EMMA - A software package for Markov model building and analysis

Martin Senne*, Christof Schütte†, Frank Noé‡

October 3, 2011

Folding and conformational changes of macromolecules often require the generation of large amounts of simulation data that are difficult to analyze. Markov state models (MSMs) address this challenge by providing a systematic way to decompose the state space of the molecular system into substates and to estimate a transition matrix containing the transition probabilities between these substates. This transition matrix can be analyzed to reveal the metastable, i.e. long-living, states of the system, its slowest relaxation timescales and transition pathways and rates e.g. from unfolded to folded, or from dissociated to bound states. To reduce the technical burden of constructing such MSMs we provide the software framework EMMA (available at `https://simtk.org/home/emma`) to construct, validate and analyse such Markov State Models.

1 Introduction

Molecular dynamics (MD) or related simulation approaches are commonly used to investigate various complex molecular processes, such as transport processes [35], polymer melts [43], electrolytes [19, 54], protein folding [42, 21, 95, 6, 62], protein-ligand binding [30, 10], macromolecular aggregration [81, 28], and intramolecular transitions [27, 60, 63]. Many of these processes are characterized by a large state spaces, and the relevant states or conformations not known *a priori*. Hence, relevant reaction coordinates are often hard to find or would necessarily be high-dimensional to include all relevant directions. Projecting the dynamics onto low-dimensional *a priori* reaction coordinates often leads to deceptive results [44, 58, 52].

Moreover, many of these molecular processes involve rare events which require a large amount of sampling. Recent technical advances have enabled a substantial increase of

*Department for Mathematics and Computer Science, FU Berlin - martin.senne@fu-berlin.de
†Department for Mathematics and Computer Science, FU Berlin - christof.schuette@fu-berlin.de
‡Department for Mathematics and Computer Science, FU Berlin - frank.noe@fu-berlin.de

the sampling, in particular fast simulation codes [31, 66, 48], public access to supercomputers, and efficient use of GPUs for molecular dynamics simulation [82, 88, 26, 29] . Up to aggregate millisecond simulation data can be generated with folding@home [80] and the Anton MD supercomputer [79].

The combination of large amounts of trajectory data and the fact that relevant states are *a priori* unknown call for efficient and objective ways to analyze the simulation data, and additionally to gain advice where more simulation will be needed. Markov (state) models (MSMs) address this problem and have received a surge of interest in the last few years [92, 58, 39, 33, 93, 11, 70, 52, 17, 76, 63, 69]. In MSMs, the molecular state space is discretized into microstates and the transition probabilities or rates between these microstates are estimated from the data. Due to the high dimensionality of macromolecular systems, microstates can usually not be defined in terms of a grid discretization, but are better obtained from clustering. The resulting transition or rate matrix can then be analyzed in order to gain insight into the relevant metastable states [59, 94, 77], the essential (slow) dynamical processes and their timescales [57, 62], transition pathways between substates of special interest (such as unfolded and folded subsets) [62, 30, 90]. It has also been shown that MSMs can be systematically compared to experimental data, e.g. obtained from kinetic experiments such as temperature-jump, fluorescence correlation or time-resolved IR data [61, 57, 78, 9, 96].

Despite the substantial advantages of MSM analysis, simple but potentially misleading analysis techniques such as principal component analysis, coordinate mapping and histogramming are still much more widely used. This may be due to the fact that the construction and analysis of MSMs is technically more challenging. In order to make MSMs more accessible, we provide the EMMA software that makes the currently most common features of MSM analysis available to public use. Another currently available MSM software package is MSMbuilder [2].

A word of warning is appropriate: The construction of Markov models still involves a lot of choices to be made (distance metric, clustering and estimation methods, several parameters), which involve some degree of experience to be made correctly. The MSM research is not yet at a stage where all choices could be safely made in an automatic fasion to hide this complexity from the user. Thus default parameters may not always be appropriate in a specific situation and EMMA is provided to "use at your own risk".

EMMA is programmed in Java. It provides a number of commands for the construction, validation and analysis of MSMs that can be either executed from command-line and thus be controlled via shell scripts, or alternatively, the EMMA application programming interface can be accessed through Java-compatible user interfaces such as Matlab or Mathematica.

Here, we focus on the command-line-based MSM construction with EMMA that can roughly be described by the following sequence of steps:

1. Generation of the MSM from simulation trajectories. This step consists of:
    a) Clustering of the simulation data to microstates. Currently we support trajectory input formats xtc (Gromacs), dcd (Charmm/NAMD) and ASCII. Available clustering methods are k-centers, k-means, and equidistant clus-

tering in space or time.

b) Determination of an appropriate lagtime τ (the time resolution of the MSM).

c) Estimation of a transition matrix $\mathbf{T}(\tau)$ describing the transition probabilities between microstates.

2. Determination of the metastable sets by means of kinetic clustering using the PCCA method.
3. Validation of the MSM by comparing long-time probabilities of states predicted by the MSM with those directly estimated from trajectory data (Chapman-Kolmogorow test).
4. Analysis of the transition matrix, e.g. by

   a) Calculation of the stationary probability distribution of states, ensemble averages of molecular observables, or free energy differences.

   b) Spectral analysis, i.e. determination of the slowest relaxation timescales of the molecular process and the associated structural rearrangements

   c) Calculation of transition pathways between subsets of special interest (e.g. unfolded→folded or dissociated→bound)

   d) Calculation of dynamical ensemble averages of molecular observables such as perturbation-relaxation or correlation functions that can be compared to kinetic experiments such as temperature-jump or fluorescence correlation spectroscopy.

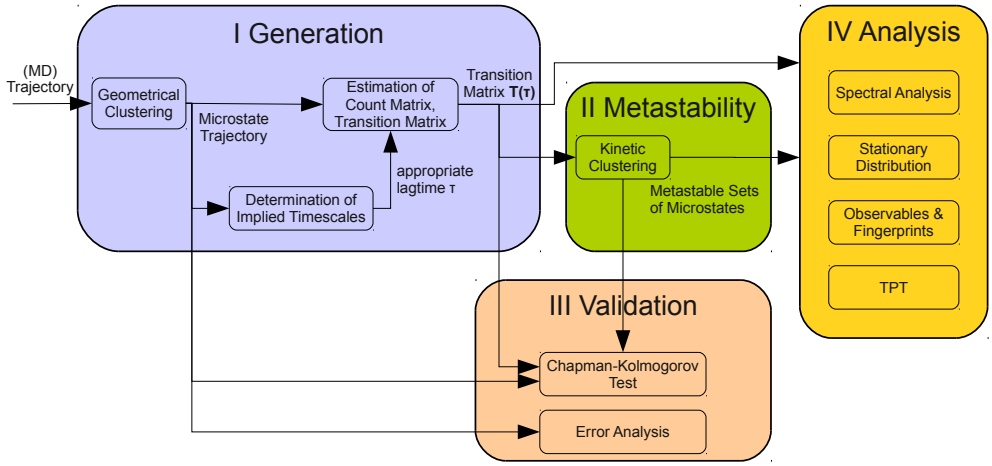

Figure 1: Sketch of the steps involved in construction, validation and analysis of Markov state models

These steps are discussed in the subsequent sections. For a detailed documentation of

the EMMA syntax please refer to the EMMA documentation and tutorial available at `https://simtk.org/home/emma`.

2 Generation of Markov State Models (I)

2.1 Determining microstates by clustering

The first step in building a MSM is to discretize the molecular state space into microstates by dividing the simulation data into clusters. A microstate is defined as the set of molecular configurations exhibiting geometrical or structural similarity. EMMA provides a number of clustering tools to do this. If the user would like to employ a specific microstate definition that cannot be provided by EMMA, the simulation trajectories may be transformed into discrete trajectories by other means (see Appendix for definition of files) and proceed with the subsequent section. In EMMA, clustering can be performed with the `mm_discretize` command, which requires three inputs: (1) simulation trajectory data, (2) the number of clusters / microstates wanted or a way to choose it, (3) the choice of the metric by which distances between structures are measured.

The *input trajectory data* may be in either Gromacs (*.xtc), Charmm/NAMD (*.dcd), or plain ASCII format. It may consist of a single trajectory or multiple/many trajectories.

The *number of clusters* and thus the number of microstates is the most critical part when trajectories are discretized into microstates. The number of microstates has a severe influence on the quality of the Markov model. A small number of microstates may lead to microstates that contain kinetically separated regions, thus leading to a poor Markov model. Increasing the number of microstates thus generally improves the quality of the MSM by reducing the discretization error [69], but may increase the negative effect of limited statistics. The impact of the number of microstates on the quality of the Markov model will be further regarded in Section 4. Typical numbers of microstates are 100's to 10.000's.

In order to cluster data, a *metric* is required which assigns a distance $d(\mathbf{x}, \mathbf{y})$ to pairs of data points $\mathbf{x}$ and $\mathbf{y}$. When the overall position and orientation of a molecular system is not of interest, it is desirable to use a metric that captures intramolecular changes. For this, the minimal RMSD metric[86] has shown to be suitable. Minimal RMSD is a proper distance metric[15] that is useful metric for protein folding, where the entire molecule can exhibit structural changes, but the global position and orientation is not of interest. When part of the simulation system is fixed or can be meaningfully aligned to a reference structure, the direct Euclidean metric is also useful. Such a metric could be useful for e.g. conformational transitions that affect only parts of the molecule or transport processes through a pore. The Euclidean metric may also be used when the input data consists of angles (e.g. dihedral angle values for investigating peptide dynamics). Note that it is not necessary to explicitly account for angle periodicity by the clustering metric - an artificial splitting of the data at the -180/+180 degree boundary

may split kinetically connected states, but such kinetic information is contained in the transition matrix that is being estimated from the transitions between microstates. Nevertheless, unnessecary splitting in angle microstates can be avoided by transforming each angle ϕ into two values $(\sin\phi, \cos\phi)$ and performing the clustering on this extended set of input coordinates [1].

The clustering of input data (usually in continuous space, here denoted as $\mathbf{z}(t) \in \mathbb{R}^n$) into discrete microstate trajectories ($s(t) \in \{1, ..., n\}$). `mm_discretize` performs two steps:

1. *Clustering*: Using the set of simulation trajectories (or a subset thereof) as an input, a clustering algorithm is used to determine a set of n representative points (molecular structures) $\mathbf{c}_0, ..., \mathbf{c}_n$. With the parameter `-stepwidth` only a subset of data points may be used for clustering by just taking every i-th trajectory frame as input. This reduction of input data is often reasonable in order to reduce memory consuption or to overcome too high computational effort when performing clustering.
2. *Assignment*: All trajectory frames $\mathbf{z}(t)$ are assigned to closest cluster centers $\mathbf{c}_i$, defining the so-called Voronoi-partitioning. The discretized trajectory or so called microstate trajectory is obtained as: $z(t) = \arg\min_i d(\mathbf{z}(t), \mathbf{c}_i)$.

