QUANTIFYING MEMORY EFFECTS IN QUASI MARKOV STATE MODELS

Franziska Teichmann

Bachelor Thesis

Supervisor: Prof. Dr. Gerhard Stock

Albert-Ludwigs-University of Freiburg
Faculty of Mathematics and Physics
Institute of Physics
June 2023

Abstract

The standard way to study conformational changes of biomolecular systems is through Markov state modelling. Since a Markov state model (MSM) is built under the assumption of well-separated timescales in the system's dynamics, it has some limitations for cases in which this condition is not satisfied. Therefore, a new approach is needed to deal with these non-Markovian dynamics. In this thesis, the quasi Markov state model (qMSM) proposed by Huang et al. [14–16] is introduced and applied to different systems. It is derived from the generalized master equation (GME) and includes the non-instantaneous response of the system in a memory kernel.

As a demonstration of the different steps needed to build a qMSM, a six-state toy model is investigated first. Out of a single transition count matrix and by applying a good and a bad lumping projector, macrostate transition probability matrices are obtained. These build the basis for the MSM and are used to calculate the memory kernel for the qMSM. The MSM and the qMSM are compared and validated by looking at the implied timescales and the Chapman-Kolmogorov test. For good lumping, both models perfectly describe the dynamics, while for bad lumping only the qMSM shows perfect agreement, indicating that qMSMs are less influenced by a bad lumping of the microstates into macrostates than MSMs.

These steps are replicated by applying them to the real biological systems HP35 and T4 lysozyme, for which MSMs were already built in previous studies [11, 17]. For the already well-understood system HP35, the qMSM shows very good agreement in the Chapman-Kolmogorov test, while the MSM fails the test for certain states. This outperformance of MSMs by the qMSM is also found when reducing the length of the trajectory to test the stability of both models. In contrast, for T4 lysozyme both models do not describe the dynamics perfectly, although the qMSM clearly showed a better agreement than the MSM.

Contents

Chapter 1

Introduction

Proteins are the workhorses of biological systems, driving and sustaining the intricate machinery of life's processes. A human cell typically contains thousands of different proteins, which are macromolecules consisting of an amino acid sequence [1]. This sequence defines the individual 3-dimensional structure of each protein. With their ability to vary their structure in conformational changes, e.g. folding [2] or ligand binding [3], proteins perform a variety of functions in organisms [4]. To understand illnesses caused by proteins' malfunctions or the advantages of different drug designs, it is therefore not sufficient to know only the structure of proteins. The dynamics are the key to the answer of the question: how do proteins work?

To observe conformational changes of macromolecules, which take place within a wide range of timescales (from picoseconds up to seconds), a high temporal resolution on an atomic scale is desirable. Therefore, molecular dynamics (MD) simulations are widely used as a complementary tool to experiments [4, 5]. They provide information about the fast motions of each atom in the system under study for up to hundreds of microseconds [6].

The resulting trajectory of atomic coordinates can be analysed with the aim of breaking the high dimensional process down to transitions between a few metastable states representing specific conformations. These states can be obtained by applying dimensionality reduction algorithms, that provide a small subset of variables, and clustering methods identifying geometrically and dynamically close regions in the resulting free energy landscape. Each time frame of the MD simulation data can be assigned to one of these states, resulting in a state trajectory. The challenge that comes with coarse-graining the complex molecular dynamics is to construct conformational states such that a timescale separation between the fast intrastate and the slow interstate processes is obtained. If this

separation is satisfied for a specific time step, called lag time, the states' evolution can be seen as memory-less or Markovian, i.e. the current conformation only depends on the previous step but not on the system's history. This allows creating a so-called Markov state model (MSM) [7–10]. An MSM is based on a transition probability matrix, which can be directly obtained from the state trajectory. However, the case of a clear timescale separation is in practice hard to achieve. Thus, MSMs often just qualitatively reproduce the process of interest but can (partially) fail to provide quantitative predictions [11].

For non-Markovian processes, the generalized master equation (GME) [12, 13] provides a framework to determine a time-dependent function, called memory kernel, considering the non-instantaneous response of the system. Depending on the quality of the timescale separation, the kernel usually decays fast. It builds the basis for a new model approach, the quasi Markov state model (qMSM), proposed by Huang et al. [14–16], which predicts time-dependent transition probabilities directly from the GME. The memory kernel lifetime and the lag time are directly connected to the length of MD simulation needed for the respective model to accurately represent the system. Since the memory kernel lifetime is usually significantly shorter than the lag time, qMSM promises the need of less MD simulation data. Moreover, by calculating the memory kernel and the transition probability matrix explicitly from the GME, the qMSM does not significantly exceed the costs of building an MSM.

In this thesis, MSMs and qMSMs will be directly compared by applying them to different systems. First, a simple six-state toy model reproduced from Huang et al. [14] will serve as a step-by-step demonstration on how to build a qMSM out of a Markovian trajectory. Subsequently, the widely studied protein HP35 will be investigated [17, 18]. As MSMs already provided good results for the system, it serves well for a quantitative comparison to a qMSM. The used state trajectory is taken from a recent work of Nagel et al. [17]. For HP35, there will also be an analysis with shortened state trajectories, assessing how the models would turn out for a reduced MD simulation length. Finally, T4 lysozyme is studied to check how well a qMSM performs for a clearly non-Markovian trajectory [11, 19].

Chapter 2

Theory and Methods

2.1 Proteins

Proteins are linear polymer molecules which consist of chained up units called amino acids. These units are also named residues because when two amino acids link together covalently, typically one oxygen and two hydrogen atoms of the original molecule are lost in the form of water. Thus, residues are the monomers in the polymeric chain. The amino acid's basic structure (main chain) is an amino group (NH_2) at one end and a carboxylic acid group (COOH) at the other, linked by a C_α-atom [20]. The main chain builds the backbone of the protein. A side chain bonds covalently at the C_α-atom [1], and the type of chain defines the type of amino acid. There are 20 different standard amino acids of which all 18000 types of protein molecules in a human cell are composed of. On average, a protein has around 500 residues which form a unique sequence defined as the primary protein structure.

The secondary structure types include mainly the α-helix and the β-sheet, which form at smaller parts of the amino acid chain [21]. In an α-helix, the backbone spirals around its long axis and hydrogen bonds form between every fourth amino acid. The β-sheet consists of two or more strands, which can be parallel or anti-parallel, connected by many hydrogen bonds. About 60% of the residues participate in α-helices and β-sheets. Secondary structure and side chain interactions result in a specific 3-dimensional arrangement, the tertiary structure [1]. There are also proteins which consist of several subunits. The spatial arrangement of these polymer chains is seen as the quaternary structure [22].

Because of the amino acids side chains interacting and the Brownian motion of solvents, proteins exhibit temperature-driven structural dynamics. The interac-

tions depend on the type of side chain, which are classified as polar or non-polar. Therefore, the polarity defines the kind and the strength of bonding between two residues. Proteins can also have an unfolded or denatured structure, i.e., they exist as a chain with many conformations (high entropy). During the folding process, the entropy decreases and the protein ends up in its singular native structure.

The relationship between the sequence and the native structure is called the folding code [1]. It is unique for every protein type, since the composition of amino acids determines the forces acting in the folding process. The native structure can vary depending on the function it should perform. However, the protein's function can often not be solely derived only from the 3-dimensional structure. Often the function results from transitions between structures, which can affect small or large conformational changes. Therefore, the protein's motion is also a part of the causal chain [1]

$$\text{SEQUENCE} \longrightarrow \text{STRUCTURE} \longrightarrow \textbf{MOTION} \longrightarrow \text{FUNCTION}.$$

Proteins perform a variety of functions within an organism, including DNA replication, providing structure to cells, transporting molecules or acting as enzymes and antibodies [22]. To understand the dynamics underlying these very important processes, especially for medical purposes, is of huge interest. The goal is to gain insight into protein motions so that illnesses due to proteins' malfunctions can be cured and drugs can be designed to be more precise and effective.

2.2 MD Simulations

Since a protein is built of numerous fast moving atoms, one wants a good time resolution as well as an atom-scale spatial resolution to capture the movement of every single atom. Standard experiments (like nuclear magnetic resonance or mass spectroscopy) may provide a detailed structural characterization but are typically limited in their time resolution. As a complementary tool, molecular dynamics (MD) simulations are widely used to overcome this challenge [23, 24]. They are a computer simulation technique which allows looking at every single atom of the investigated system and their temporal evolution in detail. For that, the protein's starting structure has to be known, which can be obtained, e.g. from crystallization experiments [25]. The protein is then placed in a simulation box with periodic boundary conditions surrounded by a solution (typically water). To get the interactions between the atoms of the system, a potential energy function is defined, which is derived from the molecular structure (atoms composition and positions). The different kinds of interaction are combined in a force field U_{tot} which by default consists of a term considering local interactions U_{bonded} and one for long range interactions $U_{\text{non-bonded}}$ [26]

$$U_{\text{tot}} = U_{\text{bonded}} + U_{\text{non-bonded}}. \tag{2.1}$$

The first term includes bond stretching, bond bending and bond torsion, while the second, non-bonded term considers the Coulombian and the Lennard-Jones interactions. Out of that, the force which every atom undergoes can be determined as

$$\mathbf{F}_i = \nabla_i \left[U_{\text{bonded}}(\mathbf{R}) + U_{\text{non-bonded}}(\mathbf{R}) \right], \tag{2.2}$$

where $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_N)$ includes the Cartesian coordinates of all N atoms. Numerical integration methods such as Euler or Verlet are used to integrate over time steps shorter than the fastest motion in the molecules, iteratively, providing a trajectory of the atoms in the protein. One data point of the trajectory includes all the atom's Cartesian coordinates at the respective time frame.

Since macromolecular processes happen on a wide range of timescales (from a few picoseconds up to seconds), one usually strives for time resolutions of a few femtoseconds to capture all relevant dynamics. This makes long MD simulations (up to a few seconds or longer) computationally very expensive. A more realistic and common dimension is an MD simulation length of a few hundreds of microseconds [6].

2.3 Markov State Models

Understanding complex configurational changes directly by looking at MD simulation data is challenging [27]. Biomolecular processes such as the folding of proteins or ligand binding are therefore often described as stochastic processes happening on a slower timescale than single atom motions. A common way of visualizing the dynamics is by looking at the free energy landscape along some chosen reaction coordinate $\mathbf{x}$ [28]

$$\Delta G(\mathbf{x}) = -k_B T \ln P(\mathbf{x}), \tag{2.3}$$

where T is the temperature and $P(\mathbf{x})$ the probability distribution for the coordinate. Regions of low energy represent metastable states and the transitions between them describe the stochastic and memory-less evolution of the process. The relevant slow (interstate) processes are thus separated from the fast (intrastate) dynamics. The fast processes within the metastable states allow the system to reach thermodynamic equilibrium before transition to another state. Therefore, the evolution of the variable $\mathbf{x}$ only depends on the current location, not on its history. The idea of Markov state modelling is to coarse-grain the complex molecular dynamics by defining such metastable states and transitions between the states, which qualitatively describe the biomolecular process.

The workflow to obtain a Markov State Model (MSM) out of MD simulation data usually includes the following steps:

1. Choosing input coordinates, to describe the internal motions of interest
2. Dimensionality reduction, to identify collective variables essential for the process
3. Clustering, to define metastable states
4. Calculating a transition probability matrix (TPM) as the basis of the MSM
5. Defining a lag time, for which the dynamics are Markovian

Choosing Coordinates

Simply using Cartesian coordinates (provided by MD simulations) as input features is not convenient since they depend on the axial and the spatial orientation of the protein, i.e., they describe the overall motion as well as the internal motion, while one is usually interested in the latter, and separating them afterwards is unfavourable. In addition, with Cartesian coordinates, the number of variables is very high ($3N$ for a number of N atoms). That is why usually the protein's coordinates are transformed into a set of internal coordinates which can be again

multidimensional like the Cartesian coordinates (including all three spatial orientations), but one-dimensional variables such as distances between, e.g. pairs of C_α-atoms, or dihedral angles are preferred. A good choice of coordinates is very important because some are more advantageous for describing the looked at process than others [28, 29]. Moreover, different coordinates can be suitable depending on the system .

Dimensionality Reduction

Even with the change from multi- to one-dimensional and external to internal coordinates, one faces a high number of dimensions (equal to N in the case of one-dimensional coordinates), while many of the variables are not really essential for the dynamics of interest [30]. To further reduce the dimension reasonably by choosing so-called *collective variables*, different algorithms can be used, like principal component analysis (PCA) and time-lagged independent component analysis (tICA).

PCA is a dimensionality reduction method that projects the MD simulation data points in the direction of their maximum covariance [31]. Therefore, instantaneous linear correlations of the variables are removed by diagonalizing the correlation matrix. Projecting the coordinates on the eigenvectors of that matrix, sorted by decreasing eigenvalues, provides the principal components. The first components represent the largest covariance of the data [17]. An alternative dimensionality reduction method is tICA which maximizes the timescale instead of the covariance [1]. Usually, the goal is to end up with $\leqslant 10$ variables as input features for the clustering algorithm. With dimensionality reduction, one gets rid of noise while preserving as much information as possible. For a good choice of variables, the free energy landscape visualizes the pathway of the investigated process. In the case of 10 variables, one has a 10-dimensional free energy landscape.

Clustering

The selected variables are then fed into an algorithm that clusters the MD simulation data into metastable states. To do so, again different methods can be used. The challenge is to determine the energy landscape barriers separating one state from another as precisely as possible. Since a system always strives for the lowest free energy, these local maxima are, in contrast to the energy minima, very low sampled. In general, there are two types of clustering: geometrical and dynamical.

The most common methods for geometrical clustering are k-means and robust density based clustering (RDC). k-means starts with placing k centroids ran-

domly in the free energy landscape [32]. Iteratively, all data points are assigned to the closest centroid, forming clusters. After every step, the centroids position is changed into the current centre of the cluster. The algorithm stops after less than a chosen threshold number of data points are reassigned to a new cluster and the clusters at that point are then defined as states. The drawback of k-means is that the clusters are separated at the mean distance between the centroids. In that way, the energy barriers are possibly not reproduced correctly, leading to a bad timescale separation between intrastate and interstate transitions.

RBC is an alternative geometrical method which solves this problem [33–35]. It computes the density of points within a radius R around each point of the trajectory and estimates the free energy. In that way, the minima of the free energy landscape can be identified. By subsequently increasing the energy threshold, metastable clusters can be defined as states which are separated by their local energy barrier, which are often significantly more well-defined than the ones resulting from k-means. One ends up with a set of metastable states and each time frame of the MD simulation data can be assigned to one of these states, resulting in a state trajectory.

When using geometrical clustering, a larger set of states results in a better resolution of each state. Usually one ends up with a few hundred states, called microstates, providing all relevant information. But by increasing the amount of states, the dynamics of the applied model will be harder to understand from the biological perspective. In addition, numerous states increase the probability of one state being split into many lower populated ones, which leads to worse sampling. Since the aim is not to investigate the free energy landscape as detailed as possible but to gain some qualitative information about conformational changes, the dynamical information obtained from clustering is used to reduce the number of states even further. Dynamical clustering algorithms are used to define a set of macrostates which are composed of dynamically close microstates.

The most probable path (MPP) lumping algorithm makes use of the transition probabilities obtained from the state trajectory [36]. For the first step, a threshold value $Q_{\min}$ is chosen. All states with self-transition probability lower than $Q_{\min}$ are reassigned to the state which they most probably jump into. With every step, the threshold is increased until one obtains the desired amount of states (in general less than 20). MPP can be visualized with a dendrogram, where branches starting from each microstate on the x-axis merge together with increasing $Q_{\min}$ (y-axis).

Another common dynamical clustering method is Perron-cluster cluster analysis (PCCA) which deals with the Perron eigenproblem [37]. The Perron eigenvalue $\lambda_0 = 1$ (only for detailed balance) is the largest eigenvalue of the transition

probability matrix (see section (2.3)) corresponding to the equilibrium conformation (the slowest timescale). PCCA first merges all microstates into one single macrostate and separates them iteratively based on the next slowest right eigenvector, so the kinetically most diverse macrostates are broken down into smaller states [7]. Dynamical clustering provides a set of macrostates and a macrostate trajectory out of the microstate one, where simply all microstates are assigned to a few different macrostates.

There are many possibilities of combining dimensionality reduction and clustering algorithms, which partly lead to divergent results [29]. For every system, one starts anew to discuss the right choice and research in improving and developing new methods goes on.

Transition Probability Matrix

The obtained state trajectory allows determining a so-called transition probability matrix (TPM). In order to do so, one needs to simply count all transitions between the different states for a chosen time step, called lag time τ [38]. The transition count matrix can be written as an $M \times M$ matrix $\mathbf{C}(\tau)$, where M is the number of states and the entries C_{ij} are the number of transitions from state i to state j. The transition probabilities between the states are then calculated by row-normalizing $\mathbf{C}$

$$T_{ij}(\tau) = \frac{C_{i,j}(\tau)}{\sum_j C_{i,j}(\tau)}, \tag{2.4}$$

where T_{ij} is the probability to transition to state j when starting in state i after a lag time τ. The diagonal elements represent the self-transitions, i.e., the probability to stay in the current state. The TPM is the basis of an MSM.

For processes in thermal equilibrium, one finds so-called detailed balance. This means, that for every transition from state i to state j there is also a backward transition and the states dynamics satisfy the detailed balance condition, expressed via the equilibrium populations π_i as

$$\pi_k T_{kl} = \pi_l T_{lk}. \tag{2.5}$$

For poor sampling, it can happen that detailed balance is not satisfied. In that case, one ends up with directed evolution as for non-equilibrium processes, where the system is externally driven out of its equilibrium conformation.

