User Manual and Source Code for a LAMMPS Implementation of Constant Energy Dissipative Particle Dynamics (DPD-E)

by James P. Larentzos, John K. Brennan, Joshua D. Moore, and William D. Mattson

ARL-SR-290

June 2014

Approved for public release; distribution is unlimited.
NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer’s or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.
User Manual and Source Code for a LAMMPS Implementation of Constant Energy Dissipative Particle Dynamics (DPD-E)

James P. Larentzos
Engility Corporation

John K. Brennan, Joshua D. Moore, and William D. Mattson
Weapons and Materials Research Directorate, ARL

Approved for public release; distribution is unlimited.
A user manual and source code files are provided for the implementation of the constant energy Dissipative Particle Dynamics method into the highly scalable Large-Scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) simulation software. The current LAMMPS velocity-Verlet (VV) integration scheme is extended to model systems under isoenergetic cases. In addition, the Shardlow-splitting algorithm is provided as an alternative integration scheme that enables longer time steps with comparable accuracy to the VV integration scheme.
Appendix. Large-Scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) User Manual for Commands Related to the Constant Energy Dissipative Particle Dynamics (DPD-E) Implementation With the Velocity-Verlet (VV) and VV-Shardlow-Splitting Algorithm (SSA) Integration Schemes 3

Distribution List 20
The constant energy dissipative particle dynamics (DPD-E) method is implemented into the Large-Scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) simulation software to efficiently model systems under isoenergetic conditions using the current LAMMPS velocity-Verlet (VV) integration scheme and the Shardlow-splitting algorithm (SSA). The relevant source code files that are current with the 7 February 2014 release of LAMMPS are provided, along with the user manual documentation. The contents of this material are described in detail in U.S. Army Research Laboratory (ARL) technical report ARL-TR-6863.¹

The DPD-E method using the VV and VV-SSA integration schemes has been implemented within LAMMPS as user packages under the directory src/USER-DPD in the accompanying compact disc (CD).

The LAMMPS user package USER-DPD can be added to the LAMMPS source files by issuing the command

```
make yes-USER-DPD
```

within the LAMMPS src/ directory. A copy of the modified user manual containing the new features added to LAMMPS can be found in the appendix, and the source txt, html, tex, and jpg files can be found within the LAMMPS doc/ and doc/Eqs directory in the accompanying CD. The full LAMMPS user manual can be found at http://lammps.sandia.gov/doc/Manual.html.

Intentionally left blank.
Appendix. Large-Scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) User Manual for Commands Related to the Constant Energy Dissipative Particle Dynamics (DPD-E) Implementation With the Velocity-Verlet (VV) and VV-Shardlow-Splitting Algorithm (SSA) Integration Schemes

This appendix appears in its original form, without editorial change.
atom_style command

Syntax:

atom_style style args

- style = angle or atomic or body or bond or charge or dpd or dipole or electron or ellipsoid or full or line or meso or molecular or peri or sphere or tri or hybrid

args = none for any style except body and hybrid
body args = bstyle bstyle-args
bstyle = style of body particles
bstyle-args = additional arguments specific to the bstyle
see the doc page for details
hybrid args = list of one or more sub-styles, each with their args

Examples:

atom_style atomic
atom_style bond
atom_style full
atom_style body nparticle 2 10
atom_style hybrid charge bond
atom_style hybrid charge body nparticle 2 5

Description:

Define what style of atoms to use in a simulation. This determines what attributes are associated with the atoms. This command must be used before a simulation is setup via a read_data, read_restart, or create_box command.

Once a style is assigned, it cannot be changed, so use a style general enough to encompass all attributes. E.g. with style bond, angular terms cannot be used or added later to the model. It is OK to use a style more general than needed, though it may be slightly inefficient.

The choice of style affects what quantities are stored by each atom, what quantities are communicated between processors to enable forces to be computed, and what quantities are listed in the data file read by the read_data command.

These are the additional attributes of each style and the typical kinds of physical systems they are used to model. All styles store coordinates, velocities, atom IDs and types. See the read_data, create_atoms, and set commands for info on how to set these various quantities.