With this general approach given, different clustering algorithms (EMMA option `-algorithm`) can be applied to determine cluster centers. The following clustering algorithms are implemented:

- **K-Means**: a well-known and established clustering approach that partitions the input data points $\mathbf{z}(t)$ into sets $S = \{S_1, ..., S_n\}$ such that S is a (local) optimum of the minimization problem $S = \underset{S}{arg\,min} \sum_{t=1}^{N} \sum_{\mathbf{z}(t)\in S_i} d(\mathbf{z}(t), \mathbf{c}_i)^2$. This is done by iterating two steps:
  - Voronoi assignment of $\mathbf{z}(t)$ to cluster representatives $\mathbf{c}_i$
  - Update clusters by: $\mathbf{c}_i = \frac{1}{|S_i|} \sum_{\mathbf{z}(t)\in S_i} \mathbf{z}(t)$.

  Note that K-means can not be used with minimal RMSD metric. If minimal RMSD metric is desired, please use one of the subsequent clustering algorithms.
- **K-Centers** [16] is a fast algorithm to partition the input data points $\mathbf{z}(t)$ into sets $S = \{S_1, ..., S_n\}$ in such a way as to approximate the optimization problem $\sum_{i,j,k,l} \max_{\mathbf{x}_i \in S_j} \min_{\mathbf{y}_k \in S_l} d(\mathbf{x}, \mathbf{y})$.
  Here, a simple and fast greedy approximation algorithm is used: The first iteration chooses an arbitrary data point, defined as $\mathbf{c}_1$. Then, for $n-1$ iterations, a data point $\mathbf{c}_i$ which maximizes $\sum_{j=1}^{i} d(\mathbf{c}_i, \mathbf{c}_j)$ is sought. Note that this version of K-centers has a tendency to find outliers as representatives which might only provide a good clustering for large numbers of clusters [2]. We do therefore not recommend K-Centers and prefer regular spatial or regular temporal clustering (below).
- **Regular spatial clustering**: Cluster clusters are chosen to be approximately equally separated in the conformation space with respect to the distance metric

used. The distance between cluster centers is controlled by the parameter $d_{\min}$. In detail the cluster centers are determined as follows:

- The first data point $\mathbf{z}(0)$ is taken to be the first cluster center $\mathbf{c}_1$. Let $n = 1$ be the current number of clusters.
- Iterate datapoints $\mathbf{z}(t)$:
  When a data-point $\mathbf{z}(t)$ is found, for which $d(\mathbf{z}(t), \mathbf{c}_i) > d_{\min}$ is fulfilled for all existing $\mathbf{c}_i$, then $\mathbf{c}_{n+1} = \mathbf{z}(t)$ becomes a new cluster center and n is incremented.

Regular spatial clustering guarantees that the conformation space is partitioned in a roughly equidistant manner. Despite its simplicity we have found it to be a good way to build microstates for an MSM [69]. Note that the number of clusters n that will be obtained by the method strongly depends on the choice of the threshold $d_{\min}$.

- **Regular temporal clustering**: Given a step length s, every s-th data point of the input trajectory set is selected as cluster center. If the trajectory has length N, then $n = \lfloor \frac{N}{s} \rfloor$ cluster centers are selected. This approach makes most sense when the trajectory data can be assumed to be close to equilibrium simulations, i.e. they consist of trajectories that are long compared to the system's slowest relaxation times, or are generated by multiensemble simulations. In this case the cluster centers are picked from the approximate Boltzmann distribution sampled by these simulation methods.

Once clustering is performed, the discretized output trajectories will be determined and be written out to a user-defined target directory. For typical settings, the clustering step will be faster than the assignment step and thus the runtime of the clustering is $O(Nn)$ where N denotes the number of data points (trajectory frames) and n the number of clusters. Nevertheless, clustering is often the most expensive step in MSM construction and it is often desirable to reduce the dimensionality of the input data to the coordinates that are relevant for describing the process of interest before clustering (e.g. α-Carbon coordinates, Domain centers, backbone dihedrals, etc.).

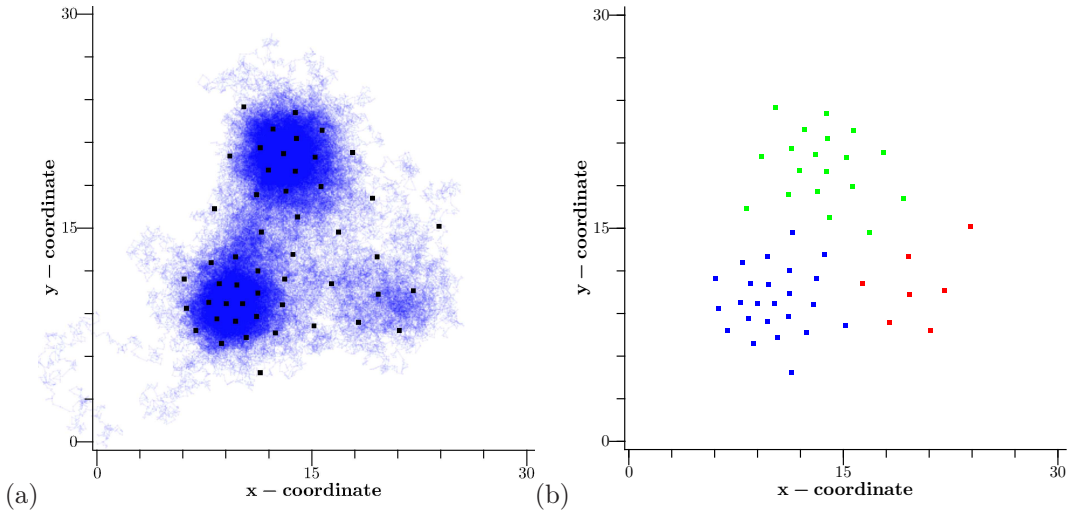

Figure 2: (a) Illustration of a two-dimensional model trajectory in a three-basin potential landscape included in the EMMA examples. The green curve shows the trajectory $\mathbf{z}(t_j)$. The 30 cluster centers $\mathbf{c}_i$ determined by the k-means algorithms are shown as black dots. (b) Assignment of the thirty microstates to three metastable sets (colored red, green, blue) determined via kinetic clustering (PCCA) of the transition matrix.

2.2 Connectivity Test

The program `mm_connectivity` tests which microstates are dynamically connected and can output the largest connected subset of microstates, which is usually used to conduct the Markov model estimation on.

Two microstates i and j are said to be connected, if trajectories exist that go from i to and j and from j to i. A set of states is said to be connected when all states in this set can be reached from all other states. It is important to have dynamical connection between the states used to build a Markov model because only within a connected component one can calculate a well-defined stationary probability distribution, and this is a prerequesite for the correct functionality of many MSM algorithms.

If all microstates used are dynamically connected, the analysis can be continued without intervention. When the data is found to be split into several connected sets, the largest set can be output and used as an input into the programs `mm_estimate`, `mm_impliedtimescales` and `mm_cgtest`.

There may be different reasons why the state space Ω is seperated in several connected sets:

- During the discretization step, cluster centers may have been determined, but without being assigned in the partitioning step. This can happen with k-means clustering. These microstates are never visited and thus are not belonging to the

main connected set of states. Empty states will appear as connected sets with single states, and can be removed without loosing information.

- Especially in the beginning and in the end of microstate trajectories it can happen that sets of states are visited only once, The trajectory never "returns" to these states. E.g. the sequence of microstates $\{1, 2, 3, 2, 3, 4, 5, 6, 5, 6, 4, 5, 7, 8, 9\}$ possesses no way to return back to states $\{1, 2, 3\}$ from states $\{4, 5, 6\}$. Thus the end snippets need to be omitted from the analysis. Removing such end-states usually causes only a small amount of the simulation data to be not described by the model.
- The clustering is too fine and misses the fact that trajectories actually do visit the same kinetic regions. When the connectivity test reports strong disconnectivity, attempt to use a coarser clustering.
- The trajectories visit different parts of state space and were too short or too few to have connected these parts. This problem often arises in the simulation of biomolecules since biomolecular dynamics is often metastable. In such a case, one may still restrict the analysis to the largest connected set of states, but this analysis may miss important contributions of the neglected parts of state space. If the largest connected set does not comprise most of the simulation data, the best solution may be to attempt to generate more simulation data.

Technically, the connectivity of microstates is determined by Tarjan's algorithm[85], which determines the strong connectivity components of a directed graph $G(V, E)$. This graph G has vertices V, here given by the microstates. The set of edges E consist of edges e_{ij}, when at least one transition $i \to j$ has been found in subsequent trajectory time steps. By the application of Tarjan's algorithm to the graph G the strong connected components are determined, the time complexity is $O(|V| + |E|)$.

The output of `mm_connectivity` contains the microstates per communicating class (row-wise). The largest set of microstates is written out for further analysis steps.

2.3 Selection of the lagtime τ: Implied timescales plot

Once a dynamically connected set of microstates has been identified, it is possible to count the number of transitions occurring for any pair of microstates between times t and $t + \tau$. The resulting transition matrix $\mathbf{C}(\tau)$ is then converted into an estimate of the transition matrix $\mathbf{T}(\tau)$ which together with the microstate definition comprises the Markov model. However, until arriving at a $\mathbf{T}(\tau)$ which is useful for further analysis some tests must be conducted. The assumption that the jump process between microstates is Markovian is only approximately true. Its quality depends on two properties [69, 74]: Firstly, is the microstate definition fine enough? Secondly, is τ long enough?

In this step we test whether an MSM of good quality can be obtained for the microstate definition that has been generated, and determine the minimal lag time τ which meets this condition. Note that a transition matrix $\mathbf{T}(\tau)$ has eigenvalue-eigenvector pairs:

$$\mathbf{T}(\tau)\mathbf{r}_i = \lambda_i(\tau)\mathbf{r}_i. \quad (1)$$

Swope et al [84] have exploited that in Markovian dynamics, $\lambda_i(\tau)$ should be exponentially decaying with increasing τ and hence the "implied timescales", i.e. the decay timescales of the spectral components of the MSM should be independent of τ and given by:

$$t_i^*(\tau) = \frac{-\tau}{\ln|\lambda_i(\tau)|}.$$

Assuming sufficiently good statistics, this independence of τ should improve when τ is being increased [69, 74], and for too short lagtimes τ the timescales should always be underestimated [67]. Thus, it was suggested [84] to test whether the timescales $t_i^*(\tau)$ become approximately constant in τ after some minimal value τ' and to then use the Markov model with lag time τ'.

The test for implied timescales is included in EMMA and is conducted by the command `mm_timescales`. The main input options comprise the discretized trajectories (via `-i`), the lagtimes τ (via `-lagtimes`) to compute the implied timescales for and the number of processes (via the option `-neig`).