Since the dimensionality reduction and clustering methods usually do not provide perfectly separated timescales, one can not randomly choose the lag time for the TPM, that the MSM is build on. To obtain an evolution of the TPM that is

Markovian from one discrete time step to the next, the time step has to be coarse grained, meaning that it has to be slower than some critical value τ_L. For Markovian processes, the Chapman-Kolmogorov equation holds [38]

$$\mathbf{T}(n\tau_L) = \mathbf{T}(\tau_L)^n, \tag{2.6}$$

with $n \in \mathbb{N}$. The Chapman-Kolmogorov equation validates the MSM and provides a simple way of propagating $\mathbf{T}(\tau_L)$ such that τ_L defines the time resolution of the resulting MSM. It states that the transition probability at the time $n\tau_L$ after one step is equal to the transition probability at τ_L after n steps.

From the TPM, one can directly extract other useful information, such as the relaxation time, also called implied timescale (ITS), which defines the time a system needs to find its way back to its equilibrium state. A system is Markovian when the relaxation time stays constant after some time τ_L. Hence, ITS can be used for the determination of τ_L. For this, one usually calculates the ITS according to [7]

$$ITS_i(\tau) = \frac{-\tau}{\ln \lambda_i(\tau)}, \tag{2.7}$$

where τ is a range of lag times and λ_i are the eigenvalues of the TPMs which are sorted decreasingly. One finds the first eigenvalue corresponding to the equilibrium condition with a value $\lambda_0 = 1$ (Perron eigenvalue) for every time step. The second-highest eigenvalue, λ_1, corresponds to the slowest process of the system and is used to derive τ_L.

With the Chapman-Kolmogorov equation, the relaxation time for a lag time $n\tau_L$ should equal the relaxation time for τ_L in the Markov case [7]

$$ITS_i(\tau_L) = \frac{-n\tau_L}{\ln \lambda_i(n\tau_L)} = \frac{-n\tau_L}{\ln \lambda_i(\tau_L)^n} = \frac{-n\tau_L}{n \ln \lambda_i(\tau_L)} = \frac{-\tau_L}{\ln \lambda_i(\tau_L)}. \tag{2.8}$$

So the proper lag time for the MSM is reached as soon as the relaxation time does not vary with τ anymore. For a perfectly Markovian trajectory, where $\tau_L = 1\,\text{frame}$, the implied timescale would have a constant value.

Since the lag time is the minimal time step describing a Markovian evolution, the MD simulation data on which the model is build needs to span at least this time τ_L. A shorter simulation would not sufficiently capture the relevant slow transition dynamics and it would not be possible to obtain a representative MSM.

Estimation of the Macrostate TPM

As already mentioned, the microstate TPM, can be obtained by determining a transition count matrix for the respective trajectory (equation (2.4)). When calculating the macrostate TPM in the same way, local equilibrium of the system is assumed, which can be expressed with the equilibrium populations π_i and Π_I of the micro- and the macrostates respectively [39]

$$\sum_{i \in I, j \in J} M_{ij} \pi_i = T_{IJ} \Pi_I, \tag{2.9}$$

where M_{ij} is the transition probability from microstate i to j and T_{IJ} the transition probability from macrostate I to J. By rearranging equation (2.9), the macrostate TPM can be obtained as a projecting of the microstates transition probabilities onto a reduced macrostate space

$$\mathbf{T} = \mathcal{X}\mathbf{M}\mathbf{A}^T, \tag{2.10}$$

where the aggregation matrix $\mathbf{A}$ includes the information of the lumping

$$A_{Ji} = \begin{cases} 1 & \text{if } i \in J \\ 0 & \text{else} \end{cases} \tag{2.11}$$

and matrix $\mathcal{X}$ represents the weighting by the equilibrium populations, defined as

$$\mathcal{X}_{Ji} = \begin{cases} \pi_i / \Pi_I & \text{if } i \in J \\ 0 & \text{else} \end{cases} . \tag{2.12}$$

Both $\mathbf{A}$ and $\mathcal{X}$ are $M \times m$ matrices, where M is the number of macrostates and m the number of microstates.

Another approach was formulated by Gerhard Hummer and Attila Szabo, which works also for non-equilibrium systems [39]. When written as matrix multiplication, the resulting macrostate TPM looks as the following

$$\mathbf{T} = \mathbb{I}_M + \mathbf{\Pi}_{\text{macro}} \mathbf{1}_M^T - \mathbf{D}_{\text{macro}} \left(\mathbf{A} \left(\mathbb{I}_m + \mathbf{\Pi}_{\text{micro}} \mathbf{1}_m^T - \mathbf{M} \right)^{-1} \mathbf{D}_{\text{micro}} \mathbf{A}^T \right)^{-1}, \tag{2.13}$$

where $\mathbf{1}_x$ is an x-dimensional identity column-vector and $\mathbb{I}_x$ the x-dimensional identity matrix. $\mathbf{D}_{\text{micro}}$ and $\mathbf{D}_{\text{macro}}$ are matrices with the equilibrium populations of the micro- and macrostates on their diagonal, respectively. $\mathbf{\Pi}_{\text{micro}}$ and $\mathbf{\Pi}_{\text{macro}}$ are therefore m- and M-dimensional column-vectors with π_i and Π_I as their elements.

In this thesis, all models are built on macrostates TPMs obtained with the first method, assuming local equilibrium. For HP35, the plots resulting for the analysis using the Hummer-Szabo projection can also be found in the appendix (8).

2.4 Generalized Master Equation

In practice, the state trajectories usually do not describe a memory-less evolution, therefore often high lag times have to be chosen for the MSM to give qualitative results. But with increasing lag times, information about faster processes are lost and more data from MD simulation is needed, which increases the computational costs. Markovian conditions can also be reached when selecting a higher amount of states, but with more states the results will be harder to understand from the biological perspective. To show how to extend the simple Markov scheme by considering memory, the exact equation of motion is derived here, following [12–14].

The determination of the time evolution of physical ensembles is provided by the Liouville equation [40]

$$\frac{\delta\rho}{\delta t} = [\mathcal{H}, \rho] = \mathcal{L}\rho, \tag{2.14}$$

where $\mathcal{L}$ is the Liouville operator, $\mathcal{H}$ the Hamilton operator and $[A, B]$ the Poisson brackets. It is a statistical classical mechanics equation describing the time evolution of the probability density $\rho(\mathbf{x}, \mathbf{p}, t)$ inside the phase space and can be solved by

$$\rho = e^{\mathcal{L}t}\rho(t = 0). \tag{2.15}$$

While the Liouville equation describes the motion of all molecules in a high-dimensional phase space, slow dynamics like conformational changes can be often well described by some collective variables, i.e., they take place in a lower-dimensional manifold. One option to create an equation of motion for the collective variables is the projection operator technique, separating the relevant slow processes from the fast processes. Here, the technique formulated by Sadao Nakajima and Robert Zwanzig is shown [12, 13]. It begins with writing the density distribution into two projections as $\rho = (\mathcal{P} + \mathcal{Q})\rho$, where $\mathcal{Q} = 1 - \mathcal{P}$ and $\mathcal{P}$ projects onto the relevant part of the system. The goal is to build an equation of motion for $\mathcal{P}$. With this purpose, the Liouville equation is rewritten as

$$\frac{\delta}{\delta t}\mathcal{P}\rho = \mathcal{P}\,\frac{\delta\rho}{\delta t} = \mathcal{P}\mathcal{L}\mathcal{P}\rho + \mathcal{P}\mathcal{L}\mathcal{Q}\rho, \tag{2.16}$$

$$\frac{\delta}{\delta t}\mathcal{Q}\rho = \mathcal{Q}\,\frac{\delta\rho}{\delta t} = \mathcal{Q}\mathcal{L}\mathcal{Q}\rho + \mathcal{Q}\mathcal{L}\mathcal{P}\rho, \tag{2.17}$$

by making use of the linearity of the Poisson brackets. With the solution ansatz for the Liouville equation (2.15), the above equation (2.17) can be solved by

$$\mathcal{Q}\rho = e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\rho(t = 0) + \int_0^t e^{\mathcal{Q}\mathcal{L}t'}\mathcal{Q}\mathcal{L}\mathcal{P}\rho(t - t')dt', \tag{2.18}$$

which is easily verified by writing the function inside the integral as a derivative

$$\begin{aligned}\int_0^t e^{\mathcal{Q}\mathcal{L}t'}\mathcal{Q}\mathcal{L}\mathcal{P}\rho(t-t')dt' &= \int_0^t e^{\mathcal{Q}\mathcal{L}t'}\mathcal{Q}\mathcal{L}\mathcal{P}e^{\mathcal{L}(t-t')}\rho(0)dt \\ &= -\int_0^t d\left(e^{\mathcal{Q}\mathcal{L}t'}\mathcal{Q}e^{\mathcal{L}(t-t')}\right)\rho(0) \\ &= -\left[e^{\mathcal{Q}\mathcal{L}t'}\mathcal{Q}e^{\mathcal{L}(t-t')}\right]_0^t\rho(0) = \mathcal{Q}\rho - e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\rho(0).\end{aligned} \tag{2.19}$$

$\mathcal{Q}\rho$ can be then inserted in equation (2.16) and the Nakajima-Zwanzig equation results

$$\frac{\delta}{\delta t}\mathcal{P}\rho = \mathcal{P}\mathcal{L}\mathcal{P}\rho + \underbrace{\mathcal{P}\mathcal{L}e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\rho(0)}_{=0} + \mathcal{P}\mathcal{L}\int_0^t e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\mathcal{L}\mathcal{P}\rho(t-t')dt'. \tag{2.20}$$

If at the initial time $t = 0$ the system is in equilibrium, where the sum over all thermal fluctuation is zero, Q will be also zero such that the projector $\mathcal{P}$ is the identity at the initial time. With this assumption, the inhomogeneous second term of the Nakajima-Zwanzig equation vanishes. Therefore, the density operator at time zero will be written as ρ_{eq} in the following expressions. The resulting equation of motion (2.20) consists of an unperturbed term describing the raw dynamics of the collective variables and a perturbed one considering the impact of all other degrees of freedom.

To turn the Nakajima-Zwanzig equation (2.20) into an equation of motion for metastable states, one chooses a matrix-based formulation, where the entries of each matrix represent the transitions between the states. For a system described by a set of selected variables $\mathbf{x}$ with M states, the projector $\mathcal{P}$ is determined as [14]

$$\mathcal{P} = \sum_j^M \left|\chi_j(\mathbf{x})\rho_{\text{eq}}(\mathbf{x},\mathbf{p})\rangle\pi_j^{-1}\langle\chi_j(\mathbf{x})\right| \tag{2.21}$$

where ρ_{eq} is the equilibrium probability distribution and π_j the equilibrium population of state j. With the orthogonal indicator functions

$$\chi_j = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is in configuration } j \\ 0 & \text{else} \end{cases} \tag{2.22}$$

and the bracket operation as the ensemble average over the equilibrium distribution for the entire system, π_j can be constructed as

$$\langle\chi_j(\mathbf{x})|\chi_i\rho_{\text{eq}}(\mathbf{x},\mathbf{p})\rangle = \int \chi_j(\mathbf{x})\chi_i(\mathbf{x})\rho_{\text{eq}}(\mathbf{x},\mathbf{p})d\mathbf{p}d\mathbf{x} = \delta_{ji}\pi_j. \tag{2.23}$$

Before determining the relevant matrices, the Nakajima-Zwanzig equation is normalized by the equilibrium probability distribution ρ_{eq}

$$\frac{\delta}{\delta t}\mathcal{P}e^{-\mathcal{L}t} = \mathcal{P}\mathcal{L}\mathcal{P}e^{\mathcal{L}t} + \mathcal{P}\mathcal{L}\int_0^t e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\mathcal{L}\mathcal{P}e^{\mathcal{L}(t-t')}dt'. \tag{2.24}$$

Out of this equation, the transition probability matrix (TPM) can be identified

$$\mathbf{T} = \mathcal{P}e^{\mathcal{L}t}, \tag{2.25}$$

where $T_{ij}(t) = \langle \chi_i(\mathbf{x}) \left| e^{\mathcal{L}t} \right| \chi_j(\mathbf{x})\rho_{\text{eq}}(\mathbf{x})\rangle \pi_j^{-1}$. Furthermore, $\mathcal{P}\mathcal{L}$ can be written as the derivative of the TPM at the starting time by deriving $\mathbf{T}$ and setting $t = 0$ such that $\dot{\mathbf{T}}(0) = \mathcal{P}\mathcal{L}$. The second term of equation (2.24) describes the fluctuations' impact on the relevant slow dynamics, which is defined by the so-called memory kernel

$$\mathcal{K} = \langle \chi_i(\mathbf{x}) \left| \mathcal{L}e^{\mathcal{Q}\mathcal{L}t}\mathcal{Q}\mathcal{L} \right| \rho_{\text{eq}}(\mathbf{x})\rangle \pi_j^{-1}. \tag{2.26}$$

By using the definitions of $\mathbf{T}$, $\dot{\mathbf{T}}(0)$ and $\mathcal{K}$, the commonly used formulation of the generalized master equation (GME) can be expressed

$$\dot{\mathbf{T}}(t) = \dot{\mathbf{T}}(0)\mathbf{T}(t) + \int_0^t \mathcal{K}(\tau)\mathbf{T}(t-\tau)d\tau. \tag{2.27}$$

The convolution integral can be interpreted as a weighting of the TPM by the memory kernel. The value of $\mathcal{K}$ at time τ describes how much $\mathbf{T}(t-\tau)$ affects the resulting values. In other words, it indicates how the conditions at a previous time $t-\tau$ influence the evolution of the system at time t.

Since the TPM contains all information about the system's dynamics of the reduced state space, it can be used to determine the probability of finding the system in a certain state at a chosen time t for an initial condition $\mathbf{P}(0)$ of interest [14]

$$\mathbf{P}(t) = \mathbf{T}(t)\ \mathbf{P}(0), \tag{2.28}$$

where $\mathbf{P}$ is an M-dimensional column-vector with M being the amount of states and $\mathbf{P}(0)$ has unitary norm describing the states' relative contributions at the initial time. Due to the idempotency of the projector ($\mathcal{P}^2 = \mathcal{P}$ per definition [41]) the initial TPM is given by $\mathbf{T}(0) = \mathbb{I}_M$, where $\mathbb{I}_M$ is the $M \times M$ identity matrix, validating equation (2.28).

Markov Limit

For the sake of completeness, the bridge between the GME and the master equation as the underlaying equation of motion of an MSM is build in the following.

For Markovian dynamics, the memory kernel decays infinitely fast in comparison to the characteristic timescale of the projected dynamics, yielding $\mathcal{K}(t) \longrightarrow \mathcal{K}_0\ \delta(t)$ [14]. Only for $t = 0$ the memory kernel has a value $\mathcal{K}_0$. With this so-called Markovian condition (or Markov limit), the equation of motion (2.27) changes into

$$\dot{\mathbf{T}}_{\mathrm{MSM}}(t) = \left(\dot{\mathbf{T}}(0) + \mathcal{K}_0\right) \mathbf{T}_{\mathrm{MSM}}(t), \tag{2.29}$$

named master equation (ME). It can be solved by $\mathbf{T}_{\mathrm{MSM}}(t) = e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)t}\ \mathbf{T}(0)$. For any time step Δt the Markovian TPM results in

$$\mathbf{T}_{\mathrm{MSM}}(t + \Delta t) = e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)\Delta t}\ \mathbf{T}_{\mathrm{MSM}}(t), \tag{2.30}$$

where $e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)\Delta t} = e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)\Delta t}\ \mathbf{T}(0) = \mathbf{T}_{\mathrm{MSM}}(\Delta t)$ since the initial TPM is the $M \times M$ identity matrix. This formulation allows rewriting equation (2.30) in the common MSM notation based on equation (2.28)

$$\begin{aligned} \mathbf{P}(t + \Delta t) &= \mathbf{T}_{\mathrm{MSM}}(t + \Delta t)\mathbf{P}(0) \\ &= \mathbf{T}_{\mathrm{MSM}}(\Delta t)\mathbf{T}_{\mathrm{MSM}}(t)\mathbf{P}(0) \\ &= \mathbf{T}_{\mathrm{MSM}}(\Delta t)\mathbf{P}(t). \end{aligned} \tag{2.31}$$

For the coarse-grained time step $\Delta t = \tau_L$ of an MSM, the Champman-Kolmogorov equation (2.6) can be directly derived from equation (2.30)

$$\begin{aligned} \mathbf{T}_{\mathrm{MSM}}(n\tau_L) &= e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)n\tau_L}\ \mathbf{T}(0) \\ &= \left(e^{(\dot{\mathbf{T}}(0)+\mathcal{K}_0)\tau_L}\ \mathbf{T}(0)\right)^n \\ &= \mathbf{T}_{\mathrm{MSM}}(\tau_L)^n. \end{aligned} \tag{2.32}$$

2.5 Quasi Markov State Models

The quasi Markov state model (qMSM), proposed by Huang et al. [14–16] is directly build from the GME and therefore provides time-dependent state transitions considering the memory of the system. The following section provides a step-by-step instruction on how to obtain a qMSM with using data that would normally be used to build an MSM.

