UNSUPERVISED IDENTIFICATION OF METASTABLE MOLECULAR CONFORMATIONS WITH DEEP LEARNING METHODS

DISSERTATION

ZUR ERLANGUNG DES GRADES
"DOKTOR DER NATURWISSENSCHAFTEN"
AM FACHBEREICH PHYSIK, MATHEMATIK UND INFORMATIK
DER JOHANNES GUTENBERG-UNIVERSITÄT
IN MAINZ

SIMON LEMCKE
GEB. IN WIESBADEN AM 04.01.1990
MAINZ, DEN 29. APRIL 2025

Unsupervised Identification of Metastable Molecular Conformations with Deep Learning Methods

LOCATION:
Mainz

DATE OF DEFENSE:
September 26th 2025

ABSTRACT

The rise of compute power over the last decades, best described by Moore's empirical law, has made it possible to establish simulation as the third pillar of science in between the longstanding pillars of theory and experiment. Investigating systems 'in-silico' has since then become a wide-spread approach to research, enabling numerical insights on scales not accessible to theory and experiment. In recent years, artificial intelligence and machine learning, specifically deep learning, has emerged as one of the key technologies of the information age, fueled by the abundant availability of computation and data.

In this thesis, we show in two case studies that a deep learning approach to dimensionality reduction, called EncoderMap, is able to find better, more descriptive collective variables in the same amount of dimensions than established linear methods. In the main chapter, we concern ourselves with improving the analysis of simulation data by incorporating this deep learning method. Simulation can be considered as an experiment conducted on a computer that creates a lot of raw data from which insights can only be extracted in a nontrivial manner. This analysis follows an elaborate modeling pipeline, which consists of multiple steps and algorithms. One of these crucial steps is dimensionality reduction, in which high-dimensional data is mapped into a lower-dimensional space, retaining as much of the important information as possible and aiming to find descriptive collective variables fit for modeling. We show with a well-studied small peptide, deca-alanine, that the aforementioned deep autoencoder architecture with an additional distance metric - EncoderMap - allows to find collective variables that are at least as good as an established linear method - TICA - in the same amount of dimensions. Connecting results, obtained by simulation, back to experiment is done by identifying metastable states, long-lived structural conformations that are accessible to experiment. We compare these dimensionality reduction methods in their capabilities to find expressive collective variables that allow to find these metastable states. Lastly, as EncoderMap does not make use of the time-series character of the data and works on structure alone, our results hint towards potential applications in combination with algorithms that allow to harvest unordered data fast, e.g. Monte Carlo simulations.

