Classical MD E-CAM Modules III

E-CAM Deliverable 1.4
Deliverable Type: Report
Delivered in September, 2018

E-CAM
The European Centre of Excellence for
Software, Training and Consultancy
in Simulation and Modelling

Funded by the European Union under grant agreement 676531

Project and Deliverable Information

Project Title	E-CAM: An e-infrastructure for software, training and discussion in simulation and modelling
Project Ref.	Grant Agreement 676531
Project Website	https://www.e-cam2020.eu
EC Project Officer	Juan Pelegrín
Deliverable ID	D1.4
Deliverable Nature	Report
Dissemination Level	Public
Contractual Date of Delivery	Project Month 34(31st July, 2018)
Actual Date of Delivery	22nd September, 2018
Description of Deliverable	9 software modules delivered to the E-CAM repository in the area of Classical Molecular Dynamics responding to requests of users, and their documentation.

Document Control Information

Document	Title:	Classical MD E-CAM Modules III
	ID:	D1.4
	Version:	As of 21st September, 2018
	Status:	Accepted by WP leader
	Available at:	https://www.e-cam2020.eu/deliverables
	Document history:	Internal Project Management Link
Review	Review Status:	Reviewed
	Action Requested:	Submit
Authorship	Written by:	Donal MacKernan (University College Dublin)
	Contributors:	Jony Castagna (STFC, Sci-Tech Daresbury Laboratory)
	Reviewed by:	Christoph Dellago (University of Vienna), Alan Ó Cais (Jülich Supercomputing Centre)
	Approved by:	Christoph Dellago (University of Vienna)

Document Keywords

Keywords:	E-CAM, Molecular Dynamics, CECAM, Rare Events, Path Sampling, OpenPathSampling

21st September, 2018

[1] donal.mackernan@ucd.ie

Contents

Executive Summary

In this report for Deliverable 1.4 of E-CAM, 9 software modules in classical dynamics are presented. These modules represent improvements and new features in path sampling of rare events, and in free energy perturbation.

The selection of the modules reported here was motivated by several goals, including:

- To implement the recommendations contained in E-CAM Deliverable D1.1 (Identification/selection of E-CAM MD codes for development) [1] and by the report of the E-CAM Classical MD State of the Art Workshop[2], held 29 August – 2 September 2016 at the Lorentz Center in Leiden, Netherlands.
- To enable junior participants at the Extended Software Development Workshops to contribute modules, which required a careful selection of proposed modules based on the feasibility of accomplishing them in the context of the workshop.
- To respond to the scientific needs of several active research projects, including projects related to conformational changes in DNA and in cancer-causing proteins, and projects focused on challenges associated with processing and purification, particularly those faced by pharmaceutical and food industries

The modules in this report are mainly based on the Python application OpenPathSampling (OPS) and the software package LAMMPS Python interface. For cases where performance is critical, these modules inherit the simulation performance and scaling from the underlying molecular dynamics (MD) engine that the code wraps around.

The 9 modules presented here are:

1. Contact Map
2. Contact Map Parallelization
3. Spring Shooting
4. Contact Concurrences
5. Web throwing
6. PLUMED wrapper for OPS
7. Shooting Range Shooter
8. Particle insertion core module
9. Particle insertion - hydration

Each module is thoroughly tested, includes in-code documentation as well as external documentation which in the case of OPS modules are in the form of Jupyter notebook examples.

Section 1 of this report gives a brief description of E-CAM modules and the role of this deliverable in the broader goals of E-CAM Work Package 1 (WP1). Section 2 provides background material on rare events and applications of path sampling, free energy perturbation, and general strategies regarding the practical use of the modules. In section 3, we describe each of the modules and provide links to their source codes, documentation and testing. Section 4 describes performance aspects of these modules, and section 5 summarizes the deliverable and describes the outlook for future development of modules within WP1, including the increasing importance of transverse actions across simulation communities and work packages.

[2]Report available for download from the E-CAM website: https://www.e-cam2020.eu/scientific-reports/

1 Introduction

Notwithstanding the exponential increase in computing power over the last few decades and the development of efficient molecular dynamics algorithms, many processes are still beyond the reach of simulation, especially those associated with very long and disparate timescales. For instance, the folding of a protein may occur on the time scale of seconds, a liquid can exist in an under-cooled state almost indefinitely, but the fastest motion may correspond to a time of the order of the femtosecond. Moreover, when the folding or the freezing occurs, it does so quickly. That is, events of interest are not slow but rather rare, causing long waiting times before they can be observed, and require an impractical number of time steps to be simulated directly. Addressing such time scale problems and developing scientific software able to overcome them is one of the central goals of Work Package 1 (WP1) of the E-CAM-Project. It does so by providing academic and industrial users the means to address such questions using open source software with verified quality standards, appropriate documentation and testing, disseminated and in part generated through state of the art workshops; industry scoping workshops; Extended Software Development Workshop (ESDW) events; and industry pilot projects. E-CAM software is produced primarily through E-CAM pilot industry projects funded directly by E-CAM, and through the ESDW's.

ESDW's typically of 2 weeks duration are a unique approach to combine software development with training, i.e. "training by doing", and of engaging with the wider simulation community. In E-CAM Deliverable D1.1 [1], we gave an overview of existing software for rare events and future needs. In particular, in subsections 4.1 and 4.2 of D1.1, several areas were highlighted where E-CAM could make useful contributions to free energy perturbation methods (FEP) and OpenPathSampling respectively and has guided the selection of most of the software modules presented here.

1.1 This Deliverable in the context of E-CAM

This report covers the third group of nine modules delivered as part of E-CAM WP1. As described in the grant agreement, they are "in the area of classical molecular dynamics responding to requests of users, and their documentation."

Five of the modules are associated with the problems highlighted by E-CAM industry partners as being important to them. Another four. selected by E-CAM ESDW participants, are in the area of in transition path sampling including the use of better (more easily defined) descriptors, and were started at a meeting that took place in Leiden August 2017, and completed in Amsterdam in June 2018. Additional modules have also been developed which are not part of this deliverable.

As mentioned above, this report includes four modules that were contributed from ESDWs. Selection of modules for the ESDW was driven by the combination of feasibility and relevance to the goals of the Deliverable and of the project, as laid out in previous reports. These software development tasks were also used as part of a practical introduction to advanced programming techniques and hardware environments for the participants of the ESDWs. Therefore, they have high value for the training component of E-CAM, as well as their intrinsic software value.