Usually, in a biomolecular system, interstate processes evolve on a much faster timescale than the investigated configurational changes. Therefore, the value of the memory kernel's lifetime τ_K is significantly smaller than the characteristic timescale of the projected dynamics. In this case, the GME (2.27) can be slightly adjusted by changing the upper limit of the convolution integral because $\mathcal{K}(\tau \leqslant \tau_k) = 0$:

$$\dot{\mathbf{T}}(t) = \dot{\mathbf{T}}(0)\mathbf{T}(t) + \int_0^{\min[\tau_K, t]} \mathcal{K}(\tau)\mathbf{T}(t-\tau)d\tau. \tag{2.33}$$

The memory kernel $\mathcal{K}$ can then be obtained by inverting this equation and using the left Riemann sum to discretize the convolution of the memory kernel and the transition probability matrix, providing

$$\mathcal{K}(n\Delta t) = \frac{\dot{\mathbf{T}}(n\Delta t) - \dot{\mathbf{T}}(0)\ \mathbf{T}(n\Delta t)}{\Delta t} - \sum_{m=1}^{n-1} \mathcal{K}(m\Delta t)\ \mathbf{T}((n-m)\Delta t), \tag{2.34}$$

where $t = n\Delta t$ for $n \geqslant 1$. Such discretization can be done here because for the investigated systems only discrete time steps Δt are used. Thus, $\mathcal{K}$ is completely defined by the TPM and its derivative [14]

$$\dot{\mathbf{T}}(n\Delta t) \approx \frac{\mathbf{T}((n+1)\Delta t) - \mathbf{T}(n\Delta t)}{\Delta t}. \tag{2.35}$$

In the above equations (2.34, 2.35) $\mathbf{T}(n\Delta t) = \mathbf{T}_{\mathrm{MD}}(n\Delta t)$ is the transition probability matrix directly obtained from MD simulation data through the count matrix for each time step (equation (2.4)). With equation (2.34) the memory kernel at $t = 0$ is undetermined. One has to be careful that this is taken into account when integrating the generalized master equation to obtain the time-dependent TPMs.

For a better understanding of the memory kernel, the first term of equation (2.34) is rewritten using the definition of the derivative (2.35)

$$\begin{aligned} \dot{\mathbf{T}}(n\Delta t) - \dot{\mathbf{T}}(0)\ \mathbf{T}(n\Delta t) & \\ &= [\mathbf{T}((n+1)\Delta t) - \mathbf{T}(n\Delta t)] - [(\mathbf{T}(\Delta t) - \mathbb{I})\ \mathbf{T}(n\Delta t)] \\ &= [\mathbf{T}((n+1)\Delta t) - \mathbf{T}(n\Delta t)] - [\mathbf{T}(\Delta t)\ \mathbf{T}(n\Delta t) - \mathbf{T}(n\Delta t)]. \end{aligned} \tag{2.36}$$

In the Markovian case, $\mathbf{T}((n+1)\Delta t) = \mathbf{T}(\Delta t)^{n+1} = \mathbf{T}(\Delta t)\ \mathbf{T}(n\Delta t)$ (using equation (2.6)), so the memory kernel becomes zero for the first (and all following) time steps. For the non-Markovian case, the difference (2.36) increases with time but is corrected by the influence of previous memory kernel values (second term of equation (2.34)), leading to a general decay of the kernel. How fast a single element K_{ij} (representing the memory of the transition from state i to state j) decays depends on the quality of the timescale separation between intra- and interstate dynamics after clustering.

Finally, one needs to determine the memory kernel lifetime τ_K for the upper limit of the convolution integral of the GME (2.33). To obtain the kernel lifetime one can simply look at the evolution of the memory kernel elements $\mathcal{K}_{ij}$. Some will decay faster than others, since the local energy landscape looks different for all states. So τ_K is chosen as the time at which all kernel elements have decayed to zero. Another way of determining τ_K is to calculate the mean integral of the memory kernel (MIK) [14]

$$MIK(t) = \frac{1}{N}\sqrt{\sum_{i,j=1}^{N}\left(\int_0^t \mathcal{K}_{i,j}(t')dt'\right)^2}, \tag{2.37}$$

which converges to a stable value if there is no variation of the elements, and therefore indicating the time when all memory kernel elements have decayed, thus, the memory kernel lifetime τ_K.

To finally obtain time-dependent TPMs for the qMSM, the GME (2.33) is solved and integrated for every time $t = n\Delta t$ ($n \geqslant 1$), iteratively

$$\dot{\mathbf{T}}_{\mathrm{qMSM}}(t) = \dot{\mathbf{T}}(0)\mathbf{T}_{\mathrm{qMSM}}(t) + \int_0^{\min[\tau_K,t]} \mathcal{K}(\tau)\mathbf{T}_{\mathrm{qMSM}}(t-\tau)d\tau. \tag{2.38}$$

One starts with the first matrix directly obtained from the MD simulation data $\mathbf{T}_{\mathrm{qMSM}}(\Delta t) = \mathbf{T}_{\mathrm{MD}}(\Delta t)$ and uses the derivative at $t = 0$, $\dot{\mathbf{T}}(0) = \dot{\mathbf{T}}_{\mathrm{MD}}(0)$. For discrete time steps, the integral can be again written as a Riemann sum.

Since the memory kernel lifetime defines the time when the fast intrastate dynamics do not affect the relevant slow conformational processes anymore ($\mathcal{K}(t \geqslant \tau_K) = 0$), τ_K is linked to the minimal length of MD simulation needed to build an accurate qMSM. As the qMSM does not try to find a Markovian description for the studied dynamics, its time resolution is not limited to a lag time, so usually $\Delta t = 1$ frame of the state trajectory is chosen as the time interval for the propagation.

2.6 Validating the Models

Once one has obtained a model, this should be validated in order to verify that it accurately reproduces the dynamics under study.

For the validation of an MSM, the ITS (equation (2.7)) can be used. For a Markovian model, the timescales should converge to a constant value, so that they do not depend on the chosen lag time after a certain time τ_L. In other words, the relaxation time of the system as an intrinsic physical property should not depend on the model.

Furthermore, the Chapman-Kolmogorov test is an illustrating tool to validate the quality of a model. It shows the evolution of the self-transition probability of each state when starting in that state for the dynamics predicted by the applied model. The values directly obtained from the MD simulation data serve as reference. In the Chapman-Kolmogorov equation (2.6), the left term refers to the MD simulation data, on which the model is built and the right term to the MSM, where the propagation is performed by simple matrix multiplication. A good agreement of the MD- and MSM-curves indicates that the model describes accurately a Markovian behaviour. An MSM is constructed in such a way that the evolution of the self-transition probability of a certain state has the same behaviour for every chosen lag time: the curves are just shifted in time and therefore decay faster or slower because of different timescales. When plotting the transition probabilities obtained from the qMSM, they will match the MD reference well by construction, at least up to the kernel lifetime, because of the way the kernel is calculated. All further predicted dynamics are what reveals how well the qMSM describes the studied dynamics.

Related to the Chapman-Kolmogorov test, one can calculate the evolution of the states' population probabilities for different initial conditions out of equation (2.28). When plotting all the states evolutions together for one initial state, the result provides an outlook on probable paths the system takes through the states. This can also be used for the validation of the models, as equation (2.28) uses the TPMs to determine the states' population probabilities. Therefore, one can build 'kinetic plots' with the reference TPMs directly obtained from MD simulation data and with the TPMs of the respective models.

Chapter 3

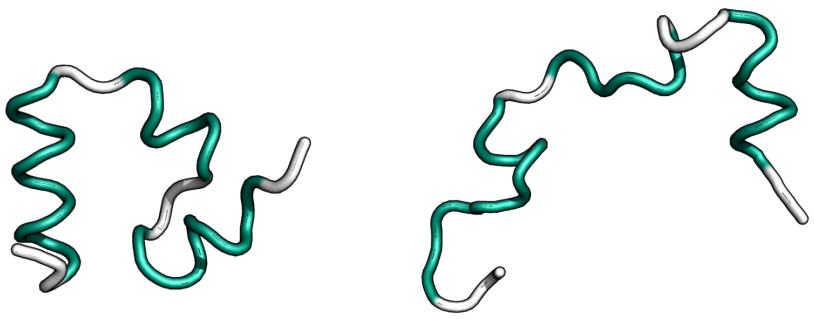
Investigated Systems

After performing the qMSM procedure to a six-state toy model, its performance on actual biological systems will be investigated.

3.1 HP35

The first system is the NLE-NLE mutant of the villin headpiece HP35, a small subdomain of a chicken villin. The backbone of HP35 consists of 35 amino acids which form three α-helices (residues 3 to 10, 14 to 19 and 22 to 32) in the folded conformation. Since it is a small and fast folding protein, many folding events can be reproduced in a single MD simulation [42–44]. The here used state trajectory was obtained by Nagel et al. [17] and is based on the 300 μs-long MD simulation by Piana et al. [18], which includes several folding events. As input features, they chose contact distances defined according to the following: 1. The two residues are three residues apart from each other, 2. The distance between the closest non-hydrogen atoms of two residues is shorter than 4.5 Å, 3. The so-defined contacts are populated at least 30 % of the simulation time. These criteria led to a final set of 42 native contacts. With PCA as dimensionality reduction method, the first five principal components were found to describe 80 % of the total correlation. By applying RDC with a radius of $R = 0.124$, a set of 522 microstates was

obtained, which were further clustered into 12 metastable macrostates by using MPP lumping with a metastability threshold of $Q_{\min} = 0.5$, [29]. The states were then ordered by decreasing fraction of native contacts, i.e., state 1 is the native state, where all contacts are formed and state 12 is the completely unfolded one, seen in figure 3.1.



(a) **(b)**

Figure 3.1: Rendering of HP35 in its crystal structure. (a) The native structure and (b) the completely unfolded conformation. The residues that form the three α-helices are shown in turquoise.

3.2 T4 Lysozyme

The second investigated system is the 154-residue enzyme T4 lysozyme [45–47]. It destroys bacterial cell walls with a mouth like open-close motion and it has a two-domain structure with ten α-helices and one β-sheet in its folded conformation, shown in figure 3.2. The used macrostate trajectory built by Post et al. [11] is based on an 61 μs MD simulation by Ernst et al. [19] capturing six closing and six re-opening processes. Here, a combination of contact distances and side-chain dihedral angles was chosen as input features. Applying the community detection technique called Leiden algorithm provided 85 internal coordinates (82 contact distances and 3 dihedral angles) associated with the open-close transition of T4 lysozyme [48]. To further reduce the number of variables, Post et al. focused on hydrogen bonds and salt bridges, which change during the open-close mechanism. In the end, six coordinates (5 contact distances and 1 dihedral angle) were obtained. The most important coordinates were found to be the distance $d_{20,145}$ (salt bridge between residue 22 and residue 145), describing the mouth opening width x and the locking coordinate $p = d_{4,60}$ (salt bridge between residue 4 and residue 60). The latter describes if residue 4 in the hinge region is in its locked or free state, which is of interest because this residue seems to leave the binding

pocket always alongside the closing of the mouth. With RDC, five metastable states were identified by employing the six coordinates: 1_R, 1_E, 2, 4_R and 4_E with populations of $33.5\,\%, 35.5\,\%, 1.3\,\%, 14.5\,\%, 15.1\,\%$. These can further be separated into two main states, 1 (open) and 4 (closed), and one transition state 2. In the substates, the index R accounts for the fully relaxed equilibrium state and E (for excited) represents the starting position of the open-close transition.

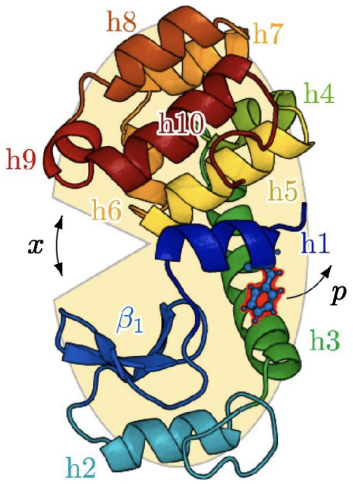


Figure 3.2: Structure of T4 lysozyme from [11], indicating the opening coordinate x of the mouth region and the locking coordinate p that describes the motion of the key residue 4 (red) in the hinge region.

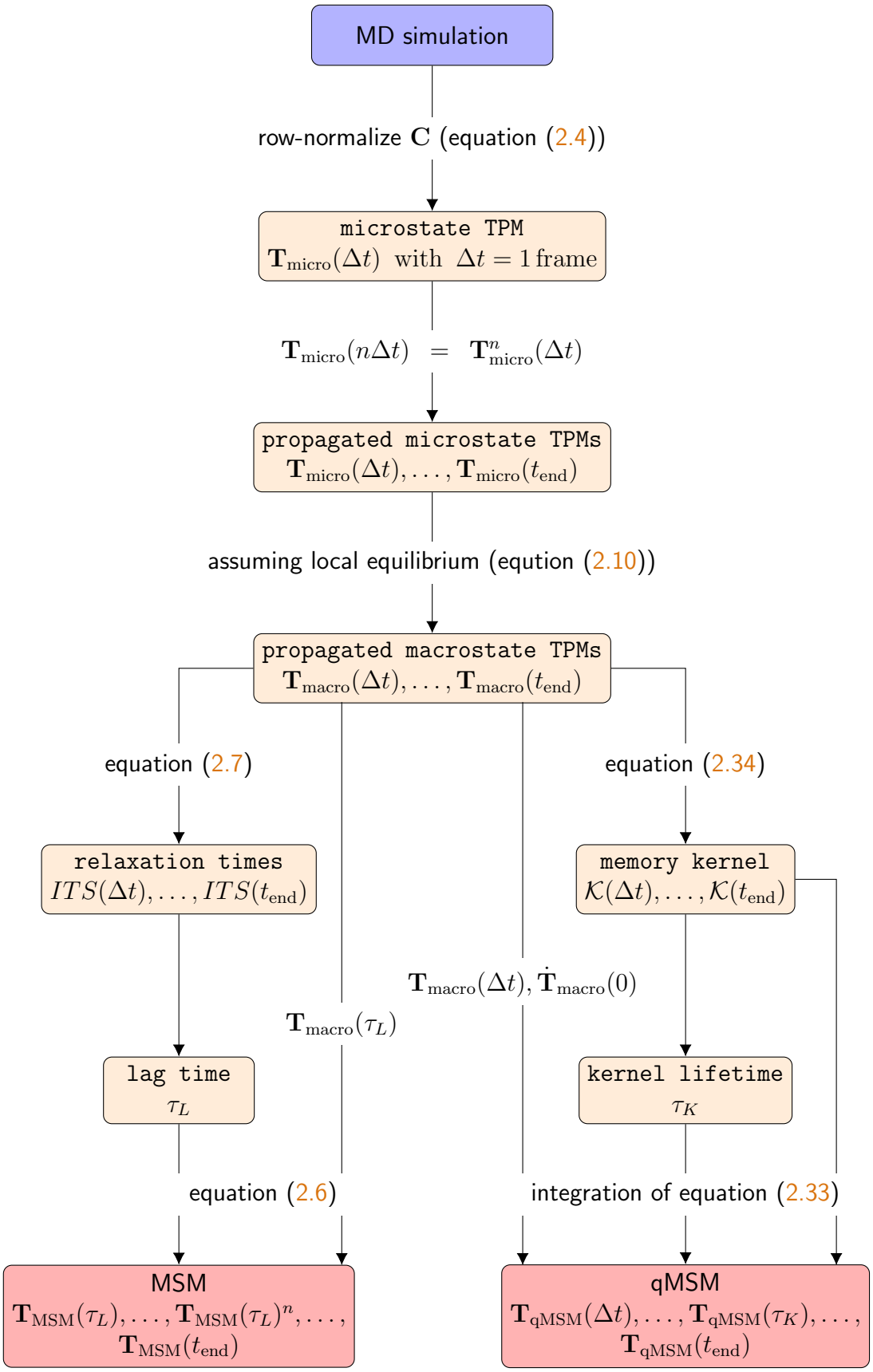
Chapter 4

Six-State Toy Model

4.1 Workflow

The six-state toy model is a simple system of six microstates constructed by Huang et al. [14]. It is built in the way that the microstate transitions are Markovian, so the time step is $\Delta t = 1\,\text{frame}$. Therefore, a single transition count matrix is sufficient to calculate the time-dependent micro- and macrostate TPMs. The idea is to lump the six microstates together into three macrostates and to study the macrostate dynamics of the resulting MSM and qMSM.

The workflow illustrated in the following scheme makes use of equations introduced and explained in chapter 2, providing a step-by-step guide of how to build an MSM and a qMSM for the toy model. It can also be applied to more complex systems when adjusting some steps, e.g. considering non-Markovian microstate dynamics or applying other projection methods.


