Dint – version 2.0

Direct Nonadiabatic Trajectories:
A code for non-Born–Oppenheimer molecular dynamics

MANUAL

Ahren W. Jasper
Argonne National Laboratories

Rui Ming Zhang
Tsinghua University and
University of Minnesota

and
Donald G. Truhlar
University of Minnesota

Dint – version 2.0 was finalized December 10, 2020.
This document was last updated December 16, 2020.
Dint is closely related to the trajectory code *ANT*.
For recent versions of *ANT*, see http://comp.chem.umn.edu/ant

LICENSE……………………………………………………………………………….2
REFERENCES…………………………………………………………………………2
I. SUMMARY 4
II. LICENSING 5
III. THE POTENTIAL ENERGY SUBROUTINE 6
IV. INITIAL CONDITIONS 11
V. INTEGRATION 14
VI. NON-BORN–OPPENHEIMER TRAJECTORY METHODS 15
VII.SPECIAL OPTIONS 17
VIII. TERMINATION CONDITIONS 18
IX. INSTALLATION AND COMPILATION 19
X. INPUT FILE 21
XI. OUTPUT FILES 29

XII. REVISION HISTORY.. 33

License:

Publications of results obtained with this program should cite the program. Optionally one should also cite the book chapter that explains many of the methods used in the program.

Reference to the program:

A. W. Jasper, R. M. Zhang, and D. G. Truhlar, *Dint* – version 2.0, 2020, https://comp.chem.umn.edu/dint

Reference to the book chapter:

A. W. Jasper and D. G. Truhlar, Non-Born–Oppenheimer molecular dynamics for conical intersections, avoided crossings, and weak interactions. In *Conical Intersections: Theory, Computation, and Experiment*, edited by W. Domcke, D. R. Yarkony, and H. Koppel (World Scientific, Singapore, 2011), pp. 375–412. DOI:10.1142/9789814313452_0010

Additional references are given in Section V.

The book chapter mentioned above and a list of unpublished errata may be found at (for personal use; do not post or cite this book chapter URL):

https://comp.chem.umn.edu/Truhlar/docs/C82proofs.pdf
https://comp.chem.umn.edu/Truhlar/docs/C82U.pdf

I. SUMMARY

Dint is a parallel Fortran computer program for performing classical and semiclassical trajectory simulations of electronically adiabatic and nonadiabatic processes.

The user may provide an analytic potential energy subroutine that returns the electronic energy (or energies and couplings for multistate systems) and gradients when passed a nuclear geometry. A selection of potential energy subroutines is available at POTLIB-online (http://comp.chem.umn.edu/potlib). *Dint* supports several potential energy subroutine interfaces (i.e., calling protocols), as described in Section II.A.

The user may perform direct dynamics (sometimes called ab initio dynamics or on-the-fly dynamics) simulations using the *Gaussian* or *Molpro* electronic structure packages. These options are described in Section II.B.

Users may specify one or two atom groups (AGs). Each AG is treated as an isolated group of atoms when its initial conditions are determined. Different AGs may be prepared using different initial-condition prescriptions, and several initial-condition prescriptions are supported. The user then specifies a scheme for initially orienting the AGs with respect to each other and/or for specifying the initial collision parameters. Alternatively one can specify only a single initial atom group and do unimolecular dynamics or unimolecular reactions. See Section III for details.

Variable and fixed-step-size integration options are available. See Section IV for details.

Propagation may be carried out adiabatically (i.e., on a single potential energy surface) or – if excited-state surfaces and their couplings are available – electronically nonadiabatically. Several options exist for incorporating electronic transitions into trajectory simulations including the coherent switches with decay-of-mixing method, the self-consistent decay of mixing method, semiclassical Ehrenfest, and several surface hopping methods. See Section V for details.

A limited set of special options is available. For example, the momenta may be zeroed at every step, resulting in a steepest-descent trajectory. As another example, the nuclear kinetic energy may be rescaled at regular intervals to simulate heating. See Section VI for details.

Trajectories are propagated until a termination condition is met. Several options are available for specifying termination, including running trajectories for a fixed simulation time or monitoring bond-breaking events. See Section VII for details.

Dint produces several output files containing information about each trajectory but does not generally perform detailed ensemble average analyses. It is expected that external codes will be used to read and analyze the raw data obtained from a *Dint* simulation. The limited set of final-state analysis options is described in Section VIII.

The code is designed to be as modular as possible with the expectation that some sections will be improved.

II. LICENSING

Dint - version 2.0 is licensed under the Apache License, Version 2.0.

The manual of *Dint* - version 2.0 is licensed under CC-BY-4.0.

Publications of results obtained with the Dint - version 2.0 software should cite the program and/or the article describing the program.

No guarantee is made that this software is bug-free or suitable for specific applications, and no liability is accepted for any limitations in the mathematical methods and algorithms used within. No consulting or maintenance services are guaranteed or implied.

The use of the Dint - version 2.0 implies acceptance of the terms of the licenses.

III.THE POTENTIAL ENERGY SUBROUTINE

III.A. Standard interfaces

Calculations within *Dint* are carried out using unscaled Cartesian coordinates, but it is often convenient to use other sets of coordinates when formulating the potential energy in potential energy subroutines. An interface between the two coordinate systems is therefore required. A series of potential energy subroutine interfaces are provided to handle the coordinate transformations (and those for the derivates) for several system types. In addition to handling coordinate transformations, these interfaces also handle potential energy surface conventions such as the specific ordering of atoms, etc.

The subroutine PREPOT is called once at the beginning of the simulation. This call may be used to set up quantities for subsequent potential calls. The user may supply a dummy routine if this call is not needed. The PREPOT subroutine should be included in the same file as the potential energy subroutine (so it is available to the compiler).

All potential energy subroutine calls go through the subroutine GETPEM, where the various potential energy subroutine interfaces are coded. New potential energy subroutine interfaces may be added by modifying GETPEM. Atomic units are used throughout this section.

The user selects which interface is used during a simulation with the input variable POTFLAG. The following interfaces are currently supported:

POTFLAG = 0: HO-MM-1 interface

This interface is described at POTLIB-online, http://comp.chem.umn.edu/potlib. This is a single surface (adiabatic), homonuclear, molecular mechanics (i.e., variable number of atoms) interface. Subroutine calls with this interface have the general form:

POT(X, Y, Z, E, DEDX, DEDY, DEDZ, NATOM, MAXATOM)

X, Y, Z (input, double precision) One-dimensional arrays containing the Cartesian components of NATOM atoms.

E (output, double precision) The potential energy.

DEDX, DEDY, DEDZ (output, double precision) One-dimensional arrays containing the first derivatives of the energy with respect to the Cartesian coordinates.

NATOM (input, integer) The number of atoms.

MAXATOM (input, integer) Sets the dimensions of the variables X, Y, Z, DEDX, DEDY, and DEDZ. Must be greater than or equal to NATOM.

POTFLAG = 1: 3V-2 interface

This interface is described at POTLIB-online, http://comp.chem.umn.edu/potlib. This interface returns a 2 x 2 diabatic potential energy surface matrix for a triatomic system.

PREPOT is called once, and POT is called when an energy and/or gradient is needed. Subroutine calls with this interface have the general form:

POT(R, E, DE, NVALS, NSURF)

R (input, double precision) A two-dimensional array containing the internuclear bond distances. The first index labels the NVALS different geometries, and the second index labels the three internuclear distances.