<table>
<thead>
<tr>
<th>style</th>
<th>bonds and angles</th>
<th>bead-spring polymers with stiffness</th>
</tr>
</thead>
<tbody>
<tr>
<td>atomic</td>
<td>only the default values</td>
<td>coarse-grain liquids, solids, metals</td>
</tr>
<tr>
<td>body</td>
<td>mass, inertia moments, quaternion, angular momentum</td>
<td>arbitrary bodies</td>
</tr>
<tr>
<td>bond</td>
<td>bonds</td>
<td>bead-spring polymers</td>
</tr>
<tr>
<td>charge</td>
<td>charge</td>
<td>atomic system with charges</td>
</tr>
<tr>
<td>dpd</td>
<td>internal temperature and energies</td>
<td>DPD particles</td>
</tr>
</tbody>
</table>

atom_style command
<table>
<thead>
<tr>
<th>Particle Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>dipole</em></td>
<td>charge and dipole moment</td>
</tr>
<tr>
<td><em>electron</em></td>
<td>charge and spin and radius</td>
</tr>
<tr>
<td><em>ellipsoid</em></td>
<td>shape, quaternion, angular momentum</td>
</tr>
<tr>
<td><em>full</em></td>
<td>molecular + charge</td>
</tr>
<tr>
<td><em>line</em></td>
<td>end points, angular velocity</td>
</tr>
<tr>
<td><em>meso</em></td>
<td>rho, e, cv</td>
</tr>
<tr>
<td><em>molecular</em></td>
<td>bonds, angles, dihedrals, impropers</td>
</tr>
<tr>
<td><em>peri</em></td>
<td>mass, volume</td>
</tr>
<tr>
<td><em>sphere</em></td>
<td>diameter, mass, angular velocity</td>
</tr>
<tr>
<td><em>tri</em></td>
<td>corner points, angular momentum</td>
</tr>
<tr>
<td><em>wavepacket</em></td>
<td>charge, spin, eradius, etag, cs_re, cs_im</td>
</tr>
</tbody>
</table>

All of the styles define point particles, except the *sphere*, *ellipsoid*, *electron*, *peri*, *wavepacket*, *line*, *tri*, and *body* styles, which define finite-size particles. See Section 14 for an overview of using finite-size particle models with LAMMPS.

All of the styles assign mass to particles on a per-type basis, using the `mass` command, except for the finite-size particle styles. They assign mass to individual particles on a per-particle basis.

For the *sphere* style, the particles are spheres and each stores a per-particle diameter and mass. If the diameter is greater than 0.0, the particle is a finite-size sphere. If the diameter equals 0.0, it is a point particle.

For the *ellipsoid* style, the particles are ellipsoids and each stores a flag which indicates whether it is a finite-size ellipsoid or a point particle. If it is an ellipsoid, it also stores a shape vector with the 3 diameters of the ellipsoid and a quaternion 4-vector with its orientation.

For the *electron* style, the particles representing electrons are 3D Gaussians with a specified position and bandwidth or uncertainty in position, which is represented by the eradius = electron size.

For the *peri* style, the particles are spherical and each stores a per-particle mass and volume.

The *meso* style is for smoothed particle hydrodynamics (SPH) particles which store a density (rho), energy (e), and heat capacity (cv).

The *dpd* style is for dissipative particle dynamics (DPD) particles which store the particle internal temperature (dpdTheta), internal conductive energy (uCond), internal mechanical energy (uMech), as well as other particle properties such as heat capacity (cv) and density (rho).

The *wavepacket* style is similar to *electron*, but the electrons may consist of several Gaussian wave packets, summed up with coefficients cs = (cs_re, cs_im). Each of the wave packets is treated as a separate particle in LAMMPS, wave packets belonging to the same electron must have identical etag values.

For the *line* style, the particles are idealized line segments and each stores a per-particle mass and length and orientation (i.e. the end points of the line segment).

For the *tri* style, the particles are planar triangles and each stores a per-particle mass and size and orientation (i.e. the corner points of the triangle).

atom_style command
For the *body* style, the particles are arbitrary bodies with internal attributes defined by the "style" of the bodies, which is specified by the *bstyle* argument. Body particles can represent complex entities, such as surface meshes of discrete points, collections of sub-particles, deformable objects, etc.

The *body* doc page describes the body styles LAMMPS currently supports, and provides more details as to the kind of body particles they represent. For all styles, each body particle stores moments of inertia and a quaternion 4-vector, so that its orientation and position can be time integrated due to forces and torques.

Note that there may be additional arguments required along with the *bstyle* specification, in the *atom_style* body command. These arguments are described in the *body* doc page.

Typically, simulations require only a single (non-hybrid) atom style. If some atoms in the simulation do not have all the properties defined by a particular style, use the simplest style that defines all the needed properties by any atom. For example, if some atoms in a simulation are charged, but others are not, use the *charge* style. If some atoms have bonds, but others do not, use the *bond* style.

