Non-Equilibrium Reaction Diffusion Self-Assembly Simulator (NERDSS) User Guide

Updated: April 8, 2020

Table of Contents

Dependencies 2

Building NERDSS 2

Starting a Simulation With NERDSS 2

Running a Restart Simulation 3

Input and Output files of NERDSS 4

- **File formats Used** 4
  - MOL files 4
  - INP files 4
  - DAT files 5
  - XYZ files 5
  - Standard output 6
- **NERDSS parameters** 6
  - Parameters in MOL file 6
  - Parameters in INP file 7
- **NERDSS and Important Rule-based Reaction-Rate considerations** 11
- **NERDSS and time-step selection** 12

Creating Input Files With GUI 12

- **General** 13
- **The Molecule Screen** 13
- **The Reactions Screen** 13
- **The Parameters Screen** 16

Troubleshooting 16

- **Debugging** 16
- **Build Errors** 16

Citations 17

Dependencies

- C++ compiler, one of the following
  - GCC ≥ 4.9.0
  - Intel ≥ 16.0
  - Clang ≥ 3.3
- GSL ≥ 1.5
- Make ≥ 4.2.1
- Java 8 or higher (for GUI)

Building NERDSS

Simply run 'make' within the main directory NERDSS/ to use build using the provided make file.

The executable will be placed in the NERDSS/bin directory.

If you wish to build slightly faster, or are having issues building NERDSS as above, you can try using cmake. To do this:

```
mkdir build; cd build
cmake ..
make (may use –j flag to specify number cores to use)
```

The NERDSS Executable will be in the build directory.

Starting a Simulation With NERDSS

In order to start a new simulation, you need a parameter file with .inp extension, and all .mol file required by that parameter file. mol files specify the structure of specific molecule, including the location of all its interfaces, and both translational and rotational diffusion constants. inp files control reaction rules as well as simulation settings. (see Input and Output file section for more info). All input files can be automatically generated using the provided GUI.

If all the input files are correctly formatted, you may run NERDSS with the following command:

./nerdss –f $YOUR_PARAM_FILE

Note that all necessary .mol files (that are referred to by the param file) must be in the same directory as the executable.

Below are some flags that may be helpful:

Flag	Description
-f <filename>, --parmfile <filename>	declares the parameter file (required)
--seed <integer>, -s <integer>	manually declare a seed for RNG (optional)
--debug-force-dissoc	force dissociation to occur whenever possible (optional, for debug only)
--debug-force-assoc	force association to occur whenever possible (optional, for debug only)
-r <filename>, --restart <filename>	declares the restart file (optional)
-a <filename>, --add <filename>	During a restart, will read updated and new parameters, added to or overriding from the original parameter file

Running a Restart Simulation

During a normal run of NERDSS, a special file called restart.dat is written at intervals specified by the user (see .inp file parameters section), A restart simulation allows you to start a new simulation from a timestep in a previous simulation at which a restart file was written. .inp and .mol files are not required for a restart simulation.

To restart a simulation successfully:

- Make sure the following are in the same directory as the executable:
    - Trajectory file (optional, a new trajectory file will be created if not providing trajectory file for restart; If restart the simulation from a checkpoint, modify the input trajectory file to make it consistent with the restart timepoint; or restart the simulation without the input trajectory file and concatenate all the trajectory files after the simulation)
    - RNG state file
    - Restart File (by default restart.dat)

Run restart simulation via: `./nerdss -r $RESTART,` where $RESTART is the restart file.

Restarting and modifying the system:

When restarting, it can be beneficial to adapt the system by changing the parameters or by adding molecules and/or reactions. This can be done via: ./nerdss –r $RESTART –a add_parameters.inp

Input and Output files of NERDSS

File formats Used

MOL files

The MOL format (*.mol) is used for storing molecule template information being read by NERDSS. The *.mol file has the prefix being the molecule with named identically to that used in the reactions, which are defined in the INP format file. The information that can be included in the MOL file are listed here.

- Required: Molecule name, copy numbers of the molecule, translational diffusion constants, rotational diffusion constants, center of mass coordinates, at least one interface coordinates;
- Optional: whether it is a lipid, whether it is an implicit lipid, interface states, pre-defined bonds.

INP files

The INP format (*.inp) is used for storing system information being read by NERDSS, and the format is, as much as possible, shared with BioNetGenLanguage (BNGL) for model portability. It can be used for a new simulation. It can also be used for a restart simulation to change the simulation parameters or adding new molecules and reactions to the previous system. The information that can be included in the INP file are listed here. The information in the *.inp file is stored in different blocks. Blocks are started with the 'start' keyword and ended with the 'end' keyword. Four blocks are used: parameters, boundaries, molecules and reactions. Molecules must be defined before reactions.