E (output, double precision) An array containing the potential energies of surface NSURF at NVALS geometries.

DE (output, double precision) A two-dimensional array of the first derivatives of surface NSURF with respect to the three internuclear distances. The first index labels the three internuclear distances, and the second index labels the NVALS different geometries.

NVALS (input, integer) The energy and derivatives are computed for NVALS different geometries.

NSURF (input, integer) Labels the potential energy surface. For a single-surface potential, NSURF = 1. For a two-state potential, NSURF = 1 and 3 for the two diagonal diabatic potential energy surfaces, respectively, and NSURF = 2 for the diabatic coupling surface.

POTFLAG = 2: HE-MM-1 interface

This interface is described at POTLIB-online, http://t1.chem.umn.edu/potlib. This is a single surface (adiabatic), heteronuclear, molecular mechanics (i.e., variable number of atoms) interface. Subroutine calls with this interface have the general form:

POT(SYMB, X, Y, Z, E, DEDX, DEDY, DEDZ, NATOM, MAXATOM)

SYMB (input, character*2) One-dimensional array containing the atomic symbols of all the atoms.

X, Y, Z (input, double precision) One-dimensional arrays containing the Cartesian components of NATOM atoms.

E (output, double precision) The potential energy.

DEDX, DEDY, DEDZ (output, double precision) One-dimensional arrays containing the first derivatives of the energy with respect to the Cartesian coordinates.

NATOM (input, integer) The number of atoms.

MAXATOM (input, integer) Sets the dimensions of the variables X, Y, Z, DEDX, DEDY, and DEDZ. Must be greater than or equal to NATOM.

POTFLAG = 3: Multistate diabatic HE-MM-1 interface

Subroutine calls with this interface have the general form:

POT(SYMB, X, Y, Z, PEMD, GPEMD, NATOM, MAXATOM, NSURF, MNSURF)

SYMB (input, character*2) One-dimensional array containing the atomic symbols of all the atoms.
NATOM (input, integer) The number of atoms.
MAXATOM (input, integer) Sets the dimensions of the variables X, Y, Z, DEDX, DEDY, and DEDZ. Must be greater than or equal to NATOM.
NSURF (input, integer) The number of electronic surfaces.
MNSURF (input, integer) Sets the dimensions of the variables PEMD and GPEMD.
X, Y, Z (input, double precision) One-dimensional arrays containing the Cartesian components of NATOM atoms. Must be greater than or equal to NSURF.
PEMD (output, double precision) The diabatic potential energy matrix of NSURF x NSURF dimensions.
GPEMD (output, double precision) The gradient of the diabatic potential energy matrix. A four dimensional array where the first two indices are the Cartesian component (1–3) and the atom number (1–NATOM) and the third and fourth indices label the element of the diabatic potential matrix.

Note: Adiabatic energies and nonadiabatic coupling vectors are computed from the diabatic energies and gradients. Using this interface, trajectories may be carried out in either the diabatic or adiabatic representations.

POTFLAG = 4: Multistate adiabatic HE-MM-1 interface

Subroutine calls with this interface have the general form:

POT(SYMB, X, Y, Z, PEMA, GPEMA, DVEC, NATOM, MAXATOM, NSURF, MNSURF)

SYMB (input, character*2) One-dimensional array containing the atomic symbols of all the atoms.
X, Y, Z (input, double precision) One-dimensional arrays containing the Cartesian components of NATOM atoms.
PEMA (output, double precision) A vector with the NSURF adiabatic potential energies.
GPEMA (output, double precision) The gradients of the adiabatic potential energies. A three dimensional array where the first two indices are the Cartesian component (1–3) and the atom number (1–NATOM) and the third index labels the adiabatic state.
DVEC (output, double precision) The nonadiabatic coupling elements. A four dimensional array where the first two indices are the Cartesian component (1–3) and the atom number (1–NATOM) and the third and fourth indices label the adiabatic states.
NATOM (input, integer) The number of atoms.
MAXATOM (input, integer) Sets the dimensions of the variables X, Y, Z, DEDX, DEDY, and DEDZ. Must be greater than or equal to NATOM.
NSURF (input, integer) The number of electronic surfaces.

MNSURF (input, integer) Sets the dimensions of the variables PEMD and GPEMD.

III.B.Direct dynamics

Direct dynamics simulations may be carried out with *Gaussian* or *Molpro*. Either code can be used for electronically adiabatic trajectories. Multiple calls to *Gaussian* can be used to return multiple energy surfaces and their gradients, but the computation of coupling matrix elements or nonadiabatic coupling vectors using *Gaussian* is not enabled. The interface to *Molpro* is more flexible and can return multiple energies and their gradients as well as their coupling matrix elements and/or nonadiabatic coupling vectors.

Calls to *Gaussian* or *Molpro* require POTFLAG = 2 or 3 and a sample subroutine is distributed as dd_pot/dd.f. A few settings near the top of the subroutines in dd.f may need to be edited. In particular, see the variables NC, NP, NS, and TNAME, where

NC = number of separate quantum chemistry calls per geometry

NP(*i*) = 1 for *Gaussian* 03/09/16, 2 for *Molpro* for call *i*

NS(*i*) = number of electronic states read in for call *i*. Must be set to 1 for a *Gaussian* call, i.e., if NP(i)=1. For NP(i)=2 (*Molpro* can return multiple energy and gradient calculations from one input template. Be sure to compute the energy and forces for a state before moving on to the next state.

TNAME(*i*) = string (max 10 characters) containing the name of the template *Gaussian*/*Molpro* input file for call #i. It can be convenient for initial state preparation and for final state analysis to specify different template files for the different reactant and product fragments that may be formed. For example,

```
if ( nat.eq.6 ) then     ! CH4 + H
  TNAME(1) = "qc.1"
elseif ( nat.eq.2 ) then ! H2
  TNAME(1) = "qc.2"
else                     ! CH3
  TNAME(1) = "qc.3"
endif
```

An example of a *Gaussian* template file is

```
# mp2/6-31g* force fchk NoSym Units=bohr scf=(tight,xqc)

TEST

0 1
GEOMETRY
```

The code will replace "`GEOMETRY`" with the appropriate geometry as needed. The flags "`force fchk NoSym Units=bohr`" are required. The formatted checkpoint file (Test.FChk) is read by *Dint* to obtain the calculated energy and gradient. This approach is I/O intensive. For *Molpro*, a sample template for a two-state system is

```
***,title
memory,200,m

geomtyp=xyz
geometry
```

```
nosym
noorient
GEOMETRY
end

basis=vdz

{rhf;wf,21,1,3}
{multi;occ,12;closed,9;wf,21,1,1;wf,21,1,3
cpmcscf,grad,1.1,spin=0.5,accu=1.d-7,record=5101.1
cpmcscf,grad,1.1,spin=1.5,accu=1.d-7,record=5102.1}
molpro_energy=energy(1)
{force
samc,5101.1}
molpro_energy=energy(2)
{force
samc,5102.1}
ftyp,f,d,d,d

---
```

The standard output is read by *Dint* to obtain the calculated energies and gradients. The “geomtyp” must be set to “xyz” (Cartesian). The “nosym” & “noorient” are optional. The GEOMETRY string must appear next in the template; this string will be replaced by: a line containing the number of atoms, a generic title line, and lines specifying the Cartesian coordinates at the geometry of interest. A line containing the word “end” must appear next. The input file must contain a line setting the desired energy to the variable “molpro_energy”. Derivatives must immediately follow, via *Molpro*'s “force” keyword. The program will look for a line containing “GRADIENT FOR STATE” in the standard output to read gradients. For multiple state calls, do both energies and gradients for a state before moving on to the next state. This approach is I/O intensive.