In the next section, we will describe: rare events and how path sampling methods are used; free energy perturbation methods; and, give an overview of how the modules are used in practise. In section 3, we describe each of the individual modules delivered as part of this report, along with links to the E-CAM documentation, which in turn contains links to examples, further documentation, and the source code for the module. Section 4 will describe the role of high performance computing in these modules, and section 5 concludes and provides outlook for future modules to be delivered by E-CAM WP1.

2 Background

It can be argued that the dramatic increase in the potency of simulation is due at least as much to advance in statistical mechanics methods and associated algorithms as developments in hardware. Rare events, path sampling methods, and free energy perturbation methods lie at the heart of much of this, and the modules presented here.

2.1 Rare-events, path sampling and free energy perturbation

The desire for high resolution in space (and therefore time) is inherently in conflict with the desire to study long-time dynamics. To study molecular dynamics with atomistic detail, we must use timesteps on the order of the femtosecond. However, many problems in biological chemistry, materials science, and other fields involve events that only spontaneously occur after a millisecond or longer (for example, biomolecular conformational changes, or nucleation processes). That means that we would need around 10^{12} time steps to see a single millisecond-scale event. This is the problem of "rare events" in theoretical and computational chemistry and physics. An apparently different statistical method is free energy perturbation (FEP) which can be likened in a popular sense to alchemy of old, where base metals were transformed as if by magic to gold. At a statistical mechanical level this is achieved by introducing additional terms to a systems Hamiltonian which are zero when a control parameter λ equals zero, and are fully turned on when it equals one. FEP can be used in principle literally to change lead into gold, mutate amino-acids, add new atoms or molecules to a substance, or change a solvent mixture and compute the corresponding change in free energy. The latter example is deeply connected to industrial purification and processing.

While the focus of FEP and the rare-event methods of bridging time scales are physically and chemically very different, at a mathematical and practical computational level they are often intimately related. For example, free energy (rare-event based methods typically sample a system in specific locations delineated through a set of order parameters (and corresponding additional interactions in the systems' potential), and combine the data from the different locations together to estimate the "global" free energy properties of the system. Using additional interactions is not very different in principle from FEP. Moreover FEP and free energy rare-event methods can use similar methods of statistical analysis, e.g. the Multiple Bennet Acceptance Ratio (MBAR).

While modern supercomputers are beginning to make it possible to obtain trajectories long enough to observe some of these processes (such as millisecond dynamics of a protein [2]), even then, we may only find one example of a given transition. To fully characterize a transition (with proper statistics), we need many examples. This is where path sampling comes in. Path sampling approaches obtain many trajectories using a Markov chain Monte Carlo approach as follows. First an existing trajectory is perturbed (usually using a variant of the "shooting" move); and second, the resulting trial trajectory is accepted or rejected according to conditions that preserve the distribution of the path ensemble. As such, path sampling is Monte Carlo in the space of paths (trajectories). Conceptually, this enhances the sampling of transitions by focusing on the transition region instead of the stable states. In direct MD, trajectories spend much more time in stable states than in the transition region (exponential population differences for linear free energy differences); path sampling skips over that time in the stable states.

The main path sampling approaches used in the modules below are transition path sampling (TPS) [3] and transition interface sampling (TIS) [4], and FEP. In practice, TPS is mainly used to characterize the mechanism of a transition, while TIS (which is more expensive than TPS) is used to calculate rates and free energy landscapes. Overviews of these methods, as well as other rare events methods, can be found in the following review articles:

- 2010 review by Bolhuis and Dellago in Reviews in Computational Chemistry [5]
- 2008 review by Dellago and Bolhuis in Advances in Polymer Science [6]

In addition, several other resources are available on the web to teach path sampling, including:

- Wikipedia entry on path sampling [7]
- Aaron Keys's tutorial on path sampling [8]

In the context of FEP a multitude of methods have been developed over several decades since the time of Robert Zwanzig[9] and even longer if one includes thermodynamic integration by Kirkwood[10]. For a long time these methods were complicated by the "insertion catastrophe", a singularity which occurs when particles are inserted in a dense medium using standard potentials and overlap with particles already present. This difficulty was resolved by introducing one para-mater families of soft-core potentials, which when $\lambda = 0$ are zero, and particles can fully overlap, and correspond to Lennard-Jones interactions (or similar potentials) when $\lambda = 1$. The fact that particles can overlap means that corresponding charged particles would give rise to a singularity in the electrostatic interactions. A simple way to avoid this is to insert (or remove)) particles in a two stage and two parameter process so that electrostatic interactions with the inserted particles are strictly zero during the inserting stage. However this is computationally in-efficient.

We have developed a new method that deals with the singularity without using soft-core potentials, and the first pair of several corresponding software modules. This approach also begins to responds to key difficulties pointed out by industry during an E-CAM industry scoping workshop on solvation held in Lyon in the spring of 2018.

2.2 Applications and modules in this report

Of the nine modules presented in this deliverable, five are rooted in two E-CAM pilot industry projects:

- Contact Map (OPS)
- Contact Map Parallelization (OPS)
- Contact Concurrences (OPS)
- Particle Insertion Core
- Particle Insertion Hydration

with Biki Technologies, and Kerry Group and the Applied Processing Company, respectively. These three companies were identified as industry partners in the original E-CAM proposal having a significant interest in advanced molecular dynamics due to their activities in drug discovery and development, food technologies, and pharmaceutical processing. Here, (OPS) denotes that the module was developed in the context of open path sampling. Four modules were developed in an ESDW held in Leiden in 2017, and it's follow-up in 2018.

- Spring Shooting (OPS)
- Web Throwing (OPS)
- PLUMED (OPS)
- Shooting Range Shooter (OPS)

They were chosen because they addressed important implementations of new methods or capabilities into OPS, and because the ESDW participants selected them. The OPS modules are largely path sampling codes, whereas the Particle Insertion modules are implementations of new free energy perturbation methods. We shall describe each in turn.

2.2.1 Path Sampling

As computational resources become more powerful, path sampling has the promise to provide insight into rare events in larger systems, and into events with even longer timescales. For example:

- Drug/protein binding and unbinding (timescales of minutes), which is essential for predicting the efficacy of drugs
- Association processes of proteins (large systems), which is at the core of communication in biochemical pathways
- Self assembly processes for complex systems (many intermediates), which can be important for the design of new materials

Further, applying the known successes of path sampling methods to larger systems can also be quite valuable. Path sampling can shed light on the networks of conformational dynamics for large proteins and protein complexes, and on the mechanisms and rates of complex reactions and phase transitions. The range of possibilities is too broad to enumerate, instead here we focus on some representative examples drawn from the present OPS modules.