- The parameters block can include following things.
  - Required: requested number of iterations, timestep length;
  - Optional: interval to write timestep information, interval to write to coordinates file, interval to update the latest restart file, interval to write separate restart files, interval to write individual pdb files.
- The boundaries block can include following things.
  - Required: water box dimensions, type of the boundary conditions.
  - Optional: isSphere, radius of sphere.
- The molecules block includes the name and starting copy number of the molecules in the system line by line. These should be consistent with the molecule name in the MOL files.
  - One note is that if the system has implicit lipid molecule, you must put it in the first position of the molecules list.
- The reactions block includes all the information of the reactions in the system.
  - Each reaction starts with a declaration like this: A(a) + B(b) <-> A(a!1).B(b!1), where A and B are the reacting molecules, a and b are the reacting interfaces. A(a!1).B(b!1) is the product, where "!" denotes an interaction with index 1, and "." Indicates the two molecules are interacting. Reversible reactions are denoted by a double-headed arrow <->. Interfaces must be uniquely named, at least on each molecule type. States are allowed and are not required to be binary. Denoted with a tilde. An interface can only change its state or interaction, not both.
  - Ancillary interfaces are allowed. These can include interfaces with/without states/interaction that do not change their state or interaction in the particular reaction but are required for the reaction to occur. For example, a molecule A has two interfaces a1 and a2, with a2 having two states P and U. If an interaction between a1 and some interface b on molecule B is dependent on a2 being in the P state, we can write the reaction as: A(a1,a2~P) + B(b) <-> A(a1!1,a2~P).B(b!1). If an interaction between a1 and some interface b on molecule B is dependent on a2 being bound to something, we can write the reaction as: A(a1,a2!*) + B(b) <-> A(a1!1,a2!*).B(b!1). Here we use wildcard * to represent ancillary interactions in the reactants. Another note is that wildcard states are allowed in the reactants/products by omitting the state of an interface which has states. If the state of an ancillary interface does not affect the reaction, it should not be listed. If it is listed, it will be required to be in the state listed.

- The supported reaction types include reversible binding reactions (A(a) + A(a) <-> A(a!1).A(a!.1)), bimolecular association (A(a) + B(b) -> A(a!1).B(b!.1)), bimolecular state change (enzyme facilitated state change)(A(a) + B(a~U) <-> A(a)+B(a~P)), unimolecular state change (A(a~S)->A(a~O)), dissociation (A(a!1).B(b!.1)->A(a)+B(b)), creation from concentration (0 -> A(a)), creation from molecule (A(a) -> A(a) + B(b)), destruction (A(a) -> 0).
- The parameters for a reaction are given below the declaration line by line. Required: 3D microscopic binding rate or macroscopic binding rate, microscopic dissociation rate or macroscopic dissociation rate (only required for reversible reaction). Optional: distance between the two reacting interfaces for a bimolecular reaction (required for bimolecular association), angles for bimolecular association (required for bimolecular association), vector used to calculate the phi1 and phi2 angles (and sometimes omega), label for tracking the reaction product.

DAT files

NERDSS produces most of output files in the DAT format. The DAT files whose name ending with _time.dat include the quantities of the system as a function of time in a specified aspect. The first line of these _time.dat files is the headline and the rest lines are the data recording in a specified interval (the interval parameters are given in the *.inp file). The information of different _time.dat files are listed here.

- observables_time.dat : the time dependence of the numbers of different 'observeLabel's given in the .inp file for each reaction.
- copy_numbers_time.dat: the time dependence of the copy numbers for all the species in the system. This includes all reactant species, and all product species. If products are part of larger complexes, they are still included in this count.
- histogram_complexes_time.dat: the time dependence of the copy numbers for all the complexes in the system.
- mono_dimer_time.dat: the time dependence of the copy numbers for all the monomers and perfect dimers in the system. If a dimer forms another bond (three-mer, etc) it is excluded from this count. If A can dimerize with B, or with C, these are both included in the count.
- bound_pair_time.dat: the time dependence of the copy numbers for all the bound pairs in the system. Bound pairs are the count of all directly bound pairs of protein A and its partner protein B. The fact that A or B may also be bound to other proteins does not matter, so it counts all A-B bonds that exist in the system.

The restart.dat stores all the system information to restart a simulation from the latest step. The rng_state file stores the state of the Random Number Generator of the latest step, which is also needed to restart a simulation. If the corresponding rng_state file is used, the trajectory will follow identically. The restart$timeStep$.dat and rng_state$timeStep$ files include the information for restarting a simulation from a specified timestep. And the interval to write these restart$timeStep$.dat and rng_state$timeStep$ files is determined by the checkpoint parameter in the *.inp file. One thing needed to do before a restart simulation is renaming the corresponding rng_state$timeStep$ to rng_state, and the restart simulation should run in a new directory.

XYZ files

The XYZ format is used for coordinate and trajectory written by NERDSS. These files can be visualized by VMD and Ovito. The initial coordinate of all the molecules in the system are stored in the file named initial_crds.xyz and the final coordinate of all the molecules are stored in the file named final_coords.xyz. The whole trajectory of the system is stored in trajectory.xyz. The interval to write coordinates in the trajectory is determined by the parameter trajWrite in the *.inp file. NOTE: If the total copies of species can change per step, due to creation and destruction, for example, then these will not work in VMD.

PSF files

The PSF format is used to specify the rigid molecules in the system, bonds between them, and their copies, for visualization in VMD. There is no way to update these files if the total copies of species can change per step, due to creation and destruction, for example. Then use PDB files with Ovito.

PDB files
The optional PDB format output is used for coordinate and trajectory written by NERDSS. Unlike the XYZ file that contains the entire trajectory, this creates an individual file for each frame. The advantage is that these files can be read by Ovito, and do not require the same copy numbers per frame. Thus they are needed to visualize open systems where total copies of species can change.

Standard output
NERDSS logs a variety of information of the simulation system to standard output, including the information that is parsed from the input files, the reaction happened in each step, the simulation time information in a fixed timestep interval, and warnings.