To execute the quantum chemistry package, an additional executable file is needed. These are typically named g.x and m.x for *Gaussian* and *Molpro*, respectively, and are executed via SYSTEM calls in dd.f. They should contain something like:

g.x

```
#!/bin/csh
if (-e Test.FChk) then
mv -f Test.FChk Test.FChk_1
endif
g09 < qc.in > qc.out
```

m.x

```
molpro -o qc.out -s qc.in
```

The executable file g.x or m.x and the dummy input file should be in the directory from which the code is run.

IV. INITIAL CONDITIONS

Dint can deal with unimolecular processes and bimolecular processes. For unimolecular processes (processes in which the initial entity is a single species), there is only one atom group (AG), and the user should set **NMOL** as 1. For bimolecular processes, the user should set two atom groups, i.e., set **NMOL** to 2. Setting up an AG for either unimolecular or bimolecular processes is discussed in Section III.A. For a bimolecular process, the user should set the relative orientation of the two AGs, and this is discussed in Section III.B.

IV.A. Methods for preparing each atom group (AG)

This section describes the general capabilities for preparing initial conditions. For details of the input file parameters, see Section X.

Several methods for preparing the initial conditions (i.e., the initial coordinates and momenta) for each AG are available. The input variables INITx and INITp specify the scheme for selecting the initial coordinates and momenta, respectively. For calculations with more than one AG, each AG may be prepared using different scheme. Some combinations of INITx and INITp are not allowed. Sometimes INITx also determines the initial momenta, and INITp is ignored. The initial rotational state can be controlled using the input variable INITj.

In the code, some quantities that are required for computing the initial conditions that are the same for all trajectories are precomputed by calling PREMOL. The specific initial conditions for each trajectory are determined by calling INITMOL. Throughout this section, comments in brackets denote the names of the variables used in the code.

INITx = 0: Fixed structure, e.g., crystal structure

The user supplies Cartesian coordinates in Å [XX0] for each atom. The initial coordinates are the same for every trajectory when this option is selected.

INITx = 1: Random, approximately spherical clusters

1. The user supplies the distance R_{nn} [RDUM] in Å.
2. The following quantities are determined: $R_{\text{rad}} = R_{nn} N_{\text{atom}}^{1/3}$, $R_{\text{min}} = 0.7 R_{nn}$, and $R_{\text{max}} = 1.3 R_{nn}$ [RRAN(1–3), respectively], where N_{atom} is the number of atoms in the AG.
[Steps 3–5 are done in RANCLU.]
3. A trial set of Cartesian coordinates X_{ij} (i = x,y,z) is generated for atom j [XX] randomly inside a sphere of radius R_{rad}.
4. The set is rejected if the minimum atom-atom distance between the new atom j and all previously placed atoms is less than R_{min} or greater than R_{max}.
5. Steps 3 and 4 are repeated until N_{atom} atomic locations are determined.

Note: This method was used to prepare aluminum clusters of various sizes. See, Lloyd and Johnston, Chem. Phys. 236, 107 (1998).

INITx = 2: Quasiclassical state-selected normal modes using the harmonic approximation

Note: When INITx = 2, the scheme determines both the coordinates and momenta simultaneously, and INITp is ignored. The potential energy surface used for this analysis corresponds to the potential energy surface indicated by REPFLAG and NSURF0.

INITx = 3: Atom-diatom initial conditions

When INITx = 3, the scheme determines both the coordinates and momenta simultaneously, and INITp is ignored. This scheme works for triatomic systems only.
This scheme is based on the methods used by the computer code NAT; see its documentation for details. Briefly, the user specifies an initial molecular arrangement, a quantized rovibrational state for the diatomic fragment, a total energy, and an initial atom-diatom separation. Precomputed information is obtained for all trajectories in PREATOMDIATOM. The initial conditions for each trajectory are computed in ATOMDIATOM.

INITx = 4: Not available

INITx = 5: Thermally distributed, state-selected quantized normal modes using the harmonic approximation or quasiclassical microcanonical

For each trajectory, n_i quanta are assigned (in the subroutine RANSTATE) randomly from a thermal distribution at some temperature T to each mode i according to the Boltzmann weights

$$P(n_i) = \exp(-E_{\mathrm{HO},i}/k_\mathrm{B}T)/\Sigma_j \exp(-E_{\mathrm{HO},j}/k_\mathrm{B}T)$$

where $E_{\mathrm{HO},i}$ is the harmonic energy for mode i with n_i quanta. Currently, j is summed to 100 when normalizing $P(n_i)$. This scheme is available for wells only, i.e., this scheme is not available for transition states.

Alternatively, an approximate microcanonical ensemble can be generated.

When INITx = 5, the scheme determines both the coordinates and momenta simultaneously, and INITp is ignored.

INITx = 6: Randomly select the initial geometry and momenta from a list of geometries stored in separate files

The user may then scale the total energy and/or add/remove rotational energy. When INITx = 6, the scheme determines both the coordinates and momenta simultaneously, and INITp is ignored. After the initial conditions are read, the user may scale the total energy and/or add/remove rotational energy.

INITp = 0: Random thermal distribution or random fixed-total-energy distribution

INITp = 1: Zero initial momenta

The initial momenta are set to zero.

INITp = 2: Read in initial momenta

The user supplies the Cartesian momenta (in a.u.) [PP0] for each atom. The momenta for every trajectory are the same when this option is selected.

INITj=0: Angular momentum is not adjusted

INITj=1: Randomly select the initial rotational state J from a thermal or even distribution over some J range

The initial vibrational energy be subsequently scaled to some *J*-dependent energy.

IV.B. Relative orientation of the AGs

After the coordinates and momenta are assigned for each AG using the methods described in III.A, the center of mass is placed at the origin, and center-of-mass motion is removed.

The AGs are oriented with respect to each other according to the control flag INITm. The following options are currently available. Note: INITm is not required for calculations with only one AG.

INITm = 0: Cartesian coordinates

The user specifies an initial set of Cartesian coordinates (Å) and momenta (a.u.) for the center of mass locations and motions for each AG.

INITm = 1: Quasiclassical bimolecular collisions

This options works for simulations with two AGs only. The user specifies the initial separation (in Å) [REL0QC], a range of impact parameters (in Å) [BMINQC to BMAXQC], and an initial relative energy (in eV) [ERELQC].

V. INTEGRATION

All $3N_{\text{atom}}$ Cartesian coordinates and momenta for the nuclei, the real and imaginary parts of the electronic wave function coefficients (for electronically nonadiabatic simulations), and other quantities are integrated in the subroutine TAKESTEP. The integration algorithm calls DERIVS whenever a set of time derivatives is needed. DERIVS packs the nuclear coordinates and electronic state coefficients into a single array Y, calls GETGRAD to obtain the information necessary to compute the time derivatives of Y (called DY) and returns DY. The integrator then uses one or more values of Y and DY to advance the system in time.

Two integration schemes are available, and the desired integration scheme is selected by the input variable INTFLAG.