ZUSAMMENFASSUNG

Die Zunahme der Rechenleistung in den letzten Jahrzehnten, die am besten durch das empirische Gesetz von Moore beschrieben wird, hat es ermöglicht, die Simulation als dritte Säule der Wissenschaft zwischen Theorie und Experiment zu etablieren. Die Untersuchung von Systemen "in-silico" ist seitdem zu einem weit verbreiteten Forschungsansatz geworden, der numerische Erkenntnisse auf Größenskalen ermöglicht, die für Theorie und Experimente nicht zugänglich sind. In jüngerer Zeit haben sich künstliche Intelligenz und maschinelles Lernen, insbesondere Deep Learning, als eine der Schlüsseltechnologien des Informationszeitalters herauskristallisiert, die durch die reichliche Verfügbarkeit von Rechenleistung und Daten ermöglicht werden. In dieser Arbeit zeigen wir in zwei Fallstudien, dass ein Deep-Learning-Ansatz zur Dimensionalitätsreduktion namens EncoderMap in der Lage ist, bessere, aussagekräftige kollektive Variablen in der gleichen Anzahl an Dimensionen zu finden als die etablierten linearen Methoden. Im Hauptkapitel beschäftigen wir uns mit der Verbesserung der Analyse von Simulationsdaten durch die Einbeziehung von Deep-Learning-Methoden. Simulationen können als Experimente betrachtet werden, die auf einem Computer durchgeführt werden und eine große Menge an Rohdaten erzeugen, aus denen nur auf nicht-triviale Weise Erkenntnisse gewonnen werden können. Diese Analyse folgt einer aufwendigen Modellierungspipeline, die aus mehreren Schritten und Algorithmen besteht. Einer dieser entscheidenden Schritte ist die Dimensionalitätsreduktion, bei der hochdimensionale Daten in einen niedrigdimensionalen Raum abgebildet werden, wobei so viele wichtige Informationen wie möglich erhalten bleiben sollen, um aussagekräftige kollektive Variablen zu finden, die sich für die Modellierung eignen. Anhand eines gut untersuchten kleinen Peptids, Deca-Alanin, zeigen wir, dass eine tiefe Autoencoder-Architektur mit einer zusätzlichen Distanzmetrik - EncoderMap - es ermöglicht, kollektive Variablen zu finden, die mindestens so gut sind wie eine etablierte lineare Methode - TICA - bei der gleichen Anzahl von Dimensionen. Die Verknüpfung von Simulationsergebnissen mit Experimenten erfolgt durch die Identifizierung metastabiler Zustände, also langlebiger Strukturkonformationen, die für Experimente zugänglich sind. Wir vergleichen diese Methoden zur Dimensionalitätsreduktion hinsichtlich ihrer Fähigkeit, aussagekräftige kollektive Variablen zu finden, die das Auffinden dieser metastabilen Zustände ermöglichen. Da EncoderMap den Zeitreihencharakter der Daten nicht nutzt und nur mit der Struktur arbeitet, deuten unsere Ergebnisse auf potenzielle Anwendungen in Kombination mit Algorithmen hin, die es ermöglichen, ungeordnete Daten schnell zu erzeugen, z.B. Monte-Carlo-Simulationen.

CONTENTS

I INTRODUCTION 1

II THEORY & METHODS 7
II.1 Statistical physics & simulation . . . 8
II.2 Stochastic dynamics . . . 10
II.2.1 Langevin equation . . . 11
II.2.2 Fokker-Planck equation . . . 13
II.3 Markov state modeling . . . 15
II.3.1 Motivation . . . 16
II.3.2 Clustering/Discretization . . . 18
II.3.2.1 K-means algorithm . . . 19
II.3.2.2 Fuzzy c-means . . . 19
II.3.2.3 Robust Perron-Cluster Cluster Analysis (PCCA++) 20
II.3.3 Dimensionality reduction techniques . . . 21
II.3.3.1 Curse of Dimensionality . . . 21
II.3.3.2 Linear dimensionality reduction . . . 21
II.3.3.3 Deep learning . . . 25
II.3.4 Estimation of the transition matrix . . . 29
II.3.5 Chapman-Kolmogorov test . . . 31
II.4 Multi-Ensemble Methods . . . 32
II.4.1 Multistate Bennett Acceptance Ratio (MBAR) . . . 33
II.4.2 Transition-based reweighting analysis method (TRAM) . . 36

III FOLDING OF DECA-ALANINE 39
III.1 Introduction/Overview . . . 41
III.2 The modeling pipeline . . . 43
III.3 The molecule . . . 46
III.4 Molecular dynamics simulation . . . 47
III.5 Descriptors . . . 48
III.5.1 Pairwise distances with H-bonds . . . 49
III.5.2 Torsion angles . . . 49
III.5.3 H-bonds . . . 50
III.6 Dimensionality reduction . . . 50
III.6.1 Pairwise distances descriptor space . . . 52