- Contacts can be an important tool for defining (meta)stable states in processes involving bio-molecules. For example, an analysis of contacts can be particularly useful when defining bound states during a binding processes between proteins, DNA, and small molecules (such as potential drugs). The practical application of this software module is the pilot project in collaboration with BiKi Technologies on "Binding Kinetics". The project aims at investigating the binding/unbinding of a selective reversible inhibitor for protein $GSK3\beta$
- Transition path sampling is most efficient when paths are generated from the top of the free energy barrier. However, complex (bio-molecular) activated processes, such as nucleation or protein binding/unbinding, can have asymmetric and peaked barriers. Using uniform selection on these type of processes is not efficient, as it, on average, results in selected points that are not on the top of the barrier. Paths generated from these points have a low acceptance probability and accepted transition paths decorrelate slowly, resulting in a low overall

efficiency. The Spring shooting and shooting range modules were developed to increase the efficiency of path sampling.

- Transition path sampling simulations and analysis rely on accurate state definitions. PLUMED is a widely used and versatile rare-event sampling and analysis code that can be used with various MD engines. It has a very intuitive and versatile syntax for the definition of collective variables, and a wide variety of sampling methods, which accounts for its widespread use. Many of PLUMED's dozens of CVs have a biomolecular focus, but they are also general enough for other applications. PLUMED's popularity (over 500 citations in 4 years after the release of PLUMED2) is greatly based on the fact that it works with many MD codes. OPS is now added to that list.

2.2.2 Particle Insertion through free energy perturbation

Particle insertion can be used to compute the free energy associated with hydration/drying, the insertion of cavities in fluids/crystals, changes in salt levels, changes in solvent mixtures, and alchemical changes such as the mutation of amino-acids. It can also be used to compute the free energy of solvent mixtures and the addition of salts, which is used in the purification processing industrially, for instance in the purification of pharmaceutical active ingredients. Particle insertion can in principle also be used to compute the free energy associated with changes in the pH, that is the proton transfer from a titratable site to the bulk, for example in water.

The applications of this module in upcoming modules include computing the free energy changes associated with the following problems of industrial and scientific importance.

1. Hydration and drying. The food, pharmaceutical and diagnostic industries frequently need to dry proteins and other active ingredients for safe long term storage and transport. Such substances however must be re-hydrated before use. This can often be very problematic, for instance proteins may denature during drying with hydrophobic residues being exposed to the exterior, and in such a way, that on rehydration the proteins cannot be solvated and drop out of solution. It also can be an issue for the handling for instance of antibodies for use in immuno-diagnostics, as the antibodies usually need to be dried to prevent interaction until their time of use.
2. The addition of multiple molecules into a condensed environment. A lot of pharmaceutical and material processing includes the addition of one molecular species in a matrix of other molecules. Being able to know whether such insertions are thermo-dynamically favorable is important.
3. Residue mutation and alchemy. The ability to mutate protein residues is a key feature underlying bio-technology and genetic engineering. Being able to simulate the effects of such changes on the structure of the protein is very important, in particular free energy properties, as it allows development and testing of novel protein forms prior to experimental validation, which is generally very expensive. Moreover, experimentally, it can be difficult to discern molecular mechanisms at play which the "designer" may wish to tune.
4. Free energy changes in chemical potentials associated with changes in solvent mixtures. Modifying solvent mixtures so as to control purification processes including nucleation is widely used in industry, particular in the processing of pharmaceutical active ingredients, or for that matter to predict their bio-availability.
5. Constant pH simulations. Although pH titration is one of the oldest methods used for processing a wide variety of materials, pharmaceuticals and foods, it frequently remains at a theoretical/simulation level very difficult to model accurately, particularly for large molecules. A future module will use particle insertion and free energy perturbation combined with quantum surface hopping modules created in E-CAM work package 3 (quantum dynamics).

The role of modules in the particle insertion work-flow

Our approach to particle insertion has focused on developing a rigorous methodology which can be easily implemented in widely available molecular dynamics engines known to scale very well on massively parallel computing platforms. For this purpose, LAMMPS turns out to be ideal. This is because in addition to its excellent scaling, it is also possible to modify forcefield paramaters "on the fly", during simulation runs which we need to be able to do so as to modify perturbations and data collection without significant loss of computational efficiency. The overall workflow for our modules comprises the following.

1. Selection of λ values to be used for perturbation - for this we have a python code which uses Chebychev methods to optimize the choice of λ values
2. Statistical sampling using production runs with LAMMPS. Here we currently use LAMMPS scripting, and a rather complex loop structure. In the future we will replace this with a C++ code to drive LAMMPS as a library.

3. Python based codes for statistical analysis of production data. We have created two types of analysis, one based on thermodynamic integration and the other based on MBAR. The latter exploits the pymbar set of codes developed by the Chodera laboratory.

3 Modules

Seven of the nine software modules described here have been produced in the context of OPS, with a focus on dynamical properties and path sampling. Two modules are completely independent of OPS, and have been developed to compute the free energy changes associated with particle insertion and deletion, which are important in various contexts, including industrially for instance hydration, drying, solubility, and purification.

Material in this section is largely drawn from the detailed module documentation files hosted at Classical MD section of the E-CAM Library. Links are provided for more information about each module. Modules that have been completed and accepted into the E-CAM library also have a link to the specific module page in the E-CAM library documentation website. Further details about the code contributed and the development process can be found through those links. In addition, each module consists of at least one example of showing how to use it, linked in the "Examples" section of the linked module documentation.

3.1 Contact Map

Frequently, we characterize states (especially in bio-molecular systems) in terms of the contacts between specific residues or atoms. When trying to identify the specific contacts of interest, it can be useful to look at all the contacts. This module provides tools for mapping and identifying contacts in trajectories. It builds on the excellent tools provided by MDTraj. The module is compatible with Python (2.7, 3.4, 3.5, 3.6). The software licence is LGPL 2.1 or later.

3.1.1 Module description

Contacts can be an important tool for defining (meta)stable states in processes involving biomolecules. For example, an analysis of contacts can be particularly useful when defining bound states during a binding processes between proteins, DNA, and small molecules (such as potential drugs).

The contacts analyzed by contact_map can be either intermolecular or intramolecular, and can be analyzed on a residue-residue basis or an atom-atom basis.

This package makes it very easy to answer questions like:

- What contacts are present in a trajectory?
- Which contacts are most common in a trajectory?
- What is the difference between the frequency of contacts in one trajectory and another? (Or with a specific frame, such as a PDB entry.)
- For a particular residue-residue contact pair of interest, which atoms are most frequently in contact?

