NUFEB User Manual
Version 1.0

Bowen Li, Jayathilake Gedara, Prashant Gupta, Curtis Madsen

September 14, 2016

Contents

1 Introduction **1**

2 LAMMPS **1**
2.1 Introduction to LAMMPS 1
2.2 LAMMPS working methodology 1
2.3 Operating systems 2
2.4 Pre-compilation instructions 2

3 NUFEB Compilation Instructions **2**
3.1 Downloading NUFEB 3
3.2 Compiling NUFEB 3
3.3 Running an Input Script with NUFEB 4
3.4 Add the NUFEB Executable to Your Path (Optional) 5
3.5 Post-processing 5

4 Input Script **6**
4.1 Input script structure 6
4.1.1 Initialization 6
4.1.2 Microbe and simulation domain definition 7
4.1.3 Settings 7
4.1.4 Run a simulation 8
4.2 atom_style command 8
4.3 create_atom command (atom data file) 9
4.4 diffnugrowth command 10
4.5 divide command 12
4.6 eps_extract command 13
4.7 death command 14
4.8 epsadh command 15
4.9 walladh command 16
4.10 shear command 17

4.11 thermo_style command 18

1 Introduction

This document provides information on how to download, compile, and start using NUFEB.

NUFEB is an open source tool for Individual Based model (IBm) simulation. The tool is implemented as a user package within LAMMPS - a molecular dynamics simulator offering basic functionalities for Discrete Element Method (DEM) simulations. NUFEB aims to improve those capability with the goal to apply it to biological modelling.

NUFEB is a freely-available open-source code, distributed under the terms of the GNU Public License.

NUFEB development has been funded by the EPSRC project An New Frontier in Design: The Simulation of Open Engineered Biological Systems (NUFEB).

2 LAMMPS

2.1 Introduction to LAMMPS

LAMMPS is a classical molecular dynamics code developed at Sandia labs and primarily built to solve the particle physics including wide range of inter-particle interactions and potentials. The code treats each particle as an individual discrete unit, much similar to the popular IB approach. Sandia Labs distributes LAMMPS under the terms of the GNU Public License (http://lammps.sandia.gov/). The current version of the code is written in C++ with an open architecture and provides an opportunity to couple with other open-source codes. LAMMPS can run efficiently in both serial and parallel versions depending upon the computational facilities available to the users. The LAMMPS code is designed to modify and extend it with newer capabilities as desired by the user. While only 25% of the 140K line code in LAMMPS forms the core of the solver, rest of the code is contributed by a large user database across the globe in order to extend its capabilities. An overview can of current LAMMPS capabilities can be found at LAMMPS-feature.

2.2 LAMMPS working methodology

LAMMPS solves the motion of every single particle by simply integrating Newton's equations of motion in response to sum of the forces (short or long range based on their interaction with neighbours). At a particular time instance, motion of each particle is collectively solved when subjected to initial or boundary conditions. In order to maintain computational tractability while calculating the interaction forces, LAMMPS maintains a neighbourhood list for each particle which gets updated every so often. These lists

are optimized so that local densities and particle overlaps never becomes non-physical. For parallel simulations, LAMMPS spatially partition the domain into smaller sub-domains assigned to each processors. Interprocessor communications are maintained by storing ghost atom interactions with the sub-domain boundaries. LAMMPS development can be helped by two user manuals: User manual and developer manual. The following links will be helpful for the users to get started on LAMMPS:

1. User manual: `http://lammps.sandia.gov/doc/Manual.pdf`
2. Developers guide: `http://lammps.sandia.gov/doc/Developer.pdf`
3. Tutorials: `http://lammps.sandia.gov/tutorials.html`
4. Commands: `http://lammps.sandia.gov/doc/Section_commands.html`
5. Features: `http://lammps.sandia.gov/features.html`

In the present study, LAMMPS-Feb14 version is developed and newer IB features and capabilities added, this version will be now on referred as NUFEB.

2.3 Operating systems

In general, LAMMPS can be run on Windows, Linux, Mac OS using pre-built executables. NUFEB can be compiled with almost any Linux or Mac OS (instructions in the user manual). It is emphasized that present NUFEB version 1.0 has been rigorously tested on Ubuntu-14.10 and Fedora-22. In near future, pre-built executables,binaries or RPMS will be provided to be used on any OS.