Apart from these main options, the countmatrix estimation can be adapted by an additional prior to the case of rarely visited microstates (via `-prior`), for details, please refer to Section 2.4. The counting mode for count matrix estimation can be adapted via `-sampling`.

The dynamically connected set of microstates is set by the `-restrictToStates` option and is advised to contain the largest set of microstates determined by the `mm_connectivity`.

In practice, this test may behave in an unexpected way for a number of reasons:

1. When $\tau \gg t_i^*$, the numerical solution of the eigenvalue problem (1) may fail, sometimes leading to an apparently linear increase of t_i^*. Such a behavior is sometimes observed for very long lagtimes τ, especially for fast timescales, and is not necessarily a signature of a poor Markov model.
2. The convergence of $t_i^*(\tau)$ to the true implied timescales occurs asymptotically with τ^{-1} [67], i.e. relatively slow. This effect makes it sometimes hard to find a lag time where convergence can be safely assumed.
3. In the case of poor statistics convergence in τ may not be observable.

Despite these issues, the implied timescale plot still provides a useful way to determine whether a good MSM can be found. Fig. 3 shows implied timescales plots of Markov models of the 2D-test dataset and the MR121-GSGS-W dataset, both showing a reasonable convergence of the dominant implied timescales and allowing an appropriate choice of τ to be made.

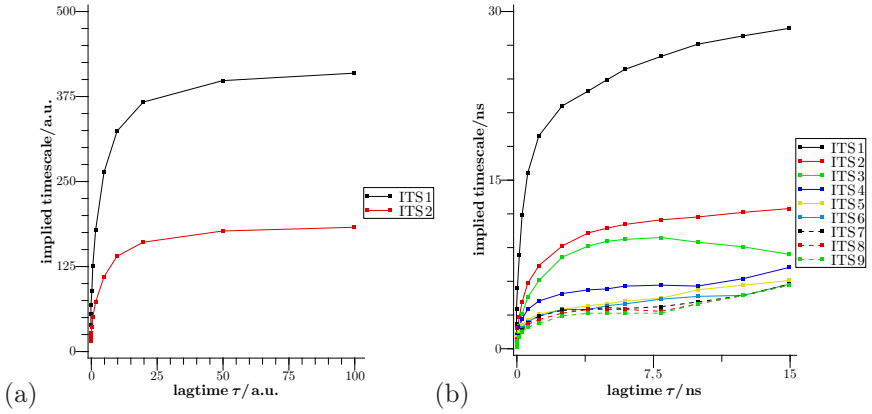

Figure 3: (a) Implied timescales for the 30-state discretization of the two-dimensional diffusion model in a three-well potential shown in Fig. 2. (b) Implied timescales for Markov models $\mathbf{T}(\tau)$ the 800-microstate discretization of MR121-GSGS-W.

2.4 Transition matrix estimation

The main purpose of the `mm_estimate` command is to generate a transition matrix $\mathbf{T}(\tau)$ using the discretized simulation trajectories and the selected lagtime τ as inputs. $\mathbf{T}(\tau)$ together with the definition of the microstate discretization is the actual Markov model and the main object of interest.

Basically, a series of transition matrix estimations has already been conducted for different values of τ when generating the implied timescale plot above. In this step, we just conduct the estimation step once again but select a specific lag time τ for which the implied timescales are approximately converged, in order to generate the transition matrix $\mathbf{T}(\tau)$ that will be subsequently analyzed. The discrete input trajectories should be dynamically connected so as to yield a dynamically connected transition matrix. If multiple disconnected sets of states had been found with the `mm_connectivity` command, the largest connected set of states should be selected with the option `-restrictToStates`.

We briefly summarize the theory and options involved in transition matrix estimation. Let us consider a discretized trajectory $s(t)$ with n observations at time resolution Δt. For the sake of transition matrix estimation we can transform this trajectory into an $n \times n$ count matrix $\mathbf{C}^o$ from a given discrete trajectory:

$$c_{ij}^o(\tau) = \sum_{t=0}^{T-\tau} \delta_i[s(t)]\, \delta_j[s(t+\tau)] \tag{2}$$

Each count matrix element contains the number of transitions seen in the trajectory at time lag τ between pairs of such clusters. When we have multiple trajectories,

their count matrices are simply added. The counting approch given by Eq. (2) is to shift a window of length τ over the data, which is the prefered count mode for the estimation of the transition matrix [69]. However, the resulting counts are not statistically independent and if a count matrix with statistically independent counts is desired, the count mode can also be changed to sample the input data at lag τ.

The user can choose to add a prior matrix $\mathbf{C}^p$ to avoid numerical problems with states that were rarely visited or never left [56, 68]:

$$c_{ij} = c_{ij}^p + c_{ij}^o.$$

The purpose of the prior $\mathbf{C}^p$ is to avoid numerical problems in scenarios with little data. It adds a bias which vanishes when much data is accumulated and c_{ij} is dominated by c_{ij}^o. However, the prior should be chosen as small as possible and as large as necessary to ensure numerical stability. As shown in [68, 2], a useful prior is the neighbor prior: $c_{ij}^p = \alpha$ when $c_{ij}^o(1) + c_{ji}^o(1) > 0$, i.e. when ever two states have been seen adjacently in the trajectory, they are considered as neighbors and a small pseudocount is added to both in order to make sure that the stationary probability is nonzero in all states. The value of α can be set by the `-prior` option, available in `mm_estimate` and `mm_impliedtimescales`. The count matrix $\mathbf{C}$ can be output for additional analysis.

Letting $c_i := \sum_{k=1}^m c_{ik}$, be the sum of counts leaving one state, the trivial estimator, which is in fact the estimator of maximum likelihood given the count matrix $\mathbf{C}$ is given by:

$$\hat{T}_{ij}(\tau) = \frac{c_{ij}}{c_i}. \tag{3}$$

When unbiased molecular dynamics trajectories are used the system is assumed to be in equilibrium, and in this situation we expect detailed balance to hold:

$$\pi_i T_{ij} = \pi_j T_{ji}, \tag{4}$$

however the estimated matrix will generally not fulfill $\pi_i T_{ij} = \pi_j T_{ji}$, simply as a result of statistical deviations from detailed balance for finite amounts of data. Therefore, in cases where the dynamics are in equilibrium is is useful to *enforce* detailed balance on the matrix. Earlier works had suggested to avoid this problem by biasing the observation by adding forward- and backward-count matrices [59, 32], but a better approach is to use an estimator which finds the maximum likelihood $\hat{\mathbf{T}}$ under the constraints (4) [8, 69]. We have implemented the optimal reversible estimator described in [69] in EMMA, and this is available for the `mm_estimation` and `mm_impliedtimescales` commands with the `-reversible` option.

3 Metastability (II): Lumping of microstates to metastable sets

An important analysis tool is the grouping of microstates to metastable set, i.e. the sets of states where there is rapid interconversion within the sets, and rare transitions

between them. Such a kinetic clustering is a useful way to reduce the complexity of a possible several-thousand-state MSM to few kinetically relevant sets and thus aids visualization and further analysis [59]. Note that metastable sets are not used to calculate kinetic properties, i.e. to generate a coarse transition matrix. Unless such a coarse-graining is done in a very special manner [71], it would increase the error of the MSM dramatically [73]. Thus, the fine microstate model is always kept as a numerical means to approximately solve the molecular kinetics on a "fine grid", while kinetically grouped macrostates are useful for visualization of relevant sets or grouping of additive properties, such as the probability of states or transition fluxes (see below).

The method of choice for determining kinetic clusters from a transition matrix is PCCA, invented by Schütte et al [77], and later improved by Weber and Deuflhard [20, 93]. The robust version of PCCA (also called PCCA+) [20] is implemented in EMMA and described subsequently. Given a transition matrix $\mathbf{T}(\tau)$ and a number of states $m < n$, PCCA assigns each of the microstates to one of $1, .., m$ clusters or metastable sets. PCCA makes this assignment in a fuzzy way, i.e. the primary result is not a clustering but a membership matrix $\chi \in \mathbb{R}^{m \times n}$ indicating by its elements χ_{ij} to what degree each microstate j belongs to metastable set i, where

$$\sum_i \chi_{ij} = 1 \, \forall i.$$

This matrix can then be used to generate a hard clustering by assigning each microstate to the metastable set it belongs to most:

$$j \in C_i \quad \text{if} \quad \arg\max \chi_{*j} = i$$

This assignment is made based by first finding m microstates that are kinetic centers and therefore representative states for the metastable sets, and then assigning kinetic distances by the coordinates all microstates have in the m-dimensional space of dominant eigenvectors of $\mathbf{T}(\tau)$. A more detailed description can be found in [20, 93, 59].

The `mm_pcca` command implemented in EMMA can provide both the microstate clustering and the fuzzy memberships. The main input parameter is the transition matrix $T(\tau)$ and the number of kinetic clusters which is set by the option `-nclusters`. The determined cluster assignments, either crisp or fuzzy, are written out by the options `-ocrisp` and `-ofuzzy` respectively.

The result of PCCA clustering of the two-dimensional diffusion in a three-state energy landscape is shown in Fig. 2. The PCCA clustering for an MSM obtained from 6 μs simulation data of the MR121-GSGS-W peptide is shown in Fig. 7

4 Validation (III)

4.1 Chapman-Kolmogorov Test

A number of ways to test the validity of Markov models have been proposed in previous papers [65, 83, 12, 69]. A rather direct and easy-to-interpret set of tests compares

long-time observations generated from the model transition matrix $\hat{\mathbf{T}}(\tau)$ with long-time information available from the trajectory data [12, 69]. Here we describe the Chapman-Kolmogorov test proposed in [69] which has been implemented in the EMMA program `mm_chapman`. The Chapman-Kolmogorow test is a strong test of the Markov model, which essentially tests how well the following approximation holds:

$$[\hat{\mathbf{T}}(\tau)]^k \approx \hat{\mathbf{T}}(k\tau),$$

where $\hat{\mathbf{T}}(\tau)$ is the transition matrix estimated for the time series at lag time τ and $\hat{\mathbf{T}}(k\tau)$ is the transition matrix which is estimated for the time series for lagtimes $k\tau$. The matrix $\hat{\mathbf{T}}(\tau)$ is taken to the power of k to simulate a propagation of the system for time $k\tau$. Thus, the test consists of checking whether the Markov model estimated at τ can be propagated to match observed transition probabilities at longer times $k\tau$. This approximate turns to an equality only if the microstate transition dynamics are exactly Markovian.