It also facilitates visualization of the contact matrix, with colors representing the fraction of trajectory time that the contact was present.

3.1.2 Motivation and exploitation

This is an independent module, but it builds on tools developed in MDTraj.

Additional Details

Direct Documentation Link	readme.rst of `Contact Map` module.
Merge Request	Merge-Request of `Contact Map` module.

3.2 Contact Map Parallelization

This module adds the ability to parallelize the calculation of contact frequencies (see the contact map module). It includes improvements to the core of the contact_map package to facilitate parallelization.

3.2.1 Module description

Contacts are defined as when two atoms, or atoms within two groups of atoms (residues), are within some cutoff distance of each other. The contact map is the set of all contacts in a given snapshot. The contact frequency is the fraction of a trajectory in which each pair of contacts is present. The contact frequency therefore requires calculation of the contact map for each individual frame in the trajectory.

3.2.2 Motivation and exploitation

The original contact_map code included OpenMP (shared-memory) parallelization of the calculation of a single contact map (a loop over atoms). Each contact map in a contact frequency (the loop over the frames of a trajectory) was done sequentially. However, each frame is completely independent, and can be processed on a separate node. This module implements that parallelization.

This module interfaces with the disk-distributed package for task-based parallelization. The trajectory is separated into segments, with the disk network calculating the contact frequency of each segment in parallel (reading from a common file source). Then the partial contact frequencies are combined into one Contact-Frequency object. This also includes methods, such as serialization into JSON strings, that would be useful for parallelization by other tools

Additional Details

Direct Documentation Link	readme.rst of `Contact Map Parallelization` module.
Merge Request	Merge-Request of `Contact Map Parallelization` module.

3.3 Contact concurrences

This module deals with the analysis of contacts between parts of biomolecules based on "contact concurrences," i.e., what contacts occur simultaneously during a trajectory. This is useful when using contacts as a definition of a metastable state in a trajectory.

3.3.1 Module description

Contact frequencies, as developed in the module contact-map, are a useful tool for studying bio-molecular systems, such as binding/unbinding of a ligand from a protein. However, they suffer from one problem when trying to use them to define meta-stable states: since they are averaged over time, they don't show time-dependent behavior. To identify a stable state, time-dependent behavior must be considered.

For example, a particular contact pair might have a frequency of 0.1 during a 100ns trajectory. But this could be achieved in several ways. If the contact events are randomly distributed through time, this contact probably isn't characteristic of a meta-stable state. On the other hand, if the contact is constantly present during the last 10 ns (and not otherwise present), it might represent a meta-stable state. More importantly, there might be multiple contacts that are *all* present during those last 10 ns. Those concurrent contacts could be used to define a metastable state. This module helps identify and analyze those concurrent contacts by providing a tool to visualize them.

3.3.2 Motivation and exploitation

This is an important tool for identifying stable states based on long-lived groups of contacts, and is being used as part of the 'E-CAM pilot project on binding-kinetics. It has also been used a part of a bachelor's thesis project to develop an automated approach to identifying metastable intermediates during binding/unbinding processes.

Additional Details

Direct Documentation Link	readme.rst of `Contact concurrences` module not yet public
Merge Request	Merge-Request of `Contact concurrences` module.

3.4 Spring Shooting

This module implements the spring shooting method in OpenPathSampling OPS. It runs under Python (2.7, 3.5, 3.6) and is distributed under Licence LGPL, v. 2.1 or later.

3.4.1 Module description

Transition path sampling is most efficient when paths are generated from the top of the free energy barrier. However, complex (biomolecular) activated processes, such as nucleation or protein binding/unbinding, can have asymmetric and peaked barriers. Using uniform selection on these type of processes will not be efficient, as it, on average, results in selected points that are not on the top of the barrier. Paths generated from these points have a low acceptance probability and accepted transition paths decorrelate slowly, resulting in a low overall efficiency. Spring shooting was developed to increase the efficiency of path sampling of these types of barriers, without any prior knowledge of the barrier shape. The spring shooting algorithm uses a shooting point selector that is biased with a spring potential. This bias pulls the selection of points towards the transition state at the top of the barrier. The paths that are generated from points selected by this biased selector therefore have an increased acceptance probability and decorrelation between accepted transition paths is also increased. This results in a higher overall efficiency. The spring shooting algorithm is described by Brotzakis and Bolhuis here.

The spring shooting selection algorithm is a selector for the one-way shooting method for transition path sampling, which uses a bias. This bias is of the shape $\min[1, e^{s\kappa\Delta\tau}]$. Where $s = -1$ for forward shooting and $s = 1$ for backward shooting, κ is the given spring constant and $\Delta\tau = \tau' - \tau$ is the number of shifted frames of the new shooting point τ' compared to the previous accepted shooting point τ. The choice of τ' is limited to the interval $[-\Delta\tau_{max}, \Delta\tau_{max}]$. The shooting move is rejected if a τ' is selected that is outside of the current path and is accepted if the trajectory satisfies the path ensemble.

3.4.2 Motivation and exploitation

The main difference of this module compared to the paper is that instead of using a rejection algorithm to sample from the correct distribution, the correct distribution is sampled directly. This module was developed as part of the second WP1 ESDW, held in Leiden in August 2017, and was selected based on a participant's specific interest and experience with this method. As this is a very recently developed module, it has no practical exploitation yet, but it is an excellent example of using an ESDW participant's previous experience to meet E-CAM's goal of training through module development.

Additional Details

Direct Documentation Link	readme.rst of `Spring Shooting` module.
Merge Request	Merge-Request of `Spring Shooting` module.

3.5 Web throwing

This module implements the web throwing method in OPS. It runs under Python (2.7, 3.5, 3.6) and is distributed under Licence LGPL, v. 2.1 or later.

3.5.1 Module description

Web throwing and stone skipping are two types of MC moves that were originally proposed by Riccardi, Dahlen, and van Erp to improve the speed of convergence. In these moves, trajectories are constructed via a sequence of sub-paths obeying super-detailed balance. By a re-weighting procedure, almost all paths can be accepted. Whereas the generation of a single trajectory becomes more expensive, the reduced correlation results in a significant speedup. For a study on DNA denaturation, the increase was found to be a factor 12.