The only scenario where the *hybrid* style is needed is if there is no single style which defines all needed properties of all atoms. For example, if you want dipolar particles which will rotate due to torque, you would need to use "atom_style hybrid sphere dipole". When a hybrid style is used, atoms store and communicate the union of all quantities implied by the individual styles.

LAMMPS can be extended with new atom styles as well as new body styles; see this section.

**Restrictions:**

This command cannot be used after the simulation box is defined by a *read_data* or *create_box* command.

The *angle*, *bond*, *full*, and *molecular* styles are part of the MOLECULAR package. The *line* and *tri* styles are part of the ASPHERE package. The *body* style is part of the BODY package. The *dipole* style is part of the DIPOLE package. The *peri* style is part of the PERI package for Peridynamics. The *electron* style is part of the USER-EFF package for electronic force fields. The *dpd* style is part of the USER-DPDE and USER-DPDE-SHARDLOW packages for dissipative particle dynamics (DPD). The *meso* style is part of the USER-SPH package for smoothed particle hydrodynamics (SPH). See this PDF guide to using SPH in LAMMPS. The *wavepacket* style is part of the USER-AWPMD package for the antisymmetrized wave packet MD method. They are only enabled if LAMMPS was built with that package. See the Making LAMMPS section for more info.

**Related commands:**

*read_data*, *pair_style*

**Default:**

*atom_style* atomic

*atom_style command* 3
**compute dpd command**

**Syntax:**

```
compute ID group-ID dpd
```

- ID, group-ID are documented in compute command
- dpd = style name of this compute command

**Examples:**

```
compute 1 all dpd
```

**Description:**

Define a computation that accumulates the total internal conductive energy ($U_{\text{cond}}$), the total internal mechanical energy ($U_{\text{mech}}$), the total internal energy ($U$) and the average internal temperature ($\Theta$) of the entire system of particles. See the compute dpd/atom command if you want per-particle internal energies and internal temperatures.

The system internal properties are computed according to the following relations:

\[
U_{\text{cond}} = \sum_{i=1}^{N} u_{i}^{\text{cond}}
\]

\[
U_{\text{mech}} = \sum_{i=1}^{N} u_{i}^{\text{mech}}
\]

\[U = \sum_{i=1}^{N} (u_{i}^{\text{cond}} + u_{i}^{\text{mech}})\]

\[\Theta_{\text{avg}} = \left( \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\theta_{i}} \right)^{-1}\]

where $N$ is the number of particles in the system.

**Output info:**

This compute calculates a global vector of length 4 ($U_{\text{cond}}$, $U_{\text{mech}}$, $U$, $\Theta$), which can be accessed by indices 1-4. See this section for an overview of LAMMPS output options.

The vector values will be in energy and temperature units.

**Restrictions:**

The `compute dpd` is only available if LAMMPS is built with the appropriate USER-DPD or...
USER-DPDE-SHARDLOW package.

Related commands:

compute dpd/atom, thermo style

Default: none

compute dpd/atom command

Syntax:

compute ID group-ID dpd/atom

- ID, group-ID are documented in compute command
- dpd/atom = style name of this compute command

Examples:

compute 1 all dpd/atom

Description:

Define a computation that accesses the per-particle internal conductive energy (u_cond), internal mechanical energy (u_mech) and internal temperatures (theta) for each particle in a group. See the compute dpd command if you want the total internal conductive energy, the total internal mechanical energy, and average internal temperature of the entire system of dpd particles.

Output info:

This compute calculates a per-particle array with 3 columns, which can be accessed by indices 1-3 by any command that uses per-particle values from a compute as input. See Section howto 15 for an overview of LAMMPS output options.

The per-particle array values will be in energy and temperature units as discussed above.

Restrictions:

The compute dpd is only available if LAMMPS is built with the appropriate USER-DPDE or USER-DPDE-SHARDELOW package.

Related commands:

compute dpd

Default: none

fix dpde command

Syntax:

fix ID group-ID dpde

ID, group-ID are documented in fix command dpde = style name of this fix command

Examples:

fix 1 all dpde

Description:

Perform constant energy dissipative particle dynamics (DPD-E) integration to update position, velocity and internal energy for particles in the group at each timestep.

For fix dpde, the particle internal temperature is related to the particle internal energy through a mesoparticle equation of state. An additional fix must be specified that defines the equation of state for each particle.

