mkdoctest

Jun 12, 2019

Contents

CHAPTER 1

Energy

The system energy, or Hamiltonian, consists of a sum of potential energy terms,

$$ \mathcal{H}_{sys} = U_1 + U_2 + … $$

The energy terms are specified in `energy` at the top level input and evaluated in the order given. For example:

```
energy:
    - isobaric: {P/atm: 1}
    - sasa: {molarity: 0.2, radius: 1.4 }
    - confine: {type: sphere, radius: 10,
                molecules: [water]}
    - nonbonded:
        default: # applied to all atoms
        - lennardjones: {mixing: LB}
        - coulomb: {type: plain, epsr: 1, cutoff: 12}
        Na CH:    # overwrite specific atom pairs
        - wca: { mixing: LB }

    - maxenergy: 100
    - ...
```

The keyword `maxenergy` can be used to skip further energy evaluation if a term returns a large energy change (in kT), which will likely lead to rejection. The default value is *infinity*.

Note: *Energies* in MC may contain implicit degrees of freedom, *i.e.* be temperature-dependent, effective potentials. This is inconsequential for sampling density of states, but care should be taken when interpreting derived functions such as energies, entropies, pressure etc. {: .notice–info}

1.1 Infinite and NaN Energies

In case one or more potential energy terms of the system Hamiltonian returns infinite or NaN energies, a set of conditions exists to evaluate the acceptance of the proposed move:

- always reject if new energy is NaN (i.e. division by zero)

- always accept if energy change is from NaN to finite energy
- always accept if the energy *difference* is NaN (i.e. from infinity to minus infinity)

Note: These conditions should be carefully considered if equilibrating a system far from equilibrium. {: .notice–notice}

1.2 External Pressure

This adds the following pressure term[^frenkel] to the Hamiltonian, appropriate for MC moves in $\ln V$:

$$ U = PV - k_BT\left (N + 1 \right) \ln V $$

where N is the total number of molecules and atomic species.

[^frenkel]: *Frenkel and Smith, 2nd Ed., Chapter 5.4.*

1.3 Nonbonded Interactions

This term loops over pairs of atoms, i, and j, summing a given pair-wise additive potential, u_{ij},

$$ U = \sum_{i=0}^{N-1}\sum_{j=i+1}^N u_{ij}(\textbf{r}_j-\textbf{r}_i)$$

Using `nonbonded`, potentials can be arbitrarily mixed and customized for specific particle combinations. Internally, the potential is *splined* in an interval [`rmin`,`rmax`] determined by the following policies:

- `rmin` is decreased towards zero until the potential reaches `u_at_rmin=20` kT
- `rmax` is increased until the potential reaches `u_at_rmax=1e-6` kT

If outside the interval, infinity or zero is returned, respectively. Finally, the spline precision can be controlled with `utol=1e-5` kT.

Below is a description of possible nonbonded methods. For simple potentials, the hard coded variants are often the fastest option.

1.3.1 Mass Center Cut-offs

For cut-off based pair-potentials working between large molecules, it can be efficient to use mass center cut-offs between molecular groups, thus skipping all pair-interactions. A single cut-off can be used between all molecules (`default`), or specified for specific combinations:

```
- nonbonded:
    cutoff_g2g:
        default: 40
        "protein water": 60
```

1.3.2 OpenMP Control

If compiled with OpenMP, the following keywords can be used to control parallelisation for non-bonded interactions. The best combination depends on the simulated system size and composition. Currently, parallelisation is disabled by default.

```
- nonbonded:
    openmp: [g2g, i2all]
```

1.4 Electrostatics

This is a multipurpose potential that handles several electrostatic methods. Beyond a spherical real-space cutoff, R_c, the potential is zero while if below,

$$ u_{ij} = \frac{e^2 z_i z_j }{ 4\pi\epsilon_0\epsilon_r r_{ij} }\mathcal{S}(q) $$

where $\mathcal{S}(q=r/R_c)$ is a splitting function:

Note: Internally $\mathcal{S}(q)$ is *splined* whereby all types evaluate at similar speed. {: .notice–info}

1.4.1 Ewald Summation

If type is `ewald`, terms from reciprocal space; surface energies; and self energies are automatically added to the Hamiltonian, activating additional keywords:

The added energy terms are:

$$ \small \begin{aligned} U =& \overbrace{\frac{2\pi f}{V}\sum_{ {\bf k} \ne {\bf 0}} A_k\vert Q^{q\mu} \vert^2}^{\text{reciprocal}}

- \overbrace{ f \sum_{j} \left(\frac{\alpha}{\sqrt{\pi}}q_j^2 + \frac{2\alpha^3}{3\sqrt{\pi}}\vert{\boldsymbol{\mu}}j\vert^2 \right)}^{\text{self}}\ &+ \underbrace{\frac{2\pi f}{(2\varepsilon{surf} + 1)V}\left(\vert \sum_{j}q_j{\bf r}j \vert^2 + 2\sum{j}q_i{\bf r}j \cdot \sum{j}{\boldsymbol{\mu}}j + \vert \sum{j}{\boldsymbol{\mu}}j \vert^2 \right)}{\text{surface}}\ \end{aligned} $$

where

$$ f = \frac{1}{4\pi\varepsilon_0\varepsilon_r} \quad\quad V=L_xL_yL_z $$

$$ A_k = \frac{e^{-k^2/4\alpha^2}}{k^2} \quad \quad Q^{q\mu} = \sum_{j}q_j + i({\boldsymbol{\mu}}_j\cdot {\bf k}) e^{i({\bf k}\cdot {\bf r}_j)} $$

$$ {\bf k} = 2\pi\left(\frac{n_x}{L_x} , \frac{n_y}{L_y} ,\frac{n_z}{L_z} \right),;; {\bf n} \in \mathbb{Z}^3 $$

In the case of isotropic periodic boundaries (`ipbc=true`), the orientational degeneracy of the periodic unit cell is exploited to mimic an isotropic environment, reducing the number of wave-vectors by one fourth compared with PBC Ewald. For point charges, IPBC introduce the modification,

$$ Q^q = \sum_jq_j\prod_{\alpha\in{x,y,z}}\cos\left(\frac{2\pi}{L_{\alpha}}n_{\alpha} r_{\alpha,j}\right) $$

while for point dipoles (currently unimplemented),

$$ Q^{\mu} = \sum_j\boldsymbol{\mu}j\cdot\nabla_j\left(\prod{\alpha \in{x,y,z}}\cos\left(\frac{2\pi}{L_{\alpha}}n_{\alpha}r_{\alph
$$

Limitations: Ewald summation requires a constant number of particles, i.e. $\mu V T$ ensembles and Widom insertion are currently unsupported. {: .notice–info}

1.4.2 Mean-Field Correction

For cuboidal slit geometries, a correcting mean-field, external potential, $\varphi(z)$, from charges outside the box can be iteratively generated by averaging the charge density, $\rho(z)$, in dz-thick slices along z. This correction assumes that all charges interact with a plain Coulomb potential and that a cubic cutoff is used via the minimum image convention.

To enable the correction, use the `akesson` keyword at the top level of `energy`:

The density is updated every `nstep` energy calls, while the external potential can be updated slower (`nphi`) since it affects the ensemble. A reasonable value of `nstep` is system dependent and can be a rather large value. Updating the external potential on the fly leads to energy drifts that decrease for consecutive runs. Production runs should always be performed with `fixed=true` and a well converged $\rho(z)$.

At the end of simulation, `file` is overwritten unless `fixed=true`.

1.5 Pair Potentials

In addition to the Coulombic pair-potentials described above, a number of other pair-potentials can be used. Through the C++ API, it is easy to add new potentials.