The present module is an implementation of web throwing into OPS to improve the efficiency of transition interface sampling (TIS). It consists of a smart selection of shooting points and shooting moves that respects super detailed balance. The web throwing algorithm generates paths that have a much lower correlation with their original paths, even if it is computational more expensive than standard shooting. The strategy thus can significantly reduce the computational time required to study transition events and to quantify their rates.

3.5.2 Motivation and exploitation

The web throwing method is a shooting method for transition interface sampling that decorrelates the trajectory between an interface λ and an associated surface of unlikely return λ_{SOUR}. It does this by doing n_cycles of Transition Path Sampling (TPS) from λ_{SOUR} to λ. The first frame in this volume is selected for a forwards shot, and the last frame in this volume is selected for a backwards shot. After the n_cycles the new sub-trajectory is extended in both ways to satisfy the TIS ensemble. The sampling efficiency is increased significantly if λ_{SOUR} and λ are positioned before and after the barrier in the potential, respectively.

Additional Details

Direct Documentation Link	readme.rst of `Web throwing` module
Merge Request	Merge-Request of `Web throwing` module.

3.6 PLUMED wrapper for OPS

3.6.1 Module description

PLUMED is a widely used and versatile rare-event sampling and analysis code that can be used with various md engines. It has a very intuitive and versatile syntax for the definition of collective variables, and a wide variety of sampling methods, which accounts for its widespread use. The present module developed by Alberto Pérez de Alba Ortíz allows PLUMED and OPS to be used together. Details on how the module is used are here.

3.6.2 Motivation and exploitation

Transition path sampling simulations and analysis rely on accurate state definitions. Such states are typically defined as volumes in a Collective Variables (CV)-space. OPS already supports a number of CVs, including the ones defined in the MDTraj python library. PLUMED offers a wide variety of extra CVs, which are enabled in OPS by this module.

Many of PLUMED's dozens of CVs have a biomolecular focus, but they are also general enough for other applications. PLUMED's popularity (over 500 citations in 4 years after the release of PLUMED2) is greatly based on the fact that it works with many MD codes. OPS is now added to that list. The PLUMED code is well-maintained and documented for both users and developers. Several tutorials and a mailing list are available to address FAQs. For more information about PLUMED is available here.

The class *PLUMEDInterface* is a subclass of the cython wrapper class *Plumed* contained in the PLUMED installation. For initialization, the *PLUMEDInterface* requires an *MDTrajTopology* and accepts additional PLUMED keywords. The initialized *PLUMEDInterface* can be subsequently used to make functions that calculate CVs for a given Trajectory. This is done via the *PLUMEDCV* class, a subclass of CoordinateFunctionCV.

In PLUMED, the syntax for all commands is: **label: keywords**. The class PLUMEDCV takes name and definition as arguments, which are respectively equivalent to label and keywords. The PLUMEDCV class also takes the PLUMEDInterface as argument. This allows for a single PLUMEDInterface to contain the MDTrajTopology, additional PLUMED keywords and previously defined CVs that can be reused for the same system. Both PLUMEDInterface and PLUMEDCV are storable.

This module supports (as listed in PLUMED documentation):

- Groups and Virtual Atoms: are directly set in the PLUMEDInterface via the PLUMEDInterface.set(name, definition) function. The PLUMEDInterface.get() function allows to consult the commands that have been already set.
- CV Documentation: all CVs are created by calling PLUMEDCV(name, PLUMEDInterface, definition). The returned function can be appied to a Trajectory. CVs with components should specify the components=["c1", "c2", "c3"] keyword and the corresponding PLUMED keywords in the definition.

- Distances from reference configurations: are also created by calling PLUMEDCV(name, PLUMEDInterface, definition). Most of them require external files with the reference configurations.
- Functions: are also created by calling PLUMEDCV(name, PLUMEDInterface, definition). They should be created using the same PLUMEDInterface that contains the previously defined CVs that are part of the function.
- Multicolvar and Exploiting contact matrices are not tested.

Additional Details

Direct Documentation Link	readme.rst of `Plumed wrapper for OpenPathSampling` module
Merge Request	Merge-Request of `Plumed wrapper for OpenPathSampling` module.

3.7 Shooting range shooter

This module implements the Shooting range shooter method in OpenPathSampling OPS. It runs under Python (2.7, 3.5, 3.6) and is distributed under Licence LGPL, v. 2.1 or later.

3.7.1 Module description

Transition path sampling is a powerful tool in the study of rare events. Shooting trial trajectories from configurations along existing transition paths has proved particularly efficient in the sampling of reactive trajectories. However, most shooting attempts tend not to result in transition paths, in particular in cases where the transition dynamics has diffusive character. To overcome the resulting efficiency problem, Hendrik Jung, Kei-ichi Okazak and Gerhard Hummer developed an algorithm for "shooting from the top". The first step is to define a shooting range through which all paths have to pass and then shoot off trial trajectories only from within this range. For a well chosen shooting range, nearly every shot is successful, resulting in an accepted transition path. To deal with multiple mechanisms, weighted shooting ranges can be used. To cope with the problem of unsuitably placed shooting ranges, they developed an algorithm that iteratively improves the location of the shooting range. The method should be particularly useful in cases where the transition paths are long so that only relatively few shots are possible, yet reasonable order parameters are known. The present module implements that algorithm in openpathsampling and was written by Hendrik Jung.

3.7.2 Motivation and exploitation

The purpose of this algorithm is to increase the number of generated transitions in a transition path sampling simulation by exclusively shooting from the transition state ensemble (TSE)/the top of the barrier (hence the name). Naturally this only works if the approximate location of the TSE is already known and can be given as a function of the atomic coordinates. In this module any object can be used by the user to define the shooting range volume. This enables the user to define the shooting range for example as a function of one or more collective variables. The implementation in this module includes a ShootingRangeSelector subclass of ShootingPointSelector to pick shooting points only in the predefined shooting range volume.

Additional Details

Direct Documentation Link	readme.rst of `Shooting range shooter` module
Merge Request	Merge-Request of `Shooting range shooter` module.

3.8 Particle Insertion Core Module

This software module computes the change in free energy associated with the insertion or deletion of Lennard-Jones particles in dilute or dense conditions in a variety of thermodynamic ensembles. It consists of a collection of codes written in Python (2.7), C and the LAMMPS scripting language, and is distributed under the MIT software licence. It will be extended to other molecular dynamics engines at a later date.