NERDSS parameters
Parameters in MOL file
is an indicator for comment in the *.mol file.

- name
  - Acceptable Values: String (must provide)
  - Description: molecule name, must match that used in the *.inp file; also consistent with the name of this *.mol file
  - Sample: name = A
- isLipid
  - Acceptable Values: Boolean (optional)
  - Default Value: false
  - Description: Is the molecule restricted to the 2D surface? I.e., is it a lipid, or transmembrane protein, etc. Also, the Dz value of the molecule
  - Sample: isLipid = true
- isImplicitLipid
  - Acceptable Values: Boolean (optional)
  - Default Value: false
  - Description: Is the molecule an implicit lipid? This is used for simulating binding to a membrane with many lipid binding sites using the implicit lipid model.
  - Sample: isImplicitLipid = true
- checkOverlap
  - Acceptable Values: Boolean (optional)
  - Default Value: false
  - Description: is steric overlap checked for during association for this molecule type? Steric overlap checks are applied then to the two molecules binding, and with each of those molecules against the rest of the system.
  - Sample: checkOverlap = true
- D
  - Acceptable Values: Array, [x, y, z] (must provide)
  - Description: the molecule's xyz translational diffusion constants
  - Unit: $\mu m^2/s$
  - Sample: D = [25.0, 25.0, 25.0]
- Dr
  - Acceptable Values: Array, [x, y, z] (must provide)

    - Description: the molecule's xyz rotational diffusion constants
    - Unit: rad^2/s
    - Sample: Dr = [0.5, 0.5, 0.5]
- COM
    - Acceptable Values: coordinates block (must provide)
    - Description: COM (center-of-mass) denotes the start of the internal coordinates block for the center of mass and all interfaces. interfaceName must be identical to that used in *.inp file.
    - Unit: nm
    - Sample: COM 0.0 0.0 0.0
      interfaceName1 0.0 0.0 1.5
      interfaceName2 0.0 0.0 -1.5
- bonds
    - Acceptable Values: bonds block (optional)
    - Description: Bonds for the molecule are declared strictly for visualization purposes, not to propagate the system (written to PSF file). A pair of interface names separated by whitespace, the first line is the number of bonds.
    - Sample: bonds = 2
      com interfaceName1
      com interfaceName2
- state
    - Acceptable Values: Single character (optional)
    - Description: define interface states with the format $interfaceName~X~Y, must be consistent with the name used in the *.inp file
    - Sample: state = interfaceName1~P~U
- mass
    - Acceptable Values: Float (optional)
    - Default Value: Set to the molecule radius. Molecule radius is calculated from the largest distance COM to interfaces.
    - Description: mass is used only to determine the geometric center of mass of a multi-component complex. Rotation occurs relative to this COM. Mass is effectively unitless, as total mass is divided out.
    - Sample: mass = 1

Parameters in INP file

is an indicator for comment in the *.inp file.

parameters block (between start parameters and end parameters):

- nItr
    - Acceptable Values: Integer (must provide)
    - Description: requested number of iterations
    - Sample: nItr = 10000
- timeStep
    - Acceptable Values: Float (must provide)
    - Description: timestep length per iteration
    - Unit: μs
    - Sample: timeStep = 0.1
- timeWrite
    - Acceptable Values: Integer (optional)
    - Default Value: 10
    - Description: iteration interval to print running time information to standard output and to record the copy numbers in the *_time.dat files.

    - Sample: timeWrite = 100
- trajWrite
    - Acceptable Values: Integer (optional)
    - Default Value: 10
    - Description: iteration interval to write coordinates to trajectory file
    - Sample: trajWrite = 100
- restartWrite
    - Acceptable Values: Integer (optional)
    - Default Value: 10
    - Description: iteration interval to write restart files
    - Sample: restartWrite = 100
- pdbWrite
    - Acceptable Values: Integer (optional)
    - Default Value: -1
    - Description: iteration interval to write pdb files; -1 means no pdb file output
    - Sample: pdbWrite = 100
- checkPoint
    - Acceptable Values: Integer (optional)
    - Default Value: nItr / 10, where nItr is the total number of iterations.
    - Description: iteration interval to write checkpoint for restart
    - Sample: checkPoint = 1000
- overlapSepLimit
    - Acceptable Values: Float (optional)
    - Default Value: 1.5
    - Description: COM-COM distance less than this value is cancelled, for the molecules whose checkOverlap is true.
    - Unit: nm
    - Sample: overlapSepLimit = 3.0

boundaries block (between start boundaries and end boundaries):

- WaterBox
    - Acceptable Values: Array, [x, y, z] (must provide)
    - Description: the xyz dimensions of the simulation system
    - Unit: nm
    - Sample: WaterBox = [500.0, 500.0, 500.0]
- xBCtype/yBCtype/zBCtype
    - Acceptable Values: reflect (optional)
    - Default: reflect
    - Description: the boundary conditions for each dimension
    - Sample: xBCtype = reflect
      yBCtype = reflect
      zBCtype = reflect

- isSphere
    - Acceptable Values: Boolean, True/False (optional)
    - Default: False
    - Sample: isSphere = false
- sphereR
    - Acceptable Values: Float (optional)
    - Default: 0

- Unit: nm
- Sample: sphereR=1000

molecules block (between start molecules and end molecules):

This block includes all the molecules' types in the simulation system, and their starting copy numbers. The name of the molecules should be consistent with the corresponding *.mol file. One note is that the implicit lipid molecule must be placed firstly if exists.