Since it is practically difficult to obtain an easily interpretable comparison of matrices $\hat{\mathbf{T}}(\tau)^k$ and $\hat{\mathbf{T}}(k\tau)$ directly, we instead choose to propagate a distribution $\mathbf{p}_0$ to $\hat{\mathbf{T}}(\tau)^k$ while we are estimating the quantity $\mathbf{p}_0\hat{\mathbf{T}}(k\tau)$ directly from the trajectory, yielding:

$$\mathbf{p}_0 \cdot [\hat{\mathbf{T}}(\tau)]^k \approx \mathbf{p}_0 \cdot \hat{\mathbf{T}}(k\tau). \tag{5}$$

Here, $\mathbf{p}_0$ is chosen such that it sums up to 1 on a set S_i of interest while being 0 otherwise. Within S_i, $\mathbf{p}_0 \propto \boldsymbol{\rho}$ where $\boldsymbol{\rho}$ is the a set of weights, which usually is taken to be the stationary distribution of $\hat{\mathbf{T}}$. The test is visualized by plotting the total probability on each set S_i tested over times $k\tau$, and checking whether Markov model and direct calculation agree within statistical error (see [69] for further details).

The program `mm_chapman` expects three main input parameters: (1) the transitionmatrix $\mathbf{T}(\tau)$, (2) the input trajectories, which are used to construct the matrix $\mathbf{T}(k \cdot \tau)$ from counting, (3) the sets of states on which the propagation 5 is tested. We suggest to use the PCCA sets calculated with `mm_pcca` at this point, as the metastable states are the ones which by definition have the fewest transitions between them, thus presenting an especially hard test. Alternatively, one can also defined user-specific sets, such as unfolded and folded sets in protein folding [61], or random sets (`-randomsets`). The initial probability $\mathbf{p}_0$ may be either specified by the user (option `-pinit`), or else the stationary distribution of $\mathbf{T}(\tau)$ is used. The propagation length k is given in multiples of the lagtime τ is selected by `-kmax`.

Fig. 4 shows the test results for the 2D diffusion example that comes with EMMA as well as for the MR121-GSGS-W peptide. The sets chosen here are the 3 or 4 metastable sets that have been previously identified with PCCA (see above). Metastable sets are an especially hard test of the Markov model quality, as the transitions between the metastable sets are those which have the least statistics. Note that the agreement at the times $t = \tau$ and at $t \to \infty$ is always expected and is no test of the Markov model quality. The critical test is whether the Markov model reproduces the relaxation for $t > \tau$.

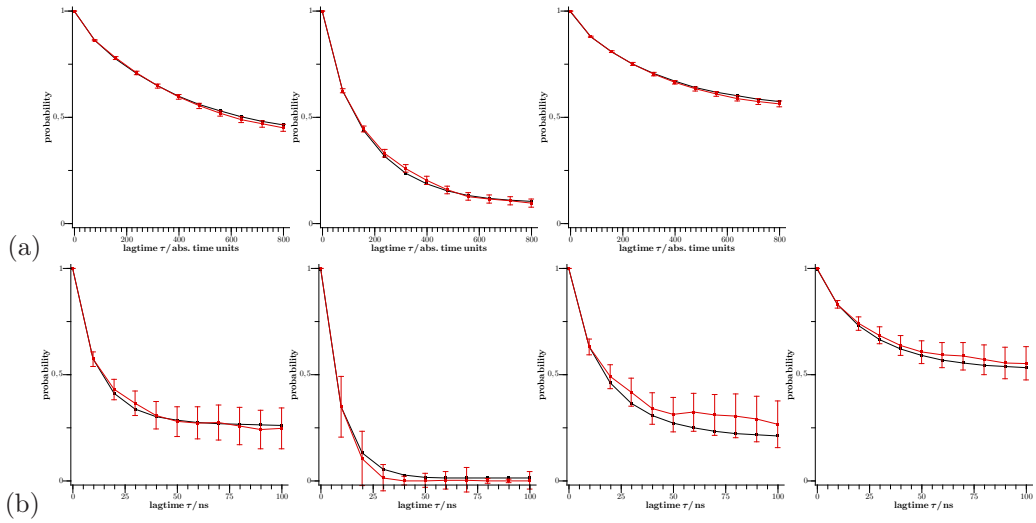

Figure 4: Chapman-Kolmogorov Test. (a) Results for the relaxation out of three metastable states of the 2d-diffusion example. (b) Results for the relaxation out of four metastable states of the MR121-GSGS-W peptide.

5 Analysis (IV)

5.1 Stationary Distribution

A basic quantity of interest is the stationary probability π_i of any microstate i. When the molecular system investigated is in equilibrium, this is given by the Boltzmann distribution on the set of microstates. Denoting the vector of all stationary probabilities as $\boldsymbol{\pi}$, we can estimate them via:

$$\boldsymbol{\pi}^T = \boldsymbol{\pi}^T \mathbf{T}(\tau),$$

i.e. by calculating the eigenvector of the transition matrix with eigenvalue 1 which is subsequently normalized such that $\sum_i \pi_i = 1$. Stationary probabilities can be calculated by the `mm_transitionmatrixAnalysis` program with the `-stationarydistribution` option.

Many scientists prefer to characterize the stability of states via their free energies. The free energies can be directly calculated (via option `-freeenergies`) from the stationary probabilities, via:

$$F_i = -k_B T \ln \frac{\pi_i}{\max \pi_j}$$

which measures the free energy differences with respect to the most stable microstate. See Fig. 5a for an illustration of the relationship between stationary distribution and energy in a one-dimensional model potential.

5.2 Spectral Analysis

The EMMA command `mm_transitionmatrixAnalysis` can be used to decompose the transition matrix into eigenvalues and eigenvectors. Such a spectral analysis is useful in order to gain insight about the slowest conformational transitions of a molecule that take place between its metastable states, and the timescales on which these transitions occur.

To see that, consider the propagation of the probability vector $\mathbf{p}$ in time via the transition matrix:

$$\mathbf{p}_{k\tau} = \mathbf{p}_0 \mathbf{T}^k(\tau).$$

This propagation can be expressed as a sum of single-exponential terms when the transition matrix is diagonalized into $\mathbf{T}(\tau) = \mathbf{R}\mathbf{\Lambda}\mathbf{L}$, where $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues and $\mathbf{R}$ and $\mathbf{L}$ are right and left eigenvector matrices with eigenvectors $\mathbf{l}_i$ and $\mathbf{r}_i$:

$$\begin{aligned} \mathbf{T}(\tau)\mathbf{r}_i &= \lambda_i \mathbf{r}_i \\ \mathbf{l}_i^T \mathbf{T}(\tau) &= \lambda_i \mathbf{l}_i^T \end{aligned}$$

that are properly normalized ($\langle \mathbf{l}_i, \mathbf{r}_i \rangle = \delta_{ij}$) and related by $l_{i,j} = \pi_j r_{i,j}$. We can then write:

$$\begin{aligned} \mathbf{p}_{k\tau}^T &= \mathbf{p}_0^T \sum_{i=1}^{n} \lambda_i^k \mathbf{r}_i \mathbf{l}_i^T \\ &= \sum_{i=1}^{n} \lambda_i^k \langle \mathbf{p}_0, \mathbf{r}_i \rangle \mathbf{l}_i^T, \end{aligned} \tag{6}$$

i.e., the relaxation of any initial distribution to the equilibrium distribution can be understood in terms of a sum of single-exponential relaxations, each of which relaxes with a rate set by eigenvalue λ_i and the associated implied timescale:

$$t_i = -\frac{\tau}{\ln \lambda_i} \tag{7}$$

and the structural rearrangment associated with this timescale is expressed by the sign structure of the corresponding left eigenvector $\mathbf{l}_i$ or right eigenvector $\mathbf{r}_i$ (Both left and right eigenvector carry the same qualitative information, they are just differently weighted. With the use of $\boldsymbol{\pi}$, Eq. 6 can be written in terms of left eigenvectors only

or right eigenvectors only). Fig. 5 shows the diffusion on an one-dimensional energy surface as an example. The yellow eigenvector corresponds to the largest implied timescale, i.e. the slowest relaxation process, and indicates that the corresponding transition occurs between energy basins (A+B) and (C+D), i.e. across the largest energy barrier. The green and blue eigenvectors further subdivide the (A,B) and (C,D) basins.

Eigenvectors are useful to obtain information which structural rearrangements can be associated to a specific relaxation timescale. Since the relaxation timescales are measurable in kinetic experiments, an eigenvalue decomposition (spectral decomposition) of a Markov model transition matrix provides direct structural insight. In many protein folding studies[61, 7, 90], the second eigenvector is analyzed because in kinetic experiments the corresponding slowest relaxation timescale is usually *assumed* to correspond to the folding transition [38] (although this assumption may be wrong [40, 57]). In [57], the first five eigenvectors were analyzed in order to understand the dominant conformation dynamics of glycine-serine peptides of different lengths.

Eigenvectors and eigenvalues can be calculated directly with the `mm_transitionmatrixAnalysis` program. The implied timescales can be calculated from the eigenvalues using Eq. (7).