3.8.1 Module description

Lennard-Jones type interactions are the key source of difficulty associated with particle insertion or deletion, which is why this module is a core module, as other interactions including Coulombic and bond, angle and dihedral interactions are comparatively easy to handle in a perturbative scheme (i.e. they do not introduce divergences in the limit of small λ). It differs from the main community approach used to date to compute such changes as it does not use soft-core potentials. Its key advantages over soft-core potentials are: (a) electrostatic interactions can in principle be performed simultaneously with particle insertion (this and other functionalities will be added in a new module); and, (b) essentially exact long-range dispersive interactions using dispersion Particle Mesh Ewald (PMME) or EWALD if desired can be selected at run-time by the user. We believe that both of these advantages, but in particular (a) will greatly facilitate the speed, flexibility and robustness of particle insertion in a wide variety of contexts.

General Formulation

Consider a system consisting of $N+M$ degrees of freedom and the Hamiltonian

$$H(r,p,\lambda) = H_0 + KE_{insert} + \Delta V(r,\lambda)$$

where H_0 corresponds to an unperturbed Hamiltonian, and the perturbation $\Delta V(r,\lambda)$ depends nonlinearly on a control parameter λ, and KE_{insert} corresponds to the kinetic energy of the inserted particles. The first set of N degrees of freedom is denoted by A and the second set of M degrees of freedom is denoted by B. To explore equilibrium properties of the system, thermostats, and barostats are used to sample either the NVT (canonical) ensemble or the isobaric-isothermal ensemble. The perturbation is devised so that when $\lambda = 0, \Delta V(r,\lambda) = 0$, B is in purely virtual. When $\lambda = 1$, B corresponds to a fully physical augmentation of the original system.

In the present software module, we consider only interaction Lennard Jones atoms.

$$\Delta V(r,\lambda) = V_{lj}(r,\lambda)$$

where for each inserted atom $\imath$

$$\hat{\sigma}(\lambda)_i = \lambda\sigma_i \qquad \hat{\epsilon}(\lambda)_i = \lambda\epsilon_i$$

and the mixing rule for Van der Waals diameters and binding energy between different atoms uses the geometric mean. The dependence of σ on λ has the consequence that the mean σ between a pair of inserted atoms scales as λ, but scales as $\sqrt{\lambda}$ when one atom in the pair is inserted and the other is already present. These choices of perturbations guarantees that the particle insertion and deletion catastrophes are avoided.

Algorithms

At the core of the PI core module there are four functions/codes. The first written in python generates a set of λ values corresponding to the zero's of suitably transformed Chebyshev functions, and reduces very substantially the number of interpolation points required for thermodynamic integration. This choice is less important when the Multiple Bennet Acceptance Ratio (MBAR) is used. The second code written in LAMMPS scripting language performs the simulation in user-defined ensembles at the selected interpolation values of λ, at a user-specified frequency, computing two-point central difference estimates of derivatives of the potential energy needed for thermodynamic integration, computing the energy functions for all values of λ in the context of MBAR. The user also specifies the locations of the inserted particles. The user also specifies whether Particle Mesh Ewald or EWALD should be used for dispersive interactions. The third code written in python takes the output data from LAMMPS, prepares it so that free energy differences in the selected ensemble can be computed using MBAR provided by the pymbar suite of python codes of the Chodera group. The fourth code, also written in python take the LAMMPS output and performs the thermodynamic integration. A code written in C is also included in the module for testing/validation purposes using published equation of state data.

3.8.2 Motivation and exploitation

Particle insertion can be used to compute the free energy associated with hydration/drying, the insertion of cavities in fluids/crystals, changes in salt levels, changes in solvent mixtures, and alchemical changes such as the mutation of amino-acids. It can also be used to compute the free energy of solvent mixtures and the addition of salts, which is used in the purification processing industrially, for instance in the purification of pharmaceutical active ingredients. Particle insertion can in principle also be used to compute the free energy associated with changes in the pH, that is the proton transfer from a titratable site to the bulk, for example in water.

Practical exploitation: This module will be central to the following practical applications (and associated modules):

1. hydration and drying;
2. the addition of multiple molecules into a condenses environment;
3. residue mutation and alchemy;
4. constant pH simulations, this also will also exploit modules created in E-CAM work package 3 (quantum dynamics); and,
5. free energy changes in chemical potentials associated with changes in solvent mixtures.

Additional Details

Direct Documentation Link	readme.rst of `Particle insertion Core` module
Merge Request	Merge-Request of `Particle insertion Core` module.

3.9 Particle Insertion - Hydration and Drying Module

This software module computes the change in free energy associated with the insertion or deletion of water in dilute or dense conditions in a variety of thermodynamic ensembles, where statistical sampling through molecular dynamics is performed under LAMMPS but will be extended to other molecular dynamics engines at a later date. It consists of a collection of codes written in Python (2.7), C and the LAMMPS scripting language, and is distributed under the MIT software licence.

3.9.1 Module description

This module builds on the PI Core software module by adding electrostatic, bond, and angle λ dependent interactions including SHAKE to the Lennard-Jones interactions that were dealt with in PIcore.

Our approach consists of re-scaling electrostatic charges of inserted atoms so that they converge to zero faster than inserted Van der Waals atoms where the later uses the geometric mean for Lennard Jones diameters and binding energies, and that bond, and angle spring constants and where necessary also bond lengths scale to zero in the same fashion effective size of inserted atoms through a parameter λ so that all interactions between inserted atoms and interactions between inserted atoms and atoms already present in the system are zero when $\lambda = 0$.

General Formulation

In the present software module, we include in the perturbation interaction Lennard Jones potentials, harmonic bond and angle interactions, and electrostatic interactions:

$$\Delta V(r,\lambda) = V_{lj}(r,\lambda) + V_b(r,\lambda) + V_a(r,\lambda) + V_{el}(r,\lambda)$$

where where for each inserted atom $\imath$,

$$\hat{\sigma}(\lambda)_i = \lambda\sigma_i \qquad \hat{\epsilon}(\lambda)_i = \lambda\epsilon_i \qquad \hat{q}(\lambda)_i = q\lambda^p$$

and the mixing rule for Van der Waals diameters and binding energy between different atoms uses the geometric mean for atoms pairs where one or more of the atoms is inserted, but retains the standard mixing rule for atoms already present. The dependence of σ on λ has the consequence that the mean σ between a pair of inserted atoms scales as λ, but scales as $\sqrt{\lambda}$ when one atom in the pair is inserted and the other is already present. The dependence of ϵ on λ ensures that forces behave regularly when λ is very small. These choices of perturbations guarantees that the particle insertion and deletion catastrophes are avoided. Regarding electrostatic interactions, the exponent p allows the product of charges present in electrostatic potentials and forces to be tuned to converge to zero faster than the rate at which effective distances between corresponding Lennard Jones atoms go to zero as λ becomes very small, thus ensuring divergences are avoided. Currently $p = 1.5$, which was chosen based on the estimate that the Coulomb potential between inserted and already present atoms scales as $\sim \frac{q^2\lambda^{1.5}}{\sigma\sqrt{\lambda}} \sim \lambda$. The spring constants for harmonic, angular and torsional interactions involving inserted atoms are currently simply multiplied by λ. It is also possible to replace bond, angle and torsional interactions involving only inserted atoms with shake constraints. In such cases, the shake constraints are continuously on. Finally it is very easy to extend this perturbative treatment to Lennard Jones interaction where arithmetic mixing rules are used (e.g. CHARMM) where the functional form of the interactions are the same as above, but in the limit $\lambda = 1$ converge to the arithmetic mean.

