k.inp file that uses the MOL_SET parameters to specify the connectivity for the Co+PI electron with 2 water molecules.

2.1.2 MixPI Input Files

2.1.2.1 rp_molecule_info.inp A file, called rp_molecule_info.inp, contains information about which molecules contain atoms treated as path integrals and the number of beads for each of those atoms. The structure is as follows:

The number of total molecules in the simulation that contain at least one PI atom
***The following block is repeated for as many molecule types as are present ***
Number of atoms in the molecule number of molecules of this type PSF file that describes this molecule type
Number of beads for atom 1 in molecule optional kinetic mass
Number of beads for atom 2 in molecule optional kinetic mass
...

In Fig. 6, we show the rp_molecule_info.inp file for a bulk water simulation with 90 water molecules where the O atoms are treated as 2 bead RPs and the H atoms are treated as 4 bead RPs. This file will be used by the sample make_pimd_input.F program (which will convert classical input files to the RP input files) and the MixPI program itself.

2.1.2.2 md_run.inp This file contains the information about the molecular dynamics methods. In order, the following variables are specified.

- nsteps (int): number of molecular dynamics steps
- print_freq (int): frequency to save and print and postions and energies

```
2                          !number of molecules with RPs
3     2      SPCFw.psf     !number of atoms in molecule type  number of times this molecule is repeated      psf file name
      2                       !number of RP beads of atom no. 1
      4                       !number of RP beads of atom no. 2
      4                       !number of RP beads of atom no. 3
```

Figure 6: The rp_molecule_info.inp file for a simulation of 2 water molecules described by the SPCFw.psf file with 2 bead RPs for the oxygen atoms and 4 bead RPs for the hydrogen atoms.

- dt (real): size of time step in fs
- method (string): pimd (specific kinetic mass) or rpmd (use harmonic mass)
- ensemble (string): nve or nvt
- thermostat (string): if ensemble = nvt, specific thermostat as pile (path-integral Langevin equation) or pileg (path-integral Langevin equation global where only the centroid velocities are themostat). If ensemble=nve, skip this line.
- dof (int): 3 (do not change)
- force_eval (string): cp2k (do not change)
- model_no (int): 1 (do not change)
- temperature (real): temperature to run simulation and initialize velocities
- propagation (string): method for propagation of PI normal modes (none=manually evaluate springs and propagate with velocity verlet; normal_manual=propagate using normal mode propagation).
- init_vel (logical): initial velocities to Boltzmann distribution or read in from file (rp-vel.dat)
- constraints (logical): are constraints present (frozen atoms; constrained centroids)
- tau_classical (real): classical thermostat variable τ_0
- tau_PI (real): PI thermostat variable τ_0
- print_level (string): low/high

Figure 7 shows an example md_run.inp file.

2.1.2.3 file_info.inp A file which outlines information about the CP2K input files. The input is as follows:

- cp2k_input (string): name of CP2K input file
- cp2k_output (string): name of output file for CP2K
- structure_file (string): name of initial structure file (usually xyz) (IGNORE: does not matter)

```
200             !nsteps for md run
1               !how frequent to save/print the positions
0.5             !dt in fs
pimd            !method
nvt             !ensemble (nvt or nve)
pileg           !thermostat (pile, pileg)
3               !dof for system;ignore
cp2k            !location of external forces;ignore
1               !model number;ignore
300.0           !temperature
normal_manual   !Use normal; progagation: none (single bead velocity verlet), normal (normal mode prop)
.true.          !init_vel_rp: initialize RP velocities from Boltzmann. If .F., read in from file rp-vel.dat
.false.         !Are contraints present (frozen atoms, centroid contraints)
300.0           !Thermostat contraint for Classical (tau0 in fs)
250.0           !Thermostt contraint for PI beads  (tau0 in fs)
low             !Print level (low/high)
```

Figure 7: The md_run.inp file for a simulation of an example simulation.

- classical_atoms (logical): do purely classical molecules exist? (I THINK CAN REMOVE THIS: NOTE TO BAJ to CHECK)
- molset_file (string): name of classical molset file (COULD JUST ENFORCE FILE NAME)

An example file_info.inp file is shown in Fig. 8 The molset file, which is

```
'cp2k.inp'              !cp2k input file
'output.out'            !cp2k output file
'RP_structure.xyz'      !structure file
.true.                  !logical for presence of classical atoms;ignore
'classical_molset.pot'  !cp2k_structure_type: file type for cp2k_structure_file
```

Figure 8: The file_info.inp file for a simulation of an example simulation.

specified in the final line of the file_info.inp file is a CP2K input file that describes the structure of a system using multiple copies of the same PSF file. See the relevant section of the CP2K manual for additional information. Additional information will also be given in section 2.1.1

2.1.3 Input Generation Script

The script make_pimd_input.F, located in the examples/generate_input directory, will create the above input files from the corresponding classical input. To run this program, compile using the following:
gfortran -ffree-form -O2 make_pimd_input.F -o make_pimd_input.out
to create an execuatable. You will need the following input files:

- structure.xyz: the structure of the classical system with RP's represented by their centroid positions
- NAME.psf: PSF files for the classical systems that contain PI atoms. The file name is up to the user and is specified in the rp_molecule_info.inp file.
- FF_info.inp: the input file for the classical force-field parameters. This file is currently a list of bond, bend, and Lennard-Jones parameters and the format is shown in Fig. 9.
- rp_molecule_info.inp: an input file that outlines the specific atoms and RP sizes for the system. This file format is also used by MixPI and is outlined in Sec. 2.1.2.1.