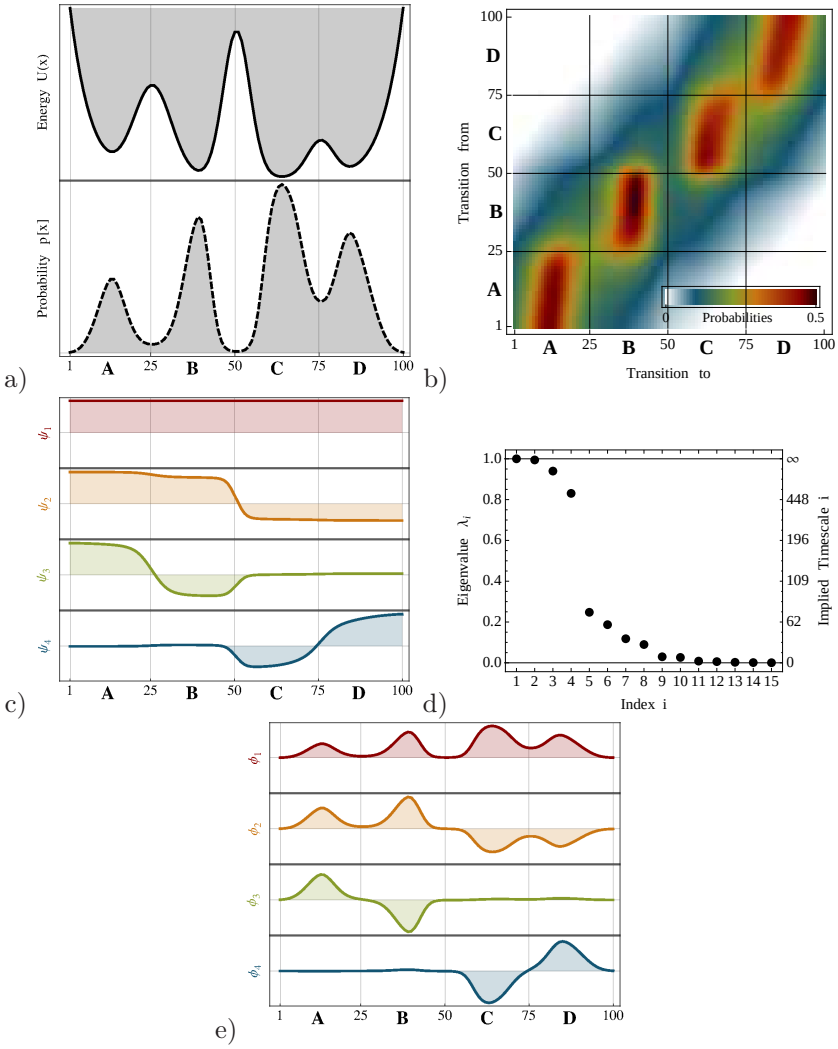

Figure 5: (a) Potential energy function with four metastable states and corresponding stationary density $\mu(\mathbf{x})$, (b) Density plot of the transfer operator for a simple diffusion-in-potential dynamics defined on the range $\Omega = [0, 100]$ (see Supplementary Information), black and red indicates high transition probability, white zero transition probability. Of particular interest is the nearly block-diagonal structure, where the transition density is large within blocks allowing rapid transitions within metastable basins, and small or nearly zero for jumps between different metastable basins. (c) The four dominant eigenfunctions of the transfer operator, $\psi_1, ..., \psi_4$, which indicate the associated dynamical processes. The first eigenfunction is associated to the stationary process, the second to a transition between $A + B \leftrightarrow C + D$ and the third and fourth eigenfunction to transitions between $A \leftrightarrow B$ and $C \leftrightarrow D$, respectively. (d) Eigenvalues of the transfer operator, The gap between the four metastable processes ($\lambda_i \approx 1$) and the fast processes is clearly visible. e) The four dominant eigenfunctions of the transfer operator weighted with the stationary density, $\phi_1, ..., \phi_4$. Figure adapted from [69]

5.3 Transition path theory (TPT)

Transition path theory allows us to analyze the essential statistical features of the reactive transitions between two chosen subsets A and B [89, 49, 50, 61], especially the set of transition pathways between A and B, their relative probilities, the total $A \to B$ flux and the $A \to B$ rate. It was used the first time to investigate the ensemble of protein folding pathways in [61] and protein-ligand binding pathways in [30]. Since then several applications have followed **[90, 10, 9]**. The results of transition path theory can be calculated with the `mm_tpt` program. Transition path properties can be either calculated on the microstates, or on selected coarse states, such as the PCCA sets (option `-coarsesets`).

The essential object needed to calculate statistics pertaining to the $A \to B$ reaction is the committor probability, also called splitting probability or probability of folding [25, 18, 5, 47, 4, 23, 36, 46]. The committor q_i^+ is a state function that provides the probability at any microstate i of next moving to B rather than to A under the action of the system dynamics. By definition, $q_i^+ = 0$ for $i \in A$ and $q_i^+ = 1$ for $i \in B$, while $q_i^+ \in [0, 1]$ for all other states. The committor thus defines a dynamical reaction coordinate, which has the advantage over *ad hoc* reaction coordinates that it does not bring the danger of concealing relevant dynamics of the system [25, 4, 46]. Of special interest is often the isocommittor surface of $q_i^+ \approx 0.5$, which has been interpreted as the transition state ensemble in protein folding theory [64].

We also need the backward committor probability, q_i^-, which is the probability that the process came from A last instead of B. In the case where dynamics are reversible, i.e. the transition matrix fulfills detailed balance, it is simply the counterprobability of the committor: $q_i^- = 1 - q_i^+$.

From a practical point of view, an especially interesting property that can be calculated with TPT is the reactive flux (or "current"), which is generally given by [50, 61]:

$$f_{ij} = \begin{cases} \pi_i q_i^- T_{ij} q_j^+ & i \neq j \\ 0 & i = j \end{cases} \tag{8}$$

which counts the number of transition pathways per time unit that move from state i to state j. f_{ij} does include transitions from unproductive recrossings and transitions into and out of dead ends.

Since it is often desirable to only analyze path probabilities without recrossings or dead ends, the net flux can be calculated via:

$$f_{ij}^+ = \max(q_i^+ - q_i^-). \tag{9}$$

in the special case of reversible dynamics (detailed balance transition matrix), the net flux is given by [3]:

$$f_{ij}^+ = \pi_i (q_j^+ - q_i^+) T_{ij}$$

when $q_i^+ \leq q_j^+$. The net flux is available via the option `-onetflux`.

Since f_{ij}^{+} is defined between microstates, it potentially a very large and detailed network of many small fluxes. In order to get a visual understanding it is often useful to investigate the fluxes between coarse sets of states. The coarse-grained flux and net-flux, as well as the coarse-grained forward- and backward commitor are available for computation by the apropriate command via a preceding `coarse`, i.e. `-coarseflux`. Fluxes can be trivially added for any set partition of the microstates. An especially interesting set partition is the partition into metastable states that has been generated with PCCA (see Sec. 3). As examples, we show the net fluxes amongst the four metastable states of MR121-GSGS-W and the folding flux network of PinWW (taken from [61]) in Fig. 6.

f_{ij}^{+} defines a flux-conserving network flow out of A and into B that can be decomposed into a set of $A \to B$ reaction pathways along with their probabilities [50, 89, 61].

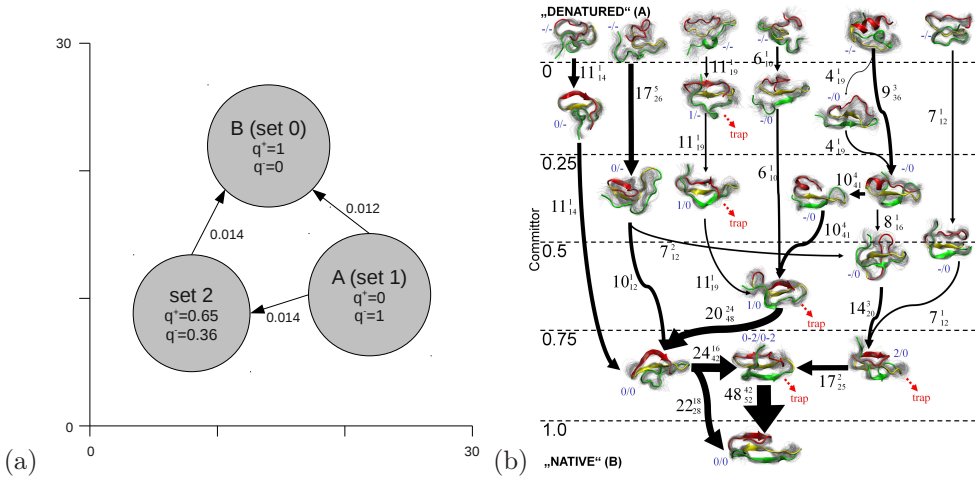

Figure 6: (a) Flux of the transition from the least-populated to the most-populated metastable state of the two-dimensional diffusion in the EMMA tutorial. (b) Flux of the folding transitions amongst the metastable states of the PinWW protein (Figure adapted from [61])

Finally the total flux of the $A \to B$ reaction is given by

$$F_{AB} = \sum_{i \in \mathbb{A}, j \notin \mathbb{A}} f_{ij} = \sum_{i \in \mathbb{A}, j \notin \mathbb{A}} f_{ij}^{+}. \tag{10}$$

and the $A \to B$ reaction rate [61] is given by:

$$k_{AB} = \frac{F_{AB}}{\sum_i \pi_i q_i^{-}}. \tag{11}$$

5.4 Expectation values, Correlation functions, Fingerprints

The EMMA program `mm_observables` is a useful tool to analyze MSMs in a way that allows comparison to experimental measurements. We here assume that a scalar observable a_i is defined for each microstate, resulting in a vector $\mathbf{a}$. Such an observable vector must be stored in a file with a single column containing the elements a_i in rows. Such observables may be any function of the molecular state, such as a fluorescence, a FRET efficiency, a distance, etc.

Many experiments measure ensemble averages. Given a transition matrix $\mathbf{T}(\tau)$ with associated stationary distribution $\boldsymbol{\pi}$ and observable vector $\mathbf{a}$, the ensemble average can be calculated with the `-expectation` command. It is simply estimated by:

$$\mathbb{E}[a] = \sum_{i=1}^{n} \pi_i a_i. \tag{12}$$

The most interesting features of `mm_observables` however allow dynamical observables to be calculated such as perturbation-relaxation and correlation curves as they can also be measured in kinetic experiments. Importantly, `mm_observables` can help to interpret these curves in terms of *dynamical fingerprints* which can be dissected into dynamical features that are associated with individiual relaxation timescales and structural rearrangement processes [57, 40]. Here, we differentiate between two types of kinetic experiments: perturbation and correlation experiments.

In **perturbation experiments**, the ensemble average of an observable is tracked over time while the ensemble relaxes from some perturbed or triggered initial state at time $t = 0$ towards its stationary distribution. The initial trigger may consist of e.g. a jump in temperature [37, 72], pressure [24], a change in the chemical environment [14] or a photoflash [91, 75, 13]. Such time-dependent ensemble averages can be calculated with the commands `-perturbation` and `-relax` *via*:

$$\mathbb{E}[a(k\tau)]_{\mathbf{p}_0} = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{0,i} T_{ij}(k\tau) a_j \tag{13}$$

A special perturbation experiment is the temperature-jump experiment where an ensemble is prepared at temperature T_1 at $t < 0$ and is then suddenly changed to temperature T_2 at $t = 0$. The system is kept at T_2 and relaxes from its old stationary distribution $\mathbf{p}_0 = \boldsymbol{\pi}(T_1)$ to its new stationary distribution $\boldsymbol{\pi}(T_2)$. In cases where simulations have been conducted at both temperatures T_1 and T_2, the stationary distribution at T_1 can be straightforwardly plugged in as initial distribution to a perturbation experiment using the transition matrix at T_2 and the result can be calculated with the `-perturbation` command. If only a single-temperature simulation has been made at temperature T_2 we can still estimate the temperature-jump relaxation curve with the command `-Tjump`. This makes the assumption that the microstates clustering is fine enough such that

$$\begin{aligned}\pi_i(T_1) &\approx Z_1^{-1}\exp(-E_i/k_BT_1)\\ \pi_i(T_2) &\approx Z_1^{-1}\exp(-E_i/k_BT_1)\end{aligned}$$

with some common energy E_i. Based on this assumption, the initial distribution $\mathbf{p}_0$ for an initial temperature T_1 that is close to the target temperature T_2 is approximated, and the temperature-jump relaxation curve is calculated via Eq. 13.