Sample:

ImplicitLipid : Ncopies
moleculeName1 : Ncopies
moleculeName2 : Ncopies

If a molecule has more than one state, those can be initialized with distinct copy numbers. In this example, molecule Kinase with site 'a' will be initialized with 100 copies in state P, and 200 copies in state U.

Sample:

Kinase : 100 (a~P), 200 (a~U)

Here is an example for a molecule pip2 that has two sites 'head' and 'tail', each of which can exist in 2 states, head~U~P and tail~S~D:

Sample

pip2 : 60 (head~U, tail~S), 10 (head~P, tail~D), 10 (head~U, tail~D), 10 (head~P, tail~S)

reactions block (between start reactions and end reactions):

Each reaction starts with the declaration and followed by the corresponding parameter values for this reaction. The syntax of the declaration and the supported reaction types have been given in the INP files Section. Here is the description of the parameters for each reaction.

- onRate3Dka
  - Acceptable Values: Float (one of onRate3Dka and onRate3DMacro must provide)
  - Description: 3D microscopic binding rate
  - Unit: $nm^3/\mu s$ (for 2D will be converted to: $nm^2/\mu s$ by length3Dto2D; for Creation: M/s)
  - Sample: onRate3Dka = 1.0
- onRate3DMacro
  - Acceptable Values: Float (one of onRate3Dka and onRate3DMacro must provide)
  - Description: Macroscopic binding rate, the relationship between 3D microscopic binding rate and macroscopic binding rate for different system can be found in the supporting information of NERDSS paper.
  - Unit: $\mu M^{-1}s^{-1}$ (the relationship between $nm^3/\mu s$ and $\mu M^{-1}s^{-1}$ is $1\ \mu M^{-1}s^{-1} = 1/0.602214076\ nm^3/\mu s$)
  - Sample: onRate3DMacro = 1.0
- offRatekb
  - Acceptable Values: Float (one of offRatekb and offRateMacro must provide for the reversible reaction)
  - Description: Microscopic dissociation rate
  - Unit: s^{-1}
  - Sample: offRatekb = 1.0
- offRateMacro
  - Acceptable Values: Float (one of offRatekb and offRateMacro must provide for the reversible reaction)

    - Description: Macroscopic dissociation rate, the relationship between microscopic dissociation rate and macroscopic dissociation rate for different system can be found in the supporting information of NERDSS paper.
    - Unit: s^{-1}
    - Sample: offRateMacro = 1.0
- rate
    - Acceptable Values: Float
    - Description: Use for zeroth order and for first order. If used for bimolecular, will map to onRate3Dka.
    - Unit: Reaction Order dependent: Zeroth (M/s), First (1/s).
    - Sample: rate = 10.0
- sigma
    - Acceptable Values: Float (optional)
    - Default Value: 1.0
    - Description: distance between the two reacting interfaces for a bimolecular reaction
    - Unit: nm
    - Sample: sigma = 1.0
- norm1/norm2
    - Acceptable Values: Vector (optional)
    - Default: [0, 0, 1]
    - Description: vectors used to calculate the phi and phi2 angles (and sometimes omega) for a bimolecular reaction, the definition can be found in the supporting information of NERDSS paper.
    - Sample: norm1 = [0, 0, 1]
      norm2 = [0, 0, 1]
- assocAngles
    - Acceptable Values: Vector (optional)
    - Default: [nan, nan, nan, nan, nan]
    - Description: five angles for bimolecular association, the definition can be found in the supporting information of NERDSS paper. If the angle does not exist, the angle should read "nan". M_PI is Pi (3.14159). For all 'nan', the binding partners are placed at the orientation they are prior to the association event.
    - Unit: rad
    - Sample: assocAngles = [1.5707963, 1.5707963, nan, nan, M_PI]
- length3Dto2D
    - Acceptable Values: Float (optional)
    - Default Value: 2*sigma
    - Description: length scale to convert 3D rate to 2D rate
    - Unit: nm
    - Sample: length3Dto2D = 30
- bindRadSameCom
    - Acceptable Values: Float (optional)
    - Default Value: 1.1
    - Description: scalar multiple of sigma, determines distance between two reactants to force reaction within the same complex
    - Unit: unitless
    - Sample: bindRadSameCom = 1.1
- loopCoopFactor
    - Acceptable Values: Float (optional)
    - Default Value: 1.0 (e.g. exp(-0))