MD simulation
row-normalize $\mathbf{C}$ (equation (2.4))
microstate TPM
$\mathbf{T}_{\text{micro}}(\Delta t)$ with $\Delta t = 1\,\text{frame}$
$\mathbf{T}_{\text{micro}}(n\Delta t) = \mathbf{T}^n_{\text{micro}}(\Delta t)$
propagated microstate TPMs
$\mathbf{T}_{\text{micro}}(\Delta t), \ldots, \mathbf{T}_{\text{micro}}(t_{\text{end}})$
assuming local equilibrium (eqution (2.10))
propagated macrostate TPMs
$\mathbf{T}_{\text{macro}}(\Delta t), \ldots, \mathbf{T}_{\text{macro}}(t_{\text{end}})$
equation (2.7)
equation (2.34)
relaxation times
$ITS(\Delta t), \ldots, ITS(t_{\text{end}})$
memory kernel
$\mathcal{K}(\Delta t), \ldots, \mathcal{K}(t_{\text{end}})$
$\mathbf{T}_{\text{macro}}(\Delta t), \dot{\mathbf{T}}_{\text{macro}}(0)$
$\mathbf{T}_{\text{macro}}(\tau_L)$
lag time
τ_L
kernel lifetime
τ_K
equation (2.6)
integration of equation (2.33)
MSM
$\mathbf{T}_{\text{MSM}}(\tau_L), \ldots, \mathbf{T}_{\text{MSM}}(\tau_L)^n, \ldots, \mathbf{T}_{\text{MSM}}(t_{\text{end}})$
qMSM
$\mathbf{T}_{\text{qMSM}}(\Delta t), \ldots, \mathbf{T}_{\text{qMSM}}(\tau_K), \ldots, \mathbf{T}_{\text{qMSM}}(t_{\text{end}})$

4.2 From Microstates to Macrostates

For the six-state toy model, the transition count matrix is given by [14] as

$$\mathbf{C} = \begin{pmatrix} 30000 & 100 & & & & \\ 100 & 10000 & 3 & & & \\ & 3 & 3000 & 10 & & \\ & & 10 & 1000 & 1 & \\ & & & 1 & 300 & 3 \\ & & & & 3 & 100 \end{pmatrix},$$

which satisfies detailed balance. The matrix represents a process in which the system is most likely found in the first or second state, since they are quite stable (high self transitions counts C_{11}, C_{22}) and the transitions between them are the most probable interstate transitions. In contrast, state 5 and 6 are the least populated states. The count matrix also reveals that only transitions between 'neighbour' states take place, resulting in a block-diagonal kind of shape.

The corresponding microstate TPM can be calculated from $\mathbf{C}$ via equation (2.4). Since the macrostate dynamics are Markovian for $\Delta t = 1\,\text{frame}$ by construction, the TPM can be directly propagated for multiple steps n of Δt with $\mathbf{T}_{\text{micro}}(n\Delta t) = \mathbf{T}^n_{\text{micro}}(\Delta t)$.

To show the importance of good lumping, the projection from micro- to macrostates will be performed with two different aggregation matrices $\mathbf{A}_{\text{good}}, \mathbf{A}_{\text{bad}}$ which are $M \times m$ matrices with $M = 3$ and $m = 6$. An entry $A_{Ji} = 1$ indicates if microstate $i \in [1, 2, 3, 4, 5, 6]$ is assigned to macrostate $J \in [1, 2, 3]$. For good lumping, a grouping by the strongest transition probabilities is chosen (considering the block-like structure of $\mathbf{C}$). For bad lumping, the grouping is done randomly [14]:

$$\mathbf{A}_{\text{good}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \quad \mathbf{A}_{\text{bad}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

For the projection, local equilibrium is assumed (equation (2.10)). The equilibrium populations of the six microstates are given by [14] as

$$\mathbf{\Pi}_{\text{micro}} = (0.674, 0.226, 0.0675, 0.0227, 0.00681, 0.00226)^T \tag{4.1}$$

and the macrostate equilibrium populations are calculated as the sum over the corresponding microstate populations

$$\begin{aligned} \mathbf{\Pi}_{\text{macro}}^{\text{good}} &= (0.9, 0.0902, 0.00907)^T, \\ \mathbf{\Pi}_{\text{macro}}^{\text{bad}} &= (0.67626, 0.2935, 0.02951)^T. \end{aligned} \tag{4.2}$$

The macrostate TPM $\mathbf{T}_{\text{macro}}(n\Delta t)$ can now be calculated for each time step $n \in [1, \ldots, t_{\text{end}}/\Delta t]$. $t_{\text{end}}^{\text{good}} = 4000$ frames and $t_{\text{end}}^{\text{bad}} = 8000$ frames are taken as the respective end times, determining the length of the models below. These TPMs are the basis for the models which are going to be built and also work as the reference when validating these models.

4.3 Building the Models

qMSM

With equation (2.34) the memory kernel can be determined for $n \geqslant 1$ with a fixed time step of $\Delta t = 1$ frames. In case of good lumping, the memory kernel elements for transitions to state 1 and 2 decay very quickly, while the lifetimes of K_{31}, K_{32} and K_{33} are significantly longer (figure 4.1a, b). This indicates a less well-defined timescale separation between the intra- and the interstate processes associated with state 3. The kernel elements resulting from bad lumping decay in general more slowly than the ones from good lumping, also indicating that here the transitions within the states happen on a similar timescale as the transitions between them. Again, processes associated with state 3 have the most long living, and highest, memory elements. This is reasonable since with bad lumping this state ends up being composed of the most and the least populated microstates. So the timescale separation is the worst for state 3.

The diagonal kernel elements, i.e., the ones corresponding to the self-transitions, always start positive while off-diagonal elements can also be negative. Positive values for self-transitions are due to decreasing transition probabilities, which lead to negative values in the TPMs' derivatives $\dot{\mathbf{T}}(n\Delta t)$ (equation (2.35)), and therefore in particular to $\dot{\mathbf{T}}(n\Delta t) - \dot{\mathbf{T}}(0)\mathbf{T}(n\Delta t) \geqslant 0$, which is the first term in the memory kernel equation (2.34). In contrast, the interstate transitions increase with time, so that $\dot{\mathbf{T}}(n\Delta t) \geqslant 0$. Still, not all off-diagonal kernel elements are negative at the start. To understand why, one can have a look again at the reshaping of the first term of the memory kernel equation (2.36). $T_{ij}((n+1)\Delta t) < [\mathbf{T}(\Delta t)\ \mathbf{T}(n\Delta t)]_{ij}$ results in a negative memory kernel element and for $T_{ij}((n+1)\Delta t) > [\mathbf{T}(\Delta t)\ \mathbf{T}(n\Delta t)]_{ij}$ the kernel element is found positive. In other words, when assuming Markovian dynamics would overestimate the resulting transition probabilities, the negative kernel causes a reduction of the value, and the other way around.

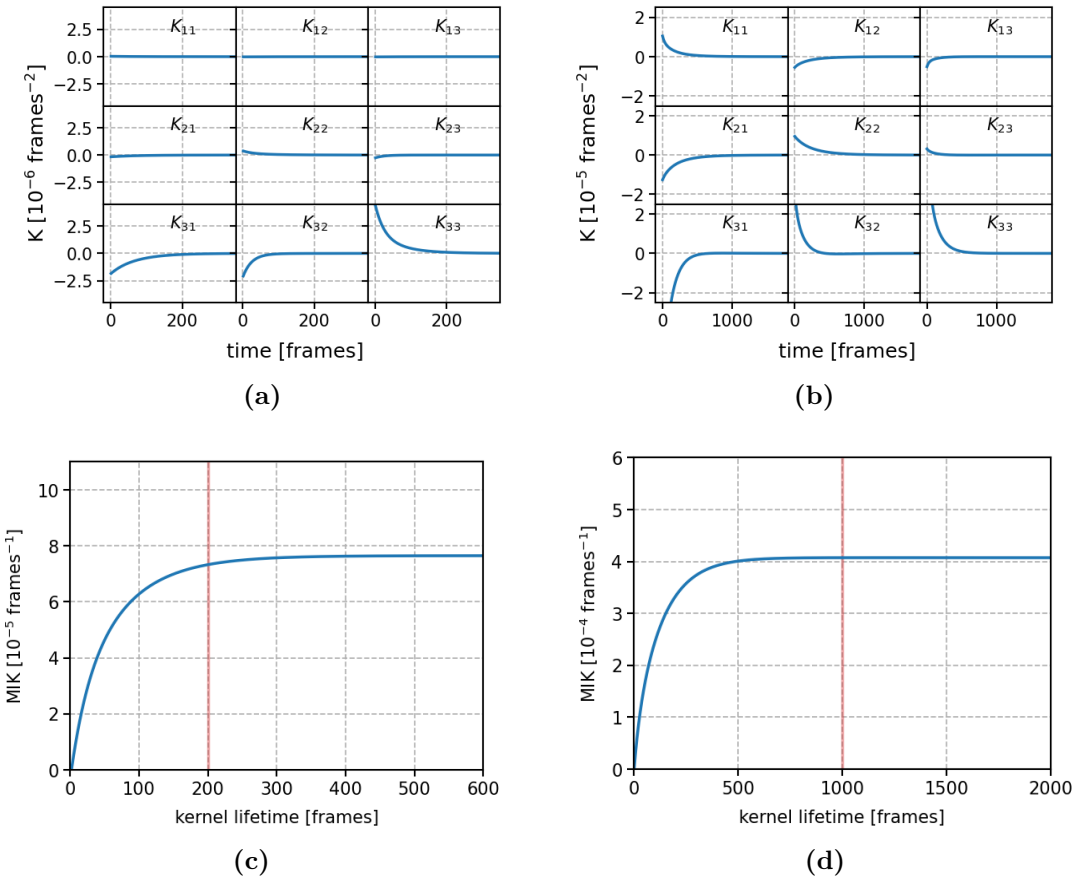


Figure 4.1: Time-dependent memory kernel determined for the toy model. The kernel elements for (a) good and (b) bad lumping. $K_{ij}(0)$ cannot be calculated with equation (2.34). The kernel lifetimes are found as $\tau_K^{\text{good}} = 200\,\text{steps}$ and $\tau_K^{\text{bad}} = 1000\,\text{steps}$ because that is where all elements are decayed. Mean integral of the memory kernel for (c) good and (d) bad lumping. The chosen kernel lifetimes are shown as a red line.

The point where all kernel elements are decayed defines the memory kernel lifetime τ_K. By looking at figure 4.1a, $\tau_K^{\text{good}} = 200\,\text{steps}$ is found for good lumping and $\tau_K^{\text{bad}} = 1000\,\text{steps}$ for bad lumping. To verify the kernel lifetime τ_K, on which the qMSM will be built, the mean integral of the determined memory kernel (MIK) is calculated with equation (2.37) where for this toy model $N = 3$. The kernel lifetime τ_K, is the minimum time when the MIK converges to a stable value. As it is seen in figure 4.1b the earlier determined kernel lifetimes $\tau_K^{\text{good}} = 200\,\text{steps}$ and $\tau_K^{\text{bad}} = 1000\,\text{steps}$ lay respectively slightly before and after the point, where the plateaus start. Still, they appear to be reasonable values.

The finite lifetime of the memory kernel allows rewriting the generalized master equation (2.27) such that the upper limit of the convolution integral is set as $\min[\tau_K, t]$, yielding equation (2.33) from which the derivatives of the qMSM

TPMs can be calculated for $t = n\Delta t$ with $n \in [1, \ldots, t_{\text{end}}/\Delta t]$. For the toy model, it is again sufficient to write the convolution integral as a sum over discrete values. Starting with $\mathbf{T}_{\text{qMSM}}(\Delta t) = \mathbf{T}_{\text{macro}}(\Delta t)$ and $\dot{\mathbf{T}}_{\text{macro}}(0)$, the iterative integration of $\dot{\mathbf{T}}_{\text{qMSM}}(t)$ provides $\mathbf{T}_{\text{qMSM}}(t)$ for every time $t = n\Delta t$ ($\Delta t = 1\,\text{frame}$).

MSM

For the MSM the lag time is chosen to equal the kernel lifetime, so $\tau_L^{\text{good}} = \tau_K^{\text{good}} = 200\,\text{frames}$ for good lumping and $\tau_L^{\text{bad}} = \tau_K^{\text{bad}} = 1000\,\text{frames}$ for bad lumping. The lag time and the memory kernel lifetime are directly linked to the length of MD simulation needed to build the models, respectively. In this way, one can directly compare, if qMSM provides better results for the same conditions. The lag time defines the temporal resolution of the MSM ($\Delta t = \tau_L$) and the Chapman-Kolmogorov equation (2.6) is used to obtain the MSM dynamics starting with $\mathbf{T}_{\text{macro}}(\tau_L)$.

4.4 Validating the Models

Implied Timescales

The quality of the state lumping and more in general of the models can be judged by looking at the implied timescales (equation (4.2)), since the ITS curves should ideally converge to the values of the microstate timescales. For a perfect agreement of microstate and macrostate timescales, the relaxation time does not change through the state reduction, indicating a good timescale separation of intra- and interstate dynamics for the macrostates. For the microstate timescales, the eigenvalues of the microstate TPMs are calculated, sorted from slowest to fastest process, and the ITS are again determined with equation (2.7). This six-state model is constructed in the way that the microstate processes are Markovian and therefore have constant implied timescales, which is not the general case for real biological systems.

The fact that the ITS curves for the bad lumping differ a lot from the microstate timescales indicates emerged kinetic barriers within states due to this kind of lumping. For good lumping, the slow macrostate processes almost perfectly reproduce the microstate timescales and at $\tau_L^{\text{good}} = 200\,\text{frames}$ the relaxation time is already converged, seen in figure 4.2a. In addition, the ITS-plot (figure 4.2b) shows that for bad lumping the timescales of the two slowest processes do not converge within the plotting range. Hence, for $\tau_L^{\text{bad}} = 1000\,\text{steps}$ the relaxation time is still dependent on the lag time. So the MSM does not correctly reproduce these processes.

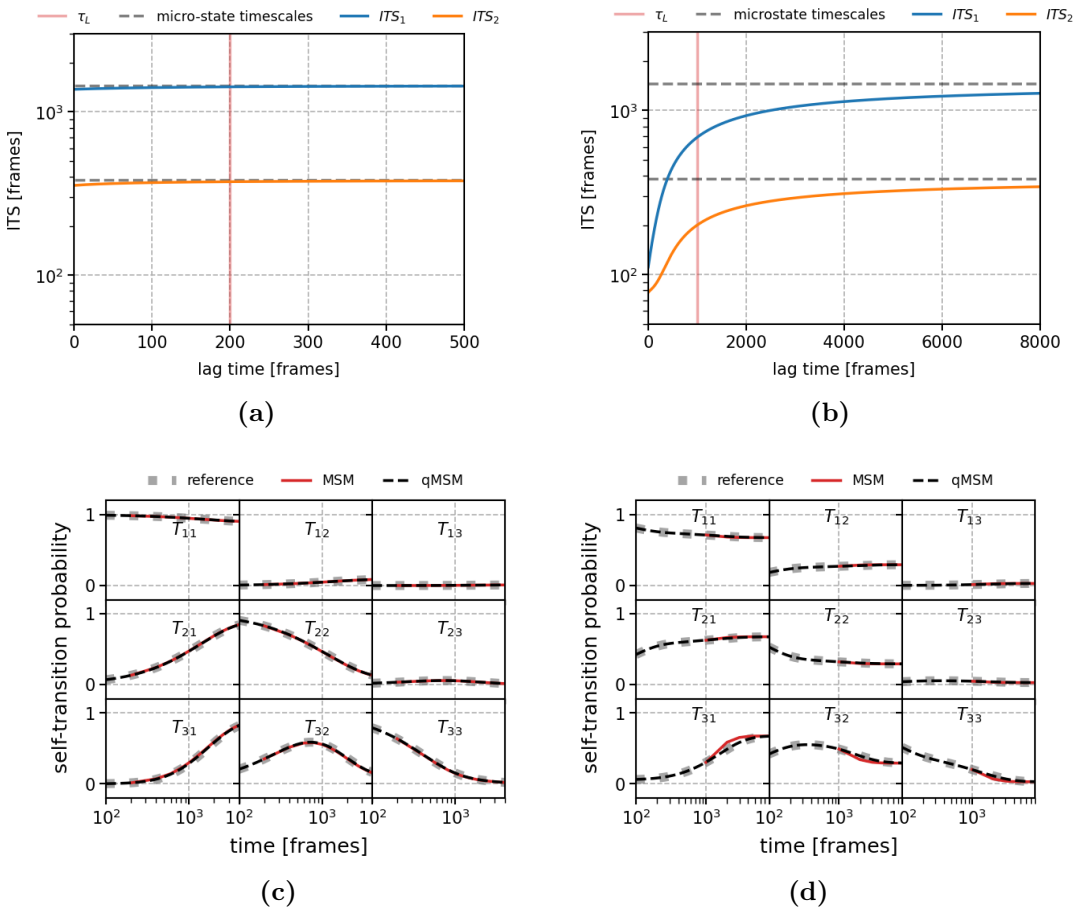


Figure 4.2: Validation of MSM and qMSM built for the toy model. Implied timescales of the two slowest processes for (a) good lumping and (b) bad lumping. The microstate timescales are constant because the microstate processes are perfectly Markovian. The chosen lag time $\tau_L^{\text{good}} = 200$ frames lays already in a constant region of the macrostate relaxation time, while for $\tau_L^{\text{bad}} = 1000$ frames the relaxation time is still dependent on the lag time. Chapman-Kolmogorov test for (c) good lumping and (d) bad lumping. Only the MSM deviates, when using bad lumping, for transitions out of state 3.

Chapman-Kolmogorov Test

For the Chapman-Kolmogorov test, the transition probabilities of the MSM and the qMSM are plotted together with the ones directly obtained from the reference dynamics. Usually one would only look at the diagonal elements because of statistic reasons (more self-transitions than interstate transitions), but here also the off-diagonal elements are presented and show good agreement.