A very common type of kinetic experiments are **correlation experiments**. Correlation experiments may either be realized through scattering techniques such as inelastic neutron scattering [22], or *via* low concentration or single molecule experiments accumulating auto- or cross-correlations of fluctuations, e.g. correlation spectroscopy of the fluorescence intensity [45, 55, 51, 87, 34] or Förster resonance energy transfer efficiency [41, 53]. The `-autocorrelation` command calculates equilibrium autocorrelations of observables **a** via:

$$\mathbb{E}[a(0)\, a(k\tau)]_{\boldsymbol{\pi}} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i\pi_i T_{ij}(k\tau)a_j \tag{14}$$

and the `-crosscorrelation` command calculates the cross-correlation between two observables **a** and **b** via:

$$\mathbb{E}[a(0)\, b(k\tau)]_{\boldsymbol{\pi}} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i\pi_i T_{ij}(k\tau)b_j \tag{15}$$

Instead of directly printing the perturbation-relaxation or correlation curve via `-relax`, one can output the dynamical fingerprint of a perturbation or correlation experiment via the `-fingerprint` command. As explained in detail in [57, 40], the long-timescale part of Eqs. (12), (13) and (14) can each be written in the form

$$\mathbb{E}(...,\, k\tau) = \sum_{i=1}^{m}\gamma_i \exp\left(-\frac{k\tau}{t_i^*}\right)$$

where t_i^* is the i-th implied timescale and γ_i is an amplitude that depends on the specific experiment conducted. The amplitudes γ_i are derived in [57, 40] and can be calculated from scalar products of initial or stationary probability distributions, properly normalized left or right eigenvectors $\mathbf{l}_i$, $\mathbf{r}_i$ and observable vectors **a**, **b**:

$$\begin{aligned}\gamma_i^{\text{perturbation}} &= \langle \mathbf{p}_0, \mathbf{r}_i\rangle\langle \mathbf{a}, \mathbf{l}_i\rangle\\ \gamma_i^{\text{autocorrelation}} &= \langle \mathbf{a}, \mathbf{l}_i\rangle^2\\ \gamma_i^{\text{crosscorrelation}} &= \langle \mathbf{a}, \mathbf{l}_i\rangle\langle \mathbf{b}, \mathbf{l}_i\rangle.\end{aligned}$$

The command `-fingerprint` outputs the amplitudes γ_i and timescales t_i^* for all spectral components with positive eigenvalues. This result can then be plotted into a

dynamical fingerprint plot that can be directly compared to the fingerprint calculated from the experimental measurement. When modeling and sampling errors are small, one can match peaks between experimental and simulated fingerprints. Based on this match one can assign structural processes to experimentally-detectable relaxation timescales (see Fig. 7) or even propose new experiments that optimally amplify low-amplitude features [57, 40]. Note that for comparing the dynamical fingerprint from the MSM to its experimental counterpart, the experimental relaxation curve must somehow be transformed into a spectral density. One way to do this suggested in [57, 40] is implemented in the SciMex package available at `simtk.org/home/scimex`.

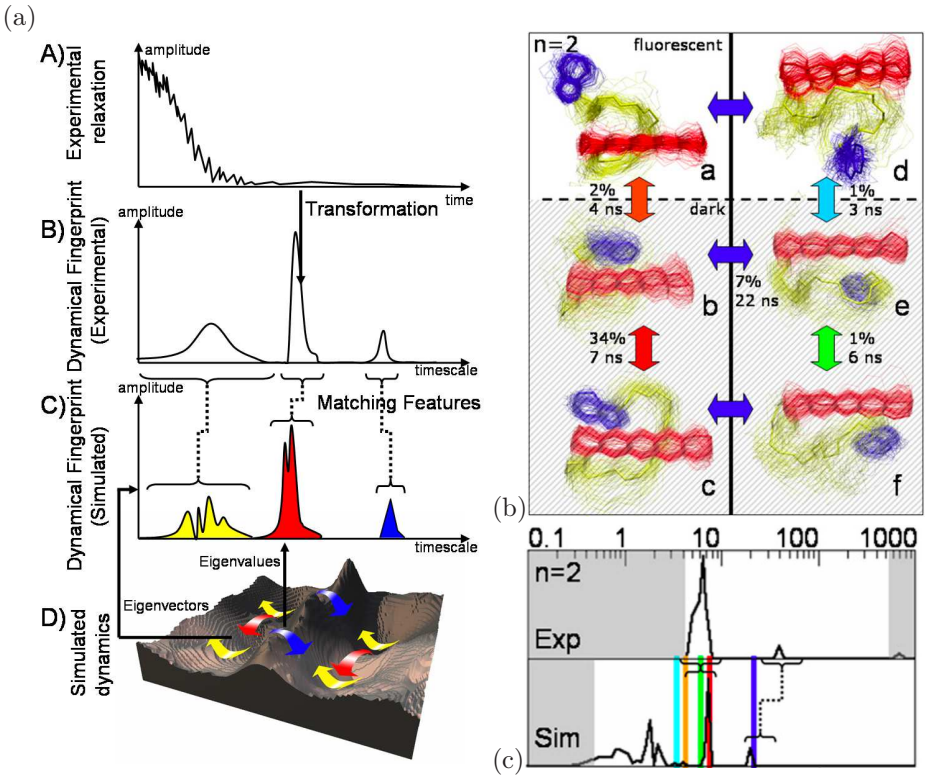

Figure 7: (a) Experimental relaxation profiles (A) can be transformed into dynamical fingerprints (B), which represent the timescales and amplitudes of the relaxation processes in the data without having to predetermine a particular model or number of processes. On the theoretical side, the dynamics on an energy landscape (D) also generates dynamical fingerprints (C), but here each feature can be uniquely assigned to a particular transition or diffusion process on the landscape. If the simulation model is a sufficiently accurate model of the experimental system, structural processes can be assigned to the experimental data by matching features in the experimental and theoretical fingerprint. (b) partition of the conformation space for the peptide MR121-GSGS-W into the 6 most stable metastable (PCCA) states and the associated 5 slowest relaxations (as indicated by the transition matrix eigenvectors). Representative structures are shown in cartoon representation with flexibility indicated by the overlay of line structures. The fluorescent states are shown bright, the dark states are shaded. The slow relaxation processes are indicated by colored arrows. (c) Dynamical fingerprint for a fluorescence quenching experiment of MR121-GSGS-W extracted from single molecule FCS data directly (top), and from the MD simulation Markov model (bottom). Features in experiment and simulation that can be qualitatively associated with each other are linked with dashed lines. The mean positions and amplitudes of the 5 slowest relaxations are marked in colors and correspond to the structural transitions shown on the left. Regions that are unreliable due to measurement or analysis artifacts are grayed out. Figures are taken from [57])

6 Conclusions

Markov state models are used by an increasingly wide community to model and analyze Molecular dynamics data. EMMA provides a number of tools for Markov model construction of molecular kinetics. Molecular dynamics data can be partitioned into microstates using different clustering techniques. Different methods are available to estimate the Markov model transition and validate it by testing its ability to reproduce long-time dynamics of the trajectory data. EMMA has a number of analysis tools that help to calculate stationary probabilities, free energies, relaxation timescales, transition pathways and kinetic experimental observables.

In the future, we plan to support MSM software developers by releasing an API documentation that will permit direct Java access to the functionality of the EMMA classes, and by further extending EMMA's functionality.

Appendix: EMMA file types

Here, a concise definition of the file formats used in EMMA is given. All file formats used for input and output are based on ascii-files. The sole exception are trajectory data, which can also be binary data, according to the Gromacs (xtc), and Charmm / NAMD (dcd) format. The usage of ascii allows a high transparency, since all files can be inspected visually and be easily modified by a user where appropriate. In the following section the file formats are listed.

Simulation / MD Trajectories

Given a time-descrete trajectory $\mathbf{z}(t_j)$ with n frames, where $j \in 0, \ldots, \mathrm{n}-1$ denotes the frame index of the trajectory. Each frame $\mathbf{z}(t_j)$ is of dimension d, thus $\mathbf{z}(t_j) \in \mathbb{R}^d$. The trajectory is stored in row-wise orientation, each line of the file contains one frame of the trajectory:

```
<t_0:double>   <z_1_frame_0:double>   ... <x_d_frame_0:double>
<t_1:double>   <z_1_frame_1:double>   ... <x_d_frame_1:double>
 ...
<t_n-1:double> <z_1_frame_n-1:double> ... <x_d_frame_n-1:double>
```

Multiple trajectories can be stored in one single file, when using ascii file format. A new trajectory is indicated by a line starting with ! as the first character. Example:

```
0.0 2.35 4.5 3.5
1.0 4.6  2.1 7.9
!
0.0 5.3 5.2 5.1
1.0 3.4 5.1 5.0
2.0 5.4 2.5 2.1
```

indicates two trajectories of dimension 3, which a length of 2 for the first trajectory, a length of 3 for the second trajectory

Discrete-state Trajectories

Given a time-descrete microstate trajectory $\mathbf{z}_\mu(t_j)$ with n frames, where $j \in 0, \ldots, \mathrm{n}-1$ denotes the frame index of the trajectory. Each frame $\mathbf{z}_\mu(t_j)$ deontes a single microstate. The trajectory is stored in row-wise orientation, each line of the file contains one frame of the trajectory:

```
<t_0:double>    <z_mu_frame_0:int>
<t_1:double>    <z_mu_frame_1:int>
 ...
<t_n-1:double> <z_mu_frame_n-1:int>
```

Example:

```
0.0 3
1.0 4
!
0.0 4
1.0 5
2.0 2
```

Matrix files

Dense matrix format

A matrix $M \in \mathbb{R}^{r \times c}$ is stored in dense format as defined below. The header line is required.

```
DENSE <r:int> <c:int>
<m_0_0:double>   <m_0_1:double>   ... <m_0_c-1:double>
<m_1_0:double>   <m_1_1:double>   ... <m_1_c-1:double>
  ...
<m_r-1_0:double> <m_r-1_1:double> ... <m_r-1_c-1:double>
```

Sparse matrix format

A matrix $M \in \mathbb{R}^{r \times c}$ is stored in a sparse format as defined below. The header line is required. The file contains only entries $m_{i,j}$, for which is $m_{i,j} \neq 0.0$. Such an entry is defined as i, j, $m_{i,j}$.

```
SPARSE <r:int> <c:int>
<row(m_i_j):int> <col(m_i_j):int> <m_i_j:double>
<row(m_i_j):int> <col(m_i_j):int> <m_i_j:double>
... lines of the above format for every entry not zero ...
```

Vector files

All vectors $\mathbf{v}$, e.g. a stationary distribution vector, are written column-wise and in dense format. A vector $\mathbf{v} \in \mathbb{R}^d$ is stored as defined below. Currently there is no header. The i-line represents the i-th entry v_i of $\mathbf{v}$.

```
<v_0:double>
<v_1:double>
    ...
<v_d:double>
```

State selection files

The state selection file contains a set of microstates. A file of this type is generated by the command `mm_connectivity` and contains, if generated by the above command, these microstates, which belong to the largest connected component.

This file can be used for input, where the option `-restrictToStates` is available, thus for the command `mm_timescales, mm_transitionmatrixEstimation` and `mm_chapman`.