III.6.2 TICA . . . 52
III.6.3 EncoderMap . . . 52
III.6.3.1 Torsion angle descriptor . . . 55
III.6.3.2 H-bond descriptors . . . 56
III.6.4 TICA-4D-E2E . . . 57
III.7 Excursus: H-bonds . . . 60
III.7.1 Criteria for closed H-bonds . . . 60
III.7.2 Distribution of closed H-bonds in latent space . . . 63
III.8 Discretization . . . 65
III.9 MSM modeling . . . 67
III.10 Metastable states . . . 69
III.11 Validation . . . 71
III.11.1 Chapman-Kolmogorov test . . . 71
III.11.2 Comparison of metastable sets . . . 75
III.11.2.1 TICA . . . 76
III.11.2.2 EncoderMap . . . 79
III.11.2.3 TICA-4D-E2E . . . 79
III.12 Summary & Conclusions . . . 81

IV SOCIAL COMPLEXITY 83
IV.1 Structure of the data . . . 84
IV.2 Connection to molecular dynamics . . . 87
IV.3 Comparison of different methods . . . 88
IV.3.1 Fraction of variance explained . . . 88
IV.3.2 Crossvalidation on imputed datasets . . . 89
IV.3.3 Performance in different dimensions . . . 91
IV.3.4 PCA vs. EncoderMap (1D) . . . 93
IV.3.5 PCA vs. EncoderMap (2D) . . . 96
IV.4 Exploratory data analysis in the latent space . . . 98
IV.4.1 Independence of the data from region . . . 98
IV.4.2 Time series behavior . . . 99
IV.4.2.1 Influence of time on the latent space . . . 99
IV.4.2.2 Transitions between clusters . . . 101
IV.4.2.3 Region time series in latent space . . . 105
IV.4.3 Interpreting the latent space . . . 107
IV.4.3.1 Correlations between complexity characteristics . . . 107
IV.4.3.2 PCA specific analysis methods . . . 108
IV.4.3.3 Correlation with latent space variables . . . 110
IV.4.3.4 Model-Agnostic interpretation methods . . . 112
IV.5 Summary . . . 121

V PERSPECTIVES 123

BIBLIOGRAPHY 125

A APPENDIX 139
A.1 List of publications 140
A.2 Acknowledgements 141
A.3 Curriculum Vitae 142

I

INTRODUCTION

The evolution of the natural sciences has largely been driven by human curiosity to understand the composition and behavior of the matter around us. To remove human bias from the equation, the process of investigation into the natural world has been formalized as the scientific method, initially resting upon the back and forth between theory and experiment. Ideally, theories allow to produce falsifiable predictions which are then tested in experiment. The results of experiments allow to rule out false theories or inform improvements of theories, while theories inform experiments by giving guidance towards interesting phenomena to examine. In practice, however, both theory and experiment have to deal with severe limitations. It is often hard to extract falsifiable predictions from a theory. Possible reasons include equations, which lack closed-form solutions, or complex systems, where countless variables interact in non-linear, emergent ways that can not be described by simple equations. It is even harder to extract falsifiable predictions from a theory that can be tested in an experiment. Experiments are often limited by the time- and length-scales the available instruments have access to. The challenge for theorists then is to come up with new plausible theories as well as to extract novel predictions in clever ways, which can be tested in experiments with available instruments. The challenge for the experimenters is on the other hand to come up with clever experimental designs or instruments to test the predictions from the theory side. This interplay between theory and experiment leads to new insights, theory improvements and knowledge gain overall. [1, 2]

A prominent example of this procedure is the debate on the existence of atoms. Though already hypothesized by Democritus in the 5th century BC, the debate could only be settled around 2500 years later in the 20th century [3, 4]. Shortly after the adaptation of the microscope in the 17th century, the irregular motion of small grains suspended in liquids was observed. In the 19th century, botanist Robert Brown then was one of the first to carry out detailed observations of the movement of pollen grains suspended in water. He later showed that the same phenomenon can be observed

for inorganic matter, suggesting a physical origin of 'Brownian motion'. The bigger controversy at that time was – after the successful theories of Maxwell and Newton – whether a 'continuous' description of all natural phenomena was possible, which stood in direct opposition to the discontinuous nature of matter put forth by the molecular-kinetic theory. In his famous paper from 1905 [5], Einstein developed the first testable theory of Brownian motion by deriving a diffusion law, which was experimentally measurable. The theory was further developed by Smoluchowski [6] and Langevin [7] and marked the development of the field of stochastic dynamics. The predictions finally were confirmed in experiment by Perrin in 1909 [8], who was awarded the Nobel price for his contributions in 1926. This scientific achievement was crucial in establishing the atomic nature of matter and consequently statistical mechanics as the statistical basis of thermodynamics [3].