Restrictions:

The fix dpde is only available if LAMMPS is built with the appropriate USER-DPDE package.

The fix dpde requires the dpd atom_style to be used in order to properly account for the particle internal energies and temperature.

The fix dpde must be used with an additional fix that specifies the mesoparticle equation of state for each particle.

Related commands:

fix nve fix cos/cv

Default: none

**fix dpde/shardlow command**

**Syntax:**

```
fix ID group-ID dpde/shardlow
```

ID, group-ID are documented in fix command dpde/shardlow = style name of this fix command

**Examples:**

```
fix 1 all dpde/shardlow
```

**Description:**

Perform constant energy dissipative particle dynamics (DPD-E) integration using the Shardlow splitting algorithm (SSA) to update position, velocity and internal energy for particles in the group at each timestep. The SSA splits the integration into a stochastic and deterministic integration step. The stochastic integration of the dissipative and random forces is performed prior to the deterministic integration of the conservative force. Further details regarding the method are provided in [Lisal](#) and [Lorentzos](#).

For fix dpde/shardlow, the particle internal temperature is related to the particle internal energy through a mesoparticle equation of state. An additional fix must be specified that defines the equation of state for each particle.

**Restrictions:**

The fix dpde/shardlow is only available if LAMMPS is built with the appropriate USER-DPDE-SHARDLOW package.

The fix dpde/shardlow must be used with the dpd/conservative pair style command to properly initialize pair coefficients for sigma and kappa.

The fix dpde/shardlow requires the dpd atom style to be used in order to properly account for the particle internal energies and temperature.

The fix dpde/shardlow must be used with an additional fix that specifies the mesoparticle equation of state for each particle.

**Related commands:**

fix dpde fix eos/cv

**Default:** none

---


fix dpde/shardlow command
fix eos/cv command

Syntax:

fix ID group-ID eos/cv cv

• ID, group-ID are documented in fix command
• eos/cv = style name of this fix command
• cv = constant-volume heat capacity

Examples:

fix 1 all eos/cv 0.01

Description:

Fix eos/cv applies a mesoparticle equation of state to relate the particle internal energy \( u_i \) to the particle internal temperature \( (\text{theta}_i) \). The eos/cv mesoparticle equation of state requires the constant-volume heat capacity, and is defined as follows:

\[
   u_i = u_i^{\text{mech}} + u_i^{\text{cond}} = C_V \theta_i
\]

where \( C_V \) is the constant-volume heat capacity, \( u_{\text{cond}} \) is the internal conductive energy, and \( u_{\text{mech}} \) is the internal mechanical energy. Note that alternative definitions of the mesoparticle equation of state are possible, but not currently implemented.

Restrictions:

The fix eos/cv is only available if LAMMPS is built with the appropriate USER-DPDE or USER-DPDE-SHARDLOW packages.

The fix eos/cv must be used with the atom style dpd.

Related commands:

fix dpd

Default: none

pair_style dpde command

Syntax:

pair_style dpde kappa_flag cutoff seed

- kappa_flag = 0 to turn off/on the energy dependence of kappa (integer)
- cutoff = global cutoff for DPD interactions (distance units)
- seed = random # seed (positive integer)

Examples:

pair_style dpde 0 2.5 34387
pair_style dpde 1 2.5 34387
pair_coeff * * 1.0 1.0 1.0
pair_coeff 1 1 3.0 1.0 1.0 1.0

Description:

Style dpde computes a force field for dissipative particle dynamics (DPD) under isoenergetic conditions. The force on atom \( i \) due to atom \( j \) is given as a sum of 3 terms:

\[
\vec{F} = (F^C + F^D + F^R)\vec{r}_{ij}^* \quad r_{ij} < r_c
\]

\[
F^C = A\omega_{ij}
\]

\[
F^D = -\gamma\omega^2_{ij}(\vec{r}_{ij}^* \cdot \vec{u}_{ij}^*)
\]

\[
F^R = \sigma_{ij}\omega_{ij}\zeta(\Delta t)^{-1/2}
\]

where \( F^C \) is a conservative force, \( F^D \) is a dissipative force, and \( F^R \) is a random force. \( \vec{r}_{ij}^* \) is the unit vector in the direction \( \vec{R}_i - \vec{R}_j \), \( \vec{v}_{ij} \) is the vector difference in velocities of the two atoms \( \vec{v}_i - \vec{v}_j \), \( \zeta \) is a Gaussian random number with zero mean and unit variance, and \( \Delta t \) is the timestep size. \( \gamma \) is set equal to \( (\text{sigma}^2/\text{sigma})/(2 \text{ Kb Theta}) \), where \( \text{Kb} \) is the Boltzmann constant and \( \text{Theta} \) is the particle internal temperature.