In case of good lumping, the test reveals a perfect match of both the MSM and the qMSM with the reference, seen in figure 4.2c. The well-defined timescale

separation of the microstate dynamics is maintained after lumping, therefore with $\tau_L = 200$ steps the system's dynamics can be reproduced with the MSM. For bad lumping, the qMSM still fits the reference quite perfectly, while the MSM deviates at certain points (figure 4.2d). This is due to poorly separated timescales resulting from random lumping, which appears to mostly affect transitions out of state 3. Also, the increased lag time leads to a worse temporal resolution of the MSM dynamics. The lag time was chosen to equal the kernel lifetime for both kinds of lumping, so one can directly compare the results of MSM and qMSM as these parameters are linked to the length of MD simulation needed to construct the respective models. The fact that qMSM still performs perfectly for the bad lumping indicates that it can be applied for cases where an MSM is not descriptive.

Population Probabilities

For a better insight into the actual dynamics of the toy model system, the probability to find the system in a certain state depending on time and on the initial condition can be calculated with equation (2.28). The resulting 'kinetic plots' give an insight on how the system evolves when starting in a certain state, i.e., they provide an outlook on the paths the system takes from one state to another. The state population probabilities are therefore determined with the reference TPMs $\mathbf{T}_{\text{macro}}(n\Delta t)$, as well as with the MSM TPMs and the qMSM TPMs.

The plots in figure 4.3 reveal that for good lumping, when starting in state 1 the system most likely stays in there. If the initial state is state 2, it probably changes into state 1 after 500 to 1500 time frames and state 3 is most likely not visited. Because state 3 is the least populated one, the system changes quite quickly into the neighbouring state 2 and finally back to 1. For bad lumping, the equilibrium populations are reached much faster, although showing similar evolutions (figure 4.4).

As the kinetic plots are directly linked to the Chapman-Kolmogorov test, they highlight the same issues when comparing MSM and qMSM to the reference. For good lumping, the MSM reproduces well this population evolution, but for bad lumping it shows deviations from the reference, which are mainly found in the 'kinetic plot' starting in state 3, i.e., the last one reaching equilibrium. Here, the time restriction of τ_L leads to shifts in the population probabilities around a time of 1000 frames to 2000 frames. In contrast, the qMSM reproduces the reference data perfectly.

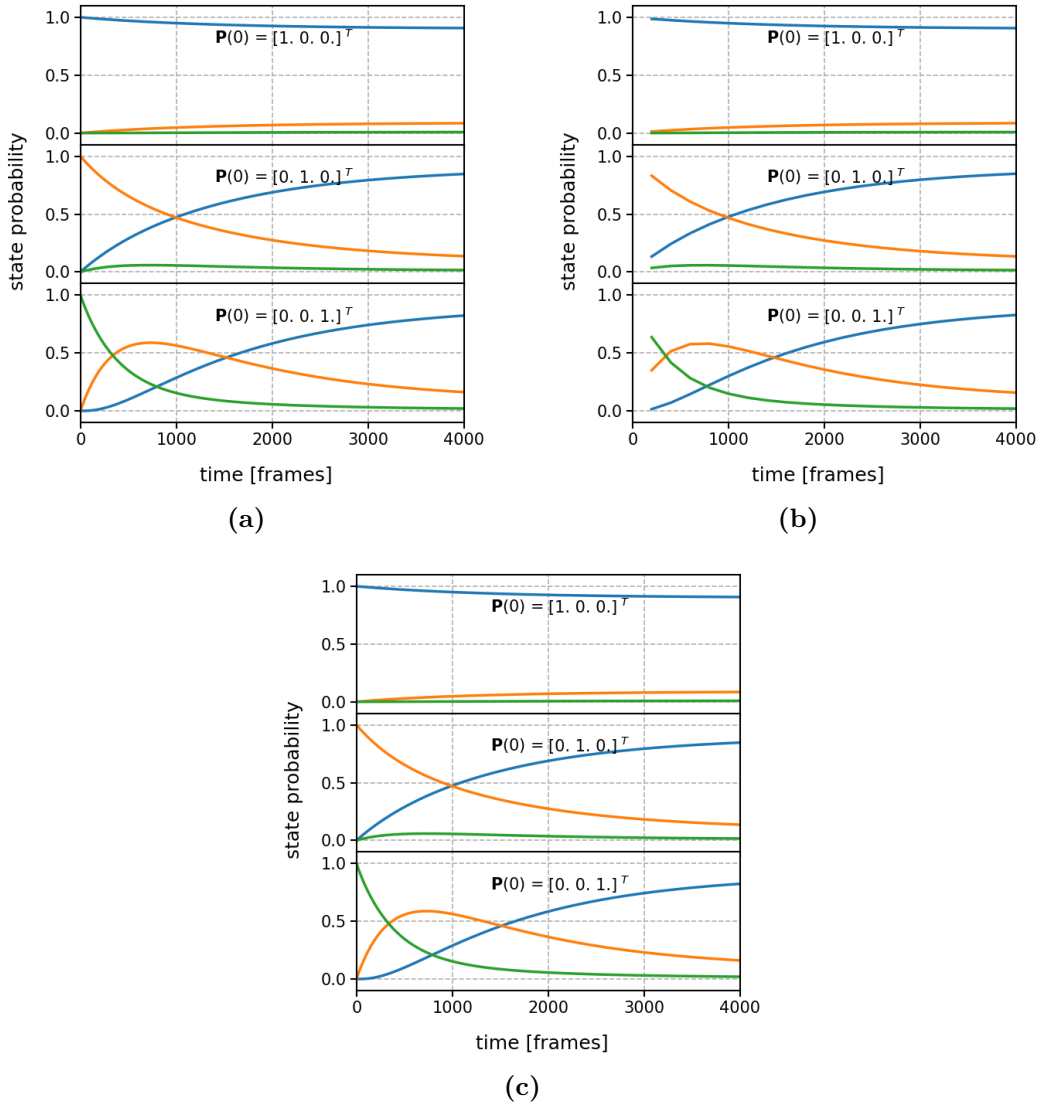


Figure 4.3: Population probabilities when using good lumping. Evolution of the states population probabilities determined with (a) the reference macrostate TPMs, (b) the MSM dynamics and (c) the qMSM TPMs. The following colour code is used: state 1 (blue), 2 (orange), 3 (green).

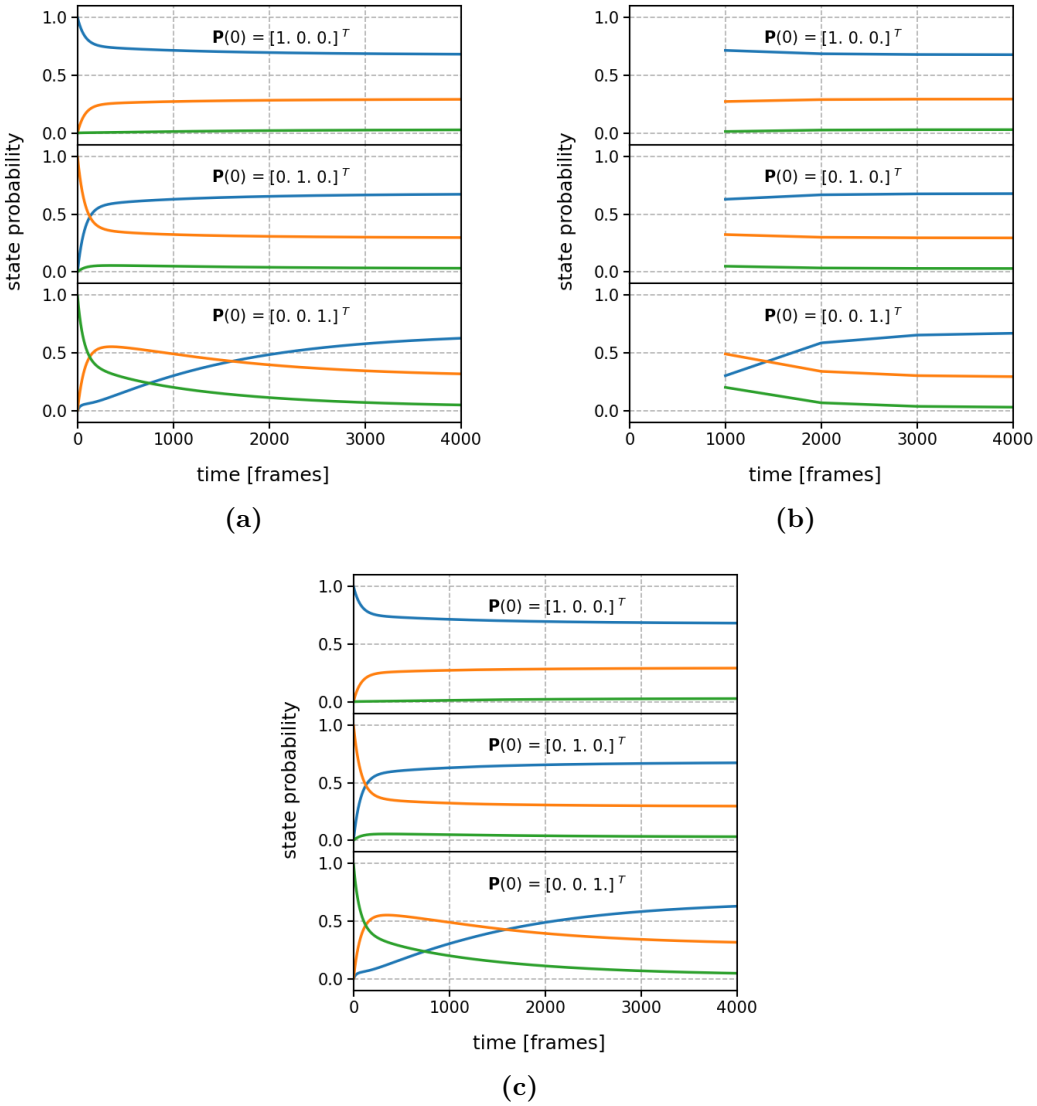


Figure 4.4: Population probabilities when using bad lumping. Evolution of the states population probabilities determined with (a) the reference macrostate TPMs, (b) the MSM dynamics and (c) the qMSM TPMs. The following colour code is used: state 1 (blue), 2 (orange), 3 (green).

Chapter 5

The Folding Process of HP35

5.1 Workflow

For HP35, the qMSM is built on the macrostates trajectory by Nagel et al. [17]. After performing PCA on the input features (minimal contact distances), the trajectory was clustered into a few hundreds of microstates with RDC. The microstates were subsequently lumped into 12 macrostates with dynamical clustering. Nagel et al. built an MSM on HP35 that well describes the dynamics of the system, and makes it a good benchmark to test the qMSM as a new approach. The corresponding kinetic network of the MSM is shown in figure 5.1, including a native basin (state 1-5), an unfolded basin (state 9-12) and a few intermediate states.

As a first step, the macrostate TPMs are built from the state trajectory for lag times $\tau \in [1, ..., N]$ frames, using the python package `msmhelper` [50]. Here $1\,\text{frames} = 0.2\,\text{ns}$ and N is the maximal considered lag time, in units of frames. Choosing $N = 3500$, $t_{\text{end}} = 0.7\,\mu\text{s}$ is the end time used to build the models.

With the resulting TPMs, the qMSM approach can be applied to HP35 by calculating the time-dependent memory kernel with equation (2.34). The MIK is then calculated (equation (2.37)) to obtain the kernel lifetime. Feeding the first TPM and the kernel lifetime back into the GME (2.33) finally provides the qMSM. By calculating the ITS of the qMSM dynamics and performing the Chapman-Kolmogorov test, MSMs calculated for different lag times $\tau_L \in [10\,\text{ns}, 20\,\text{ns}, 30\,\text{ns}, 50\,\text{ns}, 100\,\text{ns}]$ and the qMSM are compared to the transition probabilities directly obtained from the MD data.

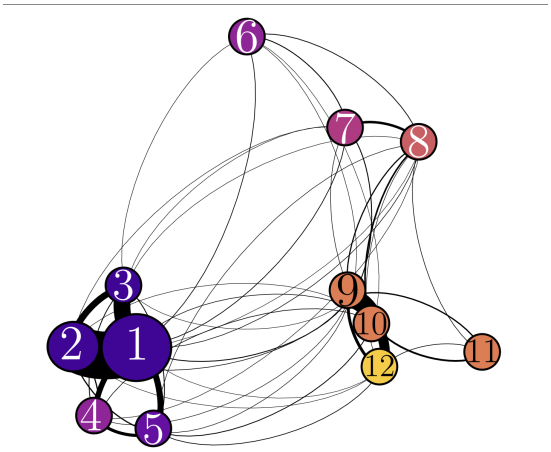


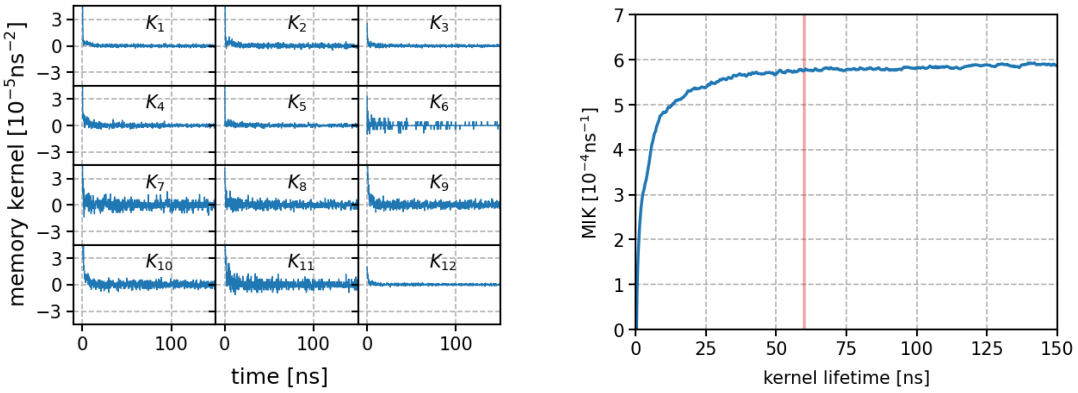
Figure 5.1: Kinetic network of the MSM from [17]. The colour code reflects the fraction of native contacts of the state, and the edges represent the flux between different states.

5.2 Comparing MSM and qMSM

To build the qMSM, the time-dependent memory kernel is determined as a first step with equation (2.34). Plotting the evolution of all elements, seen in figure 8.1 in the appendix, reveals a block-like pattern: dynamically close states (seen in the kinetic network 5.1) show more often memory, which does not decay instantaneously because transition within a basin are usually faster than inter-basin transitions and therefore are less well-separated from the intrastate timescales. The kernel elements show more fluctuation for less populated states, indicating that with lower sampling, the memory kernel values are influenced by statistical noise resulting from numerical fluctuations in the simulation data. In figure 5.2a, only the diagonal elements of the memory kernel are shown.

In comparison to the toy model, the kernel lifetime is much harder to determine directly by looking at the kernel elements, because they show no smooth curve. The kernel lifetime is therefore chosen by calculating the MIK, shown in figure 5.2b. At $\tau_K = 60\,\text{ns}$ the MIK reaches a constant value, so this time is chosen as the kernel lifetime to build the qMSM. Here, it should be noted that for HP35 the input data still allows for a clearly defined MIK-plateau, which is not the case for every system [16]. If one finds is more fluctuation in the memory kernel for slow timescales, it might be reasonable to test different kernel lifetimes to obtain an accurate qMSM.

The qMSM and the MSMs are now validated and compared by looking at the ITS and the Chapman-Kolmogorov test. The ITS-plot in figure 5.3a shows quite slowly converging timescales for the three slowest processes. Nevertheless, the



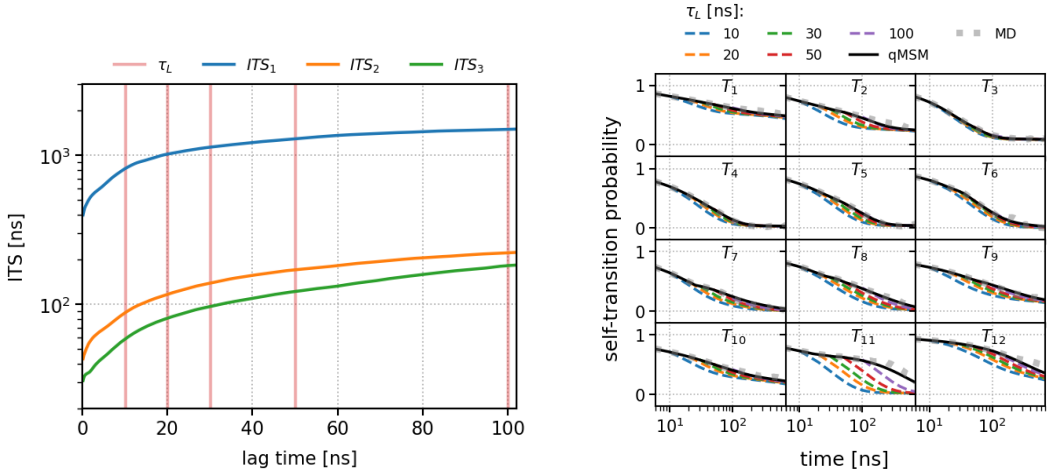
(a) **(b)**

Figure 5.2: (a) Only diagonal elements of the time-dependent memory kernel for HP35 ($K_i = K_{ii}$ for $i \in [1, \ldots, 12]$). $K_{ij}(0)$ cannot be calculated with equation (2.34). (b) Mean integral of the memory kernel. The memory kernel decays at $\tau_K = 60\,\text{ns}$ (vertical red line).

relaxation time becomes constant around 60 ns. Therefore, only the slowest lag time $\tau_L = 100\,\text{ns}$ provides good results for the MSM.