Although the existence of atoms was established in the early 20th century, the aforementioned limitations on the theoretical side of science remained. The study of many-particle-systems remained limited, because Newtonian mechanics becomes virtually intractable for more than two particles. This changed, when scientific computing became available in the early 1950s. The two corner stones of molecular simulation were introduced around the same time, Monte Carlo (MC) simulation [9] and molecular dynamics (MD) simulation [10]. Monte Carlo simulations aim to estimate average system properties by sampling and averaging over configurations from the statistical mechanical ensemble corresponding to the system's thermodynamic state. Molecular dynamics simulations determine the movements of every atom in a molecular system over time by solving its equations of motion. By now, computer simulation has established itself as a third pillar of science in between theory and experiment [1, 2, 11, 12]. Running 'computer experiments' allows to inform theory as well as experiment by uncovering functional mechanisms, often in atomic detail, or by predicting the properties of systems. Simulation enables to investigate the behavior of bio-molecules, especially proteins, on time- and length-scales not accessible to experiment. Computing contributes to the scientific method by complementing theory and experiment and alleviating some of their shortcomings. In particular, computers make it possible (in principle) to approximate solutions to equations to any desired accuracy, thus extracting predictions from theories that are not possible analytically. Since its inception, computer simulation has been developed into a mature discipline that has been instrumental in the interpretation of experiments and sometimes even in the prediction of their results [13, 14].

It is important to note, however, that simulations are not without their own inherent limitations. Firstly, simulation does not have a direct connection to the physical world. Rather, it can only explore the theory that was put in, and consequently may produce unphysical results if the theory was incorrect in the first place. In this regard, simulation is closer to theory than experiment, and all of its results must be checked against nature or, if not possible, rigorous internal validation needs to be performed. Secondly, above all, simulation is limited by the availability of

computational resources and the efficiency of the algorithms employed. Although, in principle, simulation can give answers to a lot of interesting and relevant physical questions, the feasibility is frequently constrained by financial resources and the time necessary to conduct the simulations. One of these problems is protein folding. Although in principle accessible to simulation, the majority of proteins fold on the timescales of milliseconds (10^{-3}s), while timesteps in MD simulations typically are on the order of femtoseconds (10^{-15}s), necessitating enormous compute resources to bridge this gap. Furthermore, the computation needed for each timestep scales with the amount of atoms in the molecule so that the simulation of protein folding remains feasible only for relatively small peptides comprising approximately 100 residues [15–19]. These limitations are addressed in two ways. The limitation in computing power is mitigated by the ongoing growth and the general abundance of computing resources in today's world fueled by technological advancement, the digitalization of our world and the proliferation of specialized hardware like the widespread availability of GPUs due to the current AI boom [13, 20–22]. Additionally, there are initiatives aimed at the development of specialized computer chips for the purpose of molecular dynamics simulations [23]. The second approach to mitigate the limitations of MD simulations are a multitude of enhanced sampling algorithms that accelerate the sampling of the configuration space of the system under investigation, leading to an effective speedup of the simulations [24, 25]. In this work, we employ enhanced sampling techniques by combining simulations from multiple ensembles.
An approach that has been widely adopted in the field is that of Markov state models (MSMs), as it addresses both the limitations in simulation speed and facilitates the analysis of substantial amounts of simulation data. The ability of MSMs to combine multiple independent short simulations allows the problem to be naturally parallelized, enabling the exploitation of today's available massively parallel computing. Furthermore, MSMs facilitate the analysis and human understanding of simulation data by providing a low-dimensional coarse-grained model of relevant metastable states and transition probabilities between them. For MSMs to offer a good description of the physical processes, however, it is important to choose their temporal, spatial resolution and their states well. An elaborate modeling pipeline has been developed to achieve this [26–29]. In this work, we investigate the improvement potential of introducing a deep learning method into the modeling pipeline for MSMs. By now, deep learning has proven itself to be one of the most promising, capable and versatile technologies of the emerging 21st century. Besides being responsible for generative models of not before seen quality, like ChatGPT for text or Stable Diffusion for images, deep learning also has found entrance into most branches of science as a universal tool for modeling (e.g. drug discovery, medical image analysis, materials science, particle physics) [30–33]. Although domain knowledge is still relevant, there has been a longstanding effort to remove the human element from the modeling process and let algorithms find relationships in the data in a data-driven, unsupervised manner without human constraint. Artificial neural networks (ANNs) are a big