3.9.2 Motivation and exploitation

The applications of this module use in upcoming modules include computing the free energy changes associated with:

- hydration and drying of food and pharmaceutical proteins, and antibodies;
- the relative solubility of solutes in solvent mixtures which plays a central role in industrial processing and purification, for example of pharmaceutical active ingredients;
- residue mutation as occurs in the rational design and optimization of proteins, protein based sensors/diagnostics and protein/antibody based therapeutics
- constant pH simulations, this also will also exploit modules created in E-CAM work package 3 (quantum dynamics).

Additional Details

Direct Documentation Link	readme.rst of `Particle insertion Hydration` module
Merge Request	Merge-Request of `Particle insertion Hydration` module.

4 Performance Considerations

There are three stages of running a simulation: setting it up, running it, and analyzing the results. The vast majority of computational time is spent in the second stage, generating the simulation data (specifically, in performing molecular dynamics). Therefore, discussion of performance primarily focuses on the modules associated with that stage. Many of the modules in this report focus on analysis, following the description of this Deliverable in the grant agreement, and therefore scalability does not play a major role. But some of the modules in this report include both sampling approaches and analysis, and therefore they can benefit from improved sampling performance.

In E-CAM Deliverable D1.1, we discussed the variety of MD engines. As mentioned in that report, different communities have coalesced around different engines. Software for trajectory-based rare events simulations, such as OpenPathSampling, usually wraps around other MD engines, because these simulations don't change the underlying dynamics. This gives them the potential for a wider range of applications. Furthermore, since the primary cost of the calculation is in performing the dynamics, this also means that most of the responsibility for performance falls on the underlying engine. OPS already has support for OpenMM, a GPU-accelerated MD code, and support is in development for the widely-used and highly-scalable packages LAMMPS and Gromacs. Each of these dynamics engines is designed for high performance computing, and in this way, OPS inherits the performance from the underlying dynamics code.

The FEP based module Particle Insertion Core and Particle Insertion Core are designed specifically to exploit the massive parallelism capabilities of LAMMPS, and tests to date have demonstrated that the efficiency of LAMMPS is indeed retained, and is combined with a extraordinary degree of robustness due to the intrinsic formulation of the perturbation. The OPS modules Spring Shooting, Web throwing, PLUMED wrapper for OPS, Shooting Range Shooter wrap around such dynamics engines, and their performance characteristics therefore depend on the underlying engine. The objects created by these modules take an instance of the OPS `DynamicsEngine` class in their initialization. The `DynamicsEngine` abstracts specifics of the underlying engine so that the same code in these modules can be used to manage different underlying engines. Contact map parallel in particular has been resigned as its name suggests for parallel platforms at the level of analysis.

This approach means that as new generations of hardware lead to new molecular dynamics codes or changes in existing codes, OpenPathSampling, and the modules designed for it, can continue to use the cutting edge in hardware with minimal modification. The formulation of the FEP codes is such that it can be easily transferred to md engines such as GROMACS.

5 Outlook

The report of Deliverable 1.4 of E-CAM describes 9 software modules of WP1 in classical molecular dynamics. As described in the grant agreement, they are "in the area of classical molecular dynamics responding to requests of users, and their documentation." They include seven modules and their documentation for path sampling and analysis using the open-source package OpenPathSampling, and two modules focused on free energy perturbation. The latter two modules together with the OPS modules contact, contact parallelization, and contact concurrences emerged from two E-CAM industry focused pilot projects, indeed the industrial relevance of the FEP approach was also emphasized at an E-CAM industry scoping workshop on solubility held in Lyon in early 2018. The other four OPS modules were selected by participants at two ESDW meetings - one in Leiden in August 2017, and a follow up meeting held in Amsterdam in June 2018.

Several new modules are in preparation building on FEP modules provided here responding to industry priorities. One trend that is emerging from the multiple scientific packages of E-CAM is the need and opportunity presented by transverse E-CAM ESDW's and software modules. These are transverse in the sense that a module or ESDW addresses a technical need common to more than one scientific work-package, or alternatively combines methods coming from different work-packages to address questions of scientific or industry importance. An example of the former was an ESDW held in July 2018 in Turin, "Intelligent high throughput computing for scientific applications". Regarding the latter, a set of modules that have attracted great industrial interest are being developed focused on the rational design through simulation of protein based sensors, diagnostics, and therapeutics which combine state of the rare-event methods, coarse graining, machine learning, and the use of massively parallel computing platforms. Indeed an E-CAM ESDW on this topic has been proposed. Finally, the next state of the art workshop in Work Package (WP)1, Large Scale Activated Event Simulations, to be held in Vienna in October 2018 is likely also to highlight areas where the community needs additional new modules.

References

Acronyms Used

CECAM	Centre Européen de Calcul Atomique et Moléculaire
CV	collective variable
ESDW	Extended Software Development Workshop
WP	Work Package
MD	molecular dynamics
OPS	OpenPathSampling
TPS	transition path sampling
TIS	transition interface sampling
FEP	free energy perturbation
MBAR	Multiple Bennet Acceptance Ratio
CV	Collective Variables

URLs referenced

Page ii
https://www.e-cam2020.eu . . . https://www.e-cam2020.eu
https://www.e-cam2020.eu/deliverables . . . https://www.e-cam2020.eu/deliverables
Internal Project Management Link . . . https://redmine.e-cam2020.eu/issues/131
donal.mackernan@ucd.ie . . . mailto:donal.mackernan@ucd.ie
http://creativecommons.org/licenses/by/4.0 . . . http://creativecommons.org/licenses/by/4.0