1.5.1 Charge-Nonpolar

The energy when the field from a point charge, z_i, induces a dipole in a polarizable particle of unit-less excess polarizability, $\alpha_j=\left (\frac{\epsilon_j-\epsilon_r}{\epsilon_j+2\epsilon_r}\right) a_j^3$, is

$$ \beta u_{ij} = -\frac{\lambda_B z_i^2 \alpha_j}{2r_{ij}^4} $$

where a_j is the radius of the non-polar particle and α_j is set in the atom topology, `alphax`. For non-polar particles in a polar medium, α_i is a negative number. For more information, see J. Israelachvili's book, Chapter 5.

Limitations: Charge-polarizability products for each pair of species is evaluated once during construction and based on the defined atom types. {: .notice–info}

1.5.2 Cosine Attraction

An attractive potential used for coarse grained lipids and with the form,

$$ \beta u(r) = -\epsilon \cos^2 \left (\frac{\pi(r-r_c)}{2w_c} \right) $$

for $r_c\leq r \leq r_c+w_c$. For $r<r_c$, $\beta u=-\epsilon$, while zero for $r>r_c+w_c$.

1.5.3 Hard Sphere

`hardsphere`

The hard sphere potential does not take any input. Radii are read from the atomlist at the beginning of the simulation.

1.5.4 Lennard-Jones

The Lennard-Jones potential consists of a repulsive and attractive term,

$$ u_{ij}^{\text{LJ}} = 4\epsilon_{ij} \left (\left (\frac{\sigma_{ij}} {r_{ij})} \right)^{12} - \left (\frac{\sigma_{ij}}{r_{ij})}\right)^6 \right) $$

and currently only the Lorentz-Berthelot (`LB`) mixing rule is available:

$$ \sigma_{ij} = \frac{\sigma_i+\sigma_j}{2} \quad \textrm{and} \quad \epsilon_{ij} = \sqrt{\epsilon_i \epsilon_j} $$

The mixing rule can be overridden for specific pairs of atoms:

```
lennardjones:
    mixing: LB
    custom:
        "Na Cl": {eps: 0.2, sigma: 2}
        "K Cl": {eps: 0.1, sigma: 3}
```

1.5.5 Weeks-Chandler-Andersen

Like Lennard-Jones but cut and shifted to zero at the minimum, forming a purely repulsive potential,

$$ u_{ij} = u_{ij}^{\text{LJ}} + \epsilon_{ij} \quad \textrm{for} \quad r<2^{1/6}\sigma_{ij} $$

1.5.6 SASA

This calculates the surface area of two intersecting particles or radii R and r to estimate an energy based on transfer-free-energies (TFE) and surface tension. The total surface area is calculated as

$$ A = 4\pi \left (R^2 + r^2 \right) - 2\pi \left (Rh_1 + rh_2 \right) $$

where h_1 and h_2 are the heights of the spherical caps comprising the lens formed by the overlapping spheres. For complete overlap, or when far apart, the full area of the bigger sphere or the sum of both spheres are returned. The pair-energy is calculated as:

$$ u_{ij} = A \left (\gamma_{ij} + c_s \varepsilon_{\text{tfe},ij} \right) $$

where γ_{ij} and $\varepsilon_{\text{tfe},ij}$ are the arithmetic means of `tension` and `tfe` provided in the atomlist.

Note that SASA is strictly not additive and this pair-potential is merely a poor-mans way of approximately take into account ion-specificity and hydrophobic/hydrophilic interactions.

1.5.7 Custom

This takes a user-defined expression and a list of constants to produce a runtime, custom pair-potential. While perhaps not as computationally efficient as hard-coded potentials, it is a convenient way to access alien potentials. Further, used in combination with `nonbonded` there is no overhead since all potentials are splined.

The following illustrates how to define a Yukawa potential:

```
custom:
    function: lB * q1 * q2 / r * exp( -r/D ) # in kT
    constants:
        lB: 7.1  # Bjerrum length
        D: 30    # Debye length
```

The function is passed using the efficient ExprTk library and a rich set of mathematical functions and logic is available. In addition to user-defined constants, the following symbols are defined:

1.6 Custom External Potential

This applies a custom expernal potential to atoms or molecular mass centra using the ExprTk library syntax.