As the last step, the Chapman-Kolmogorov test reveals if the qMSM reproduces the states' dynamics, seen in figure 5.3b. Expectedly, for both models, state 11 is represented the worst because it is a very low populated (0.9%) state [17]. For the MSM, the self-transition probabilities decay significantly faster than for the MD reference, especially for the unfolded states 11 and 12. This improves for high lag times, as for most of the states the MSM with a lag time of $\tau_L = 100\,\text{ns}$ fits the MD values quite well, promising a better fit for even higher lag times. Since in the molecular dynamics under study there are relevant processes faster than this, taking such a slow lag time is not desirable. In contrast, the qMSM build for a kernel lifetime of $\tau_K = 60\,\text{ns}$ already matches the MD data well for every state and shows only minor deviations for states 11 and 12 starting around 100 ns.

In the work of Nagel et al. the most probable paths from the complete unfolded state 12 to the native state 1 were found to pass first state 10 and then state 5 or 9 before transitioning into state 1, or to go directly from state 12 to 5 or 9 into state 1. Nagel et al. also determined the folding time, i.e., the time it takes from state 12 to reach state 1, from the MSM as $t_{\text{fold}} = 1.7\,\mu\text{s}$ [17]. Since the models here are built for an end time of $t_{\text{end}} = 0.7\,\mu\text{s}$, the 'kinetic plots', shown in figures 8.3 and 8.2 in the appendix, do not provide a complete outlook, on whether the qMSM predicts the same paths because within this time, equilibrium is not reached yet. Still, at least for this short-time range, the qMSM's reproduction



(a) (b)

Figure 5.3: Validation of MSM and qMSM build for HP35. (a) Implied timescales of the three slowest processes. Except for $\tau_K = 100\,\text{ns}$, all chosen lag times lay in a region where the macrostate relaxation time is not constant yet. (b) Chapman-Kolmogorov test ($T_i = T_{ii}$ for $i \in [1, \ldots, 12]$). For states 11 and 12 both models show the worst agreement. Still, the qMSM matches the MD reference better than the MSMs.

of the reference kinetic evolution is significantly better than the one calculated from the MSM (with $\tau_L = 100\,\text{ns}$).

Since the lag time and the kernel decay time are directly linked to the length of MD simulation needed to build the respective model which includes all relevant processes, the test indicates, that for qMSM a smaller amount of data could be sufficient. To prove this, the length of the used trajectory is reduced first by 33 % and then by 50 % of its frames.

5.3 Shortened State Trajectory

Using shorter state trajectories (66 % and 50 % of the data), calculations and plots are performed the same way as with the full trajectory for both MSM and qMSM. The ITS- and the MIK-curves, seen in figures 5.4a, 5.4b and 8.4 in the appendix, do not show a significant deviation from the ones obtained with the full trajectory. Only the MIK-curves are a bit less smooth and also show slightly higher plateaus ($\approx 8 \times 10^{-4}\,\text{ns}^{-1}$ instead of $\approx 6 \times 10^{-4}\,\text{ns}^{-1}$ for the full trajectory) because of increased numerical fluctuation coming from sparser sampling.

The Chapman-Kolmogorov tests (figures 5.4c, d) reveal again good agreement for the qMSM. Only the self-transition probabilities of the low populated state

11 and the completely unfolded state 12, which already did not show a perfect fit for the full trajectory, decay slightly too fast. With a shorter trajectory, also the MSMs' dynamics deteriorate, but not significantly more than the qMSM's.

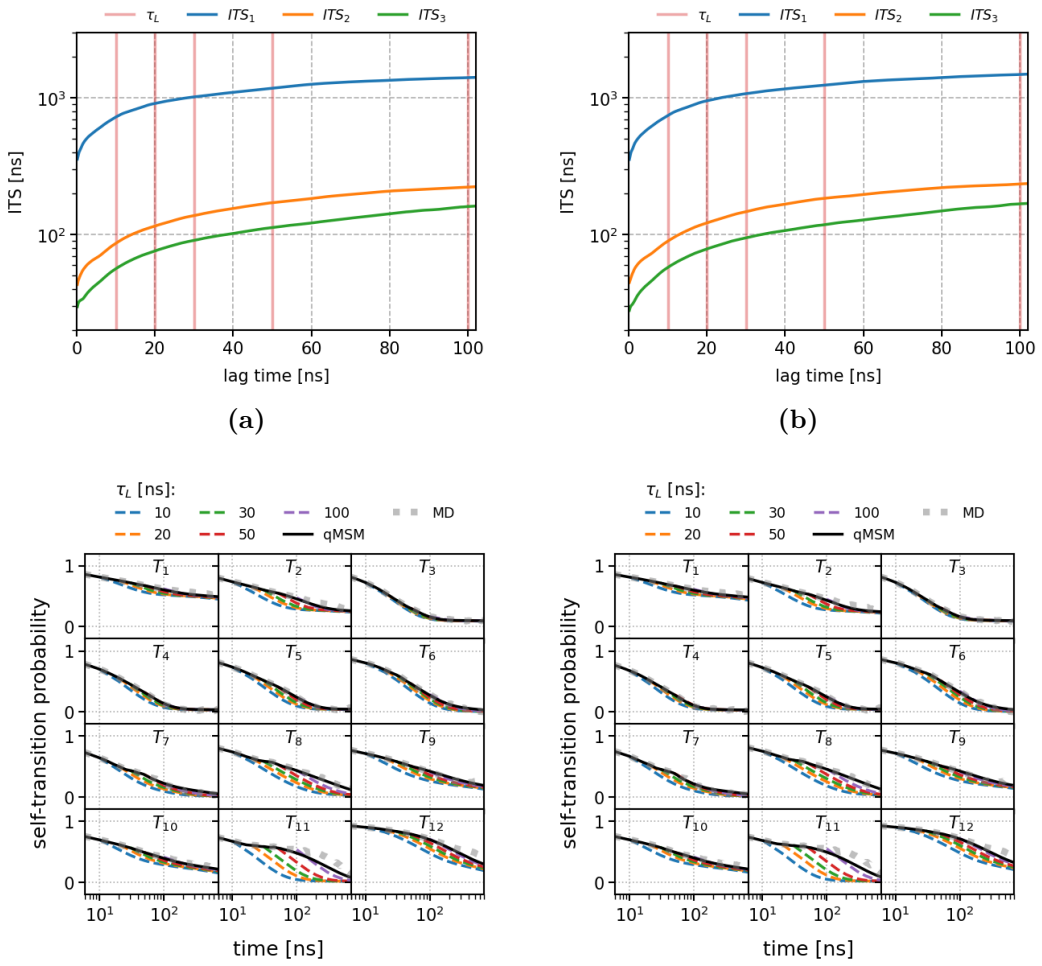


Figure 5.4: Validation of MSM and qMSM for shortened trajectories of HP35. Implied timescales of the three slowest processes for (a) 66 % of the trajectory and (b) 50 % of the trajectory. Chapman-Kolmogorov test for (c) 66 % of the trajectory and (d) 50 % of the trajectory. Both models deteriorate slightly, but still, qMSM well-describes the dynamics.

Chapter 6

Allosteric Communication in T4 Lysozyme

6.1 Workflow

The macrostate trajectory used to build a qMSM for T4 lysozyme was obtained in a study of Post et al. [11]. They chose a combination of contact distances and dihedral angles as input features with the Leiden algorithm [48] and RDC provided five metastable states: namely 1_R, 1_E, 2, 4_R and 4_E, where state 1 is the open conformation, state 4 the closed one and state 2 the transition state. By building an MSM with $\tau_L = 0.7\,\text{ns}$, Post et al. found that the main path from the open to the close conformation was $1_R \to 1_E \to 2 \to 4_E \to 4_R$. The MSM is visualized in the kinetic network, shown in figure 6.1. However, the problem with applying MSM to the open-close transition of T4 lysozyme is that it is a cooperative process, where the various motions occur almost simultaneously, which is in contrast to the timescale separation assumed for an MSM [11]. As one consequence, the transition times calculated with MSM by Post et al. exceed the ones directly obtained with MD simulation data by a factor of 2. T4 lysozyme can therefore be used here as an example system on which MSMs show their limits, and it is of interest to investigate how a qMSM instead performs for such a system.

As for HP35, the python package `msmhelper` [50] is used to obtain the macrostate TPMs for lag times $\tau \in [1, \ldots, N]$ frames, where $1\,\text{frame} = 10\,\text{ps}$ and N is chosen as 100000, i.e., $t_{\text{end}} = 1\,\mu\text{s}$. To obtain a qMSM for T4 lysozyme, the same steps are applied as for HP35. In the Chapman-Kolmogorov test, the resulting model is then compared to the MSM built for the lag times $\tau_L \in [0.7\,\text{ns}, 2\,\text{ns}]$.

The state numbering $(1-5)$ is connected to the five states presented above according to the following:

$$
\begin{aligned}
1 &\longrightarrow 1_E \\
2 &\longrightarrow 2 \\
3 &\longrightarrow 4_E \\
4 &\longrightarrow 1_R \\
5 &\longrightarrow 4_R
\end{aligned}
$$

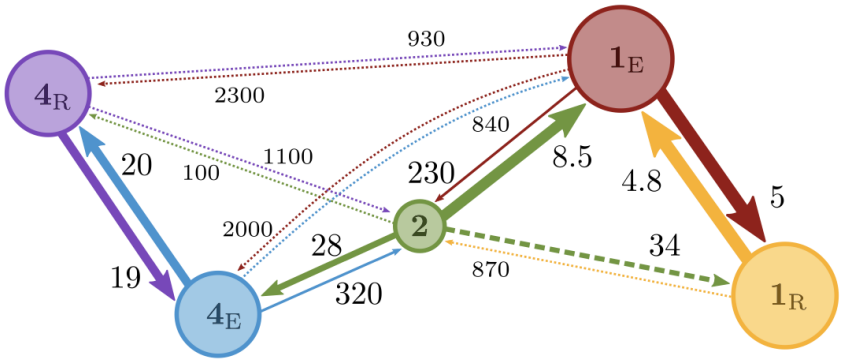
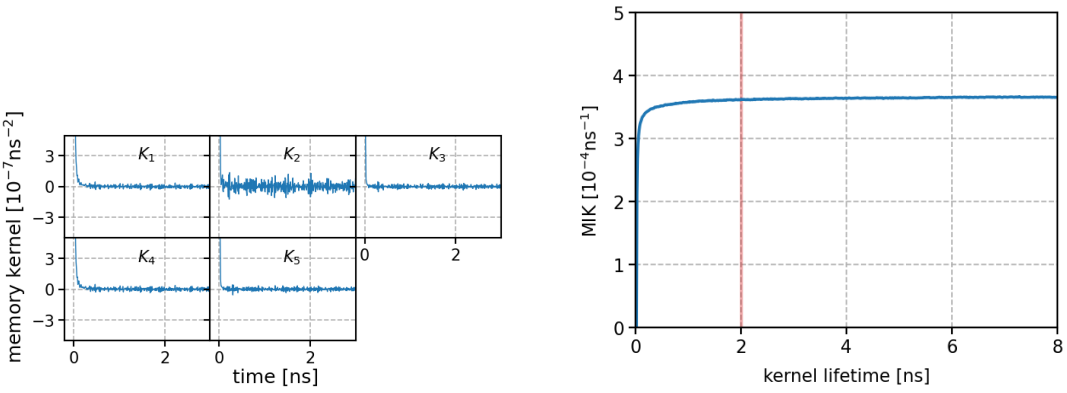


Figure 6.1: Kinetic network of the resulting Markov state model of T4 lysozyme, indicating the main transition times (in units of ns) taken from [11].

6.2 Comparing MSM and qMSM

In the first step, the memory kernel is calculated with equation (2.34). Like for HP35, the kernel lifetime is hard to determine just by looking at the time evolution of the kernel elements, (figure 8.8 in the appendix), as they seem to decay very fast and are quite noisy. Nevertheless, one can see that the memory kernel decays infinitely fast for transitions between the open states $1_R, 1_E$ and the closed states $4_R, 4_E$, indicating well-separated timescales for these main states. The kernel elements for the interstate transitions $1_R \leftrightarrow 1_E$ and $4_R \leftrightarrow 4_E$ seem to decay slower. Since these substates are dynamically very close states, it is not surprising that they show more memory, because for these states the timescales of the interstate transitions are less well-separated from the intrastate transition timescales. Finally, the transitions assigned to the low populated state 2 show the most fluctuations in the memory kernel due to statistical noise coming with poor sampling. In figure 6.2a, only the diagonal elements of the memory kernel are shown.



(a) (b)

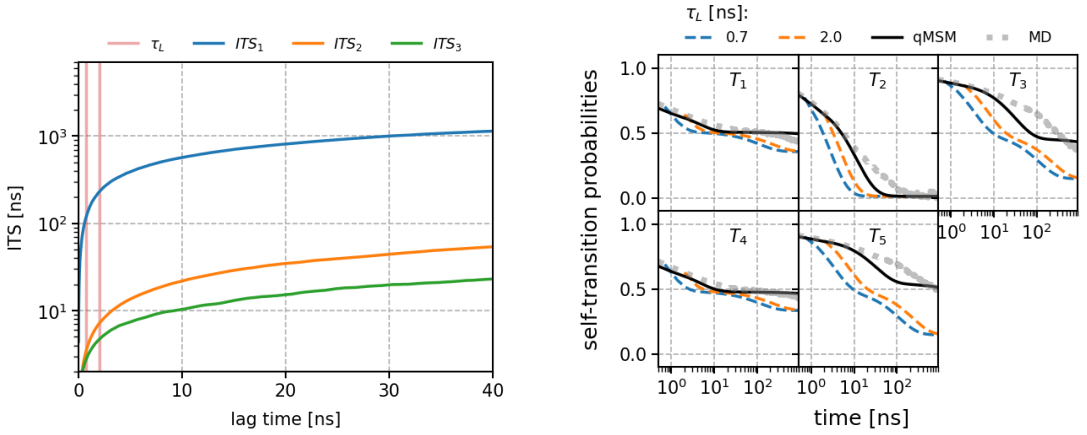
Figure 6.2: (a) Only diagonal elements of the time-dependent memory kernel for T4 lysozyme ($K_i = K_{ii}$ for $i \in [1, \ldots, 5]$). $K_{ij}(0)$ cannot be calculated with equation (2.34). (b) Mean integral of the memory kernel used to determine the kernel lifetime $\tau_K = 2\,\text{ns}$ (vertical red line).

The MIK, seen in figure 6.2b, confirms the short lifetime of the memory kernel: for $\tau_K = 2\,\text{ns}$ the MIK-curve is clearly converged. This kernel lifetime is now used to determine the qMSM TPMs by iteratively integrating the GME (2.33).

For the validation of the MSM, the ITS are calculated. The relaxation time does not converge within the plotting range of $\tau = 40\,\text{ns}$, seen in figure 6.3a. Therefore, for both chosen values of τ_L, the relaxation time is still dependent on the lag time, i.e., the MSMs is not accurately describing the dynamics of the process. The Chapman-Kolmogorov test in figure 6.3b shows that the evolution of the most populated states 1_R and 1_E can be well described by the MSMs and the qMSM, as the self-transition probabilities of the MSMs decay just slightly faster than the ones of the MD reference, while the ones of the qMSM show just minor deviations. For the other states the dynamics of both models decay too fast, but the qMSM is significantly closer to the MD reference values than the MSMs. Different kernel lifetimes could be tested here to check if they improve the qMSM.

It is not clear why the qMSM self-transition probabilities seem to flatten for all states starting around 100 ps. Especially because it looks like they have not reached equilibrium yet within the chosen end time of $t_{\text{end}} = 1\,\mu\text{s}$, it would be interesting to see how the qMSM dynamics evolve for timescales beyond this point. Nevertheless, one can say, that the qMSM already shows an improvement compared to the MSMs, at least for short-time predictions.

For cooperative processes like the open-close transition of T4 lysozyme, the population probabilities do not provide an insight into the transition pathways because



(a) (b)

Figure 6.3: Validation of MSM and qMSM for T4 lysozyme. (a) Implied timescales of the three slowest processes. The chosen lag times lay in a region where the macrostate relaxation time is not constant yet. (b) Chapman-Kolmogorov test ($K_i = K_{ii}$ for $i \in [1, \ldots, 5]$). Both models fail the test, but qMSM agrees to the MD reference better, especially for the open states 1_R and 1_E.

they just show the statistical evolution of the populations and can not capture rare events of transitioning in and out of a low populated state. Here, the 'kinetic plots' seen in figure 8.9 in the appendix just reflect that the relaxed and the excited conformations are dynamically close, as they most likely first transition into the respective 'partner' state. In addition, the population probabilities confirm that the qMSM reproduces better the reference dynamics.

Chapter 7

Conclusions

7.1 Summary

Markov state models are widely used to describe biological processes but they involve some limitations. The main challenge is to find a workflow such as dimensionality reduction and clustering in order to obtain a sufficient timescale separation between intra- and interstate dynamics in the resulting MSM. In many cases, the dynamics turn out non-Markovian when a short enough lag time (i.e., which is not longer than any relevant processes) is chosen. The aim of this study was to consider the work of Huang at al. who proposed a new method, the quasi Markov state model (qMSM), which deals with non-Markovian dynamics by considering the non-instantaneous response of the system [14], and applying it to different systems. For that model a memory kernel is determined by inverting the underlaying equation of motion, the generalized master equation (GME) (2.27).