    - Description: multiple the rate by this scale factor, used only when closing loops, such as occurs, for example, within a hexagonal lattice. lCF=exp(-$\Delta G_{coop}/k_BT$).
    - Unit: unitless
    - Sample: loopCoopFactor = 0.001
- observeLabel
    - Acceptable Values: String (optional)
    - Description: label for tracking the reaction product, the copy numbers of each observeLabel storing in obervables_time.dat
    - Sample: observeLabel = leg
- rxnLabel
    - Acceptable Values: String (optional)
    - Description: give the reaction a name. This is helpful if you want to couple a different reaction to this one.
    - Sample: rxnLabel = phosphorylateA
- coupledRxnLabel
    - Acceptable Values: String from 'rxnLabel' (optional)
    - Description: This allows the completion of a reaction to immediately cause another reaction to happen. The reaction that it triggers must already be a listed reaction, but it will not occur with the rate for that reaction. It will occur with rate 'kcat' (see next entry). If kcat is not specified, it will not occur. The reaction it is coupled to must have its own rxnLabel (see previous entry) set, so that this can be mapped to it. Only applies to products of a reaction. Couples currently to dissociation only.
    - Sample: coupledRxnLabel = phosphorylateA
- kcat
    - Acceptable Values: Float (optional)
    - Description: For a coupledRxn, will occur with this rate. Therefore, will not be used unless coupledRxnLabel is also specified (see previous entry). Michaelis-Menten reactions, for example, can be defined via a reversible bimolecular reaction, with coupledRxnLabel specified as pointing to a unimolecular reaction that changes state, and kcat specified.
    - Units: s-1.
    - Sample: kcat = 1.0 #s-1. For example see sample_inputs/VALIDATE_SUITE/michaelis_menten/
- excludeVolumeBound
    - Acceptable Values: Bool (optional)
    - Default: False
    - Description: Once two sites are in the bound site (where being bound, they cannot undergo additional bimolecular reactions), by default they will not try to bind and therefore will not exclude volume with any other sites.
    - Sample: excludeVolumeBound = false: NOT YET ACTIVE IN CODE

NERDSS and Important Rule-based Reaction-Rate considerations

- An important property must be noted when a molecule has multiple interfaces, A(a,b), and can bind itself, with a distinct interface: e.g. A(a)+A(b) <->A(a!1).A(b!1). These reactions will read in k_a^{3D} values, such that $K_D=k_b/k_a^{3D}$, with no automatic re-scaling. This is fine for a molecule like Actin, that binds to itself using two distinct interfaces (A(barbed end)+A(pointed end)->). However, for models such as clathrin, the distinction between an 'a' site and a 'b' site is purely a label to distinguish their coordinates in space. In reality, they are all structurally/chemically identical sites, and all interactions are possible (A(a)+A(a)->, A(a)+A(b)->, A(b)+A(b)->). To correct for this distinction arising purely due to

labeling, for these specific cases, **users should input rates multiplied by two**, relative to the case where both labels are the same. For example:

- A(a)+A(a) <-> A(a!1).A(a!1) onRate3Dka=1
- A(a)+A(b)<-> A(a!1).A(b!1) onRate3Dka=2

The rate for the true self interaction (i.e. onRate3Dka=1 in the example above), is the one that corresponds to the K_D given all identical sites. If this change is not done by the user, and the same rates are used for the A(a)+A(a) binding for clathrin sites, as for the A(a)+A(b) binding for clathrin sites, the equilibrium will not match what is thermodynamically expected. This is also discussed as well in Ref. 2, and applies to all rule-based, rate-based reactions (noted in NFSim—here they automatically divide self-rates by 2). What we want is that 2*A(a!).A(a!)=A(a!).A(b!) = A(a!).A(c!). Same for sites b and c. Then, each product/complex type has the same number of 'a' sites bound, due to having the same binding free energy. This result emerges due to the combinatorics of choosing two of the same molecules from $A_{tot}=N_0$, vs one molecule A from $A_{tot}=N_0$ and one molecule B from $B_{tot}=N_0$. They produce an identical equilibrium with the same number of A in bound states when $A_{eq}^{self}(K_D)=A_{eq}^{distinct}(K_D/2)=B_{eq}^{distinct}(K_D/2)$.

- As described in the NERDSS paper, the reaction rates are input by the user as the 3D values, k_a^{3D}, and the code will convert them to values needed for the appropriate reaction kinetics. We always have that $K_D= k_b/k_a^{3D}= k_{off}/k_{on}^{3D}$, based on the user-input rates. For information on what rates are needed to give to the Green's function to recover appropriate reaction kinetics, see Ref 2, SI.