In addition to user-defined `constants`, the following symbols are available:

If `com=true`, charge refers to the molecular net-charge, and `x,y,z` the mass-center coordinates. The following illustrates how to confine molecules in a spherical shell of radius, *r*, and thickness *dr*:

```
customexternal:
    molecules: [water]
    com: true
    constants: {radius: 15, dr: 3}
    function: >
        var r2 := x^2 + y^2 + z^2;
        if ( r2 < radius^2 )
           1000 * ( radius-sqrt(r2) )^2;
        else if ( r2 > (radius+dr)^2 )
           1000 * ( radius+dr-sqrt(r2) )^2;
        else
           0;
```

1.7 Bonded Interactions

Bonds and angular potentials are added via the keyword `bondlist` either directly in a molecule definition (topology) or in energy/bonded where the latter can be used to add inter-molecular bonds:

```
energy:
    - bonded:
        bondlist: # absolute index
           - harmonic: { index: [56,921], k: 10, req: 15 }
moleculelist:
    - water: # TIP3P
        structure: "water.xyz"
        bondlist: # index relative to molecule
            - harmonic: { index: [0,1], k: 5024, req: 0.9572 }
            - harmonic: { index: [0,2], k: 5024, req: 0.9572 }
            - harmonic_torsion: { index: [1,0,2], k: 628, aeq: 104.52 }
```

Bonded potential types:

Note: $\mu V T$ ensembles and Widom insertion are currently unsupported for molecules with bonds. {: .notice–info}

1.7.1 Harmonic

$$ u(r) = \frac{1}{2}k(r-r_\mathrm{eq})^2 $$

1.7.2 Finite Extensible Nonlinear Elastic

Finite extensible nonlinear elastic potential long range repulsive potential.

$$ u(r) = \begin{cases} -\frac{1}{2} k r_{\mathrm{max}}^2 \ln \left [1-(r/r_{\mathrm{max}})^2 \right], & \text{if } r < r_{\mathrm{max}} \ \infty, & \text{if } r \geq r_{\mathrm{max}} \end{cases} $$

Note: It is recommend to only use the potential if the initial configuration is near equilibrium, which prevalently depends on the value of `rmax`. Should one insist on conducting simulations far from equilibrium, a large displacement parameter is recommended to reach finite energies. {: .notice–info}

1.7.3 Finite Extensible Nonlinear Elastic + WCA

Finite extensible nonlinear elastic potential long range repulsive potential combined with the short ranged Weeks-Chandler-Anderson (wca) repulsive potential. This potential is particularly useful in combination with the `nonbonded_cached` nobonded energy.

$$ u(r) = \begin{cases} -\frac{1}{2} k r_{\mathrm{max}}^2 \ln \left [1-(r/r_{\mathrm{max}})^2 \right] + u_{\mathrm{wca}}, & \text{if } 0 < r \leq 2^{1/6}\sigma \ -\frac{1}{2} k r_{\mathrm{max}}^2 \ln \left [1-(r/r_{\mathrm{max}})^2 \right], & \text{if } 2^{1/6}\sigma < r < r_{\mathrm{max}} \ \infty, & \text{if } r \geq r_{\mathrm{max}} \end{cases} $$

Note: It is recommend to only use the potential if the initial configuration is near equilibrium, which prevalently depends on the value of `rmax`. Should one insist on conducting simulations far from equilibrium, a large displacement parameter is recommended to reach finite energies. {: .notice–info}

1.7.4 Harmonic torsion

$$ u(r) = \frac{1}{2}k(\alpha - \alpha_\mathrm{eq})^2 $$

1.7.5 Cosine based torsion (GROMOS-96)

$$ u(r) = \frac{1}{2}k(\cos(\alpha) - \cos(\alpha_\mathrm{eq}))^2 $$

1.7.6 Proper periodic dihedral

$$ u(r) = k(1 + \cos(n\phi - \phi_\mathrm{syn})) $$

1.8 Geometrical Confinement

Confines `molecules` in a given region of the simulation container by applying a harmonic potential on exterior atom positions, $\mathbf{r}_i$:

$$ U = \frac{1}{2} k \sum^{\text{exterior}} f_i $$

where f_i is a function that depends on the confinement `type`, and k is a spring constant. The latter may be *infinite* which renders the exterior region strictly inaccessible and may evaluate faster than for finite values. During equilibration it is advised to use a *finite* spring constant to drive exterior particles inside the region.