2.4 Pre-compilation instructions

Before compiling NUFEB, please make sure you are installed with these packages depending upon the operating system used:

- openmpi (http://www.open-mpi.org/)
- gcc/g++ (https://help.ubuntu.com/community/InstallingCompilers)

3 NUFEB Compilation Instructions

For the pre-existing LAMMPS commands, features and documentation, please refer to the LAMMPS user manual, listed above. The manual covers extensive instructions on compiling LAMMPS and how to get started. NUFEB is compiled the same way as you would compile LAMMPS and there is no

change in those instructions. The newer capabilities and commands will be highlighted and emphasized in the next sections and the sample input script.

3.1 Downloading NUFEB

NUFEB can be downloaded from the NUFEB Github repository by opening a terminal and executing the following set of instructions:

```
$ git clone https://github.com/nufeb/NUFEB.git
```

3.2 Compiling NUFEB

Once downloaded, the source code for NUFEB can be found in the NUFEB/src/ directory. To compile this code, go to that directory:

```
$ cd NUFEB/src/
```

and execute the following commands to compile code in the STUBS directory:

```
$ cd STUBS/
$ make clean
$ make
$ cd ..
```

Now, install the NUFEB and granular packages with the following instruction:

```
$ make yes-USER-NUFEB
$ make yes-GRANULAR
```

You should get the messages "Installing package USER-NUFEB" and "Installing package GRANULAR" with no errors. Finally, execute the following command to compile the NUFEB executable:

```
$ make serial
```

This process may take some time to complete. When finished without errors, you should have an executable "lmp_serial" in the NUFEB/src/ directory.

3.3 Running an Input Script with NUFEB

To run the flat surface example input script, go to your nufeb folder, change to the NUFEB/examples/ directory, and execute "lmp_serial" passing in the "Input.lammps" file:

```
$ cd NUFEB/examples/
$ ../src/lmp_serial < Inputscript.lammps
```

The output should look similar to this:

```
LAMMPS (1 Feb 2014)
Reading data file ...
  orthogonal box = (0 0 0) to (1e-04 1e-04 1e-04)
  1 by 1 by 1 MPI processor grid
  reading atoms ...
  5 atoms
1 atoms in group HET
1 atoms in group AOB
1 atoms in group NOB
1 atoms in group EPS
0 atoms in group DEAD
Setting up run ...
Memory usage per processor = 184.889 Mbytes
Step Atoms nHET nAOB nNOB nEPS nDEAD
       0        5    1    1    1    1    0
    1000        6    1    1    1    3    0
    2000        10   2    2    1    5    0
       .        .    .    .    .    .    .
       .        .    .    .    .    .    .
       .        .    .    .    .    .    .
    9000        534  167  9    3    355  0
    10000       991  310  12   4    665  0
Loop time of 5.19042 on 1 procs for 10000 steps with 991 atoms

Pair  time (%) = 0.233821 (4.50486)
Neigh time (%) = 0.206507 (3.97862)
Comm  time (%) = 0.0842083 (1.62238)
Outpt time (%) = 0.0106578 (0.205336)
Other time (%) = 4.65523 (89.6888)

Nlocal:    991 ave 991 max 991 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Nghost:    0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
```

```
Neighs:    5769 ave 5769 max 5769 min
Histogram: 1 0 0 0 0 0 0 0 0 0

Total # of neighbors = 5769
Ave neighs/atom = 5.82139
Neighbor list builds = 86
Dangerous builds = 0
```

After running, there should be a "output.lammps" file in the same directory as output.

3.4 Add the NUFEB Executable to Your Path (Optional)

To make your life easier, you can add the "lmp_serial" executable to your path using the following command from within the
NUFEB/src/ directory:

```
$ export PATH=$PATH:$PWD
```

This addition, however, will only last for the current session. To permanently add it to your path, add the previous line to your ".bashrc" file in your home directory replacing "$PWD" with the path to your
NUFEB/src/ directory. Once "lmp_serial" is on your path, it can simply be executed as follows replacing "input.lammps" with the input script you want to run:

```
$ lmp_serial < Inputscript.lammps
```

3.5 Post-processing

In order to post-process NUFEB output, you need to have the following software packages:

- POVray (http://www.povray.org/)
- MATLAB (http://uk.mathworks.com/products/matlab/)

Copy the "output.lammps" file to the NUFEB/post_processing/ directory, change to this directory and execute the "run.sh" script:

```
$ cp output.lammps ../post_processing/
$ cd ../post_processing/
$ ./run.sh
```

This script will process the output file to generate a collection of images for each time point as well as a time-lapse video of the simulation in the 0_images directory.