step towards this promise by learning internal representations autonomously - often incomprehensible to humans - that are frequently able to outperform other, more classical methods. The impact of deep learning on the natural sciences was recognized by the Nobel Prizes in physics and chemistry in 2024. In physics the Nobel Prize was awarded to Hinton and Hopfield for their foundational discoveries and inventions that enable machine learning with artificial neural networks. In the field of chemistry, the Nobel Prize was awarded in part to researchers from Google DeepMind, who developed AlphaFold [34], an ANN that predicts protein structures, resolving a long-standing challenge in the field through the application of deep learning.

In this thesis, we investigate the capabilities of deep learning for the use case of dimensionality reduction, which is an important step in most modeling pipelines. This is the case because high-dimensional spaces are so vast that data in them is always sparse, a fact known as the curse of dimensionality. More specifically, we use EncoderMap, an autoencoder architecture with an additional distance metric which is able to compress information from a high-dimensional into a lower-dimensional space in a non-linear and unsupervised fashion. We show in two case studies that this method is able to build better collective variables, compressing more information in the same amount of dimensions than established linear methods.

In the main chapter III, we investigate the capabilities of EncoderMap as a dimensionality reduction technique in the modeling pipeline of molecular dynamics simulations. We use a small, well studied peptide - Deca-Alanine - to compare the capabilities of EncoderMap with the established linear method of time-lagged independent component analysis (TICA).

In the second case study in chapter IV, we explore this method for sparse data using a very different dataset containing quantitative information from world history. As the modeling in this case bears similarity with the modeling in molecular dynamics, we carry over our approach, apply EncoderMap for dimensionality reduction and compare it against the established linear method of principal component analysis (PCA).

In both case studies we aim to quantitavely compare the methods involved as well as to investigate the collective variables found and gain understanding of their low dimensional spaces. The linear methods offer a more direct interpretation of the axes, while for EncoderMap the axes do not necessarily have meaning. The distance metric in the EncoderMap algorithm, however, infuses additional structure in the low-dimensional space, leading to points being close in low-dimensional latent space, which were close in the original high-dimensional space before dimensionality reduction. By investigating their low-dimensional spaces, we aim to understand which information is retained by the different dimensionailty reduction methods and assess their capabilities for modeling.

The various methods used in this thesis are summarized in chapter II, which is intended to act as both an introduction to the methods and as a reference, when more

information about the methods is of interest to the reader. In the final chapter V, we summarize our findings and give some potential directions for further research.

II

THEORY & METHODS

Contents

II.1 Statistical physics & simulation 8
II.2 Stochastic dynamics 10
II.2.1 Langevin equation 11
II.2.2 Fokker-Planck equation 13
II.3 Markov state modeling 15
II.3.1 Motivation 16
II.3.2 Clustering/Discretization 18
II.3.3 Dimensionality reduction techniques 21
II.3.4 Estimation of the transition matrix 29
II.3.5 Chapman-Kolmogorov test 31
II.4 Multi-Ensemble Methods 32
II.4.1 Multistate Bennett Acceptance Ratio (MBAR) 33
II.4.2 Transition-based reweighting analysis method (TRAM) 36