As a demonstration of how to apply a qMSM, the six-state toy model of Huang et al. was considered. For this toy model, the impact of good lumping compared with a bad lumping was then analysed. When using a grouping of the microstates into three macrostates based on the strongest transition probabilities (good lumping), the intra- and interstate timescales were well-separated. The implied timescales showed an almost perfect match of the slowest micro- and macrostate processes. Thus, for a lag time of $\tau_L = 200\,\text{frames}$ the MSM already provided a perfect description of the toy model's dynamics. For the obtained qMSM dynamics, the Chapman-Kolmogorov test showed a perfect agreement as well.

On the other hand, a bad lumping of the six states resulted in a breakdown of the timescale separation. For the chosen lag time $\tau_L = 1000\,\text{frames}$, the implied timescales were still dependent on the lag time, which indicated, that MSM did not reproduce the process perfectly. This was confirmed by the Chapman-Kolmogorov test, which showed deviations from the MD reference. The less well-defined timescale separation due to bad lumping was also expressed in the memory kernel. Here, all elements decayed significantly slower than the ones obtained from good lumping. Still, qMSM provided consistent results, as the Chapman-Kolmogorov test showed again good agreement, suggesting that the qMSM was less affected by bad lumping. By considering the memory kernel, the transition probabilities were adjusted depending on an over- or underestimation of the dynamics when using a Markovian description. Since the toy model is an extremely simplified system, the perfect results are not surprising, and it rather serves here as a step-by-step guide on how to perform a qMSM.

To test the advantages of qMSM for a real system, the method was applied on the well-studied folding trajectory of HP35 by Piana et al. [18]. Analysing a state trajectory obtained by using contact distances as input features, PCA as the dimensionality reduction method and RDC and MPP for clustering, an MSM was already obtained by Nagel et al. [17]. When applying qMSM to that trajectory, the memory kernel lifetime was found as $\tau_K = 60\,\text{ns}$. With the resulting qMSM dynamics, the Chapman-Kolmogorov test showed a perfect agreement for most states. Even the dynamics of the states, for which the MSM clearly failed the test, were well-reproduced by the qMSM. The self-transition probabilities here only decayed slightly faster than the MD reference. The MSM showed better results with increasing lag times, but an even longer lag time would have been needed, to match the MD reference as well as the qMSM did. Since the lag time and the kernel lifetime are directly linked to the length of MD simulation needed to capture the relevant processes for building the respective model, the results indicated that qMSM may still provide a good description with less available data. To validate that, the macrostate trajectory was reduced by 33 % and 50 % of its frames. For most of the states, the model again agreed well with the MD reference. Only for the states that did not show perfect agreement before, the qMSM deteriorated slightly. In summary, applying qMSM to HP35 provided a good description of the dynamics of its folding process for all states and also for a shortened trajectory, while MSM failed for some states both with a shortened and the full trajectory.

Overall, qMSM seemed to provide good results for an already quite optimized analysis of the underlaying simulation data. It was left to check how it performs for a less well-understood process. Therefore, the dynamical evolution of allosteric transitions, as studied in the protein T4 lysozyme, was investigated. The global open-closed motion of the two domains of T4 lysozyme was studied by

Post et al. [11], who obtained a state trajectory based on an 61 μs MD simulation of Ernst et al. [11]. They defined a combination of contact distances and dihedral angles as input features using the community detection technique called Leiden algorithm and applied RDC. The resulting macrostate trajectory of five configurational states was here used to build a qMSM. It appeared, that the memory kernel decayed quite fast for all states, with a lifetime of $\tau_K = 2\,\text{ns}$. The Chapman-Kolmogorov test showed that the qMSM provided significantly better results than the MSM. For the high populated open states both models were quite close to the real dynamics, due to good sampling. For the less populated closed states, they could less well reproduce the MD reference, but the better performance of qMSM compared to MSM was even more evident. This again indicates, that qMSM is less sensitive to low sampling, which allows using less MD simulation data.

In summary, qMSM describes biomolecular dynamics very well for well-known processes like the folding process of HP35, and even in cases where the MD simulation analysis still can be improved, it provides significantly better results than the more commonly used MSM, at least for short-time predictions.

7.2 Outlook

Even though qMSMs seems to perform well for the considered systems, it is still in its development stages. One thing that was not addressed in this study but can be an issue for other systems, is that the memory kernel can become very noisy because it is strongly affected by the numerical fluctuations of the MD simulation data [16]. In that case, the kernel lifetime may be hard to determine by looking at the mean integral of the memory kernel, since there will be no clear plateau. Still, a qMSM can be built by testing different kernel lifetimes. But the resulting dynamics can be numerical instable, which clearly limits the applicability of qMSM. In the meantime, there are some new model approaches which deal with this issue, e.g. the integrative generalized master equation (IGME) [51]. It uses the time integrations of the memory kernel, that are calculated implicitly instead of explicitly by writing the convolution integral of the GME as a Taylor expansion. Neglecting higher order terms promises better numerical stability in the resulting dynamics. So it would be of interest to study more systems using a qMSM to find out its limitations and to compare it with other model approaches considering memory.

With MSMs, one can obtain even more information about the process under study than discussed here. One mayor application of MSMs is using the transition probability matrix $\mathbf{T}(\tau_L)$ for running Markov Chain Monte Carlo (MCMC)

simulation [52]. With MCMC, information like most important pathways and waiting times can be extracted out of this single matrix. Knowing the most frequent pathways between, e.g., a folded and an unfolded state or an open and a closed state, is the first step to find out, how to force or inhibit these processes (e.g. for medical purposes). Furthermore, it is useful to know the time it takes from entering the starting state to reach the ending state, also known as the waiting time. In addition, most important pathways and waiting times are good quantities to validate a state model, as they also can be obtained directly from MD data. Thus, it would be of great interest to see how qMSM performs for these quantities. The challenge here is that qMSM does not provide a single transition matrix but a time dependent set of matrices and it is not yet clear how to run an MCMC-type propagation in this case.

In summary, qMSM opens the door for the study of non-Markovian dynamics, where it is challenging to construct a valid MSM with just a few states. So far, it has been shown that the qMSM successfully describes the dynamics of simple systems, but it is still not known, how to use it for pathway predictions.

Chapter 8

Appendix

HP35

Memory kernel elements

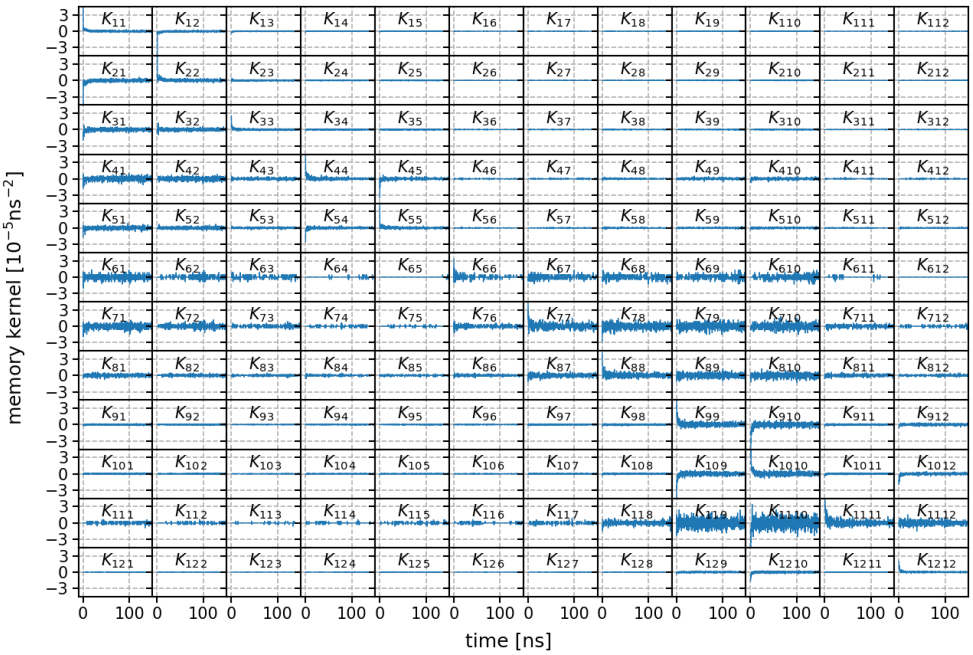
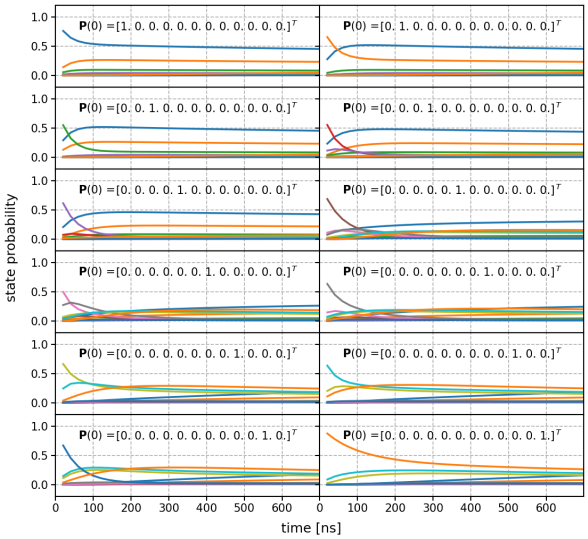
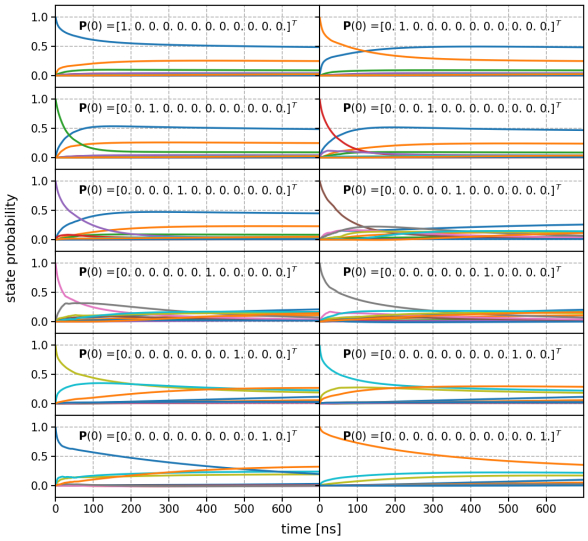


Figure 8.1: Time-dependent memory kernel elements for HP35. It can be seen, that the elements for transitions in particular between dynamical close states do not decay instantaneously.

Population probabilities



(a)



(b)

Figure 8.2: Population probabilities for HP35. (a) Using the MSM TPMs with $\tau_L = 100\,\text{ns}$ and (b) the qMSM TPMs. Each plot shows the population probability evolution for a different starting state. $\mathbf{P}(0)$ is the initial condition and $P_i(0) = 1$ indicates that the starting state is state i. The following colour code is used: state 1 (dark blue), 2 (orange), 3 (green), 4 (red), 5 (purple), 6 (brown), 7 (pink), 8 (gray), 9 (light green), 10 (light blue), 11 (blue), 12 (light orange).

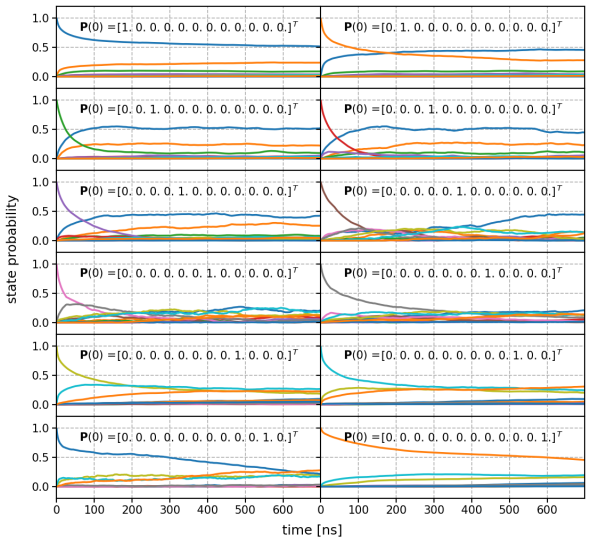


Figure 8.3: Population probabilities for HP35 when using the macrostate TPMs obtained directly from MD simulation data. Each plot shows the population probability evolution for a different starting state. $\mathbf{P}(0)$ is the initial condition and $P_i(0) = 1$ indicates that the starting state is state i. The following colour code is used: state 1 (dark blue), 2 (orange), 3 (green), 4 (red), 5 (purple), 6 (brown), 7 (pink), 8 (gray), 9 (light green), 10 (light blue), 11 (blue), 12 (light orange).

MIK for the reduced state trajectories

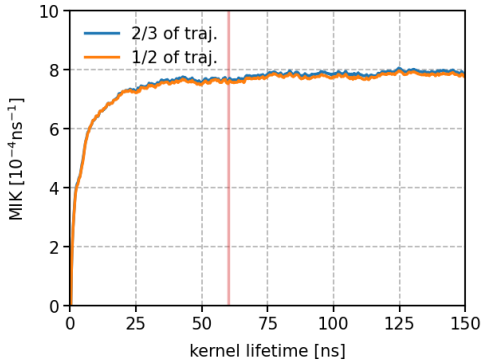


Figure 8.4: Mean integral of the memory kernel when using a reduced trajectory for HP35. $\tau_K = 60\,\text{ns}$ (red line) is the chosen kernel lifetime.

Results when using Hummer-Szabo Projection

Both models (MSM and qMSM) show slightly better results when using the Hummer-Szabo projection (equation (2.13)) to get macrostate TPMs. In the main analysis, these results are not shown to be consistent with the type of projection for all investigated systems.

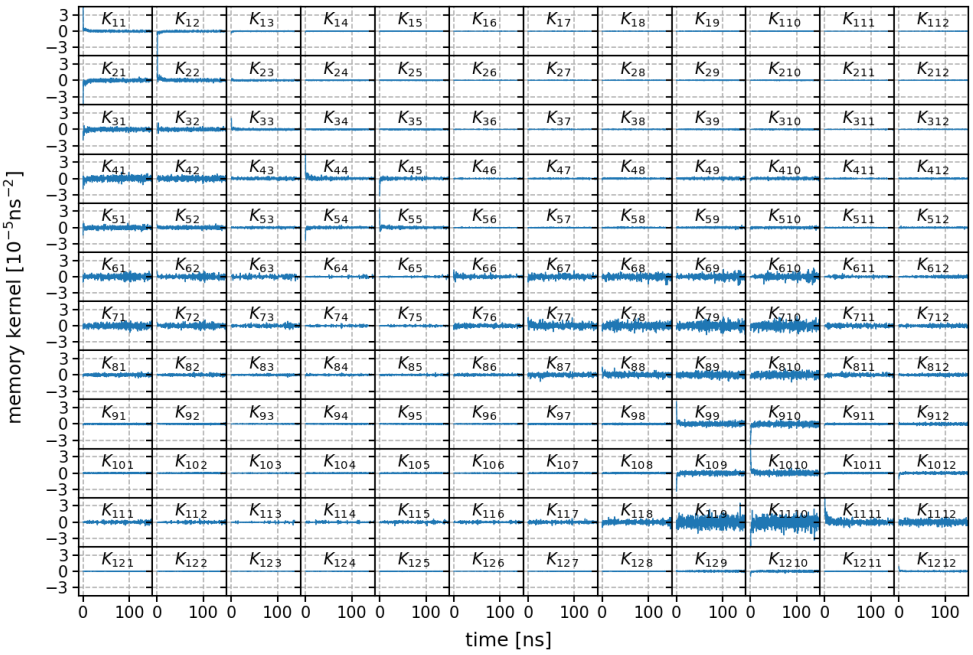


Figure 8.5: Time-dependent memory kernel elements for HP35 when using Hummer-Szabo projection. The elements for transitions between dynamical close states do not decay instantaneously.

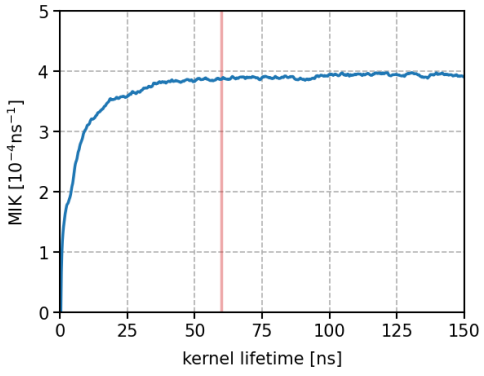
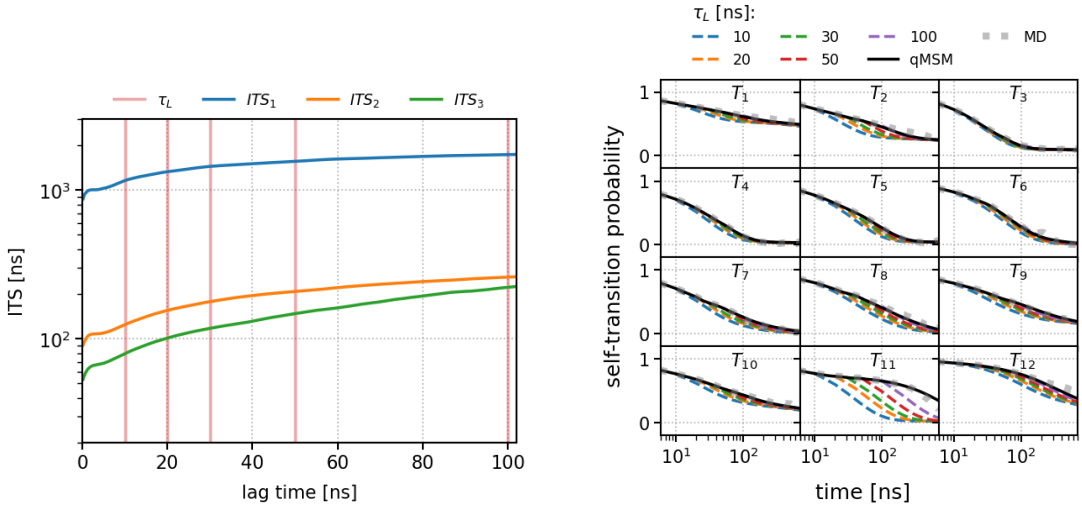


Figure 8.6: Mean integral of the memory kernel when using Hummer-Szabo projection. $\tau_K = 60\,\text{ns}$ (red line) is the chosen kernel lifetime.