In Fig. 9, column 1 shows the classical input for make_pimd_input.F and column 2 shows the generated input for MixPI for the Co+PI electron with 2 water system.

Figure 9: This figure outlines the input (Column 1) and output (Column 2) for the make_pimd_input.F script. The files in column 2 are files that are used as input for the MixPI program.

2.2 Output Files

In addition to the main output file that cp2k generates (xxxx.out) that contains information about the simulation (what exactly do we read from this in a pimd run?), MixPI generates the following kinds of files:

- energy files
- trajectory files

- final velocity and structure files
- restart files
- other files
  - run time file

We now describe the structure and content of these files in detail.

2.2.1 Energy Files

Energy files print out system energies (in atomic unit) at each time step with printing frequency specified by the user. It includes energy_virial.dat, energy_average.dat, and energy_actual.dat:

- energy_virial.dat: from left to right there are seven columns; they are the time step number, time (in femtosecond), instant_temp (in Kelvin), virial kinetic energy, virial potential energy, primitive kinetic energy, and total energy (virial kinetic plus virial potential energy), respectively.
- energy_average.dat: A running average for the values of the items printed in the energy_virial.dat file above. Currently hardcoded to ignore the first 40% of the simulation. from left to right there are six columns; they are time step number, time (in fs), run_temp, run_kinetic, run_potential, and run_total.
- energy_actual.dat (only when print_level = high): from left to right there are nine columns; they are time step number, time(fs), classical atom kinetic energy, potential energy from cp2k, PI kinetic energy, bead spring potential energy, kinetic virial energy, potential virial energy, kinetic primtive energy

0	0.00	300.00	1.1717080560	-0.6793891680	0.9619499899	0.4923188881
1	0.50	303.62	1.1789204325	-0.6660613119	0.9958790603	0.5128591206
2	1.00	298.90	1.1984372222	-0.6444638106	1.0988697609	0.5539734116
3	1.50	298.62	1.2078107298	-0.6357955952	1.0580830089	0.5720151346
4	2.00	304.20	1.2097410077	-0.6373655460	1.0571058991	0.5723754617
5	2.50	297.38	1.2175113317	-0.6355688489	0.9442008275	0.5819424828
6	3.00	293.13	1.2263880295	-0.6326029724	1.0646594156	0.5937850571
7	3.50	306.44	1.2090415799	-0.6513013979	1.1167503982	0.5577401820
8	4.00	299.17	1.2011109033	-0.6563092589	1.1906860290	0.5448016444
9	4.50	294.33	1.2025665380	-0.6514312678	1.2459015417	0.5511352702
10	5.00	296.57	1.2089412226	-0.6399940607	1.1589664023	0.5689471619

Figure 10: energy_virial.dat

2.2.2 Trajectory Files

MixPI will output trajectory files giving system coordinates at every time step with printing frequency specified by the user. The trajectory files follow the same format as the input initial configuration files (section 2.1.1.1 and Figure 1). Four kinds of trajectory file will be generated, including trajectory_run_full.dat, trajectory_run_centroid.dat, trajectory_run_com.dat (generated if print_level = high), and trajectory_all.dat (generated if print_level = high)

- trajectory_run_full.dat: Full trajectories of all particles in the system (All PI beads and all classical atoms).

- trajectory_run_centroid.dat: trajectories of the center of mass of each atom (quantized or not). For example, if a hydrogen atom is quantized with 4 ring polymer beads, it will print the coordinate of "the average hydrogen" as the center of mass of the 4 beads.
- trajectory_run_com.dat: Full trajectories of all particles in the system, with system center of mass centered as (0,0,0).
- trajectory_all.dat: Full trajectories of all particles including their velocities at each configuration. The first three columns are the positions (in Angstrom) for three dimensions and the second three columns are the velocities (in Angstrom/fs) for three dimensions. (not centered)

2.2.3 Final position and velocity files

For the user's convenience to visualize or potentially restart from the final frame of simulation, final position and velocity files are generated at the end of a simulation.