Page 1
E-CAM Deliverable D1.1 . . . https://dx.doi.org/10.5281/zenodo.841694
report of the E-CAM Classical MD State of the Art Workshop . . . https://www.e-cam2020.eu/wp-content/uploads/2017/02/SAW-WP1.pdf
https://www.e-cam2020.eu/scientific-reports/ . . . https://www.e-cam2020.eu/scientific-reports/
OPS . . . https://openpathsampling.org
LAMMPS . . . https://lammps.sandia.gov
Jupyter notebook . . . http://jupyter.org

Page 2
E-CAM Deliverable D1.1 . . . https://dx.doi.org/10.5281/zenodo.841694

Page 3
millisecond dynamics of a protein . . . http://pubs.acs.org/doi/abs/10.1021/acs.jpcb.6b02024
TPS . . . http://aip.scitation.org/doi/abs/10.1063/1.475562
TIS . . . http://aip.scitation.org/doi/abs/10.1063/1.1562614
2010 review by Bolhuis and Dellago in Reviews in Computational Chemistry . . . http://onlinelibrary.wiley.com/doi/10.1002/9780470890905.ch3/summary
2008 review by Dellago and Bolhuis in Advances in Polymer Science . . . https://link.springer.com/chapter/10.1007%2F978-3-540-87706-6_3
Wikipedia entry on path sampling . . . https://en.wikipedia.org/wiki/Transition_path_sampling
Aaron Keys's tutorial on path sampling . . . https://sites.google.com/site/aaronskeys/resources/tutorials/transition-path-sampling

Page 5
PLUMED . . . http://www.plumed.org/home

Page 6
pymbar . . . https://github.com/choderalab/pymbar

Page 7
Classical MD section of the E-CAM Library . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/

index.html#readme-classical-md
MDTraj . . . http://mdtraj.org/
MDTraj . . . http://mdtraj.org/
readme.rst of Contact Map module. . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/contact_maps/readme.html
Merge-Request of Contact Map module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/6

Page 8

readme.rst of Contact Map Parallelization module. . . . http://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/contact_maps_parallelization/readme.html
Merge-Request of Contact Map Parallelization module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/17
binding-kinetics . . . https://www.e-cam2020.eu/pilot-project-biki/
readme.rst of Contact concurrences module not yet public . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/17
Merge-Request of Contact concurrences module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/9

Page 9

here . . . http://dx.doi.org/10.1063/1.4965882
readme.rst of Spring Shooting module. . . . http://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/OpenPathSampling/ops_spring_shooting/readme.html
Merge-Request of Spring Shooting module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/24
proposed . . . http://dx.doi.org/10.1021/acs.jpclett.7b01617

Page 10

readme.rst of Web throwing module . . . http://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/OpenPathSampling/ops_web_throwing/readme.html
Merge-Request of Web throwing module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/7
PLUMED . . . https://doi.org/10.1016/j.cpc.2013.09.018
here . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/OpenPathSampling/ops_plumed_wrapper/readme.html
here . . . http://www.plumed.org/home

Page 11

readme.rst of Plumed wrapper for OpenPathSampling module . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/OpenPathSampling/ops_plumed_wrapper/readme.html
Merge-Request of Plumed wrapper for OpenPathSampling module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/8
shooting from the top . . . https://aip.scitation.org/doi/10.1063/1.4997378
module . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modules/modules/OpenPathSampling/ops_sr_shooter/readme.html
readme.rst of Shooting range shooter module . . . https://e-cam.readthedocs.io/en/latest/Classical-MD-Modul
modules/OpenPathSampling/ops_sr_shooter/readme.html
Merge-Request of Shooting range shooter module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/48
LAMMPS . . . https://lammps.sandia.gov/

Page 12

pymbar . . . https://github.com/choderalab/pymbar

Page 13

readme.rst of Particle insertion Core module . . . https://gitlab.e-cam2020.eu/mackernan/E-CAM-Library/blob/PIcore/Classical-MD-Modules/modules/PIcore/readme.rst
Merge-Request of Particle insertion Core module. . . . https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/65
LAMMPS . . . https://lammps.sandia.gov/
LAMMPS . . . https://lammps.sandia.gov/

Page 14

readme.rst of Particle insertion Hydration module . . . https://gitlab.e-cam2020.eu/mackernan/

E-CAM-Library/blob/PIhydration/Classical-MD-Modules/modules/PIhydration/readme.rst
Merge-Request of `Particle insertion Hydration` module. ... https://gitlab.e-cam2020.eu:10443/e-cam/E-CAM-Library/merge_requests/66

---

[1] Christoph Dellago, David Swenson, Donal MacKernan, Ralf Everaers, and Jony Castanga. Identification/selection of E-CAM MD codes for development, November 2016. URL https://doi.org/10.5281/zenodo.841694.

[2] Kresten Lindorff-Larsen, Paul Maragakis, Stefano Piana, and David E. Shaw. Picosecond to millisecond structural dynamics in human ubiquitin. *J. Phys. Chem. B*, 120(33):8313, 2016.

[3] Christoph Dellago, Peter G Bolhuis, Félix S Csajka, and David Chandler. Transition path sampling and the calculation of rate constants. *J. Chem. Phys.*, 108(5):1964, 1998.

[4] Titus S van Erp, Daniele Moroni, and Peter G Bolhuis. A novel path sampling method for the calculation of rate constants. *J. Chem. Phys.*, 118(17):7762, 2003.

[5] Peter G Bolhuis and Christoph Dellago. Trajectory-Based Rare Event Simulations. In *Reviews in Computational Chemistry, Volume 27*, pages 111–210. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2010.

[6] Christoph Dellago and Peter G Bolhuis. Transition Path Sampling and Other Advanced Simulation Techniques for Rare Events. In *Advanced Computer Simulation Approaches for Soft Matter Sciences III*, pages 167–233. Springer, Berlin, Heidelberg, Berlin, Heidelberg, 2008.

[7] Wikipedia. Transition path sampling — Wikipedia, the free encyclopedia, 2017. URL https://en.wikipedia.org/w/index.php?title=Transition_path_sampling&oldid=720483427. [Online; accessed 20-November-2017].

[8] Aaron Keys. Transition Path Sampling, 2017. URL https://sites.google.com/site/aaronskeys/resources/tutorials/transition-path-sampling. [Online; accessed 20-November-2017].

[9] Robert W. Zwanzig. High-Temperature Equation of State by a Perturbation Method. I. Nonpolar Gases. *J. Chem. Phys.*, 22:1420, 1954.

[10] J. G. Kirkwood. Statistical mechanics of fluid mixtures. *J. Chem. Phys.*, 3:300, 1935.