NERDSS and time-step selection

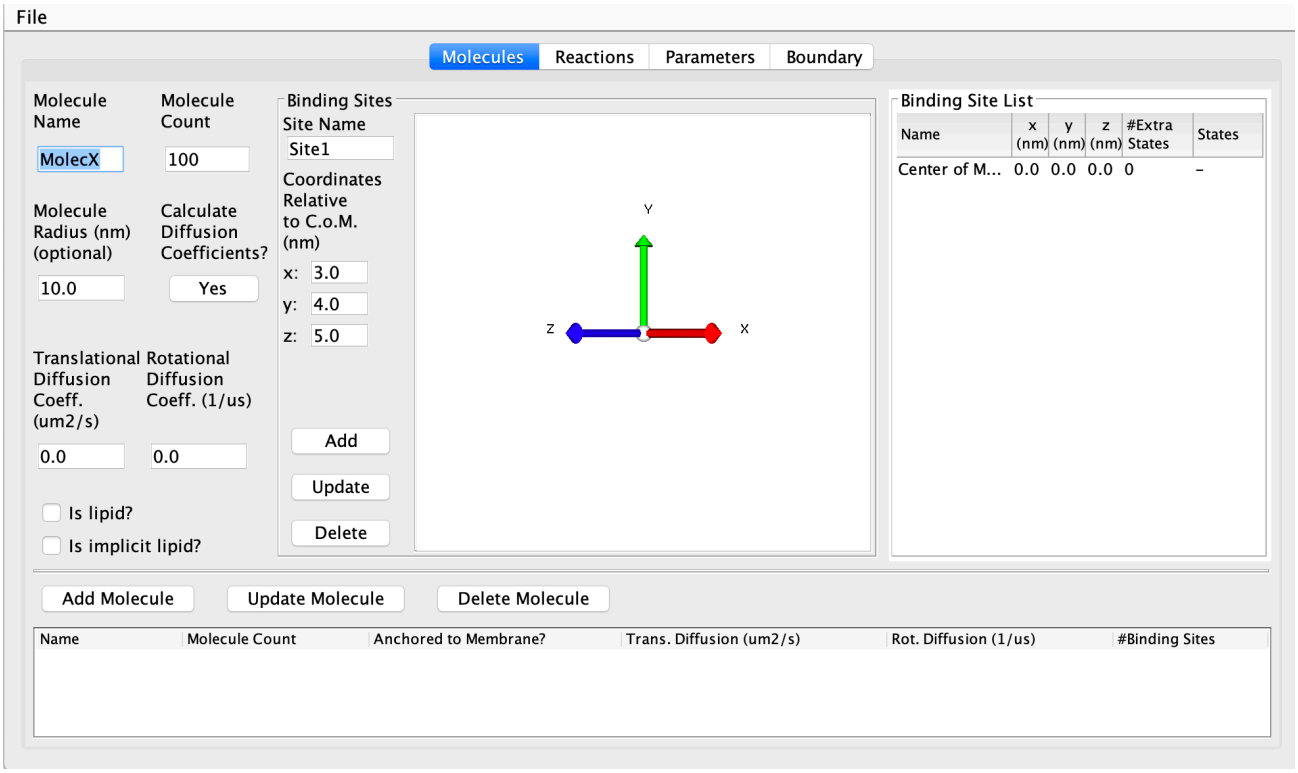
- To maintain at most two-body interactions between all reactant pairs per time-step (approximately), the time-step is limited (usually) by the bimolecular reactions and the density of those reactants, and their diffusion coefficients. Based on the definition of R_{max} for the FPR algorithm [3], or the maximum distance a particle could diffuse in a step and collide with a partner, a minimum time-step for a given reactant B due to binding to A in 3D is: $\Delta t^{3D} = \frac{1}{56(D_A+D_B)}\left[\left(\frac{3}{4\pi\rho_A}+\sigma^3\right)^{1/3}-\sigma\right]^2$, where $\rho_A = \frac{N_A}{V}$. A minimum time-step for a given reactant B due to binding to A in 2D is: $\Delta t^{2D} = \frac{1}{36(D_A+D_B)}\left[\left(\frac{1}{\pi\rho_A}+\sigma^2\right)^{1/2}-\sigma\right]^2$, where $\rho_A = \frac{N_A}{Area}$, and the diffusion coefficients are the (slower) values in 2D, on the surface. These can be approximated as comparable to the lipid diffusion coefficients (~0.5μm^2/s). The dependence on σ is weak as it nearly cancels for most systems. The time-step for the system is the minimum of all the time-steps, we typically set default values of 0.1-1μs.
- NERDSS will still propagate dynamics effectively with steps larger than the minimum suggested above. Particularly if systems spend minimal time in their densest state, the error due to choosing a larger step will be small, and empirically, the error for many-body systems is relatively small even when the two-body requirement is not closely maintained. Nonetheless, a smaller time-step will ensure more accurate propagation of dynamics.

Creating Input Files With GUI

General

This is a Graphical Interface for creating input files for NERDSS. It allows you to define all the molecules, reactions, boundary conditions, and other parameters for your simulation.