4 Input Script

In order to execute NUFEB simulation, an input script (text file) is usually prepared with certain commands and parameters list. NUFEB executes by reading those commands and parameters, one line at a time. When the input script ends, NUFEB exits. Each command causes NUFEB to take some action. It may set an internal variable, read in a file, or run a simulation.

In this section, we will explain the new commands used for IBm simulation. The detailed description of LAMMPS pre-existing commands can be found in LAMMPS user manual

4.1 Input script structure

This section describes the structure of a typical NUFEB input script. We will take the script "Inputscript.lammps" in the "examples" directory as an example for the explanation. The same directory contains many other sample input scripts.

A NUFEB input script typically has 4 parts:

- Initialization
- Microbe and simulation domain definition
- Settings
- Run a simulation

4.1.1 Initialization

Example:

```
atom_style   bio
atom_modify  map array sort 1000 5.0e-7
boundary     ff ff ff
newton       off
pair_style   gran/hooke/history 200 NULL 50 NULL 0.0 1
```

Set parameters that need to be defined before microbes are created or read-in from a file. `atom_style` and `atom_modify` commands define what style and attributes of atoms to use in the simulation. `boundary` command is used to set the style of boundaries for the simulation domain in each dimension. `newton` command turns Newton's 3rd law on or off for pairwise and bonded interactions. `pair_style` command sets the force fields being used for the simulation.

4.1.2 Microbe and simulation domain definition

Example:

```
4    atoms
5    atom types

0.0  1.0e-04  xlo xhi
0.0  1.0e-04  ylo yhi
0.0  1.0e-04  zlo zhi

Atoms
1 1 1.0839e-6 150 5e-05 5.1e-05 0.2e-05 1.4307e-6
8.0e-02 10.0e-03 9.00e-02 8.0e-03 1.0e-05
 .
 .
```

There are several ways to define microbe (atom) and simulation domain in LAMMPS. In NUFEB, however, we encourage the users to use command `read_data`. A typical example is the file "flat_surface_bottom_large.in " in `nufeb/code/examples/` directory. $xlo, xhi, ylo, yhi, zlo, zhi$ are bounds of domain in all dimensions. In order to create a microbe, attributes must be given in the following order: ID, type, radius, density, x, y, z, outer-radius, S_s, O_2, NO_3, NO_2, NH_4. The details of how to create a microbe is given in sections 4.2 and 4.3.

4.1.3 Settings

Example:

```
neighbor       5.0e-7 bin
neigh_modify   delay 0 one 5000

pair_coeff     * *
timestep 10
fix 1 all nve/limit 1e-8
.
.
fix d5 all death 500 v_deadDia
.
.
dump id all custom 10 output.lammps id type diameter x y z
```

Once initial microbes and simulation domain are defined, a variety of settings can be specified: force field coefficients, simulation parameters, output options, etc.

Force field coefficients are set by the command `pair_coeff`.

Various simulation parameters are set by these commands: `neighbor`. `neigh_modify`. `timestep`.

Fixes commands are operation that are applied to the system during the simulation. Examples include updating of microbe positions and velocities due to time integration, applying constraint forces to microbes, performing biological processes to the microbes, etc. As an example, the `fix_nve_limit` command performs constant NVE updates of position and velocity for microbes in the group each timestep.

Output options are set by the `thermo`, `dump`, commands.

4.1.4 Run a simulation

Example:

```
run  10000
```

A molecular dynamics simulation is run using the `run` command.

4.2 atom_style command

Syntax

```
atom_style bio
```

- *bio*: atom style for IBm simulation

Description

Define a biological atom style used in the IBm simulationm. Classical LAMMPS provides different atom types that could be used by user. These are specified in the input script by the command: `atom_style`. Command must be used before a simulation is setup via a `read_data`, `read_restart` or `create_box` command. A newer `atom_style` is added (named "bio") to increase the number of attributes such as nutrients available to each microbe type. The new `atom_style` is inherited from already existing `atom_style sphere`.