Each row of the file contains one state.

```
<state:int>
<state:int>
...
```

Set definition format

A set definition file contains sets $\mathbb{S}$ of state sets. E.g. all microstates belonging to metastable states. The i-th row of the file corresponds to the set $S_i \in \mathbb{S}$ and contains all states (arbitrarily many) belonging to that specific set, separated by spaces.

```
<state_of_set_0:int> <state_of_set_0:int>
<state_of_set_1:int> ...
...
```

Example, which defines three sets $S_0 = \{2, 3, 5, 6\}$, $S_1 = \{1, 4\}$ and $S_2 = \{0, 7, 8, 9\}$ respectively:

```
2 3 5 6
1 4
0 7 8 9
```

References

[1] Alexandros Altis, Phuong H. Nguyen, Rainer Hegger, and Gerhard Stock. Dihedral angle principal component analysis of molecular dynamics simulations. *The Journal of Chemical Physics*, 126(24):244111+, June 2007.

[2] Kyle Beauchamp, Gregory Bowman, Thomas Lane, Lutz Maibaum, Imran Haque, and Vijay Pande. Msmbuilder2: Modeling conformational dynamics at the picosecond to millisecond scale. *J. Comp. Theo. Chem. (submitted)*, 2011.

[3] Alexander Berezhkovskii, Gerhard Hummer, and Attila Szabo. Reactive flux and folding pathways in network models of coarse-grained protein dynamics. *The Journal of chemical physics*, 130(20), May 2009.

[4] Robert B Best and Gerhard Hummer. Reaction coordinates and rates from transition paths. *Proc. Nat. Acad. Sci. USA*, 102(19):6732–6737, Jan 2005.

[5] Peter G Bolhuis, David Chandler, Christoph Dellago, and Phillip L Geissler. Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annu. Rev. Phys. Chem.*, 53:291, Jan 2002.

[6] Gregory R. Bowman, Kyle A. Beauchamp, George Boxer, and Vijay S. Pande. Progress and challenges in the automated construction of Markov state models for full protein systems. *J. Chem. Phys.*, 131(12):124101+, September 2009.

[7] Gregory R Bowman, Kyle A Beauchamp, George Boxer, and Vijay S Pande. Progress and challenges in the automated construction of markov state models for full protein systems. *J. Chem. Phys*, 131(12):124101, Jan 2009.

[8] Gregory R Bowman, Daniel L Ensign, and Vijay S Pande. Enhanced modeling via network theory: Adaptive sampling of markov state models. *J. Chem. Theo. Comp.*, 6(3):787–794, Jan 2010.

[9] Gregory R. Bowman, Vincent A. Voelz, and Vijay S. Pande. Atomistic Folding Simulations of the Five-Helix Bundle Protein Î»6-85. *Journal of the American Chemical Society*, 133(4):664–667, February 2011.

[10] Ignasi Buch, Toni Giorgino, and Gianni De Fabritiis. Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proceedings of the National Academy of Sciences*, 108(25):10184–10189, June 2011.

[11] Nicaolae V. Buchete and Gerhard Hummer. Coarse master equations for peptide folding dynamics. *J. Phys. Chem. B*, 112:6057–6069, 2008.

[12] Nicolae-Viorel Buchete and Gerhard Hummer. Coarse master equations for peptide folding dynamics. *J. Phys. Chem. B*, 112(19):6057–6069, Jan 2008.

[13] Janina Buck, Boris Fürtig, Jonas Noeske, Jens Wöhnert, and Harald Schwalbe. Time-resolved nmr methods resolving ligand-induced rna folding at atomic resolution. *Proc. Natl. Acad. Sci. USA*, 104(40):15699–15704, October 2007.

[14] Chi-Kin Chan, Yi Hu, Satoshi Takahashi, Denis L. Rousseau, William A. Eaton, and James Hofrichter. Submillisecond protein folding kinetics studied by ultrarapid mixing. *Proc. Natl. Acad. Sci. USA*, 94(5):1779–1784, March 1997.

[15] John D Chodera, Nina Singhal, Vijay S Pande, Ken A Dill, and William C Swope. Automatic discovery of metastable states for the construction of markov models of macromolecular conformational dynamics. *J. Chem. Phys*, 126(15):155101, Apr 2007.

[16] S. Dasgupta and P. Long. Performance guarantees for hierarchical clustering. *J. Comput. Syst. Sci.*, 70(4):555–569, June 2005.

[17] B. de Groot, X. Daura, A. Mark, and H. Grubmüller. Essential dynamics of reversible peptide folding: Memory-free conformational dynamics governed by internal hydrogen bonds. *J. Mol. Bio.*, 301:299–313, 2001.

[18] Christoph Dellago, Peter G Bolhuis, and Phillip L Geissler. Transition path sampling. *Adv. Chem. Phys.*, 123:1, Jan 2002.

[19] Markus Deserno, Axel Arnold, and Christian Holm. Attraction and Ionic Correlations between Charged Stiff Polyelectrolytes. *Macromolecules*, 36(1):249–259, January 2003.

[20] P. Deuflhard and M. Weber. Robust perron cluster analysis in conformation dynamics. *ZIB Report*, 03-09, 2003.

[21] Ken A. Dill and Hue S. Chan. From levinthal to pathways to funnels. *Nature Structural and Molecular Biology*, 4(1):10–19, January 1997.

[22] Wolfgang Doster, Stephen Cusack, and Winfried Petry. Dynamical transition of myoglobin revealed by inelastic neutron scattering. *Nature*, 337(6209):754–756, February 1989.

[23] Rose Du, Vijay S Pande, Alexander Yu Grosberg, Toyoichi Tanaka, and Eugene I Shakhnovich. On the transition coordinate for protein folding. *J. Chem. Phys*, 108:334, 1998.

[24] Charles Dumont, Tryggvi Emilsson, and Martin Gruebele. Reaching the protein folding speed limit with large, sub-microsecond pressure jumps. *Nature Methods*, 6(7):515–519, May 2009.

[25] Weinan E, Weiqing Ren, and Eric Vanden-Eijnden. Transition pathways in complex systems: Reaction coordinates, isocommittor surfaces, and transition tubes. *Chem. Phys. Lett.*, 413(1-3):242–247, 2005.

[26] Peter Eastman and Vijay S. Pande. Efficient nonbonded interactions for molecular dynamics on a graphics processing unit. *J. Comput. Chem.*, 31(6):1268–1272, October 2010.

[27] S. Fischer, B. Windshuegel, D. Horak, K. C. Holmes, and J. C. Smith. Structural mechanism of the recovery stroke in the myosin molecular motor. *Proc. Natl. Acad. Sci. USA*, 102:6873–6878, 2005.

[28] Razif R. Gabdoulline and Rebecca C. Wade. Protein-protein asociation: Investigation of factors influencing association rates by brownian dynamics simulation. *J. Mol. Biol.*, 306:1139–1155, 2001.

[29] M. J. Harvey and G. De Fabritiis. An Implementation of the Smooth Particle Mesh Ewald Method on GPU Hardware. *Journal of Chemical Theory and Computation*, 5(9):2371–2377, September 2009.

[30] Martin Held, Philipp Metzner, and Frank Noé. Mechanisms of protein-ligand association and its modulation by protein mutations. *Biophys. J.*, 100:701–710, 2011.

[31] Berk Hess, Carsten Kutzner, David van der Spoel, and Erik Lindahl. GROMACS 4: Algorithms for Highly Efficient, Load-Balanced, and Scalable Molecular Simulation. *Journal of Chemical Theory and Computation*, 4(3):435–447, March 2008.

[32] Danzhi Huang and Amedeo Caflisch. The Free Energy Landscape of Small Molecule Unbinding. *PLoS Comput Biol*, 7(2):e1002002+, February 2011.

[33] Isaac A. Hubner, Eric J. Deeds, and Eugene I. Shakhnovich. Understanding ensemble protein folding at atomic detail. *Proc. Natl. Acad. Sci. USA*, 103(47):17747–17752, November 2006.

[34] Robert R. Hudgins, Fang Huang, Gabriela Gramlich, and Werner M. Nau. A fluorescence-based method for direct measurement of submicrosecond intramolecular contact formation in biopolymers: An exploratory study with polypeptides. *J. Am. Chem. Soc.*, 124(4):556–564, January 2002.

[35] G. Hummer, J. C. Rasaiah, and J. P. Noworyta. Water conduction through the hydrophobic channel of a carbon nanotube. *Nature*, 414(6860):188–190, November 2001.

[36] Gerhard Hummer. From transition paths to transition states and rate coefficients. *J. Chem. Phys*, 120(2):516–523, Jan 2004.

[37] M. Jäger, Y. Zhang, J. Bieschke, H. Nguyen, M. Dendle, M. E. Bowman, J. P. Noel, M. Gruebele, and J. W. Kelly. Structure-function-folding relationship in a ww domain. *Proc. Natl. Acad. Sci. USA*, 103:10648–10653, 2006.

[38] Marcus Jäger, Houbi Nguyen, Jason C. Crane, Jeffery W. Kelly, and Martin Gruebele. The folding mechanism of a beta-sheet: the ww domain. *J. Mol. Biol.*, 311(2):373–393, August 2001.

[39] Mary E. Karpen, Douglas J. Tobias, and Charles L. Brooks. Statistical clustering techniques for the analysis of long molecular dynamics trajectories: analysis of 2.2-ns trajectories of YPGDV. *Biochemistry*, 32(2):412–420, January 1993.

[40] Bettina Keller, Jan-Hendrik Prinz, and Frank Noé. Markov models and dynamical fingerprints: Unraveling the complexity of molecular kinetics. *Chem. Phys. (in revision)*, 2011.

[41] Harold D. Kim, G. Ulrich Nienhaus, Taekjip Ha, Jeffrey W. Orr, James R. Williamson, and Steven Chu. Mg2+-dependent conformational change of rna studied by fluorescence correlation and fret on immobilized single molecules. *Procl. Natl. Acad. Sci. USA*, 99(7):4284–4289, April 2002.

[42] D K Klimov and D Thirumalai. Progressing from folding trajectories to transition state ensemble in proteins. *Chem. Phys.*, 307:251–258, 2004.

[43] Kurt Kremer and Gary S. Grest. Dynamics of entangled linear polymer melts: A molecular-dynamics simulation. *The Journal of Chemical Physics*, 92(8):5057–5086, 1990.

[44] S. V. Krivov and M. Karplus. Hidden complexity of free energy surfaces for peptide (protein) folding. *Proc. Nat. Acad. Sci. USA*, 101:14766–14770, 2004.

[45] Lisa J. Lapidus, William A. Eaton, and James Hofrichter. Measuring the rate of intramolecular contact formation in polypeptides. *Proc. Natl. Acad. Sci. USA*, 97(13):7220–7225, June 2000.

[46] A Ma and Aaron R Dinner. Automatic method for identifying reaction coordinates in complex systems. *J. Phys. Chem. B*, 109:6769–6779, Jan 2005.