INTFLAG = 0: The Bulirsch-Stoer adaptive step size method

See *Numerical Recipes* for details. The user supplies an initial step size (in fs) and an algorithmic tolerance (in atomic units). For analytic potentials, a value of 10^{-7} for the integrator tolerance is recommended for typical cases. For direct dynamics runs, looser tolerances are typically required (e.g., 10^{-4}). An initial step size of 1 fs is typical. For production runs, the tolerance should be optimized with respect both for CPU time and for conservation of energy, angular momentum, and electronic probability.

INTFLAG = 1: The 4th order Runge-Kutta method fixed-step-size method

See *Numerical Recipes* for details. The user supplies a step size (in fs). Step sizes of 0.25–1 fs are typical.

VI.NON-BORN–OPPENHEIMER TRAJECTORY METHODS

For systems where multiple potential energy surfaces and their couplings are available, non-Born–Oppenheimer (non-BO) trajectory methods may be used. All of the non-BO methods use the electronic wave function, which is propagated along the classical trajectory as the solution to the classical path electronic Schrödinger equation.

Either the adiabatic or diabatic electronic representation may be used. The flag REPFLAG is used to select which set of potential energy surfaces (adiabatic or diabatic) to use for setting up the initial conditions and for propagating the nuclear and electronic variables.

Note: The classical path equations for the electronic state populations neglect the "kinetic energy" nonadiabatic coupling term. In this formulation, the adiabatic and diabatic representations are eaxactly equivalent (for a given nuclear trajectory) only for the case of two electronic states.

REPFLAG = 0: Adiabatic representation

REPFLAG = 1: Diabatic representation

The specific non-BO algorithm is selected using the input variable METHFLAG.

METHFLAG = 0: Propagation on a single surface

METHFLAG = 1: Tully's fewest switches (FS) method

The FS method is a version of trajectory surface hopping. It is described in J. C. Tully, J. Chem. Phys. **93**, 1061 (1990). In this version, frustrated hops are ignored. Set TFLAG(4) = 1 to turn on stochastic decoherence (SD); this yields the FS/SD method, whose elements are described in "Non-Born–Oppenheimer Molecular Dynamics of Na…FH Photodissociation," A. W. Jasper and D. G. Truhlar, Journal of Chemical Physics **127**, 194306/1-7 (2007). doi.org/10.1063/1.2798763

METHFLAG = 2: Semiclassical Ehrenfest (SE) method

Note: The final state analysis for the vibrational action does not work properly for this method. The code assumes that the system is in a single electronic state, which is not, in general, the case for this method. Only the vibrational action is affected.
See "Coherent Switching with Decay of Mixing: An Improved Treatment of Electronic Coherence for Non-Born-Oppenheimer Trajectories," C. Zhu, S. Nangia, A. W. Jasper, and D. G. Truhlar, Journal of Chemical Physics **121**, 7658-7670 (2004). doi.org/10.1063/1.1793991 for a discussion of the SE method.

METHFLAG = 3: Self-consistent decay of mixing (SCDM) method

The decay lifetime is computed with $E_0 = 0.1\ E_\mathrm{h}$ and $C = 1$.
See "Coherent Switching with Decay of Mixing: An Improved Treatment of Electronic Coherence for Non-Born-Oppenheimer Trajectories," C. Zhu, S. Nangia, A. W. Jasper, and D. G. Truhlar, Journal of Chemical Physics **121**, 7658-7670 (2004). doi.org/10.1063/1.1793991 for a discussion of the SCDM method.

METHFLAG = 4: Coherent switches with decay of mixing (CSDM) method

See “Coherent Switching with Decay of Mixing: An Improved Treatment of Electronic Coherence for Non-Born-Oppenheimer Trajectories,” C. Zhu, S. Nangia, A. W. Jasper, and D. G. Truhlar, Journal of Chemical Physics **121**, 7658-7670 (2004). doi.org/10.1063/1.1793991 for the description of the CSDM method.

METHFLAG = 5: Fewest switches with time uncertainty (FSTU) method

See “Fewest Switches with Time Uncertainty: A Modified Trajectory Surface Hopping Algorithm with Better Accuracy for Classically Forbidden Electronic Transitions,” A. W. Jasper, S. N. Stechmann, and D. G. Truhlar, Journal of Chemical Physics **116**, 5424-5431 (2002). doi.org/10.1063/1.1453404 [Erratum: **117**, 10427 (2002). doi.org/10.1063/1.1519005] for a discussion of the FSTU method.

Frustrated hops are treated using the gradV method. See “Improved Treatment of Momentum at Classically Forbidden Electronic Transitions in Trajectory Surface Hopping Calculations,” A. W. Jasper and D. G. Truhlar, Chemical Physics Letters **369**, 60-67 (2003). for a discussion of the gradV method.

Set TFLAG(4) = 1 to turn on stochastic decoherence (SD) to use the FSTU/SD method., which is described in “Non-Born–Oppenheimer Molecular Dynamics of Na…FH Photodissociation,” A. W. Jasper and D. G. Truhlar, Journal of Chemical Physics **127**, 194306/1-7 (2007). doi.org/10.1063/1.2798763

VII.SPECIAL OPTIONS

Special options are indicated using the control flag array TFLAG. The following options are supported.

Momentum related options

TFLAG(1) = 0 No special options

= 1 Steepest-descent minimization
The momenta are zeroed at every step.

= 2 Temperature ramping
After each interval of RAMPTIME fs, the momenta are scaled by a factor of RAMPFACT. The momenta are scaled NRAMP times before the trajectory ends. When this option is selected, unit 40 is written, which records information between rescalings. The time is reset at each temperature rescaling.

= 3 Andersen thermostat
With some frequency ANDERSEN_FREQ, the momenta of every atom is replaced with a momenta sampled from a random thermal distribution at the temperature ANDERSEN_TEMP. See H. C. Andersen, *J. Chem. Phys.* **72**, 2348 (1980). This option can also be used to periodically resample the momenta with the total energy conserved.

Restart trajectory options

TFLAG(2) = 0 No special options

= 1 Restart trajectory option
A list of trajectory indices may be read and propagated. Unique series of random numbers are assigned based on the seed [RANSEED] and the trajectory number for each trajectory, such that when a trajectory is restarted out of order it will use the same series of random numbers as it would have used if the entire set of trajectories had been run.

Initial electronic state preparation options

TFLAG(3) = 0 No special options

= 1 Photoexcite trajectory before propagation to some electronic state after preparing the initial conditions is some other electronic state

Electronic coherence options

TFLAG(4) = 0 No special options

= 1 Use the stochastic decoherence (SD) method. This method can be used with surface hopping methods. See *J. Chem. Phys.* 127, 194306 (2007) for details. The current implementation of SD is given as eq 3 of: *J. Chem. Phys.* 130, 234303 (2009).

VIII.TERMINATION CONDITIONS

The termination condition is selected using the control variable TERMFLAG.
Termination is monitored in the subroutine DRIVER.

TERMFLAG = 0: Fixed number of steps

Each trajectory propagates for T_NSTEP integration steps, which is specified by the user.

TERMFLAG = 1: Fixed simulation time

Each trajectory propagates for T_STIME fs, which is specified by the user. The user also specifies a maximum number of steps to catch pathological trajectories.

TERMFLAG = 2: Gradient convergence

Each trajectory propagates until the magnitude of the gradient is less than T_GRADMAG eV/A, which is specified by the user. The user also specifies a maximum number of steps to catch pathological trajectories.