Note: Should you insist on equilibrating with $k=\infty$, ensure that displacement parameters are large enough to transport molecules inside the allowed region, or all moves may be rejected. Further, some analysis routines have undefined behavior for configurations with infinite energies. {: .notice–danger}

Available values for `type` and their additional keywords:

The `scale` option will ensure that the confining radius is scaled whenever the simulation volume is scaled. This could for example be during a virtual volume move (analysis) or a volume move in the NPT ensemble.

where $\mathbf{d}=(1,1,0)$ and $\circ$ is the entrywise (Hadamard) product.

where δr are distances to the confining, cuboidal faces. Note that the elements of `low` must be smaller than or equal to the corresponding elements of `high`.

1.9 Solvent Accessible Surface Area

Calculates the free energy contribution due to

1. atomic surface tension
2. co-solute concentration (typically electrolytes)

via a SASA calculation for each atom. The energy term is:

$$ U = \sum_i^N A_{\text{sasa},i} \left (\gamma_i + c_s \varepsilon_{\text{tfe},i} \right) $$

where c_s is the molar concentration of the co-solute; γ_i is the atomic surface tension; and $\varepsilon_{\text{tfe},i}$ the atomic transfer free energy, both specified in the atom topology with `tension` and `tfe`, respectively.

1.10 Penalty Function

This is a version of the flat histogram or Wang-Landau sampling method where an automatically generated bias or penalty function, $f(\mathcal{X}^d)$, is applied to the system along a one dimensional ($d=1$) or two dimensional ($d=2$) reaction coordinate, $\mathcal{X}^d$, so that the configurational integral reads,

$$ Z(\mathcal{X}^d) = e^{-\beta f(\mathcal{X}^d)} \int e^{-\beta \mathcal{H}(\mathcal{R}, \mathcal{X}^d)} d \mathcal{R}. $$

where $\mathcal{R}$ denotes configurational space at a given $\mathcal{X}$. For every visit to a state along the coordinate, a small penalty energy, f_0, is added to $f(\mathcal{X}^d)$ until Z is equal for all $\mathcal{X}$. Thus, during simulation the free energy landscape is flattened, while the true free energy is simply the negative of the generated bias function,

$$ \beta A(\mathcal{X}^d) = -\beta f(\mathcal{X}^d) = -\ln\int e^{-\beta \mathcal{H}(\mathcal{R}, \mathcal{X}^d)} d \mathcal{R}. $$

Flat histogram methods are often attributed to Wang and Landau (2001) but the idea appears in earlier works, for example by Hunter and Reinhardt (1995) and Engkvist and Karlström (1996).

To reduce fluctuations, f_0 can be periodically reduced (`update`, `scale`) as f converges. At the end of simulation, the penalty function is saved to disk as an array ($d=1$) or matrix ($d=2$). Should the penalty function file be available when starting a new simulation, it is automatically loaded and used as an initial guess. This can also be used to run simulations with a *constant bias* by setting $f_0=0$.

Example setup where the x and y positions of atom 0 are penalized to achieve uniform sampling:

```yaml
energy:
- penalty:
    f0: 0.5
    scale: 0.9
    update: 1000
    file: penalty.dat
    coords:
    - atom: {index: 0, property: "x", range: [-2.0,2.0], resolution: 0.1}
    - atom: {index: 0, property: "y", range: [-2.0,2.0], resolution: 0.1}
```

Options:

The coordinate, $\mathcal{X}$, can be freely composed by one or two of the types listed in the next section (via `coords`).