- final_atom-vel.xyz prints out classical atom velocities (NEED TO EXPLAIN FORMAT)
- final_rp-vel.xyz prints out ring polymer bead velocities
- final_structure.xyz prints out the final configuration of all particles (including classical atoms and ring polymer beads)

2.2.4 RESTART Files

To restart from the final configuration and velocity at the end of a previous simulation, first change the input file md_run.F so that initial velocity will be read in instead of generated:

Change line 12 in *md_run.F* (init_vel_rp) to *.false.*

Then, simply change the name of the final position and velocity files (section 2.2.3) to the corresponding input file names:

cp final_atom-vel.xyz atom-vel.dat
cp final_rp-vel.xyz rp-vel.dat
cp final_structure.xyz RP_structure.xyz

Restart your simulation with these files in your working directory.

2.2.5 Other Files

- runtime_info.dat: A file that contains information about the time it takes to run each step, and the total time takes to complete the simulation. The time each component of MixPI contributes to the calculation time of each step is also shown. Time is in second.

3 Analysis Scripts

3.1 rdf calculation

Interesting properties can be calculated from a PIMD simulation. Here, we provide a Fortran script (pi_rdf.F) to calculate the radial distribution function

```
time for thermostat step    5.40000014E-02
time for update force env step    0.00000000
time for calc_force_cp2k step    7.90000036E-02
time for calc_force_pi step    0.00000000
time for propagate normal mode step    1.30000003E-02
time for one step   0.474000007
time for one and two step   0.465000004
complete run time is       285002   285.002014
```

Figure 11: runtime_info.dat

between different atoms from trajectories of an equilibrated run. We recognize that in a PIMD simulation atoms can be quantized with different number of beads. To calculate the rdf between two atoms involves taking the average of n_{max} number of rdfs between beads of the two atoms. For example, in the example simulation of a box a water with 32 bead H atoms and 4 bead O atoms, $n_{max} = 32$ rdfs between beads of hydrogen and oxygen atoms are calculated and averaged to produce the final rdf.

Example calculations using this script for rdf calculation are provided in the example folders. Three input files are needed to generate the rdf using the script:

- The trajectory file generated from the PIMD simulation for which you want to calculate the rdf. The file must be in .xyz format
- rp_molecule_info.inp. This file contains information on the system composition about which molecules contain atoms treated as path integrals and the number of beads for each of these atoms. This file is also used by MixPI and is explained in detail in Sec. 2.1.2.1.
- sim_details.inp. This file contains system related information that is required for the rdf calculation. An example is shown in 12 with comments explaining each item.

```
traj20ps_1.dat                    !Name of the trajectory file
200                               !Total number of configs in trajectory file
20                                !Number of initial configs to skip from trajectory file
14.0 14.0 14.0                    !Box size, in Angstrom
HW HW                             !Atom 1 and atom 2 label (consistent with what is in the trajectory file)
0.01 14.0                         !xmin and xmax for the rdf calcualtion, in Angstrom
```

Figure 12: sim_details.inp

4 Theory

Here we review the theory behind PIMD and MTS-PIMD; similar derivations have been presented previously [5, 2]. One particular deviation from some previous derivations is the location of the $\frac{1}{N}$ term, where N is the number of RP beads. In many prior derivations, this term was combined with the β term and, ultimately, resulted in simulations that are effectively conducted at higher temperatures. Since we can have different bead number N for different ring polymers in our mixTS derivation, we derive the RP Hamiltonian to include the $\frac{1}{N_i}$ terms, where i is the label for the ring polymer.

4.1 All Replica Ring Polymer Hamiltonian

The quantum mechanical canonical partition function can be written as

$$\begin{aligned} Z &= \mathrm{Tr}[e^{-\beta\hat{H}}] \\ &= \int dq \ \langle q| \, e^{-\beta\hat{H}} \, |q\rangle \, , \end{aligned} \tag{2}$$

where $|q\rangle$ is a position eigenstate of the one-dimensional Hamiltonian

$$\hat{H} = T(\hat{p}) + V(\hat{q}). \tag{3}$$

We use the asymmetric Trotter approximation to split the Hamiltonian into a potential and kinetic components and rewrite the partition function as

$$Z = \lim_{N\to\infty} \int dq \ \langle q| \left(e^{-\frac{\beta}{N}V(\hat{q})} e^{-\frac{\beta}{N}T(\hat{p})} \right)^N |q\rangle \, , \tag{4}$$

By inserting N-1 copies of the identity in the position basis ($\hat{I} = \int dq \ |q\rangle \langle q|$), we obtain the following expression.

$$Z = \lim_{N\to\infty} \int d\{\mathbf{q}\} \prod_{\alpha=1}^{N} \langle q_\alpha| \left(e^{-\frac{\beta}{N}V(\{\mathbf{q}\})} e^{-\frac{\beta}{N}T(\{\mathbf{p}\})} \right) |q_{\alpha+1}\rangle \, , \tag{5}$$

where $\int d\{\mathbf{q}\} = \int dq_1 \, dq_2 \, \ldots \, dq_{N+1}$, and $q_{N+1} = q_1$ to maintain the trace.