This chapter expands on methods and theory that underlie the analyses of the case studies in chapter III and chapter IV. It is intended both as an introduction and as a reference and collects further information here to not disrupt the flow of reading in the other chapters.
We start with a general short introduction into statistical physics and how it enables the study of physical systems on the computer through simulation in section II.1. As thermal fluctuations give rise to stochastic behavior in molecular systems, we introduce the description of stochastic dynamics in section II.2. We approximate the stochastic processes with Markov state models (MSMs), time-discrete stochastic models with a finite state space, to make modeling computationally feasible and interpretable. For MSMs to be a good approximation, it is important that timescales and states are well chosen and we introduce the elaborate modeling pipeline in section II.3. Lastly, we introduce multi-ensemble methods in section II.4, which allow combining information from simulations conducted in different ensembles and which can be used to speed up the sampling process.

II.1 STATISTICAL PHYSICS & SIMULATION

Statistical physics is the branch of physics that applies statistical methods to explain and predict the thermodynamic properties of systems that are comprised of a large number of particles. Unlike classical mechanics, which often focuses on the behavior of individual particles, statistical physics recognizes that the collective behavior of particles leads to emergent phenomena that can be described statistically. There exists comprehensive literature covering thermodynamics and statistical physics - we mainly use [35, 36] here.
At its core, statistical physics connects microscopic states (the individual configurations of particles) to macroscopic observables (such as temperature, pressure and entropy). The set of all accessible microstates is called the phase space

$$\Gamma = \{(\mathbf{x}_1, \ldots, \mathbf{x}_f) \,|\, \mathbf{x}_i = (q_i, p_i)\} \tag{II.1}$$

with the generalized coordinates q_i and their canonically conjugated momenta p_i. For a classical system with N particles the degrees of freedom amount to $f = 3N$.
To understand the physics of a macroscopic system, however, the knowledge of the exact microstate is often not relevant. When the system exhibits a high amount of degrees of freedom, the central limit theorem leads to well defined stable equilibria, which are characterized by a few macroscopic observables. There then exist numerous microscopic configurations which correspond to the same macrostate.
This realization leads to the notion of an ensemble: An ensemble is defined as all the thermodynamic systems (all possible microscopic realizations) which are compatible with a given set of macroscopic obervables (e.g. given energy E, volume V and number

of particles N for the microcanonical ensemble) that fully define the macroscopic equilibrium state.
The notion of an ensemble enables two distinct ways to determine observables A on the computer [1]:

- Monte Carlo (MC) Simulation:
  In general, following basic probability theory, observables A are given by ensemble averages over the state space Γ weighted by the equilibrium distribution $\mu(\mathbf{x})$:

$$\langle A \rangle = \int_\Gamma A(\mathbf{x})\, \mu(\mathbf{x})\, d\mathbf{x}. \tag{II.2}$$

  The equilibrium distribution μ and the accessible state space Γ depend on the given ensemble, e.g. $\mu = \mu_{(T,V,N)}$ and $\Gamma = \Gamma(T, V, N)$ for the canonical ensemble with given temperature T, volume V and number of particles N. Here, each configuration of the system $\mathbf{x} \in \Gamma$ needs to be prepared in such a way that it is compatible with the macroscopic equilibrium quantities describing the system. Each microstate $\mathbf{x}$ can therefore be viewed as another macroscopically equivalent system of the ensemble. Using the law of large numbers, Equation II.2 can be approximated by averaging the observable over a finite amount of configurations/systems $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of the ensemble, which are distributed according to the equilibrium distribution $\mathbf{x}_i \sim \mu$:

$$\langle A \rangle = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} A(\mathbf{x}_i) \approx \frac{1}{n} \sum_{i=1}^{n} A(\mathbf{x}_i). \tag{II.3}$$

  For efficient calculation, configurations $\mathbf{x}$ with high probability weight $\mu(\mathbf{x})$ need to be found. Here, the famous Metropolis-Hastings-Algorithm [9, 37] is often used, which is a Markov Chain Monte Carlo (MCMC) method that uses a Markov process to obtain a sequence of random samples from a probability distribution (from the equilibrium distribution μ in this case).