TERMFLAG = 3: Monitor association or dissociation

The user specifies pairs of atom types (e.g., C and H). As the trajectory propagates, the code monitors the reaction distance R between all pairs of atoms corresponding to the specified atom types (e.g., all C and H atom pairs). For dissociation, once R is less than the user specified distance T_R, the code considers the collision even to have started. The code then monitors R until R exceeds T_R, i.e., until a bond has broken. For association, the code monitors R until it is less than T_R, where it considers association to have occurred. One may specify more than one set of atom pairs to monitor, and each pair is assigned a unique outcome identifier in the output so that product branching can be computed. The user may also assign different atom labels to equivalent atom types (i.e., label some hydrogens H1 and others H2) for more control over dissociation outcomes. The user also specifies a maximum number of steps to halt pathological trajectories.

IX. INSTALLATION AND COMPILATION

Dint is distributed as a tar file, which may be untarred by executing

```
tar -xvf dint.tar
```

The dint/ distribution contains the following subdirectories:

exe/	Initially empty; fills with executables as they are compiled
pot/	Contains sample potential energy subroutines
dd_pot/	Contains sample potential energy subroutines for direct dynamics
runs/	Initially empty; provided for the user
sprng/	SPRNG random number generator routines and libraries
src/	*Dint* source code and makefile
NOTICE.txt	Contains the summary of *Dint* – version 2.0 licensing
LICENSE.txt	Contains the full text of Apache License 2.0

The random number generator is compiled by executing

```
make
```

in the sprng/SRC/ subdirectory. The user may need to modify the file sprng/make.CHOICES. See the SPRNG documentation for details. This step needs to be done once for each installation of *Dint*.

The user must place the potential energy subroutines that they wish to use in the subdirectory pot/. Let's assume that the name of the potential energy subroutine is SAMPLEPOT.f. To compile *Dint*, a MPI compiler is needed. Go to the src/ subdirectory and execute

```
make POT=SAMPLEPOT
```

This will generate an executable named dint-SAMPLEPOT.x.opt. The executable, once compiled, is stored in the exe/ subdirectory.

The code is executed using (from the runs/ subdirectory in this example)

```
mpirun -np 4 ../exe/dint-SAMPLEPOT.x.opt
```

The code will read a properly formatted input file named "input" and write general information to standard output named "output" as well as several additional output files. When the code running, each processor will create its own work directory named according to the rank of the processor, e.g. "000/", "001/" under the temporary directory. Each processor will make copies of the required run time files from the directory from which the job was submitted into its work directory, evenly divide the number of trajectories into *nproc* subsets, and run its subset of trajectories. The three important directory locations mentioned above are:

Job submission directory: The directory from which the job is submitted. All of the required input files should be placed here.

Temporary directory: This directory is set by the system environment variable `DINT_TMP_DIR`. The user can specify it in bash via `export DINT_TMP_DIR=/home/work`. If not set, the temporary directory will be sest to the job submission directory. The system environment variable `DINT_DELETE_TMP` controls whether to clean the temporary directory after the calculation. If you want the temporary directory to be cleaned, set `export DINT_TMP_DIR=yes` .

Work directory: A processor-specific directory located under the temporary directory. The code will create new subdirectories named according to the rank of the processor, e.g., “000/”, “001/” under the temporary directory and set the path of each as the work directory. Before the calculation, each processor will change the directory from the job submission directory to the work directory and copy all the files under the job submission directory into the work directory. After the calculation, the work directory will be moved from temporary directory into the job submission directory.

During the calculation, output files of a processor will located in their own work directory. After all trajectories finish, the rank 0 processor will gather the separated output files and re-print them as one under the job submission directory .

X. INPUT FILE

The input file consists of a series of records, where a record is a single line of input parameters. A description of each record follows. If the record is labeled by an index such as *n.m*, then it is read only if a parameter read in by Record *n* is set to the value *m*. Some records are read more than once (e.g., once for each atom).

Key: dp = double precision; int = integer; char**n* = character**n*; logical = logical

Record 1: POTFLAG

POTFLAG (int)   Selects the potential interface
If POTFLAG = 0, the HO-MM-1 interface is used.
= 1, the 3V-2 interface is used.
= 2, the HE-MM-1 interface is used.
= 3, the multistate diabatic HE-MM-1 interface is used.
= 4, the multistate adiabatic HE-MM-1 interface is used.

Record 2: NSURF0, NSURFT, METHFLAG,REPFLAG

NSURF0 (int)   Initial electronic surface for all trajectories
NSURFT (int)   Total number of electronic surfaces
METHFLAG (int)   Selects which method to use for nonadiabatic dynamics
If METHFLAG = 0, single surface propagation
= 1, Tully's fewest switches (TFS) non-BO method is used
= 2, the semiclassical Ehrenfest (SE) non-BO method is used
= 3, the self-consistent decay of mixing (SCDM) non-BO method is used.
= 4, the coherent switches with decay of mixing (CSDM) non-BO method is used.
= 5, the fewest switches with time uncertainty (FSTUgradV) non-BO method is used.
REPFLAG (int)   Representation flag
If REPFLAG = 0, the adiabatic representation is used.
= 1, the diabatic representation is used.

Record 3: INTFLAG

INTFALG (int)   Selects the integrator
If INTFLAG = 0, the Bulirsch-Stoer integrator is used, read Record 3.0.
= 1, the forth-order Runge-Kutta integrator is used, read Record 3.1.

Record 3.0: HSTEP0, EPS, NPRINT

HSTEP0 (dp) Initial time step in fs
EPS (dp)   Integrator tolerance in a.u.
NPRINT (int) Print information every NPRINT steps

Record 3.1: HSTEP, NPRINT

HSTEP (dp) Time step in fs
NPRINT (int) Print information every NPRINT steps

Record 4: RANSEED

RANSEED (int) Nine-digit integer used to initialize the SPRNG random number generator

Record 5: NTRAJ, TFLAG(1),TFLAG(2),TFLAG(3),TFLAG(4)

NTRAJ Number of trajectories
TFLAG(1) (int) Momentum related options
If TFLAG(1) = 0, no special options are employed.
= 1, the nuclear momenta are set to zero at every step, i.e., steepest-descent minimization.
= 2, temperature rescaling. Read Record 5.1.
= 3, Andersen thermostat. Read Record 5.2.
TFLAG(2) (int) Trajectory restart options.
If TFLAG(2) = 0, no restart option is employed.
= 1, a list of NTRAJ trajectory indices is read and restarted. Read Record 5.3.
TFLAG(3) (int) Initial photoexcitation options
If TFLAG(3) = 0, no special options are employed.
= 1, photoexcite each trajectory from electronic state NSURF0 to some other state before propagation. Read Record 5.4.
TFLAG(4) (int) Stochastic decoherence
If TFLAG(4) = 0, no special options are employed.
= 1, The SD method is used. This method is intended for use with surface hopping methods.

Record 5.1: RAMPTIME, RAMPFACT, NRAMP

RAMPTIME (dp) Time interval in fs between temperature rescalings.
RAMPFACT (dp) Temperature rescaling factor.
NRAMP (int) Number of rescalings for each trajectory.

Record 5.2: ANDERSEN_TEMP,ANDERSEN_FREQ, SCANDTH

ANDERSEN_TEMP (dp) Bath temperature in K
ANDERSEN_FREQ (dp) Frequency of “collisions” in 1/fs
SCANDTH (dp) If SCANDTH is less than 0, its value is not used and a thermal bath at temperature ANDERSEN_TEMP is simulated. If SCANDTH is greater than zero, the momenta will be rescaled after each collision such that the total energy is conserved with a value equal to SCANDTH plus the initial potential energy.