The potential term can be evaluated in the position basis:

$$\langle q_\alpha| \left(e^{-\frac{\beta}{N}V(\{q\})} e^{-\frac{\beta}{N}T(\{p\})} \right) |q_{\alpha+1}\rangle = e^{-\frac{\beta}{N}V(q_\alpha)} \langle q_\alpha| \left(e^{-\frac{\beta}{N}T(\{p\})} \right) |q_{\alpha+1}\rangle \tag{6}$$

In order to evaluate the kinetic term, we introduce identity in the momentum basis and evaluate the inner product ($\langle q|p\rangle = \frac{1}{\sqrt{2\pi\hbar}} e^{\frac{i}{\hbar}pq}$) to obtain

$$\begin{aligned} \langle q_\alpha| \left(e^{-\frac{\beta}{N}T(\{p\})} \right) |q_{\alpha+1}\rangle &= \int dp \, \langle q_\alpha| \, e^{-\frac{\beta}{N}T(\{p\})} \, |p\rangle \, \langle p| \, |q_{\alpha+1}\rangle \\ &= \frac{1}{\sqrt{2\pi\hbar}} e^{-\frac{\beta}{N}T(p)} \int dp \langle q_\alpha|p\rangle e^{-\frac{i}{\hbar}pq_{\alpha+1}} \\ &= \frac{1}{2\pi\hbar} \int dp e^{-\frac{\beta}{N}\left(\frac{p^2}{2m}\right)} e^{\frac{i}{\hbar}p(q_\alpha - q_{\alpha+1})} \end{aligned} \tag{7}$$

This integral can be solved by completing the square to obtain

$$\langle q_\alpha| \left(e^{-\frac{\beta}{N}V(\{q\})} e^{-\frac{\beta}{N}T(\{p\})} \right) |q_{\alpha+1}\rangle = \left(\frac{1}{2\pi\hbar} \right) \left(\frac{2\pi m N}{\beta} \right)^{1/2} e^{-\frac{\beta}{N}V(q_\alpha)} e^{-\frac{Nm}{2\beta\hbar^2}(q_\alpha - q_{\alpha+1})} \tag{8}$$

We can insert this solution into the above expression for the partition function and obtain

$$Z = \lim_{N\to\infty} \left(\frac{mN}{2\pi\beta\hbar^2} \right)^{\frac{N}{2}} \int d\{\mathbf{q}\} e^{-\beta \sum_{\alpha=1}^{N} \left(\frac{1}{N}V(q_\alpha) + \frac{Nm}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \right)} \tag{9}$$

We now introduce a momentum term by multiplying the above partition function by identity via N Gaussian integrals over momentum with each of them divided by itself for each bead

$$1 = \frac{\int dp_\alpha e^{-\frac{\beta}{N}\frac{p_\alpha^2}{2m_f}}}{\sqrt{\frac{2m_f\pi N}{\beta}}} \tag{10}$$

. We can rewrite the partition function as

$$Z = \lim_{N\to\infty}\left(\frac{\beta}{2\pi m_f N}\right)^{N/2}\left(\frac{mN}{2\pi\beta\hbar^2}\right)^{\frac{N}{2}}\int d\{\mathbf{q}\}d\{\mathbf{p}\}e^{-\beta H_{\mathrm{RP,N}}} \tag{11}$$

.

We now have an expression in which the exponential looks similar to a classical Hamiltonian of the form

$$H_{\mathrm{RP},0} = \frac{1}{N}\sum_{\alpha=1}^{N}\left(\frac{p_\alpha^2}{2m_f} + V(q_\alpha) + \frac{N^2 m}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2\right) \tag{12}$$

which is referred to as the one-dimensional PIMD Hamiltonian. For equilibrium properties, one can choose any quantity as m_f, the fictitious mass. The choice of m_f can be used to improve the sampling efficiency of PIMD.

4.2 MixTS Treatment of 1/N

The choice of where to include the $1/N$ term in Eq. 11 can influence the dynamics of a RPMD simulation. Traditionally, the $1/N$ term was often included in the β, which is why many RPMD simulations are considered "high temperature" simulations. When considered a mixed-time slicing (MixTS) approach for MixPI, we do not have a universal $1/N$ that can be factored out of the Hamiltonian. Therefore, we must considered how to accurately distribute the $1/N$ factor into the expression of the Hamiltonian while maintaining the same Boltzmann distributions and dynamics as obtained from Eq. 11.