(a) **(b)**

Figure 8.7: Validation of the MSMs and the qMSM when using Hummer-Szabo projection for HP35. (a) Implied timescales of the three slowest processes. They converge faster than the ones calculated when assuming local equilibrium (5.3a). For $\tau_L = 50\,\text{ns}, \tau_L = 100\,\text{ns}$ the relaxation times are converged. (b) Chapman-Kolmogorov test. The qMSM shows good agreement for all states, while the MSM is less accurate, especially for state 11.

T4 Lysozyme

Memory kernel elements

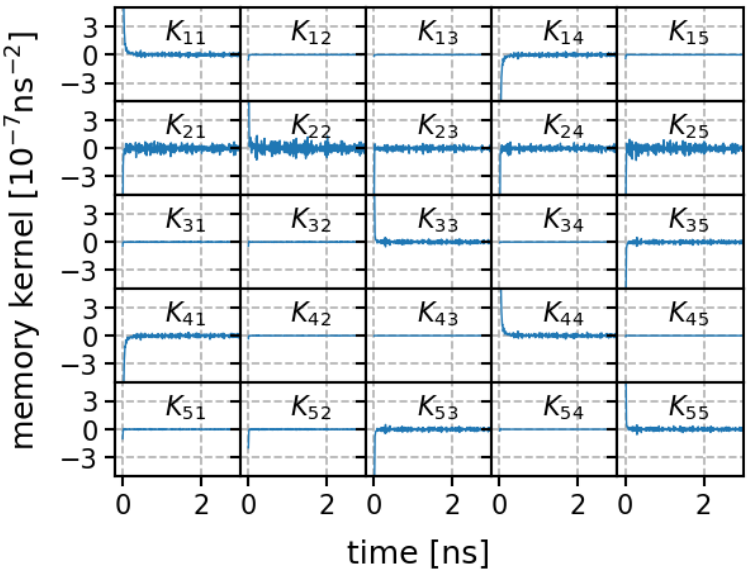


Figure 8.8: Time-dependent memory kernel elements for T4 lysozyme. $K_{ij}(0)$ cannot be calculated with equation (2.34). It can be seen, that the elements for transitions in particular between dynamically close states and all transitions assigned to state 2 do not decay instantaneously.

Population probabilities

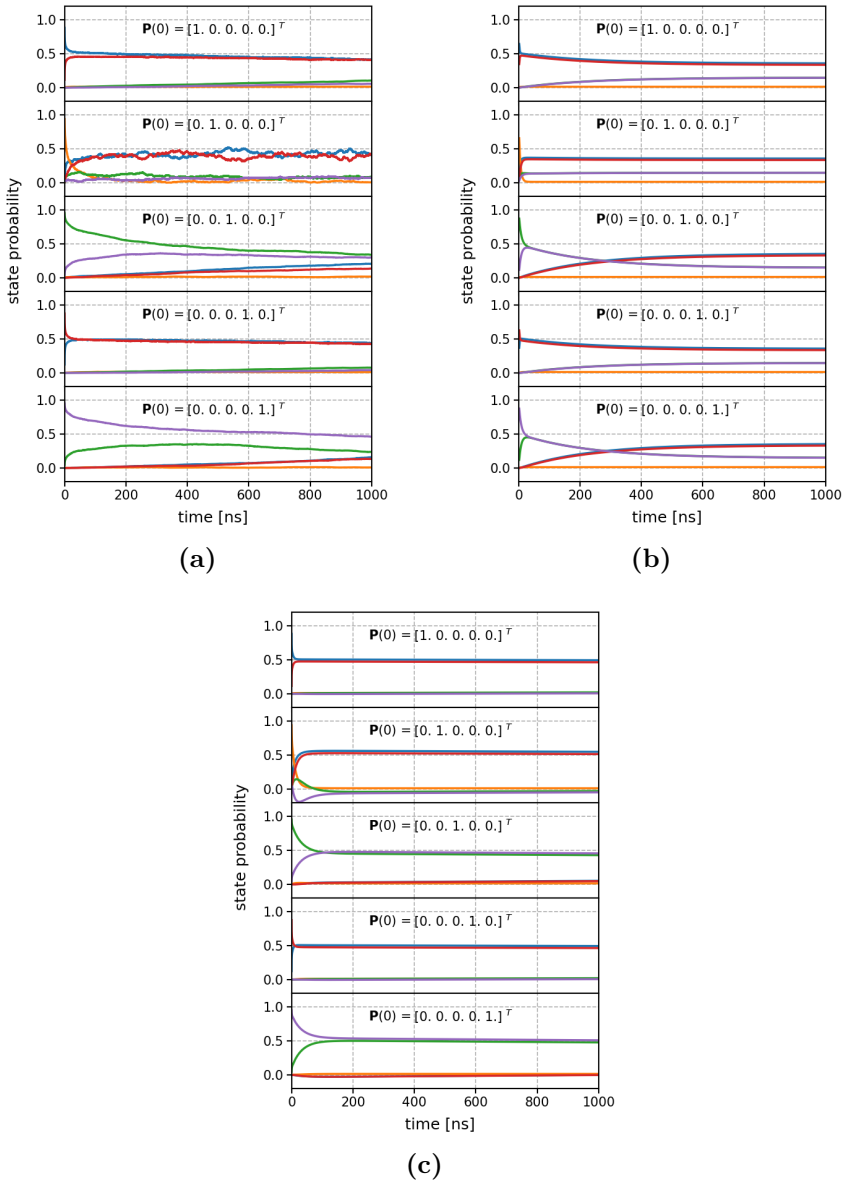


Figure 8.9: Population probabilities for T4 lysozyme. (a) Using the macrostate TPMs obtained directly from MD simulation data, (b) using the MSM TPMs and (c) the qMSM TPMs. Each plot shows the population probability evolution for a different starting state. $\mathbf{P}(0)$ is the initial condition and $P_i(0) = 1$ indicates, that the starting state is state i. The following colour code is used: state 1 (dark blue), 2 (orange), 3 (green), 4 (red), 5 (purple).

Bibliography

[1] I. Bahar, R.L. Jernigan, K.A. Dill, *Protein Actions - Principles & Modeling*, Garland Science, 2017.

[2] K.A. Dill, J.L. MacCallum, *The Protein-Folding Problem, 50 Years On*, Science, **338**, no. 6110 (2012), 1042–1046.

[3] X. Du et al., *Insights into Protein–Ligand Interactions: Mechanisms, Models, and Methods*, International Journal of Molecular Sciences, **17**, no. 2 (2016), 144.

[4] C. Branden, J. Tooze, *Introduction to Protein Structure*, Garland Publishing, Inc., 1998.

[5] M.P. Allen, D.J. Tildesley, *Computer Simulation of Liquids*, New York: Oxford University, 1987.

[6] A.R. Leach, *Molecular Modelling*, London: Pearson Education Limited, 1996.

[7] G.R. Bowman, V.S. Pande, F. Noé, *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*, Advances in Experimental Medicine and Biology (Springer), **797**, 2014.

[8] J.H. Prinz, H. Wu, M. Sarich, B. Keller, M. Senne, M. Held, J.D. Chodera, C. Schütte, F. Noé, *Markov models of molecular kinetics: Generation and validation*, Journal of Chemical Physics, **134**, no. 17 (2011), 17410.

[9] V.S. Pande, K. Beauchamp, G.R. Bowman, *Everything you wanted to know about Markov State Models but were afraid to ask*, Methods, **52**, no. 1 (2010), 99–105.

[10] G.R. Bowman, K.A. Beauchamp, G. Boxer, V.S. Pande, *Progress and challenges in the automated construction of Markov state models for full protein systems*, Journal of Chemical Physics, **131**, no.12 (2009), 124101.

[11] M. Post, B. Lickert, G. Diez, S. Wolf, G. Stock, *Cooperative Protein Allosteric Transition Mediated by a Fluctuating Transmission Network*, Journal of Molecular Biology, **434**, 2022, 167679.

[12] S. Nakajima, *On Quantum Theory of Transport Phenomena: Steady Diffusion*, Progress of Theoretical Physics, **20**, 1958, 948–959

[13] R. Zwanzig, *Ensemble Method in the Theory of Irreversibility*, Journal of Chemical Physics, **33**, 1960, 1338-1341.

[14] S. Cao, A. Montoya-Castillo, W. Wang, T.E. Markland, X. Huang, *On the advantages of exploiting memory in Markov state models for biomolecular dynamics*, Journal of Chemical Physics, **153**, 2020, 014105.

[15] A.K. Yik, Y. Qiu, I.C. Unarta, S. Cao, X. Huang, *A Step-by-step Guide on How to Construct quasi-Markov State Models to Study Functional Conformational Changes of Biological Macromolecules*, University of Wisconsin Madison, 2022.

[16] A.J. Dominic III, S. Cao, A. Montoya-Castillo, X. Huang, *Memory Unlocks the Future of Biomolecular Dynamics: Transformative Tools to Uncover Physical Insights Accurately and Efficiently*, Journal of the American Chemical Society, **145**, 2023, 9916–9927.

[17] D. Nagel, S. Sartore, G. Stock, *Selecting Features for Markov Modeling: A Case Study on HP35*, Journal of Chemical Theory and Computation, 2023.

[18] S. Piana, K. Lindorff-Larsen, D.E. Shaw, *Protein folding kinetics and thermodynamics from atomistic simulation*, Proceedings of the National Academy of Sciences, **109**, 2012, 17845–17850.

[19] M. Ernst, F. Sittel, G. Stock, *Contact- and distance-based principal component analysis of protein dynamics*, Journal of Chemical Physics, **143**, 2015, 244114.

[20] M.J. Lopez, S. Mohiuddin, *Biochemistry, Essential Amino Acids*, StatPearls Publishing, 2022.

[21] D. Eisenberg, *The discovery of the -helix and -sheet, the principal structural features of proteins*, Proceedings of the National Academy of Sciences, **100**, no. 20 (2003), 11207–11210.

[22] B. Alberts, A. Johnson, J. Lewis, *Molecular Biology of the Cell*, Garland Science, 2002.

[23] M. Karplus, J. McCammon, *Molecular dynamics simulations of biomolecules*, Nature structural biology, **9**, 2002, 646–652.

[24] S.A. Adcock, J.A. McCammon, *Molecular Dynamics: Survey of Methods for Simulating the Activity of Proteins*, Chemical reviews, **106**, no. 5 (2006), 1589–615.

[25] A. Papageorgiou, J. Mattsson, *Protein Structure Validation and Analysis with X-Ray Crystallography*, Methods in molecular biology (Clifton, N.J.), **1129**, 2014, 397–421.

[26] L. Monticelli, D. Tieleman, *Force Fields for Classical Molecular Dynamics*, Methods in molecular biology (Clifton, N.J.), **924**, 2013, 197–213.

[27] J.D. Chodera et al., *Long-Time Protein Folding Dynamics from Short-Time Molecular Dynamics Simulations*", Multiscale Modeling Simulation, **5**, no. 4 (2006), 1214–1226.

[28] F. Sittel, G. Stock, *Perspective: Identification of collective variables and metastable states of protein dynamics*, Journal of Chemical Physics, **149**, 2018, 150901.

[29] D. Nagel, S. Sartore, G. Stock, *Towards a Benchmark for Markov State Models: The Folding of HP35*, Albert-Ludwigs University of Freiburg, preprint, 2023.

[30] S. Brandt et al., *Machine Learning of Biomolecular Reaction Coordinates*, The Journal of Physical Chemistry Letters, **9**, no. 9 (2018), 2144–2150.

[31] I.T. Jolliffe, *Principal Component Analysis*, Springer Series in Statistics, Springer, 2002.

[32] A.K. Jain, *Data clustering: 50 years beyond K-means*, Pattern Recognition Letters, **31**, no. 8 (2010), 651–666.

[33] M. Ester et al., *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, **KDD'96**, 1996, 226–231.

[34] B. Keller, X. Daura, W. F. van Gunsteren, *Comparing geometric and kinetic cluster algorithms for molecular simulation data*, The Journal of Chemical Physics, **132**, no. 7 (2010), 074110.

[35] F. Sittel, G. Stock, *Robust Density-Based Clustering To Identify Metastable Conformational States of Proteins*, Journal of Chemical Theory and Computation, **12**, no. 5 (2016), 2426–2435.

[36] A.K. Jain, G. Stock, *Hierarchical folding free energy landscape of HP35 revealed by most probable path clustering*, Journal of Physical Chemistry, **118**, 2014, 7750–7760.

[37] S. Röblitz, M. Weber, *Fuzzy spectral clustering by PCCA+: application to Markov state models and data classification*, Advances in Data Analysis and Classification, **7**, no. 2 (2013), 147–179.

[38] G.R. Bowman, F. Noé, V.S. Pande, *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*, Advances in Experimental Medicine and Biology, Springer, 2014.

[39] G. Hummer, A. Szabo, *Optimal Dimensionality Reduction of Multistate Kinetic and Markov-State Models*, Journal of Chemical Physics, **119**, 2015, 9029–9037.

[40] Kubo, Ryogo, *Stochastic Liouville Equations*, Journal of Mathematical Physics, **4**, 1963, 174–183.

[41] A. Basilevsky, *Applied Matrix Algebra in the Statistical Science*, Dover Publications Inc., 1983.

[42] J. Kubelka et al., *Chemical, physical, and theoretical kinetics of an ultrafast folding protein*, Proceedings of the National Academy of Sciences, **105**, 2008, 18655–18662.

[43] A. Reiner, P. Henklein, T. Kiefhaber, *An unlocking/relocking barrier in conformational fluctuations of villin headpiece subdomain*, Proceedings of the National Academy of Sciences, **107**, 2010, 4955–4960.

[44] C.D. Snow et al., *Absolute comparison of simulated and experimental protein folding dynamics*, Nature, **420**, 2002.

[45] M. Dixon, H. Nicholson, L. Shewchuk, W. Baase, B. Matthews, *Structure of a hinge-bending bacteriophage T4 lysozyme mutant, Ile3→Pro*. Journal of Molecular Biology, **227**, 1992, 917.

[46] X. Zhang, J.A. Wozniak, B.W. Matthews, *Protein flexibility and adaptability seen in 25 crystal forms of T4 Lysozyme*, Journal of Molecular Biology, **250**, 1995, 527.

[47] M. Ernst, S. Wolf, G. Stock, *Identification and validation of reaction coordinates describing protein functional motion: Hierarchical dynamics of T4 Lysozyme*, Journal of Chemical Theory and Computation, **13**, 2017, 5076.

[48] Georg Diez, Daniel Nagel, and Gerhard Stock, *Correlation-Based Feature Selection to Identify Functional Dynamics in Proteins*, Journal of Chemical Theory and Computation, **18**, no. 8 (2022), 5079–5088.

[49] V. Traag, L. Waltman, N. van Eck, *From Louvain to Leiden: guaranteeing well-connected communities*, Scientific Reports, **9**, (2019), 5233.

[50] D. Nagel, G. Stock, *msmhelper: A Python Package for Markov State Modeling of Protein Dynamics*, Journal of Open Source Software, 2023.

[51] S. Cao, Y. Qiu, M. Kalin, X. Huang, *Integrative Generalized Master Equation: A Theory to Study Long-timescale Biomolecular Dynamics via the Integrals of Memory Kernels*, University of Wisconsin Madison, 2023.

[52] D. Nagel, A. Weber, G. Stock, *MSMPathfinder: Identification of Pathways in Markov State Models*, Journal of Chemical Theory and Computation, **16**, 2020, 78747882.

Erklärung gem. der geltenden Prüfungsordnung:

Hiermit versichere ich, dass

1. ich die eingereichte Bachelorarbeit bzw. bei einer Gruppenarbeit meinen Anteil entsprechend gekennzeichneten Anteil der Arbeit selbständig verfasst habe,
2. ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommenen Inhalte als solche kenntlich gemacht habe und
3. die eingereichte Bachelorarbeit weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens war oder ist.

_____ _____

Ort, Datum Unterschrift

Auszug Prüfungsordnung B.Sc.
Vom 31. August 2010 (Amtliche Bekanntmachungen Jg. 41, Nr. 72, S. 401–503)
in der Fassung vom 25. September 2020 (Amtliche Bekanntmachungen Jg. 51, Nr. 67, S. 338–346)

§21 (8) Bei der Einreichung der Bachelorarbeit hat der/die Studierende schriftlich zu versichern, dass

1. er/sie die eingereichte Bachelorarbeit beziehungsweise bei einer Gruppenarbeit seinen/ihren entsprechend gekennzeichneten Anteil der Arbeit selbständig verfasst hat,
2. er/sie keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommenen Inhalte als solche kenntlich gemacht hat und
3. die eingereichte Bachelorarbeit weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens ist oder war.