The new atom type "bio" have following attributes in conjunction to the sphere type: ID, type, radius, outer-radius, mass, outer-mass, x, y, z, S_s, O_2, NO_3, NO_2, NH_4. Commonly found microbes in a WWTP are grouped into functional group: Heterotrophs (HET), Ammonia oxidizing bacteria (AOB), Nitrogen oxidizing bacteria (NOB), Extracellular polymeric substances (EPS) and Dead microbes (DEAD) (see LAMMPS group command):

- group HET type 1

- group AOB type 2
- group NOB type 3
- group EPS type 4
- group DEAD type 5

4.3 create_atom command (atom data file)

```
ID type radius density x y z outer-radius
sub O2 NO3 NO2 NH4
```

- *ID*: microbe ID
- *type*: microbe type, 1 = HET, 2 = AOB, 3 = NOB, 4 = EPS, 5 = DEAD
- *density*: microbe density
- *x, y, z*: microbe position in simulation domain
- *outer-radius*: EPS shell radius
- *sub, O2, NO3, NO2, NH4*: initial nutrient concentrations of the microbe

Examples

```
Atoms
1 1 1.0839e-6 150 5e-05 5.1e-05 0.2e-05 1.4307e-6
8.0e-02 10.0e-03 9.00e-02 8.0e-03 1.0e-05

2 2 1.2839e-6 150 5e-05 4.9e-05 0.2e-05 1.2839e-6
8.00e-02 10.00e-03 9.00e-02 8.000e-03 1.0000e-05
```

Description

NUFEB uses atom data file to create initial microbes. Each microbe has unique moniker *ID* but shares its type according to the functional group. Physical parameters such as initial *radius, outer-radius* and *density* can be mentioned. Radius and density are the radius and density of the microbe. Outer radius and densities are the initial EPS-shell bound radius and densities. Density is constant throughout the simulation and time invariant. However radius and mass (calculated from density and radius) can change. EPS is only bound to the HET particles (type 1), hence outer radius should be specified as radius for other particle types initially. NUFEB will give an

error message if this is not followed. For NUFEB book-keeping, inner radius is always same as outer-radius for other types than 1 throughout the simulation. Initial configuration can be either generated randomly through NUFEB or by the user. Caution must be taken to avoid overlaps when defined by the user. Some typical user atom data files are included in the "examples" folder (e.g., flat_surface_bottom_large.in).

4.4 diffnugrowth command

Syntax

```
fix ID group-ID diffnugrowth Nevery Devery KsHET Ko2HET
Kno2HET Kno3HET Knh4AOB Ko2AOB Kno2NOB Ko2NOB MumHET MumAOB
MumNOB etaHET bHET bAOB bNOB bEPS bmHET bmAOB bmNOB bX YHET
YAOB YNOB YEPS Y1 EPSdens Do2 Dnh4 Dno2 Dno3 Ds diffT
sub o2 no2 no3 nh4 nx ny nz BCmode BCsub BCo2 BCno2
BCno3 BCnh4 Oevery
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *diffnugrowth* : style name of this fix command
- *Nevery* : call the growth every this many timesteps
- *Devery* : call the diffusion every this many timesteps
- *KsHET, Ko2HET, Kno2HET, Kno3HET* : carbon source, dissolved oxygen, nitrite and nitrate affinity constants for HET
- *Knh4AOB, Ko2AOB* : ammonia and dissolved oxygen affinity constants for AOB
- *Kno2NOB, Ko2NOB* : nitrite and dissolved oxygen affinity constants for NOB
- *MumHET, MumAOB, MumNOB* : maximum specific growth rates for HET, AOB and NOB
- *etaHET* : reduction factor in anoxic condition
- *bHET, bAOB, bNOB, bEPS, bX* : decay rates for HET, AOB, NOB, EPS and DEAD
- *bmHET, bmAOB, bmNOB* : maintenance rates for HET, AOB and NOB
- *YHET, YAOB, YNOB* : yield coefficient for HET, AOB, and NOB growth

- *Y1* : yield of inert biomass
- *YEPS* : EPS formation coefficient
- *EPSdens* : density of EPS
- *Do2, Dnh4, Dno2, Dno3, Ds* : diffusion coefficients for oxygen, ammonia, nitrite, nitrate and substrate
- *diffT* : diffusion time
- *sub, o2, no2, no3, nh4* : inlet substrate, exygen, nitrite, nitrate and ammonia concentrations
- *nx, ny, nz* : grid size for x, y and z coordinate of the simulation domain
- *BCmode* : boundary condition modes (*BCmode = dirich or neu or mixed*)
- *BCsub, BCo2, BCno2, BCno3, BCnh4* : inlet substrate, exygen, nitrite, nitrate and ammonia BC concentrations
- *Oevery* : call output data every this many timesteps

Examples