First, we defined a Hamiltonian that matches the traditional derivation of the RP Hamiltonian in which a $\frac{1}{N}$ term can be brought outside the Hamiltonian in Eq. 12. A second Hamiltonian, in which this term is factored inside the Hamiltonian is written

$$\begin{aligned} H_{\mathrm{RP,N}} &= \sum_{\alpha=1}^{N}\frac{p_\alpha^2}{2Nm_f} + \frac{1}{N}V(q_\alpha) + \frac{Nm}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \\ Z &= \lim_{N\to\infty}\left(\frac{1}{2\pi\hbar}\right)^N\left(\frac{m}{m_f}\right)^{\frac{N}{2}}\int d\{\mathbf{q}\}d\{\mathbf{p}\}e^{-\beta H_{\mathrm{RP,N}}} \end{aligned} \tag{13}$$

It was previous determined by the Ananth group [3] that the exact treatment of the $\frac{1}{N}$ term is important when generating accurate distributions and dynamics. The expression for a quantum observable $\hat{A}(t)$ using the two different Hamiltonian definitions

$$\langle\hat{A}(t)\rangle_0 = \frac{1}{Z}\int d\{\mathbf{q}\}d\{\mathbf{p}\}e^{-\frac{\beta}{N}H_{\mathrm{RP},0}(\{q(0)\}\{p(0)\})}A(\{q_t\},\{p_t\}) \tag{14}$$

$$\langle \hat{A}(t) \rangle_{\mathrm{N}} = \frac{1}{Z} \int d\{\mathbf{q}\} d\{\mathbf{p}\} e^{-\beta H_{\mathrm{RP,N}}(\{q(0)\}\{p(0)\})} A(\{q_t\}, \{p_t\}) \tag{15}$$

. The equations of motion generated for each of these Hamiltonians

$$\begin{aligned} H_0 : \dot{p} &= -\frac{dV}{dq} \\ \dot{q} &= \frac{p}{m} \\ H_{\mathrm{N}} : \dot{p} &= -\frac{1}{N}\frac{dV}{dq} \\ \dot{q} &= -\frac{p}{Nm} \end{aligned} \tag{16}$$

are different depending on whether the $\frac{1}{N}$ term is in the Hamiltonian or is included in the β term. Since the position and momentum initial distributions sampled for $\langle \hat{A} \rangle_{RP}$ and $\langle \hat{A} \rangle_0$are the same, the overall dynamics for H_0 and H_R will only match if equations of motions are also the same. If we look at the corresponding the Langrangian's

$$\begin{aligned} L_0 &= \sum_{\alpha=1}^{N} \frac{m v_\alpha^2}{2} - V(q_\alpha) - \frac{N^2 m}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \\ L_N &= \sum_{\alpha=1}^{N} \frac{m v_\alpha^2}{N2} - \frac{1}{N}V(q_\alpha) - \frac{Nm}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \end{aligned} \tag{17}$$

and derive the Euler-Langrange EOM

$$\begin{aligned} \frac{\partial L}{\partial q} &= \frac{d}{dt}\frac{\partial L}{\partial \dot{q}} \\ L_0 : -\frac{\partial V}{\partial q_\alpha} - \frac{N^2 m}{\beta^2\hbar^2}(2q_\alpha - q_{\alpha-1} - q_{\alpha+1}) &= m\ddot{q_\alpha} \\ L_N : -\frac{1}{N}\frac{\partial V}{\partial q_\alpha} - \frac{Nm}{\beta^2\hbar^2}(2q_\alpha - q_{\alpha-1} - q_{\alpha+1}) &= \frac{m\ddot{q_\alpha}}{N} \end{aligned} \tag{18}$$

we determine both treatments have the same Euler-Langrage EOM. If we use the Langrage definition for momentum

$$p = \frac{\partial L}{\partial \dot{q}} \tag{19}$$

and convert our Langragians back to Hamiltonians

$$\begin{aligned} H &= p\dot{q} - L \\ H_{\mathrm{RP},0} &= \sum_{\alpha=1}^{N} m\dot{q_\alpha}\dot{q_\alpha} - \left(\sum_{\alpha=1}^{N} \frac{m v_\alpha^2}{2} - \frac{N^2 m}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 - V(q_\alpha) \right) \\ &= \sum_{\alpha=1}^{N} \frac{p_\alpha^2}{2m} + V(q_\alpha) + \frac{N^2 m}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \\ H_{\mathrm{RP,N}} &= \sum_{\alpha=1}^{N} \frac{m\dot{q_\alpha}}{N}\dot{q_\alpha} - \left(\sum_{\alpha=1}^{N} \frac{m v_\alpha^2}{2N} - \frac{Nm}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 - \frac{1}{N}V(q_\alpha) \right) \\ &= \sum_{\alpha=1}^{N} \frac{p_\alpha^2}{2m'} + \frac{1}{N}V(q_\alpha) + \frac{N^2 m'}{2\beta^2\hbar^2}(q_\alpha - q_{\alpha+1})^2 \end{aligned} \tag{20}$$

where $m' = m/N$. The expression for the kinetic piece in H_{RP} has changed from the initial expression by using a different mass. This new Hamiltonian reproduces the same equations of motions as H_0. Therefore, MixPI chooses to use this Hamiltonian in which both the kinetic and harmonic mass terms are scaled ($m' = m/N$).