1.10.1 Reaction Coordinates

The following reaction coordinates can be used for penalising the energy and can further be used when analysing the system (see Analysis).

Atom Properties

Molecule Properties

Notes:

- the molecular dipole moment is defined w. respect to the mass-center
- for `angle`, the principal axis is the eigenvector corresponding to the smallest eigenvalue of the gyration tensor

System Properties

The enclosing cuboid is the smallest cuboid that can contain the geometry. For example, for a cylindrical simulation container, `Lz` is the height and `Lx=Ly` is the diameter.

1.10.2 Multiple Walkers with MPI

If compiled with MPI, the master process collects the bias function from all nodes upon penalty function `update`. The *average* is then re-distributed, offering linear parallellizing of the free energy sampling. It is crucial that the walk in coordinate space differs on the different nodes, i.e. by specifying a different random number seed; start configuration; or displacement parameter. File output and input are prefixed with `mpi{rank}.`

The following starts all MPI processes with the same input file and MPI prefix is automatically appended to all other input and output:

```
yason.py input.yml | mpirun --np 6 --stdin all faunus -s state.json
```

Here, each process automatically looks for `mpi{nproc}.state.json`.

1.11 Constraining the system

Reaction coordinates can be used to constrain the system within a `range` using the `constrain` energy term. Stepping outside the range results in an inifinite energy, forcing rejection. For example,

```yaml
energy:
    - constrain: {type: molecule, index: 0, property: end2end, range: [0,200]}
```

Tip: placing `constrain` at the *top* of the energy list is more efficient as the remaining energy terms are skipped should an infinite energy arise.

CHAPTER 2

Topology

The topology describes atomic and molecular properties as well as processes and reactions.

2.1 Global Properties

The following keywords control temperature, simulation box size etc., and must be placed outer-most in the input file.

```
temperature: 298.15  # system temperature (K)
geometry:
  type: cuboid       # Cuboidal simulation container
  length: [40,40,40] # cuboid dimensions (array or number)
mcloop:              # number of MC steps (macro x micro)
  macro: 5           # Number of outer MC steps
  micro: 100         # ...inner MC steps; total steps: 5x100=5000
random:              # seed for pseudo random number generator
  seed: fixed        # "fixed" (default) or "hardware" (non-deterministic)
```

2.1.1 Geometry

Below is a list of possible geometries, specified by `type`, for the simulation container, indicating if and in which directions periodic boundary conditions (PBC) are applied. Origin ($0,0,0$) is always placed in the geometric *center* of the simulation container.

2.2 Atom Properties

Atoms are the smallest possible particle entities with properties defined below.

A filename (`.json`) may be given instead of an atom definition to load from an external atom list. Atoms are loaded in the given order, and if it occurs more than once, the latest entry is used.

Example:

```
atomlist:
  - Na: {q:  1.0, sigma: 4, eps: 0.05, dp: 0.4}
  - Ow: {q: -0.8476, eps: 0.65, sigma: 3.165, mw: 16}
  - my-external-atomlist.json
  - ...
```

2.3 Molecule Properties

A molecule is a collection of atoms, but need not be associated as real molecules. Two particular modes can be specified:

1. If `atomic=true` the atoms in the molecule are unassociated and is typically used to define salt particles or other non-aggregated species. No structure is required, and the molecular center of mass (COM) is unspecified.
2. If `atomic=false` the molecule resembles a real molecule and a structure or trajectory is *required*.

Properties of molecules and their default values:

Example:

```
moleculelist:
  - salt: {atoms: [Na,Cl], atomic: true}
  - water:
      structure: water.xyz
      bondlist:
        - harmonic: {index: [0,1], k: 100, req: 1.5}
        - ...
  - ...
```

2.3.1 Structure Loading Policies

When giving structures using the `structure` keyword, the following policies apply:

- `structure` can be a file name: `file.@` where `@=xyz|pqr|aam`
- `structure` can be an *array* of atom names and their positions: `- Mg:  [2.0,0.1,2.0]`
- `structure` can be a FASTA sequence: `{fasta:  [AAAAAAAK], k:  2.0; req:  7.0}` which generates a linear chain of harmonically connected atoms. FASTA letters are translated into three letter residue names which *must* be defined in `atomlist`. Special letters: `n=NTR`, `c=CTR`, `a=ANK`.
- Radii in files are *ignored*; `atomlist` definitions are used.
- By default, charges in files are *used*; `atomlist` definitions are ignored. Use `keepcharges=False` to override.
- A warning is issued if radii/charges differ in files and `atomlist`.
- Box dimensions in files are ignored.