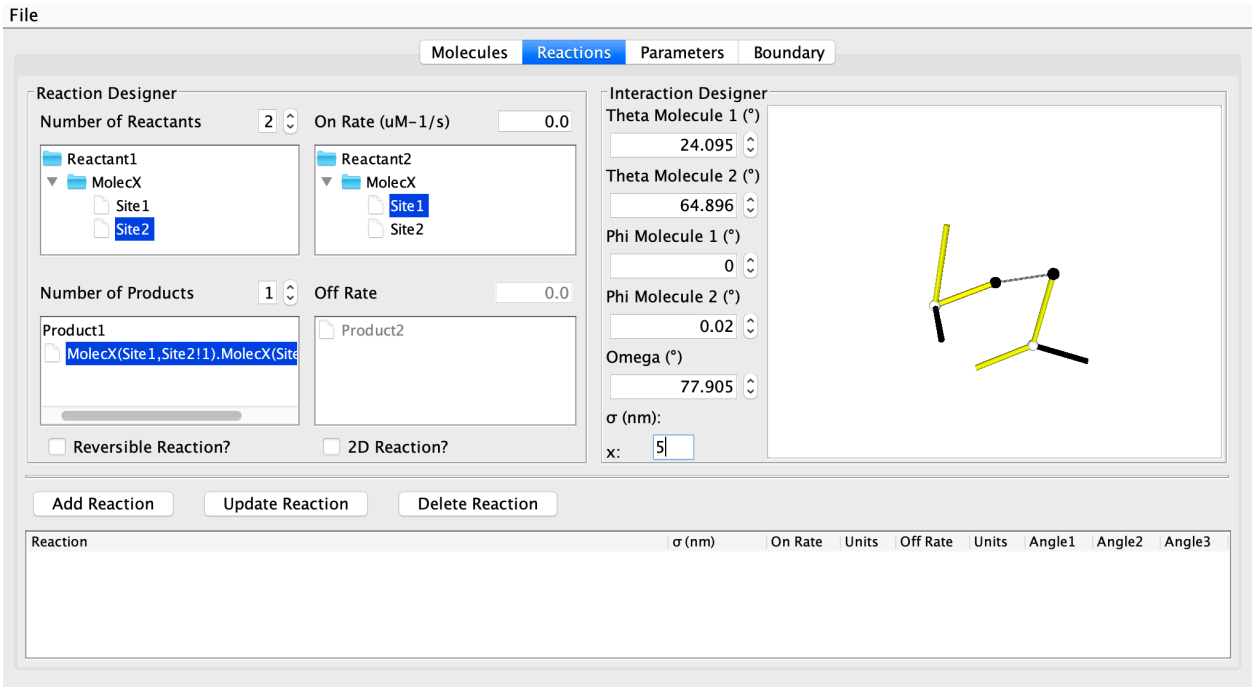
The Molecule Screen



Above is the opening screen for the GUI. Here you can add all the individual molecules that will be included in your simulation. You must also specify each binding site as a vector from its molecule's center of mass. Binding will occur at the terminal end of this interface vector. Each binding site must be added a molecule, and the molecule must be added to the molecule list, in order for it to appear on the reactions screen.

The Reactions Screen

This is where you can specify all possible reactions. You can do both unimolecular and bi molecular reaction. Since there are no association angles for a unimolecular reaction, the 3D pane on the right will only appear once a bimolecular reaction is specified.

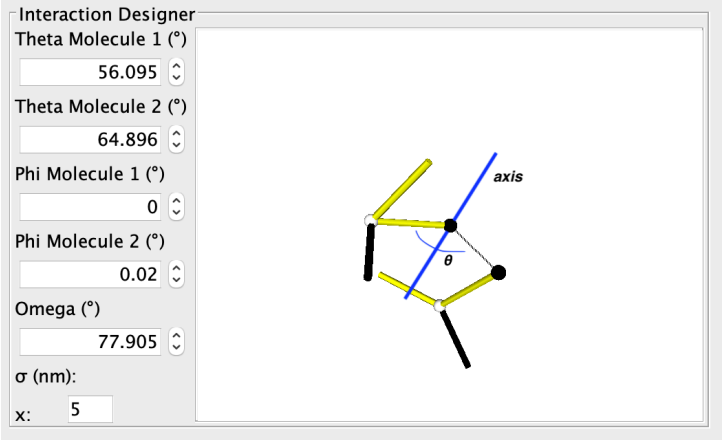


Sigma: Sigma is the association distance (in nm) for a reaction between two interfaces. It is represented as the dotted line between the two interface vectors. Only its length may be changed - positions of both other molecules are defined relative to it.

The Normal Vector: The solid black line is the normal vector. It does not have any physical significance, but is instead used as a reference point for the phi angle, which measure the rotation of the molecule about a given interface vector.

The Angles:
Five angles are used to specify the geometry of association.

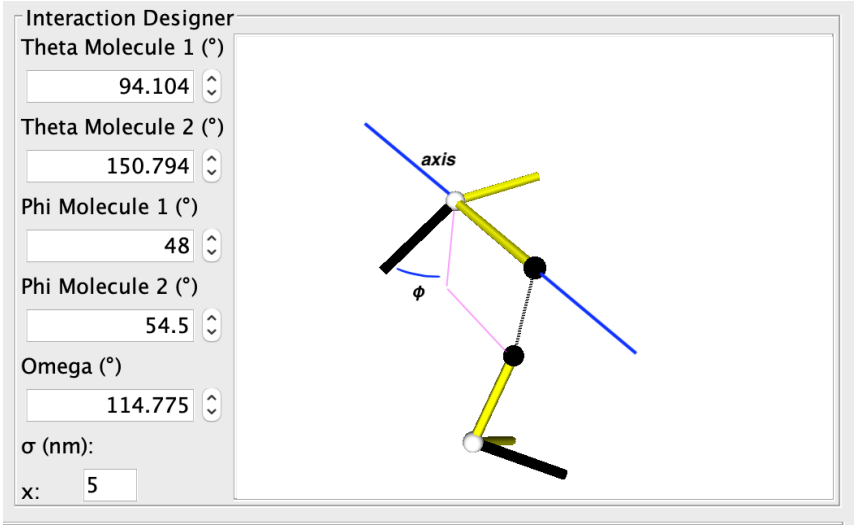
Theta (one for each molecule) ***[0 < Θ < π]***
Theta is the angle between the interface vector and sigma. It's axis of rotation is the cross product of the sigma vector and a given interface vector.