4.3 MTS-Expansion in Three-Dimensions

Here, we derive the MTS-PIMD Hamiltonian using a similar scheme as in 4.1 . We derive the Hamiltonian for three particles in one-dimension to emphasize how multiple quantization levels interact in the MTS-PIMD Hamiltonian. This Hamiltonian can be extended to handle an arbitrary number of degrees. We start with the same quantum Hamiltonian operator, written for three particles,

$$\begin{aligned}\hat{H} &= \sum_{i=1}^{3} \frac{\hat{p_i}}{2m_i} + \sum_{i=1}^{3} V_1(\hat{q_i}) + \sum_{i=1}^{3}\sum_{j<i} V_2(\hat{q_i},\hat{q_j}) + \sum_{i=1}^{3}\sum_{j<i}\sum_{k<j} V_3(\hat{q_i},\hat{q_j},\hat{q_k}) \\ &= H_1(\hat{q_1},\hat{q_2},\hat{q_3}) + H_2(\hat{q_2},\hat{q_3}) + H_3(\hat{q_3})\end{aligned} \tag{21}$$

where $\hat{p_i}$, $\hat{q_j}$, and m_i are the momentum, position, and mass of the i^{th} quantum particle. $V_1(\hat{q_i})$ is the one-body portion of the potential energy for the i^{th} degree while $V_2(\hat{q_i},\hat{q_j})$ and $V_3(\hat{q_i},\hat{q_j},\hat{q_k})$ are the two-body and three-body potential energy contributions, respectively. We assume that the q_1 degree of freedom experiences the largest quantum mechanical effect, followed by q_2 and then q_3 and define N_1, N_2, and N_3 as the number of time slices along each degre with $N_1 \geq N_2 \geq N_3$e. We split the Hamiltonian by into components based on the the highest quantized variable a term depends on. For example, $H_1(\hat{q_1},\hat{q_2},\hat{q_3}) = T_1(\hat{p_1}) + V(\hat{q_1}) + V(\hat{q_1},\hat{q_2}) + V(\hat{q_1},\hat{q_3}) + V(\hat{q_1},\hat{q_2},\hat{q_3})$. We use the asymmetric Trotter approximation on this partitioned Hamiltonian to first split H_3 from H_{12}

$$Z = \lim_{N_3\to\infty} \int d\{q\}\, \langle q_1 q_2 q_3| \left(e^{-\frac{\beta H_3}{N_3}}\, e^{-\frac{\beta H_{12}}{N_3}}\right)^{N_3} |q_1 q_2 q_3\rangle \tag{22}$$

and then apply the Trotter approximation twice more to $H_{12} = H_1 + H_2$ and, finally, to H_1 to obtain

$$Z = \lim_{N_3,N_2,N_1\to\infty} \int d\{q\}\, \langle q_1 q_2 q_3| \left(e^{-\frac{\beta H_3}{N_3}} \left(e^{-\frac{\beta H_2}{N_3 N_2}}\, e^{-\frac{\beta H_1}{N_3 N_2}}\right)^{N_2}\right)^{N_3} |q_1 q_2 q_3\rangle \tag{23}$$

$$Z = \lim_{N_3,N_2,n_1\to\infty} \int d\{q\}\, \langle q_1 q_2 q_3| \left(e^{-\frac{\beta H_3}{N_3}} \left(e^{-\frac{\beta H_2}{N_2}} \left(e^{-\frac{\beta H_1}{N_1}}\right)^{\frac{N_1}{N_2}}\right)^{\frac{N_2}{N_3}}\right)^{N_3} |q_1 q_2 q_3\rangle \tag{24}$$

We define $\Lambda_i = e^{-\frac{\beta H_i}{N_i}}$ and $\Lambda_{ij} = \Lambda_i \Lambda_j^{\frac{N_j}{N_i}}$ and rewrite the above expression as

$$Z = \lim_{N_3,N_2,N_1\to\infty} \int d\{q\}\, \langle q_1 q_2 q_3| \left(\Lambda_3\, (\Lambda_{21})^{\frac{N_2}{N_3}}\right)^{N_3} |q_1 q_2 q_3\rangle\,. \tag{25}$$

. Following a similar serious of steps as employed in the all-replica derivation in the previous section, we insert N_3 copies of identity with respect to the position

basis q_3 and evaluate Λ_{12} since it is diagonal with respect to q_3.

$$Z = \lim_{N_3,N_2,N_1\to\infty} \int d\{q\} \langle q_1 q_2| \prod_{\alpha=1}^{N_3} \langle q_{3,\alpha}| \Lambda_3 |q_{3,\alpha+1}\rangle \Lambda_{12}(q_{3,\alpha+1})^{\frac{N_2}{N_3}} |q_1 q_2\rangle . \quad (26)$$