```
variable KsHET equal 0.01
variable Ko2HET equal 0.81
.
.
variable BCnh4 equal 9.00e-02

fix g1 all diffnugrowth 1 200 v_KsHET v_Ko2HET v_Kno2HET
v_Kno3HET v_Knh4AOB v_Ko2AOB v_Kno2NOB v_Ko2NOB v_MumHET
v_MumAOB v_MumNOB v_etaHET v_bHET v_bAOB v_bNOB v_bEPS
v_bmHET v_bmAOB v_bmNOB b_bX v_YHET v_YAOB v_YNOB v_YEPS
v_Y1 v_EPSdens v_Do2 v_Dnh4 v_Dno2 v_Dno3 v_Ds v_diffT
v_sub v_o2 v_no2 v_no3 v_nh4 10 10 10 mixed v_BCsub
v_BCo2 v_BCno2 v_BCno3 v_BCnh4 1000
```

Description

Perform growth/decay for the microbes in group. Each of the microbe types (HET, AOB, NOB) consumes different nutrients and competes with each other for survival. The nutrient distribution within the rectangular computational domain is calculated by solving convection-diffusion-reaction equation (transport equation) for each of them (S, NH_4, NO_2, NO_3, O_2).

The transport equation is discretized on a Marker-And-Cell (MAC) uniform grids defined by using nx, ny and nz. The function implements three boundary conditions (*dirich* = Dirichlet BC, *neu* = Neumann BC, *mixed* = mixed BC).

The uptake or consumption rate for each microbe is calculated separately and based on the nutrient concentration of the grid where the microbe belongs to. The growth rate is defined according to Monod-kinetics and biomass decay at a constant rate. The decayed biomass converts to substrate.

4.5 divide command

Syntax

```
fix ID group-ID divide Nevery EPSdens ratio seed
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *divide* : style name of this fix command
- *Nevery* : call the function every this many timesteps
- *EPSdens* : density of EPS
- *ratio* : diameter ratio which division takes place
- *seed* : random seed for cell orientation

Examples

```
fix d1 all divide 500 30 2.0 111
fix d2 HET divide 200 30 2.0 222
```

Description

Perform division for the microbes in group. The function is implemented in following way: If the mass of a bacterial cell becomes greater than twice the mass of an inoculated individual bacterium, it divides into two daughter cells each. During the division process, the cell mass is split in a ratio randomly selected between 0.4-0.6. This generated two daughter cells from a parent cell. These daughter cells are oriented randomly around the centre of the parent cell.

The *EPSdens* setting is required for the divisions of HET particles. Value can be floating type number.

The *ratio* value is calculated as the instantaneous diameter divided by the average diameter of the microbes. The average diameter is initially defined and hard coded. Values typically floating type number greater than 1.0.

4.6 eps_extract command

Syntax

```
fix ID group-ID eps_extract Nevery EPSratio EPSdens seed
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *eps_extract* : style name of this fix command
- *Nevery* : call the function every this many timesteps
- *EPSratio* : ratio between outer-radius and inner-radius of HET
- *EPSdens* : density of EPS
- *seed* : random seed for cell orientation

Example

```
fix d1 HET eps_extract 500 1.25 30 111
```

Description

Microbes secrete extracellular polymeric substances (EPS) every so often as a waste product of their metabolic activities. EPS is secreted into their neighbouring environment and have known to lend structural integrity to the biofilms. The implementation works on the common knowledge that HETs excrete EPS, while AOB and NOB microbes do not. Initially, EPS is accumulated as a extra shell beyond the HET particle. It should be noted that the EPS density is much lower than the HET density. When the relative thickness of the EPS shell bound to HET particle exceeds a certain threshold value, i.e., *EPSratio* value, almost half (random ratio between 0.4-0.6) of the EPS mass excretes as a separate EPS particle and positions next to the HET cell.

4.7 death command

Syntax