Phi (one for each molecule) ***[-π < Φ < π]*** *

**note that this range is converted to [0 < Φ < 2π] when angles are output to .parms file.*

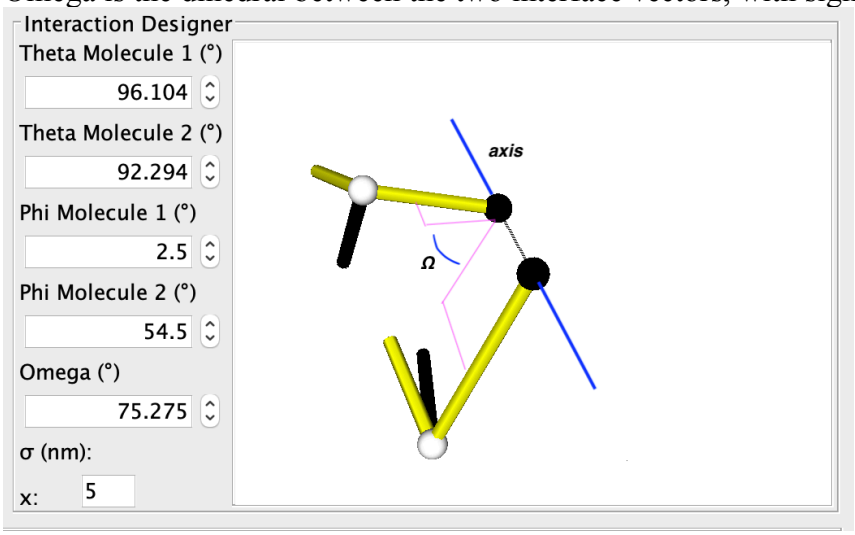
Phi is the dihedral angle between the normal vector and the sigma vector. It can be thought of as defining the rotation of a molecule about its interface vector.



Omega ***[-π < Ω < π]****

*** note that this range is converted to [0 < Φ < 2π] when angles are output to .parms file.***

Omega is the dihedral between the two interface vectors, with sigma has the axis of rotation.



***There may be some small unexpected graphical behavior when changing omega. The display is corrected quickly and this will not affect simulations.*

Note that the angle spinners may not be able exactly reach boundary conditions (e.g. theta may not get all the way to 180 degrees or all the way down to -180 degrees). When simulation input files are created, these angles will be set equal to the boundaries. The GUI will output angles at a precision of 1 degree. Also note that the angles are displayed in degrees on the spinners for easy interpretability, but are always stored internally and written to the .parm file in radians.

On Rate: onRate3Dka This is required for all reactions. Specifies the association rate constant.

Off Rate: offRatekb Only needed for reversible reactions. Specifies the disassociation rate constant. Do not specify that that a reaction is reversible and then leave the Off Rate at 0. This may cause errors.

2D Reaction: Weather the reaction is limited to species on a 2D surface (e.g. a membrane).

Managing Reactions: Use the Add reaction button to add the current reaction to the simulation. To update a reaction, first select a reaction from the list, then change its parameters, and finally click Update Reaction. To remove a reaction from the simulation, first select a reaction from the list, then click Delete Reaction.

The Parameters Screen

Time Step: The length of time in μs between each iteration of the simulation.

Number of Time Steps: The length of the simulation in iterations.

Freq to Print Configuration: How many iterations are between each print of the current configuration.

Freq to Print Statistics: How many iterations are between each print of the current statistics.

Freq to Print Restart File: How often to print a restart file (in iterations) – the restart file can be used to start a new simulation at a later date from this point in the current simulation

The Boundary Screen

As of right now, the GUI can only output simulation with a box boundary condition. The output parameter files may be edited manually for spherical boundaries.

Simply specify the box dimensions along the x, y, and z axis.

Create parameter file:
To output .mol and .parm files to run simulations with, click Create Parameter Input Files. Then select the directory to place them in. To run simulations, place these files in the same directory as the Nerdss executable.

Troubleshooting

Debugging

Can do general debugging by building NERDSS using: cmake -DCMAKE_BUILD_TYPE=Debug
Instead of just normal cmake.

- `-g`: generate debugging tables for GDB/LLDB
- `-fsanitize=address`: Uses a sanitizer to give information about memory access issues, such as out-of-range element access
- `-fno-omit-frame-pointer`: stores the stack frame pointers for functions (needed for -fsanitize)

Build Errors

Build issues may occur on system configurations that NERDSS was not tested on, particularly on super computers and large clusters. A common issue is cmake having issues finding GSL when it is installed in a non-standard location. This is easily solved by setting the GSL_ROOT_DIR environment variable using: `export GSL_ROOT_DIR="correct_path"`

If make cannot find a recipe for 'all', (Makefile:83: recipe for target 'all' failed), specify the target for NERDSS, e.g. `make nerdss`

Citations

1. Varga, M.◇, Y. Fu◇, S. Loggia, O.N. Yogurtcu, and M.E. Johnson. NERDSS: a non-equilibrium simulator for multibody self-assembly at the cellular scale. *In Revision.*

2. Johnson, M. E. Modeling the Self-Assembly of Protein Complexes through a Rigid-Body Rotational Reaction–Diffusion Algorithm. J. of Phys. Chem. B 2018, 122, 11771–11783.

3. Johnson, M.E., and G. Hummer. Free propagator reweighting integrator for single-particle dynamics in reaction-diffusion models of heterogeneous protein-protein interactions systems. Phys. Rev. X 4, 031037 (2014)

4. Yogurtcu, O.N., and M.E. Johnson. Theory of bi-molecular association dynamics in 2D for accurate model and experimental parameterization of binding rates. J. Chem. Phys. 143, 084117 (2015)

5. Yogurtcu, O.N., and M.E. Johnson. Cytoplasmic proteins can exploit membrane localization to trigger functional assembly. PLoS Comput. Biology 14, e1006031 (2018)