- Molecular Dynamics (MD) Simulation:
  Another way to view an ensemble is to think about the time evolution of the system. When a microstate $\mathbf{x}$ is prepared that adheres to a specific ensemble (e.g. the canonical ensemble (T, V, N)) then this microstate will evolve over time $\mathbf{x} = \mathbf{x}(t)$ according to the equations of motion. As the macroscopic equilibrium (i.e. the ensemble) does not change, this time evolution represents a trajectory in the state space of the respective ensemble, e.g. $\mathbf{x}(t) \in \Gamma(T, V, N)$. This allows to take a time average of an observable A along the trajectory:

$$\bar{A} = \lim_{t\to\infty} \frac{1}{t} \int_0^t A\left(\mathbf{x}(t')\right)\, dt'. \tag{II.4}$$

If this time evolution visits all possible microstates $\mathbf{x} \in \Gamma$ in a finite time, then the time average is equal to the ensemble average of Equation II.2:

$$\langle A \rangle = \bar{A}. \tag{II.5}$$

This assumption is called the "ergodic hypothesis" and it is not true in general (it is e.g. not true for glasses). It is true for a lot of systems, however, and when true it allows for another method to determine observables on the computer: In molecular dynamics simulations a system is prepared in a microstate $\mathbf{x}$ and is then evolved by solving the equations of motion repeatedly for a small timestep Δt. The simulation program needs to take care of keeping the evolution of microstates $\mathbf{x}(t)$ in a given ensemble. When the system is equlibrated properly, observables can be measured by time averages according to Equation II.5.

The equilibrium distribution of systems in thermal equilibrium with a heat bath of given temperature T is the Boltzmann distribution. For the case of the canonical ensemble with given (T, V, N), it reads:

$$\mu(\mathbf{x}) = e^{-\beta E(\mathbf{x})}/Z_c \tag{II.6}$$

with the inverse temperature $\beta = 1/k_B T$ and the energy function $E(\mathbf{x})$. The canonical partition function Z_c, represents the normalization over the state space $\Gamma(T, V, N)$:

$$Z_c = \int_{\Gamma(T,V,N)} e^{-\beta E(\mathbf{x})}\, d\mathbf{x}. \tag{II.7}$$

It can be shown that the partition function is a particular important quantity, as all thermodynamic equilibrium quantities can be derived from it. The quantity that is minimized in equilibrium in the canonical ensemble is the free energy

$$F(T, V, N) = -k_B T \ln(Z_c). \tag{II.8}$$

II.2 STOCHASTIC DYNAMICS

The trajectory $\mathbf{x}(t)$ of a point particle in an isolated system can be described in a deterministic way by Newton's equations of motion

$$m\ddot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}, \dot{\mathbf{x}}, t), \tag{II.9}$$

which is a differential equation in $\mathbf{x}$. The total force $\mathbf{F}$ acting on the particle depends on the system under consideration.
When considering more complex systems, however, thermal fluctuations and environmental interactions can give rise to stochastic behavior. This turns the previously

deterministic trajectory $x(t)$ into a realization of a stochastic process X_t. A stochastic process X_t is a sequence of random variables, indexed by a parameter, usually time t.
A classic example for such stochastic trajectories is the erratic and irregular movement of large pollen grains suspended in water (Brownian motion). Furthermore, each trajectory calculated in a molecular dynamics simulation is such a realization of a stochastic process.
There exist two equivalent ways to describe stochastic processes:

- Description on the level of realizations $x(t)$ of a stochastic process X_t:
  By introducing stochastic terms into the differential equation, we end up with a stochastic differential equation (SDE) that is called Langevin equation in physics.