Record 5.3: TRAJLIST

TRAJLIST (int) A list of NTRAJ trajectory indices for restart

<u>Record 5.4</u>: NTARGET,EPHOTON,WPHOTON

NTARGET (int) Target electronic state

EPHOTON (dp) Photon energy in eV

WPHOTON (dp) Photon energy width in eV

Initial coordinates and momenta will be assigned for each trajectory in state NSURF0 according to INITx and INITp. Before propagation, this option will attempt to photoexcite the trajectory from state NSURF0 to NTARGET. Photoexcitation is successful if the energy gap between states NSURF0 and NTARGET is equal to EPHOTON +/- WPHOTON.

<u>Record 6</u>: NMOL, EZERO

NMOL (int) Number of atom groups (AGs)

EZERO (dp) Zero of energy (eV). This value is subtracted from the energies returned from the potential energy subroutine for every surface for supermolecule calculations, i.e., for calculations involving all atom groups.

Note: Records 7–10 are read NMOL times (once for each AG).

<u>Record 7</u>: NATOM, INITx, INITp, INITj, EZEROIM

NATOM (int) Number of atoms in this AG

INITx (int) Scheme for preparing the initial coordinates for this AG

If INITx = 0, read initial coordinates from the standard input. Read Record 8.0.

= 1, random spherical coordinates. Read Records 8.1.*.

= 2, quasiclassical state-selected coordinates and momenta. Read Records 8.2.*.

= 3, atom-diatom quasiclassical coordinates and momenta. Read Records 8.3.*.

= 4, not available.

= 5, thermal, vibrational quasiclassical state-selected coordinates and momenta. Read Records 8.5.*.

= 6, sample initial coordinates and momenta from separate files. Read Records 8.6.*.

INITp (int) Scheme for preparing the initial momenta for this AG.

If INITp = –1, initial momenta is determined by INITx.

= 0, random thermal distribution of momenta. Read Record 9.0.

= 1, initial momenta set to zero.

= 2, read in initial momenta. Read Records 9.2.*.

INITj (int) Scheme for preparing the initial total angular momentum for this AG.

If INITj = 0, angular momentum is not adjusted.

= 1, thermal distribution over *J* range. Read Records 10.1.*.

EZEROIM (dp) Zero of energy for this atom group (eV). This value is subtracted from the energies returned from the potential energy subroutine for calculations involving the isolated atom groups.

Record 8.INITx

Record 8.0: SYMBOL, MASS, XX0
Note: Record 8.0 is read NATOM times.
SYMBOL (char*2) Atomic symbol for this atom
MM (dp) Mass of this atom in amu
XX0 (dp) Initial x,y,z coordinates for this atom in Å

Record 8.1.1: RDUM
RDUM (dp) Random cluster packing distance in Å

Record 8.1.2: SYMBOL, MM
Note: Record 8.1.2 is read NATOM times.
SYMBOL (char*2) Atomic symbol for this atom
MM (dp) Mass of this atom in amu

Record 8.2.1: LREADHESS, NMTYPE, NMQN
LREADHESS (logical) If true, read Hessian from unit 70. If false, compute Hessian numerically from gradients.
NMTYPE (int) Set to 0 to indicate a well and 1 to indicate a saddle point.
NMQN (dp) An array of length 3*NATOM – 6 to indicate the initial quantum number for each mode. Typically these will be integers, but they can be nonintegers. Modes are ordered from largest frequency to the lowest frequency. For saddle points, the last number in the array is the energy to be assigned to the unbound mode in eV.

Record 8.2.2: SYMBOL, MM, XX0
Note: Record 8.2.2 is read NATOM times.
SYMBOL (char*2) Atomic symbol for this atom.
MM (dp) Mass of this atom in amu.
XX0 (dp) Initial x,y,z coordinates of this atom in A.

Record 8.3.1: SYMBOL, MM
Note: Record 8.3.1 is read NATOM times.
SYMBOL (char*2) Atomic symbol for this atom
MM (dp) Mass of this atom in amu

Record 8.3.2: ESCATAD,VVAD,JJAD,RRAD,ARRAD
ESCATAD (dp) Total energy in eV
VVAD (int) Initial vibrational quantum number for the diatom
JJAD (int) Initial rotational quantum number for the diatom
RRAD (dp) Initial atom-diatom separation in A

ARRAD (int) Initial molecular arrangement,
ARRAD = 1, AB + C
= 2, BC + A
= 3, AC + B.

Record 8.5.1: LREADHESS, NMTYPE, TEMP0IM, SCALE0IM, LEMS

LREADHESS (logical) = TRUE, read the Hessian from unit 70
= FALSE, compute Hessian numerically from gradients
NMTYPE (int) Set to 0 to indicate a well and 1 to indicate a saddle point.
TEMP0IM (dp) ≥ 0, temperature for the random thermal distribution in K
< 0, use SCALE0IM to generate a microcanonical distribution
SCALE0IM (dp) Target HARMONIC total energy for the microcanonical distribution in eV relative to the well; used only if TEMP0IM < 0.
LEMS (logical) If true, use Efficient Microcanonical Sampling (EMS). Read Record 8.5.2.

Record 8.5.2: EMICR,NINC,NBREA,NEMSTOT

EMICR (dp) Target energy for EMS in eV.
NINC (int) The EMS algorithm takes NINC initial incubation steps before choosing the coordinates and momenta of the first trajectory.
NBREA (int) The EMS algorithm takes an additional NBREA steps on subsequent trajectories.
NEMSTOT (int) Maximum number of EMS points allowed.

Record 8.5.3: SYMBOL, MM, XX0

Note: Record 8.2.2 is read NATOM times.
SYMBOL (char*2) Atomic symbol for this atom
MM (dp) Mass of this atom in amu
XX0 (dp) Initial x,y,z coordinates of this atom in Å

Record 8.6.1: SAMPTOT, LBINSAMP, SAMPFILEXX, SAMPFILEPP, LEMS

SAMPTOT (int) Total number of samples to randomly sample from
LBINSAMP (logical) TRUE if SAMPFILEXX and SAMPFILEPP are binary.
SAMPFILEXX (char*10) File from which to sample coordinates. This file has the formatting of the output file FORT.80 (FORT.86 if binary).
SAMPFILEPP (char*10) File from which to sample momenta. This file has the formatting of the output file FORT.81 (FORT.87 if binary). If LEMS is true, this file will not be read, so it can be any non-blank string.
LEMS (logical) If true, use Efficient Microcanonical Sampling (EMS). Read Record 8.6.2.

Record 8.6.2: EMICR,NINC,NBREA,NEMSTOT,LNOROT

EMICR (dp) Target energy for EMS in eV.
NINC (int) The EMS algorithm takes NINC initial incubation steps before choosing the coordinates and momenta of the first trajectory.
NBREA (int) The EMS algorithm takes an additional NBREA steps on subsequent trajectories.
NEMSTOT (int) Maximum number of EMS points allowed.
LNOROT (logical) = TRUE, the angular momentum will be removed and EMICR

indicates the vibrational energy. This option generates a microcanonical ensemble for the vibrational modes.