For style dpde, the weighting factor, \( \omega_{ij} \), varies between 0 and 1, and is chosen to have the following functional form:

\[
\omega_{ij} = 1 - \frac{r_{ij}}{r_c}
\]

where \( r_c \) is the cutoff radius. Note that alternative definitions of the weighting function exist, but would have to be implemented with a separate pair_style command.

The kappa_{ij} variable can be specified with or without an energy dependence by toggling the kappa_flag. In the energy-independent model (kappa_flag = 0), kappa_{ij} is explicitly given as a pair coefficient. In the energy-dependent model (kappa_flag = 1), kappa_{ij} is given by the equation:

\[
\text{pair_style dpde command}
\]

pair_style dpde command
where $\kappa_{ij}$ is the pair coefficient that is specified in the input file, $k_B$ is the Boltzmann constant, and $\omega^2_{ij}$ is the total internal energy of particle $i$.

The differential internal conductive and mechanical energies are computed as

$$du^\text{cond}_i = \kappa_{ij} \left( \frac{1}{\theta_i} - \frac{1}{\theta_j} \right) \omega^2_{ij} + \sigma_{ij} \omega_{ij} \zeta_{ij} (\Delta t)^{-1/2}$$

$$du^\text{mech}_i = -\frac{1}{2} \gamma_{ij} \omega^2_{ij} \left( \frac{\vec{r}_{ij}}{r_{ij}} \cdot \vec{v}_{ij} \right)^2 - \frac{\sigma_{ij}^2}{4} \left( \frac{1}{m_i} + \frac{1}{m_j} \right) \omega^2_{ij} - \frac{1}{2} \sigma_{ij} \omega_{ij} \left( \frac{\vec{r}_{ij}}{r_{ij}} \cdot \vec{v}_{ij} \right) \zeta_{ij} (\Delta t)^{-1/2}$$

where

$$\alpha_{ij}^2 = 2k_B \kappa_{ij}$$

$$\sigma_{ij}^2 = 2\gamma_{ij} k_B \Theta_{ij}$$

$$\Theta_{ij}^{-1} = \frac{1}{2} \left( \frac{1}{\theta_i} + \frac{1}{\theta_j} \right)$$

$\zeta_{ij}$ is a second Gaussian random number with zero mean and unit variance that is used to compute the internal conductive energy.

For style dode, the pairwise energy associated with style dode is only due to the conservative force term $F_c$ and is shifted to be zero at the cutoff distance $R_c$. The pairwise virial is calculated using only the conservative term.

For style dode, the following coefficients must be defined for each pair of atom types via the pair_coeff command as in the examples above, or in the data file or restart files read by the read_data or read_restart commands:

- $A$ (force units)
- $\sigma$ (force*time^1/2 units)
- $\kappa_{ij}$ (energy*temperature/time units) or $\kappa_{ij}$ (1/time units)
- $\Theta_{ij}$ (distance units)

The last coefficient is optional. If not specified, the global DPD cutoff is used. Note that $\gamma$ is set equal to $\sigma^2/2 \Theta$, where $\Theta$ is the average internal temperature of the pair.

**Mixing, shift, table, tail correction, restart info:**

The pair style does not support mixing. Thus, coefficients for all LJ pairs must be specified explicitly.

The pair style does not support the pair_modify shift option for the energy of the pair interaction. Note that as discussed above, the energy due to the conservative $F_c$ term is already shifted to be 0.0 at the cutoff distance $R_c$.

pair_style dode command
The pair_modify table option is not relevant for this pair style.

The pair style does not support the pair_modify tail option for adding long-range tail corrections to energy and pressure.

The pair style writes its information to binary restart files, so pair_style and pair_coeff commands do not need to be specified in an input script that reads a restart file. Note that the user-specified random number seed is stored in the restart file, so when a simulation is restarted, each processor will re-initialize its random number generator the same way it did initially. This means the random forces will be random, but will not be the same as they would have been if the original simulation had continued past the restart time.

Restrictions:

The pair style dpde is only available if LAMMPS is built with the USER-DPDE or USER-DPDE-SHARQLOW package.

This pair style dpde requires the dpd atom_style to be used in order to properly account for the particle internal energies and temperature.