2.4 Initial Configuration

Upon starting a simulation, an initial configuration is required and must be specified in the section `insertmolecules` as a list of valid molecule names. Molecules are inserted in the given order and may be `inactive`. If a group is marked `atomic`, its `atoms` is inserted `N` times.

Example:

```
insertmolecules:
  - salt:  { N: 10 }
  - water: { N: 256 }
  - water: { N: 1, inactive: true }
```

The following keywords for each molecule type are available:

A filename with positions for the `N` molecules can be given with `positions`. The file must contain exactly N-times molecular positions that must all fit within the simulation box. Only *positions* from the file are copied; all other information is ignored.

2.4.1 Overlap Check

Random insertion is repeated until there is no overlap with the simulation container boundaries. Overlap between particles is ignored and for i.e. hard-sphere potentials the initial energy may be infinite.

2.5 Equilibrium Reactions (beta)

Faunus supports density fluctuations, coupled to chemical equilibria with explicit and/or implicit particles via their chemical potentials as defined in the `reactionlist` detailed below, as well as in `atomlist` and `moleculelist`.

```
reactionlist:
  - "AH = A + H": { pK: 4.8 }
  - "Mg(OH)2 = Mg + OH + OH": { lnK: -25.9 }
```

The initial string describes a transformation of reactants (left of =) into products (right of =) that may be a mix of atomic and molecular species.

An implicit reactant or product is an atom which is included in the equilibrium constant but it is not represented explicitly in the simulation cell. A common example is the acid-base equilibrium of the aspartic acid (treated here as atomic particle):

```
reactionlist:
  - "HASP = ASP + H": { pK: 4.0 }
```

where H is included in the `atomlist` as

```
  - H: { implicit: true, activity: 1e-7 }
```

and we set `pK` equal to the `pKa`, i.e., $$ K_a = \frac{ a_{\mathrm{ASP}} a_{\mathrm{H}} }{ a_{\mathrm{HASP}} }. $$ To simulate at a given constant pH, H is specified as an implicit atom of activity $10^{-\mathrm{pH}}$ and the equilibrium is modified accordingly (in this case K is divided by a_{H}). An acid-base equilibrium, or any other single-atom ID transformation (see the Move section), can also be coupled with the insertion/deletion of a molecule. For example,

```
reactionlist:
  - "HASP + Cl = ASP + H": { pK: 4.0 }
  - "= Na + Cl": { }
```

where Na and Cl are included in the `moleculelist` as

```
  - Cl: {atoms: [cl], atomic: true, activity: 0.1 }
  - Na: {atoms: [na], atomic: true, activity: 0.1 }
```

In this case K is both divided by a_{H} and a_{Cl}, so that the actual equilibrium constant used by the speciation move is

$$ K’ = \frac{K_a}{a_{ \mathrm{H} } a_{ \mathrm{Cl} } } = \frac{ a_{\mathrm{ASP}} }{ a_{\mathrm{HASP}} a_{\mathrm{Cl}} }. $$

In an ideal system, the involvement of Cl in the acid-base reaction does not affect the equilibrium since the grand canonical ensemble ensures that the activity of Cl matches its concentration in the simulation cell.

Note that:

- all species, +, and = must be surrounded by white-space
- atom and molecule names cannot overlap
- you may repeat species to match the desired stoichiometry

Available keywords:

Warning: This functionality is under construction and subject to change. {: .notice–warning}

CHAPTER 3

Indices and tables

- genindex
- modindex
- search