We now have the one-particle results with H_3 and can use the result of the previous derivation to obtain

$$\begin{aligned} Z = \lim_{N_3,N_2,N_1\to\infty} & \left(\frac{\beta}{2\pi m_{3,f}}\right)^{1/2} \left(\frac{m_3 N_3}{2\pi\beta\hbar^2}\right)^{\frac{N_3}{2}} \\ & \int d\{q\} e^{-\beta\sum_{\alpha=1}^{N_3}\left(\frac{p_{3,\alpha}^2}{2m_{3,f}} + \frac{1}{N_3}V_1(q_{3,\alpha}) - \frac{N_3 m_3}{2\beta^2\hbar^2}(q_{3,\alpha}-q_{3,\alpha+1})^2\right)} \\ & \langle q_1 q_2| \prod_{\alpha=1}^{N_3} \Lambda_{12}(q_{3,\alpha+1})^{\frac{N_2}{N_3}} |q_1 q_2\rangle . \end{aligned} \quad (27)$$

Next, we introduce N_3 copies of identity with respect to q_2 between each of the $\Lambda(q_{3,\alpha+1})^{N_2/N_3}$ terms

$$\begin{aligned} Z = \lim_{N_3,N_2,N_1\to\infty} & A_3 \int d\{q\} e^{-\beta H_{3,RP}(q_3)} \\ & \langle q_1| \prod_{\alpha=1}^{N_3} \langle q_{2,(\alpha-1)*N_2/N_3+1}| \Lambda_{12}(q_{3,\alpha+1})^{\frac{N_2}{N_3}} |q_{2,\alpha*N_2/N_3+1}\rangle |q_1\rangle . \end{aligned} \quad (28)$$

Since q_2 is more quanitzed than q_3, and needs a greater number of slices then q_3, we introduce additional N_2/N_3 copies of identity between each $\Lambda_{12}(q_{3,\alpha+1})$

$$\begin{aligned} Z = \lim_{N_3,N_2,N_1\to\infty} & A_3 \int d\{q\} e^{-\beta H_{3,RP}(q_3)} \langle q_1| \\ & \prod_{\alpha=1}^{N_3} \prod_{\gamma=1}^{N_2/N_3} \langle q_{2,(\alpha-1)*N_2/N_3+\gamma}| \Lambda_2(q_{3,\alpha+1}) \Lambda_1(q_{3,\alpha+1})^{\frac{N_1}{N_2}} |q_{2,(\alpha-1)*N_2/N_3+\gamma+1}\rangle |q_1\rangle . \end{aligned} \quad (29)$$

$\Lambda_1(q_{3,\alpha+1})$ is diagonal with respect to q_2 and we have obtained the one-dimensional ring-polymer solution for q_2.

$$\begin{aligned} Z = \lim_{N_3,N_2,N_1\to\infty} & A_3 A_2 \int d\{q\} e^{-\beta H_{3,RP}(\{q_3\})} e^{-\beta H_{2,RP}(\{q_2\},\{q_3\})} \\ & \prod_{\alpha=1}^{N_3} \prod_{\gamma=1}^{N_2/N_3} \langle q_1| \Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1})^{\frac{N_1}{N_2}} |q_1\rangle . \end{aligned} \quad (30)$$

We follow the same procedure to introduce N_2 copies of identity between each $\Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1})^{\frac{N_1}{N_2}}$ and N_1/N_2 copies of identity between each $\Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1}$ to obtain the all-replica Hamiltonian for q_1. Once

again, we introduce N_2 copies between each of the copies of $\Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1})^{\frac{N_1}{N_2}}$

$$Z = \lim_{N_3,N_2,N_1\to\infty} A_3 A_2 \int d\{q\} e^{-\beta H_{3,RP}(\{q_3\})} e^{-\beta H_{2,RP}(\{q_2\},\{q_3\})} \prod_{\alpha=1}^{N_3}\prod_{\gamma=1}^{N_2/N_3} \langle q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+1}| \Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1})^{\frac{N_1}{N_2}} |q_{1,((\alpha-1)*N_2/N_3+\gamma)*N_1/N_2+1}\rangle . \tag{31}$$

as well as N_1/N_2 copies of identity in q_1 between each $\Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1}$

$$Z = \lim_{N_3,N_2,N_1\to\infty} A_3 A_2 \int d\{q\} e^{-\beta H_{3,RP}(\{q_3\})} e^{-\beta H_{2,RP}(\{q_2\},\{q_3\})} \prod_{\alpha=1}^{N_3}\prod_{\gamma=1}^{N_2/N_3}\prod_{\lambda=1}^{N_1/N_2} \langle q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}| \Lambda_1(q_{3,\alpha+1}, q_{2,(\alpha-1)*N_2/N_3+\gamma+1}) |q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda+1}\rangle . \tag{32}$$