= FALSE, the angular momentum will not be removed and EMICR indicates the total internal (vibrational + rotational) energy. This option generates a microcanonical ensemble for the vibrational and rotational modes.

Record 8.6.3: SYMBOL, MM

Note: Record 8.6.3 is read NATOM times.

SYMBOL (char*2)   Atomic symbol for this atom.

MM (dp)   Mass of this atom in amu.

Record 9.INITp

Record 9.0: TEMP0IM, ESCALE0IM

TEMP0IM (dp)   Target temperature for the random thermal distribution for this AG in K.

ESCALE0IM (dp)   Target kinetic energy for the random distribution for this AG in eV. If (ESCALE0IM < 0), ESCALE0IM is not used.

Record 9.2: PSYMB,PMASS,PP0

Note: Record 8.2 is read NATOM times.

PSYMB (character*2) NOT USED! Atomic symbols read in Record 8.

PMMASS (double precision) NOT USED! Masses are read in Record 8.

XX0 (double precision) Initial x,y,z momenta of atom *i* in a.u. = $(E_\mathrm{h}\ \mathrm{u}_i)^{1/2}$, where u_i is the mass of atom *i* in atomic units.

Record 10.INITj

Record 10.1.1: SAMPTARG, LETOT, SAMPJMIN, SAMPJMAX, SAMPJTEMP1, SAMPJTEMP2, SAMPBROT1, SAMPBROT2

SAMPTARG (dp) Scaling factor

If SAMPTARG = 0 do not scale sampled momenta

If SAMPTARG < 0, scale vibrational momenta to a *J*-dependent energy using the rule specified by Record 10.1.2.

If SAMPTARG > 0, scale the vibrational momenta such that the total energy equals SAMPTARG in eV.

LETOT (logical)   If true, the total energy is scaled to SAMPTARG. If false, the vibrational energy is scaled to SAMPTARG. This energy includes the energy associated with the unconserved quantum number K.

SAMPJMIN,SAMPJMAX,SAMPJTEMP,SAMPJBROT1,SAMPJBROT2 (dp)

The initial value of J is selected from SAMPJMIN to SAMPJMAX (in units of hbar) and from a thermal distribution if SAMPEJTEMP1 > 0 or from a uniform distribution if SAMPJTEMP1 < 0.

If SAMPJTEMP1 > 0, the temperature used to define the distribution is sampled uniformly from SAMPTEMP1 to SAMPTEMP2 and SAMPJBROT1 and SAMPJBROT2 are the rotational constants to be used, given in cm-1. A symmetric top is assumed, and SAMPBROT1 is the symmetrized doubly degenerate rotational constant.

If SAMPBROT2 < 0, the equation for a spherical top is used with SAMPBROT1.Note: A proper thermal distribution is obtained only if SAMPJMIN = 0 and SAMPJMAX is large enough to converge the tail of the distribution.

Record 10.1.2: EJSC(1),EJSC(2),EJSC(3),EJSC(4)

Read if SAMPTARG < 0.

EJSC (dp) Parameters used in the subroutine EJSCALE to define the initial vibrational energy as a function of J.
EVIB=(EJSC(1)+EJSC(2) J+EJSC(3) J^2)*EJSC(4) eV.

Note: If NMOL is greater than 1, read Record 11.

Record 11: INITm, LDOFRAG

INITm (int) Selects the initial orientations of two atom groups.
= 0, user supplied center of mass coordinates and momenta. Read Records 11.0.*.
= 1, quasiclassical center of mass coordinates and momenta for two atom groups. Read Record 11.1.

LDOFRAG (logical) If TRUE, the potential energies of initial and final fragments will be computed. If FALSE, only supermolecule calls to the potential subroutine will be made.

Record 11.0.1: COMXX(3)

COMXX (double precision) Initial coordinates for the center of mass of this AG in A.

Record 11.0.2: COMPP(3)

COMPP (double precision) Initial momenta for the center of mass of this AG in a.u.

Record 11.1: REL0QC, TEMPQC, BMINQC, BMAXQC

REL0QC (dp) Initial relative center of mass separation in A.

TEMPQC (dp) > 0, the initial relative kinetic energy is selected from a distribution at TEMPQC in K.
= 0, the randomly generated temperature used to sample the rotational state in Record 10.1.1 is used to sample the relative kinetic energy.
< 0, a constant relative energy is used with a value equal to the absolute value of TEMPQC in eV.

BMINQC and BMAXQC (dp) The initial impact parameter is selected quasiclassically from BMINQC to BMAXQC in A. The impact parameter is sampled evenly from 0–$b_{\max}$, which is not the correct distribution. The present sampling scheme must be corrected for when averaging. (The correct scheme is to sample b^2 from 0–$b^2_{\max}$.)

Record 12: TERMFLAG, T_NSTEP

TERMFLAG (int) Selects the termination condition

If TERMFLAG = 0, run for T_NSTEP steps.
= 1, run for T_STIME fs, read Record 12.1.
= 2, converge the gradient, read Record 12.2.
= 3, monitor dissociation, read Records 12.3.*.

= 4, terminate at the first minimum in the magnitude of the nonadiabatic coupling vector.

T_NSTEP (int) Maximum number of steps per trajectory. If < 0, no maximum is enforced.

Record 12.1: T_STIME

T_STIME (dp) Trajectories will propagate for T_STIME fs.

Record 12.2: T_GRADMAG

T_GRADMAG (dp) The trajectory terminates when the magnitude of the gradient is less than T_GRADMAG in eV/A.

Record 12.3.0: T_NOUTCOME, LCHKDIS

T_NOUTCOME (int) Number of dissociation pairs to monitor.

LCHKDIS(logical) If True, the termination condition will only be checked when the monitored distances are less than 0.95 of the delimiter distance set by *T_R*. This prevents the trajectories from "dissociating" before the collision.

Record 12.3.1: T_SYMB(1), T_SYMB(2), T_R

Note: Record 11.3.1 is read T_NOUTCOME times.

T_SYMB (char*2) The code will monitor the maximum T_SYMB(1)–T_SYMB(2) bond distance, where T_SYMB are atom types. We provide an option to monitor the distance between two CoM of atom of groups. In this case, set T_SYMB(1) and T_SYMB(2) to be "cm".

T_R (dp) Delimiter distance at which the bond is considered formed/broken.

Record 13: IOUTPUT, ILIST(1, …, IOUTPUT)

IOUTPUT (int) Outcome control flag
If IOUTPUT = 0, write output to all units; ILIST is not read
> 0, write output to unit 6 and to IOUTPUT units listed in ILIST

ILIST (list of integers of length IOUTPUT) Write output to units contained in this list

XI.OUTPUT FILES

Unit 6 (standard output): General output
General information and error warnings are written to the standard output (unit 6).

Unit 10: Initial coordinates
The first line contains the trajectory index and potential energy in eV, followed by (one line for each atom)

1. Atomic symbol
2. Atomic mass in amu
3. The Cartesian coordinates in Å.

Unit 11: Initial momenta
The first line contains the trajectory index and kinetic energy in eV, followed by (one line for each atom)

1. Atomic symbol
2. Atomic mass in amu
3. The Cartesian components of the nuclear momentum in a.u.

Unit 20: Final coordinates
The first line contains the trajectory index and potential energy in eV, followed by (one line for each atom)

1. Atomic symbol
2. Atomic mass in amu
3. The Cartesian coordinates in Å.

Unit 21: Final momenta
The first line contains the trajectory index and kinetic energy in eV, followed by (one line for each atom)

1. Atomic symbol
2. Atomic mass in amu
3. The Cartesian components of the nuclear momentum in a.u.

Unit 30: Atom-diatom scattering output (for INITx = 3 only)
Note: The rovibrational quantum numbers are not accurate for the semiclassical Ehrenfest method because the code assumes a single potential energy surface when the simulation terminates.