```
fix ID group-ID death Nevery deadDia
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *death* : style name of this fix command
- *Nevery* : call the function every this many timesteps
- *deadDiam* : death diameter

Example

```
fix d1 HET death 500 0.8e-6
```

Description

The size of microbe decreases when there is not enough nutrient to uptake. Microbe dies if a threshold is reached. The *deadDiam* parameter defines how small a microbe may become before changing the type to DEAD. The dead microbe keeps shrinking in a constant rate until reaching a system-defined mass value (mass = 1e-18), then the microbe will be removed from the system.

4.8 epsadh command

Syntax

```
fix ID group-ID epsadh Nevery ke model
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *epsadh* : style name of this fix command
- *Nevery* : call the function every this many timesteps
- *ke* : spring stiffness
- *model* : adhesive force model flag (*model =1 or 2*)

Example

```
fix d1 HET epsadh 1 5e+10 1
```

Description

The excreted EPS mass from the HET particles can be employed as a parameter of adhesion force models between the particles. The EPS link between the particles are treated as much more stiffer springs, but only employing the attractive forces. Total effective EPS mass is calculated between the microbes ($Meps_e$), and a spring stiffness is defined per unit mass (k_e). The separation between the EPS links are calculated and the forces calculated according to the effective spring stiffness ($Meps_e$*k_e) multiplied by the separation distance between two particles (model 1), or inverse of the separation distance (model 2).

4.9 walladh command

Syntax

```
fix ID group-ID walladh kanc wallstyle lo hi
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *walladh* : style name of this fix command
- *kanc* : adhesive strength
- *wallstyle* : specify a pair of walls in a dimension (*wallstyle = xplane or yplane or zplane*)
- *lo, hi* : position of lower and upper plane

Examples

```
fix xw all walladh 50 xplane 0.0e-04 1.0e-04
fix yw all walladh 50 yplane 0.0e-04 1.0e-04
```

Description

Impose an adhesive force between the wall (the boundary of simulation domain) and the microbes attaching to the wall. The force is calculated as the product of adhesive strength and overlap distance.

4.10 shear command

Syntax

```
fix ID group-ID shear Nevery viscosity shear-rate height
direction start end
```

- *ID* : user-assigned name for the fix
- *group-ID* : ID of the group of microbes to apply the fix to
- *shear* : style name of this fix command
- *Nevery* : call the function every this many timesteps
- *viscosity*
- *shear-rate*
- *height* : distance to the stationary point from bottom wall
- *direction* : direction of the force (*direction = zx or zy*)
- *start, end* : time range for applying the force

Examples

```
variable viscosity equal 0.5
variable shearRate equal 0.6
variable height equal 5e-5
fix s1 all shear 10 v_viscosity v_shearRate v_height zx 10 500
```

Description

Impose an additional shear force each microbe in the group. The shear force is calculated according to the drag force created on a sphere in Stokes flow. The parameter *height* is a user-defined value where the directions of flow above and blow the stationary point are in opposition.

4.11 thermo_style command

Syntax

```
thermo_style custom args
```

- *args* : list of keywords

  NUFEB keywords = nhet, naob, nnob, neps, ndead mhet, maob, mnob, meps, mdead

  nhet, naob, nnob, neps, ndead = total numbers of HET, AOB, NOB, EPS and DEAD

  mhet, maob, mnob, meps, mdead = biomass of HET, AOB, NOB, EPS and DEAD

Examples

```
thermo_style custom step atoms nhet mhet
```

Description

Set the content for printing thermodynamic data to the screen and log file.

LAMMPS defines various of keywords that allow the user to specify any of them in order to print on each thermodynamic timestep. For example, the *step* keyword in the above example prints out the current timestep. The full list of keywords can be found in the LAMMPS manual: `thermo_style`.

In addition to the LAMMPS default list, the keywords *nhet, naob, nnob, neps, ndead, mhet, maob, mnob, meps* and *mdead* are implemented for monitoring total number and biomass of each microbe.