This pair style dpde requires you to use the communicate vel yes option so that velocities are stored by ghost atoms.

Related commands:

pair_coeff, pair_dpd, fix dpde

Default: none


pair_style dpde/conservative command

Syntax:

pair_style dpde/conservative kappa_flag cutoff seed

* kappa_model = 0/1 to turn off/on the energy-dependence of kappa (integer)
* cutoff = global cutoff for DPD interactions (distance units)
* seed = random # seed (positive integer)

Examples:

pair_style dpde/conservative 0 2.5 34337
pair_style dpde/conservative 1 2.5 34337
pair_coeff * * 0.0 1.0 1.0 1.0
pair_coeff 1 1 3.0 1.0 1.0 1.0

Description:

Style dpde/conservative computes the conservative force for dissipative particle dynamics (DPD). The conservative force on atom $i$ due to atom $j$ is given by

$$ F^C = \mathbf{A} \omega_{ij}, \quad r_{ij} < r_c $$

where the weighting factor varies between 0 and 1, and is chosen to have the following functional form:

$$ \omega_{ij} = 1 - \frac{r_{ij}}{r_c} $$

where $\mathbf{R}_{ij}$ is a unit vector in the direction $\mathbf{R}_i - \mathbf{R}_j$, and $r_c$ is the cutoff. Note that alternative definitions of the weighting function exist, but would have to be implemented with a separate pair style command.

The $\kappa_{ij}$ variable can be specified with or without an energy dependence by toggling the kappa_flag. In the energy-independent model (kappa_flag = 0), $\kappa_{ij}$ is explicitly given as a pair coefficient. In the energy-dependent model (kappa_flag = 1), $\kappa_{ij}$ is given by the equation:

$$ \kappa_{ij} = \frac{\kappa_0 (u_i + u_j)^2}{k_B} $$

where $\kappa_0$ is the pair coefficient that is specified in the input file, $k_B$ is the Boltzmann constant, and $u_i$ is the total internal energy of particle $i$.

This pair style differs from the other dpde styles in that the dissipative and random forces are not computed within the pair style. This style is combined with the fix dpde/chainflow, which will perform the stochastic integration of the dissipative and random forces through the Shardlow splitting algorithm approach.

For style dpde/conservative, the pairwise energy associated with style dpde/conservative is only due to the conservative force term $F^C$, and is shifted to be zero at the cutoff distance $r_c$. The pairwise virial is calculated.
using only the conservative term.

For style dpde/conservative, the following coefficients must be defined for each pair of atoms types via the pair_coeff command as in the examples above, or in the data file or restart files read by the read_data or read_restart commands:

- A (force units)
- sigma (force*time^2(1/2) units)
- kappa_ij (energy*Temperature/time units) or kappa0 (1/time units)
- cutoff (distance units)

The last coefficient is optional. If not specified, the global DPD cutoff is used. Note that gamma is set equal to sigma^2*sigma/(2 Theta), where Theta is the average internal temperature for the pair.

Mixing, shift, table, tail correction, restart info:

The pair style does not support mixing. Thus, coefficients for all IJ pairs must be specified explicitly.

The pair style does not support the pair_modify shift option for the energy of the pair interaction. Note that as discussed above, the energy due to the conservative Fc term is already shifted to be 0.0 at the cutoff distance Re.

The pair_modify table option is not relevant for these pair styles.

The pair style does not support the pair_modify tail option for adding long-range tail corrections to energy and pressure.

The pair style writes its information to binary restart files, so pair_style and pair_coeff commands do not need to be specified in an input script that reads a restart file. Note that the user-specified random number seed is stored in the restart file, so when a simulation is restarted, each processor will re-initialize its random number generator the same way it did initially. This means the random forces will be random, but will not be the same as they would have been if the original simulation had continued past the restart time.

Restrictions:

The pair style dpde/conservative is only available if LAMMPS is built with the USER-DPDE-SHARDLOW package.

The pair style dpde/conservative requires the dpd atom_style to be used in order to properly account for the particle internal energies and temperature.

The pair style dpde/conservative requires you to use the communicate vel yes option so that velocities are stored by ghost atoms.

The pair style dpde/conservative will not restart exactly when using the read_restart command, though they should provide statistically similar results. This is because the forces they compute depend on atom velocities. See the read_restart command for more details.

Related commands:

pair_style dpde/conservative command
pair_coeff pair_dpd pair_dpd4 fix_dpd/shardlow

Default: none