1. Trajectory index
2. Final surface label
3. Final arrangement label
   Note: 1 = AB + C, 2 = BC + A, 3 = CA + B
4. Total time for the trajectory in (fs)
5. Total number of integration steps
6. Final value of the electronic state density matrix for state 1
7. Final value of the electronic state density matrix for state 2
8. Final total energy (eV)

9. Potential energy of the classical minimum of the final arrangement (eV)
10. Final kinetic energy corresponding to the relative atom-diatom translational motion (eV)
11. Final internal (rovibrational) energy of the diatomic fragment (eV)
12. Final vibrational energy of the diatomic fragment (eV)
13. Final rotational energy of the diatomic fragment (eV)
14. Final classical (unquantized) vibrational quantum number.
15. Final classical (unquantized) rotational quantum number.

Unit 31: General two–atom group output

1. Trajectory index
2. Final electronic surface label
3. Final outcome (i.e., final molecular arrangement, labeled according to TERMFLAG=3 information; 0 indicates no termination condition was met before the maximum number of steps was taken)
4. Final time, fs
5. average temperature, K (for $3N_{\mathrm{atom}}$ degrees of freedom)
6. stdev of temperature, K
7. b_0, Å (initial impact parameter)
8. l_0 (initial orbital angular momentum)
9. $E_{\mathrm{orb},0}$, eV (initial orbital energy)
10. $E_{\mathrm{rel},0}$, eV (initial relative energy, excluding orbital energy)
11. PE_0, eV (initial potential energy)
12. $J_0(1)$ (initial rotational state of fragment 1)
13. $E_{\mathrm{rot},0}(1)$ (initial instantaneous classical rotational energy of fragment 1)
14. $E_{\mathrm{vib},0}(1)$ (initial instantaneous vibrational energy of fragment 1)
15. $\mathrm{PE}_0(1)$ (initial potential energy of fragment one; calculated only if LDOFRAG = TRUE, otherwise 0 is printed)
16. $J_0(2)$ (initial rotational state of fragment 2)
17. $E_{\mathrm{rot},0}(2)$ (initial instantaneous classical rotational energy of fragment 2)
18. $E_{\mathrm{vib},0}(2)$ (initial instantaneous vibrational energy of fragment 2)
19. $\mathrm{PE}_0(2)$ (initial potential energy of fragment one; calculated only if LDOFRAG = TRUE, otherwise 0 is printed)

Lines 20–31 are as above (8–19) but for the final state of the trajectory. The ordering is slightly different than above and there is no b_{f}.

20. PE_{f}, eV
21. l_{f}
22. $E_{\mathrm{orb,f}}$
23. $E_{\mathrm{rel,f}}$
24. $J_f(1)$
25. $E_{\mathrm{rot,f}}(1)$
26. $E_{\mathrm{vib,f}}(1)$
27. $\mathrm{PE}_{\mathrm{f}}(1)$
28. $J_{\mathrm{f}}(2)$
29. $E_{\mathrm{rot,f}}(2)$
30. $E_{\mathrm{vib,f}}(2)$
31. PEf(2)
32. Sampling temperature, K
33. Scattering angle, degrees

Unit 40: Temperature ramping output

Note: This output is written only if TFLAG(1) = 2.
Data is written every time the temperature is rescaled.

1. Trajectory index
2. Counts the number of temperature rescalings
3. Time average of the temperature
4. Standard deviation of the temperature
5. Lindemann parameter (the relative second moment of the bond distances, averaged over all atom pairs)

Note: Entries 3 and 4 are calculated between temperature rescalings, whereas entry 5 is averaged over the entire trajectory.

Unit 41: Radial distribution function

Data is written for every NPRINT steps and at the end of each trajectory.

1. Trajectory index
2. Time in fs
3. A series of numbers representing the radial distribution function binned into equally spaced bins and averaged over the trajectory. The radial distribution function is normalized to unit area. The bin size [RBIN] is hardcoded in RADIALDIST to be 1/8 the bulk Al nearest-neighbor distance (i.e., 1/8 of 4.02 Å)

Unit 42: Honeycutt-Anderson parameter

This output is written when the temperature is rescaled (TFLAG(1) = 2) and when each trajectory finishes.

For a definition of the Honeycutt-Anderson (HA) index see Honeycutt and Anderson, J. Phys. Chem. 91, 4950 (1987). Briefly, the HA index is a set of four indices which describes the local geometry of a bonded pair of atoms (i.e., a pair of atoms with a bond distance less than some cutoff distance R_{NN}). Here we record the fraction of bonded pairs $H(i, j, k, l)$ with HA indices (i, j, k, l). R_{NN} is hard-coded in HONEY with the value 3.5 Å.

1. Trajectory index
2. Time in fs
3. The sum of $H(1, 2, k, l)$ for all k and l
4. The sum of $H(1, 3, k, l)$ for all k and l
5. The sum of $H(1, 4, k, l)$ for all k and l
6. The sum of $H(1, 5, k, l)$ for all k and l
7. The sum of $H(1, 6, k, l)$ for all k and l
8. $H(1, 4, 2, 1)$
9. $H(1, 4, 2, 2)$

Unit 43: Bond distance matrix

Data is written for every NPRINT steps and at the end of each trajectory.

1. Trajectory index
2. Time in fs
3. Bond distances in Å. Prints upper triangle of the symmetric matrix R_{ij} for i = 1–N and j = i+1–N, i.e., the order is R_{12}, R_{13}, …, R_{23}, R_{24}, … R_{NN}.

Unit 70: Hessian

This file is written if LREADHESS is FALSE and read if LREADHESS is TRUE. There are 3*NATOM lines. Each line contains

1. An index i.
2. An index j.
3. The (i,j)th component of the Hessian in unscaled atomic units, where i and j label the degrees of freedom in the order x_1, y_1, z_1, x_2, ….

If Hessians are calculated for more than one AG, they are written sequentially to unit 70. If Hessians are read for more than one AG, they are read sequentially from unit 70. In both cases, the atom index starts from 1 for each AG.

Unit 80: Coordinates

This file contains coordinates in Molden format. Data is written for every NPRINT steps.

1. NATOM
2. NSTEP PE(hartree) time(fs)
3. NATOM lines: symbol(character*2), atomic mass (amu), x, y, z(A)

Unit 81: Momenta

Similar to FORT.80, momenta for every NPRINT steps.

1. NATOM
2. NSTEP PE(hartree)
3. NATOM lines: symbol(character*2), atomic mass (amu), ppx, ppy, ppz(a.u.)

Unit 82: Binary, coordinates

Same as FORT.80, but written in binary format for much faster direct access.

Unit 83: Binary, momenta

Same as FORT.81, but written in binary format for much faster direct access.

XII. REVISION HISTORY

Version	Date	Description
2.0	December 16, 2020	Finalized this version the manual
2.0	December 10, 2020	Finalized version 2.0 of the code
1.1	July 4, 2013	Finalized version 1.2 of the code and manual
1.0	July 1, 2010	Finalized version 1.0 of the code and manual