We obtain an expression for the three-level mixed-time slicing RPMD partition function and Hamiltonian

$$Z = \lim_{N_3,N_2,N_1\to\infty} A_3 A_2 A_1 \int d\{q\} e^{-\beta H_{3,RP}(\{q_3\})} e^{-\beta H_{2,RP}(\{q_2\},\{q_3\})} e^{-\beta H_{1,RP}(\{q_1\},\{q_2\},\{q_3\})}. \tag{33}$$

where

$$\begin{aligned}
H_{3,RP} &= \sum_{\alpha=1}^{N_3} \frac{p_{3,\alpha}^2}{2m_{3,f}} + \frac{1}{N_3} V_1(q_{3,\alpha}) - \frac{N_3 m_3}{2\beta^2\hbar^2}(q_{3,\alpha} - q_{3,\alpha+1})^2 \\
H_{2,RP} &= \sum_{\alpha=1}^{N_3}\sum_{\gamma=1}^{N_2/N_3} \frac{p_{2,(\alpha-1)*N_2/N_3+\gamma}^2}{2m_2} + \frac{1}{N_2} V_1(q_{2,(\alpha-1)*N_2/N_3+\gamma} + \frac{1}{N_2} V_2(q_{2,(\alpha-1)*N_2/N_3+\gamma}, q_{3,\alpha}) \\
&\quad + \frac{N_2 m_2}{2\beta^2\hbar^2}(q_{2,(\alpha-1)*N_2/N_3+\gamma} - q_{2,(\alpha-1)*N_2/N_3+\gamma+1})^2 \\
H_{1,RP} &= \sum_{\alpha=1}^{N_3}\sum_{\gamma=1}^{N_2/N_3}\sum_{\lambda=1}^{N_1/N_2} \frac{p_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}^2}{2m_1} + \frac{1}{N_1} V_1(q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}) \\
&\quad + \frac{1}{N_1} V_2(q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}, q_{2,(\alpha-1)*N_2/N_3+\gamma}) \\
&\quad + \frac{1}{N_1} V_2(q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}, q_{3,\alpha}) \\
&\quad + \frac{1}{N_1} V_3(q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda}, q_{2,(\alpha-1)*N_2/N_3+\gamma}, q_{3,\alpha}) \\
&\quad + \frac{N_3 m_3}{2\beta^2\hbar^2}(q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda} - q_{1,((\alpha-1)*N_2/N_3+\gamma-1)*N_1/N_2+\lambda+1})^2
\end{aligned} \tag{34}$$

5 Funding Acknowledgment

The authors are grateful to Greg Schenter for useful discussions. B.A.J. and C.J.M acknowledge support by the DOE Office of Science, Office of Basic Energy Sciences, Division of Chemical Sciences, Geosciences, and Biosciences, Condensed Phase and Interfacial Molecular Science program, FWP 16249. N.A. and B.A.J. acknowledge support from the U.S. Department of Energy, Office of Basic Energy Sciences, Division of Chemical Sciences, Geosciences and Biosciences under Award DE-FG02-12ER16362 (Nanoporous Materials Genome: Methods and Software to Optimize Gas Storage, Separations, and Catalysis). S.B. acknowledges support from Cornell University, Department of Chemistry and Chemical Biology and the New Frontiers Grant from the College of Arts and Science.

References

[1] AnanthGroup. AnanthGroup/Path-Integral-Atomistic-Driver: MixPI v1.0-beta, May 2024.

[2] Michele Ceriotti, Michele Parrinello, Thomas E. Markland, and David E. Manolopoulos. Efficient stochastic thermostatting of path integral molecular dynamics. *The Journal of Chemical Physics*, 133(12), 2010.

[3] Elliot Eklund. *Path Integral Computational Methods for Condensed Phase Systems: From Classical to Quantum.* Thesis, Cornell University, 2022.

[4] Thomas D. Kühne, Marcella Iannuzzi, Mauro Del Ben, Vladimir V. Rybkin, Patrick Seewald, Frederick Stein, Teodoro Laino, Rustam Z. Khaliullin, Ole Schütt, Florian Schiffmann, Dorothea Golze, Jan Wilhelm, Sergey Chulkov, Mohammad Hossein Bani-Hashemian, Valéry Weber, Urban Borštnik, Mathieu Taillefumier, Alice Shoshana Jakobovits, Alfio Lazzaro, Hans Pabst, Tiziano Müller, Robert Schade, Manuel Guidon, Samuel Andermatt, Nico Holmberg, Gregory K. Schenter, Anna Hehn, Augustin Bussy, Fabian Belleflamme, Gloria Tabacchi, Andreas Glöß, Michael Lass, Iain Bethune, Christopher J. Mundy, Christian Plessl, Matt Watkins, Joost VandeVondele, Matthias Krack, and Jürg Hutter. Cp2k: An electronic structure and molecular dynamics software package - quickstep: Efficient and accurate electronic structure calculations. *The Journal of Chemical Physics*, 152(19), 2020.

[5] Ryan P. Steele, Jill Zwickl, Philip Shushkov, and John C. Tully. Mixed time slicing in path integral simulations. *The Journal of Chemical Physics*, 134(7), 2011.