[47] Luca Maragliano, Alexander Fischer, Eric Vanden-Eijnden, and Giovanni Ciccotti. String method in collective variables: minimum free energy paths and isocommittor surfaces. *J. Chem. Phys*, 125(2):24106, Jul 2006.

[48] Chao Mei, Yanhua Sun, Gengbin Zheng, Eric J. Bohm, Laxmikant, James, and Chris Harrison. Enabling and Scaling Biomolecular Simulations of 100˜Million Atoms on Petascale Machines with a Multicore-optimized Message-driven Runtime. In *Proceedings of the 2011 ACM/IEEE conference on Supercomputing*, Seattle, WA, 2011.

[49] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Illustration of transition path theory on a collection of simple examples. *J. Chem. Phys*, 125(8):084110, Jan 2006.

[50] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Transition path theory for markov jump processes. *Multiscale Model. Sim.*, 7(3):1192–1219, 2009.

[51] Xavier Michalet, Shimon Weiss, and Marcus Jäger. Single-molecule fluorescence studies of protein folding and conformational dynamics. *Chem. Rev.*, 106:1785–1813, 2006.

[52] Stefanie Muff and Amedeo Caflisch. Kinetic analysis of molecular dynamics simulations reveals changes in the denatured state and switch of folding pathways upon single-point mutation of a -sheet miniprotein. *Proteins*, 70:1185–1195, 2007.

[53] Daniel Nettels, Armin Hoffmann, and Benjamin Schuler. Unfolded protein and peptide dynamics investigated with single-molecule fret and correlation spectroscopy from picoseconds to seconds†. *J. Phys. Chem. B*, 112(19):6137–6146, May 2008.

[54] Roland R. Netz and Jean-François Joanny. Adsorption of Semiflexible Polyelectrolytes on Charged Planar Surfaces: Charge Compensation, Charge Reversal, and Multilayer Formation. *Macromolecules*, 32(26):9013–9025, December 1999.

[55] Hannes Neuweiler, Marc Löllmann, Sören Doose, and M. Sauer. Dynamics of unfolded polypeptide chains in crowded environment studied by fluorescence correlation spectroscopy. *J. Mol. Biol.*, 365:856–869, 2007.

[56] Frank Noé. Probability Distributions of Molecular Observables computed from Markov Models. *J. Chem. Phys.*, 128:244103, 2008.

[57] Frank Noé, Sören Doose, Isabella Daidone, Marc Löllmann, John D. ChoderaJ, Markus Sauer, and Jeremy C. Smith. Dynamical fingerprints: Understanding biomolecular processes in microscopic detail by combination of spectroscopy, simulation and theory. *Proc. Natl. Acad. Sci. USA, in press*, 2011.

[58] Frank Noé and Stefan Fischer. Transition networks for modeling the kinetics of conformational transitions in macromolecules. *Curr. Opin. Struc. Biol.*, 18:154–162, 2008.

[59] Frank Noé, Illia Horenko, Christof Schütte, and Jeremy C Smith. Hierarchical analysis of conformational dynamics in biomolecules: Transition networks of metastable states. *J. Chem. Phys*, 126:155102, 2007.

[60] Frank Noé, Dieter Krachtus, Jeremy C. Smith, and Stefan Fischer. Transition networks for the comprehensive characterization of complex conformational change in proteins. *J. Chem. Theory and Comput.*, 2:840–857, 2006.

[61] Frank Noé, Christof Schütte, Eric Vanden-Eijnden, Lothar Reich, and Thomas R Weikl. Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations. *Proc. Nat. Acad. Sci. USA*, 106(45):19011–6, Nov 2009.

[62] Frank Noé, Christof Schütte, Eric Vanden-Eijnden, Lothar Reich, and Thomas R. Weikl. Constructing the full ensemble of folding pathways from short off-equilibrium simulations. *Proc. Natl. Acad. Sci. USA*, 106:19011–19016, 2009.

[63] Albert C. Pan and Benoît Roux. Building Markov state models along pathways to determine free energies and rates of transitions. *J. Chem. Phys.*, 129(6):064107+, August 2008.

[64] V. Pande. Pathways for protein folding: is a new view needed? *Current Opinion in Structural Biology*, 8(1):68–79, February 1998.

[65] Sanghyun Park and Vijay S Pande. Validation of markov state models using shannon's entropy. *J. Chem. Phys*, 124(5):054118, Jan 2006.

[66] James C. Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D. Skeel, Laxmikant Kalé, and Klaus Schulten. Scalable molecular dynamics with NAMD. *J. Comput. Chem.*, 26(16):1781–1802, December 2005.

[67] Jan-Hendrik Prinz, John D. Chodera, and Frank Noé. Robust rate estimate from spectral estimation. *Phys. Rev. Lett. (submitted)*, 2011.

[68] Jan-Hendrik Prinz, Martin Held, Jeremy C. Smith, and Frank Noé. Efficient computation of committor probabilities and transition state ensembles. *submitted to SIAM Multiscale Model. Simul.*, 2010.

[69] Jan-Hendrik Prinz, Hao Wu, Marco Sarich, Bettina Keller, Martin Fischbach, Martin Held, John D. Chodera, Christof Schütte, and Frank Noé. Markov models of molecular kinetics: Generation and validation. *J. Chem. Phys.*, 134:174105, 2011.

[70] F. Rao and A. Caflisch. The protein folding network. *J. Mol. Bio.*, 342:299–306, 2004.

[71] Susanna Röblitz and Marcus Weber. A coarse graining method for the identification of transition rates between molecular conformations. *J. Chem. Phys*, 126:024103, Jan 2007.

[72] Mourad Sadqi, Lisa J. Lapidus, and Victor Munoz. How fast is protein hydrophobic collapse? *Proc. Natl. Acad. Sci. USA*, 100(21):12117–12122, October 2003.

[73] Marco Sarich, Frank Noé, and Christof Schütte. On the approximation quality of markov state models. *Multiscale Model. Sim.*, (in press), 2009.

[74] Marco Sarich, Frank Noé, and Christof Schütte. On the approximation error of markov state models. *SIAM Multiscale Model. Simul.*, 8:1154–1177, 2010.

[75] Ilme Schlichting, Steven C. Almo, Gert Rapp, Keith Wilson, Kyriakos Petratos, Arno Lentfer, Alfred Wittinghofer, Wolfgang Kabsch, Emil F. Pai, Gregory A. Petsko, and Roger S. Goody. Time-resolved x-ray crystallographic study of the conformational change in ha-ras p21 protein on gtp hydrolysis. *Nature*, 345(6273):309–315, May 1990.

[76] V. Schultheis, T. Hirschberger, H. Carstens, and P. Tavan. Extracting markov models of peptide conformational dynamics from simulation data. *J. Chem. Theory Comp.*, 1:515–526, 2005.

[77] C. Schütte, A. Fischer, W. Huisinga, and P. Deuflhard. A direct approach to conformational dynamics based on hybrid monte carlo. *J. Comput. Phys.*, 151:146–168, 1999.

[78] Deniz Sezer, Jack H. Freed, and Benoit Roux. Using Markov Models to Simulate Electron Spin Resonance Spectra from Molecular Dynamics Trajectories. *The Journal of Physical Chemistry B*, 112(35):11014–11027, September 2008.

[79] David E. Shaw, Paul Maragakis, Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, Michael P. Eastwood, Joseph A. Bank, John M. Jumper, John K. Salmon, Yibing Shan, and Willy Wriggers. Atomic-Level Characterization of the Structural Dynamics of Proteins. *Science*, 330(6002):341–346, October 2010.

[80] Michael R Shirts and Vijay S Pande. Screen savers of the world unite! *Science*, 290(5498):1903–1904, Dec 2000.

[81] A. Spaar, C. Dammer, R. R. Gabdoulline, R. C. Wade, and V. Helms. Diffusional encounter of barnase and barstar. *Biophys J*, 90:1913–1924, 2006.

[82] John E. Stone, James C. Phillips, Peter L. Freddolino, David J. Hardy, Leonardo G. Trabuco, and Klaus Schulten. Accelerating molecular modeling applications with graphics processors. *Journal of computational chemistry*, 28(16):2618–2640, December 2007.

[83] William C Swope, Jed W Pitera, and Frank Suits. Describing protein folding kinetics by molecular dynamics simulations. 1. theory. *J. Phys. Chem. B*, 108:6571–6581, Jan 2004.

[84] William C Swope, Jed W Pitera, Frank Suits, M Pitman, Maria Eleftheriou, Blake G Fitch, Robert S Germain, Aleksandr Rayshubskiy, TJ Christopher Ward, and Yuriy Zhestkov. Describing protein folding kinetics by molecular dynamics simulations. 2. example applications to alanine dipeptide and a [beta]-hairpin peptide. *J. Phys. Chem. B*, 108(21):6582–6594, 2004.

[85] Robert E. Tarjan. Depth first search and linear graph algorithms. *SIAM J. COMPUT.*, 1:146–160, 1972.

[86] Douglas L Theobald. Rapid calculation of rmsds using a quaternion-based characteristic polynomial. *Acta Cryst.*, A61:478–480, 2005.

[87] Philip Tinnefeld and Markus Sauer. Branching out of single-molecule fluorescence spectroscopy: Challen ges for chemistry and influence on biology. *Angewandte Chemie Intl. Ed.*, 44:2642–2671, 2005.

[88] J. A. van Meel, A. Arnold, D. Frenkel, Portegies, and R. G. Belleman. Harvesting graphics power for MD simulations. *Molecular Simulation*, 34(3):259–266, 2008.

[89] E. Vanden-Eijnden. Transition path theory. pages 453–493. 2006.

[90] Vincent A. Voelz, Gregory R. Bowman, Kyle Beauchamp, and Vijay S. Pande. Molecular Simulation of ab Initio Protein Folding for a Millisecond Folder NTL9. *J. Am. Chem. Soc.*, 132(5):1526–1528, February 2010.

[91] A. Volkmer. One- and two-photon excited fluorescence lifetimes and anisotropy decays of green fluorescent proteins. *Biophys. J.*, 78(3):1589–1598, March 2000.

[92] D. J. Wales. *Energy Landscapes*. Cambridge University Press, Cambridge, 2003.

[93] M. Weber. Improved perron cluster analysis. *ZIB Report*, 03-04, 2003.

[94] Marcus Weber. Improved perron cluster analysis. *ZIB Report*, 04, Jan 2003.

[95] P. G. Wolynes, J. N. Onuchic, and D. Thirumalai. Navigating the folding routes. *Science*, 267:1619–6616, 1995.

[96] Wei Zhuang, Raymond Z. Cui, Daniel-Adriano Silva, and Xuhui Huang. Simulating the T-Jump-Triggered Unfolding Dynamics of trpzip2 Peptide and Its Time-Resolved IR and Two-Dimensional IR Signals Using the Markov State Model Approach. *The Journal of Physical Chemistry B*, 115(18):5415–5424, May 2011.