- Description on the level of the probability distribution $p(x,t)$ of a stochastic process X_t:
  Under the restriction that the stochastic process is Markovian (it does not depend on the past) the time evolution of the probability is described by an integral equation called Chapman-Kolmogorov equation. The differential form of the Chapman-Kolmogorov equation is called master equation and in the special case of continuous state spaces bears the name of Fokker-Planck equation (FPE).

We will elaborate on these descriptions in the following chapters and demonstrate their application using the example of Brownian motion. For this chapter we use [38–40], which are classic books on stochastic processes in the context of physics.

II.2.1 LANGEVIN EQUATION

To model the erratic and irregular movement of large pollen grains suspended in water, Langevin [7] introduced a stochastic force $\mathbf{F}_{\text{stoch}}$ into Newton's equation of motion (Equation II.9). For instructional purposes we will constrain ourselves to the 1-dimensional case here, although the formalism can well be generalized to d dimensions. Langevin's equation reads

$$\begin{aligned} m\ddot{x}(t) &= F(\mathbf{x}, \dot{x}, t) \\ &= F_{\text{drag}} + F_{\text{stoch}} \\ &= -\gamma\dot{x}(t) + \sigma\eta(t), \end{aligned} \tag{II.10}$$

where $\mathbf{F}_{\text{drag}} = -\gamma\dot{x}(t)$ is Stokes' law for the friction of a particle moving through a viscous fluid. The stochastic force $F_{\text{stoch}} = \sigma\eta(t)$ consists of the strength of the force σ and the Gaussian white noise $\eta(t)$ with the properties

$$\langle\eta(t)\rangle = 0 \tag{II.11}$$

$$\langle\eta(t)\eta(t')\rangle = \delta(t-t'). \tag{II.12}$$

These properties encode that the stochastic force has no preferred direction and that different times are uncorrelated (i.e. there exists no memory and the process is Markovian). This Langevin equation can be considered as a first order differential equation in $v(t) = \dot{x}(t)$ with the formal solution

$$v(t) = \boldsymbol{v}(0)\, e^{-\frac{\gamma}{m}t} + \frac{\sigma}{m}\, e^{-\frac{\gamma}{m}t} \int_0^t \eta(t')\, e^{\frac{\gamma}{m}t'}\, dt'. \tag{II.13}$$

Taking the average and using Equation II.11 yields

$$\langle v(t)\rangle = v(0)\, e^{-\frac{\gamma}{m}t}, \tag{II.14}$$

so on average an initial velocity $\boldsymbol{v}(0)$ decays exponentially with relaxation time $\tau = \frac{m}{\gamma}$. For $t < t'$ the velocity autocorrelation yields

$$\langle v(t)v(t')\rangle = \left(v(0)^2 - \frac{\sigma^2}{2m\gamma}\right) e^{-(t+t')/\tau} + \frac{\sigma^2}{2m\gamma} e^{-(t'-t)/\tau}. \tag{II.15}$$

As for these particles the relaxation timescale is usually very short $t, t' \gg \tau$, the first term can be neglected and the second term dominates. The velocity autocorrelation therefore decays very rapidly, but yields a finite constant value for equal times:

$$\left\langle v(t)^2 \right\rangle \approx \frac{\sigma^2}{2m\gamma}. \tag{II.16}$$

In statistical physics exists another expression for the mean square velocity in equilibrium, namely the equipartition theorem:

$$\frac{1}{2} m \left\langle v(t)^2 \right\rangle = \frac{1}{2} k_B T. \tag{II.17}$$

Combining Equation II.16 and Equation II.17 results in a relation between the strength of the stochastic force σ and the temperature of the system T:

$$\sigma^2 = 2\gamma k_B T. \tag{II.18}$$

Integrating Equation II.15 allows to determine the mean squared displacement of the particle

$$\left\langle (x(t)-x(0))^2 \right\rangle \approx \frac{\sigma^2}{\gamma^2}\tau\left(e^{-t/\tau}-1\right)+\frac{\sigma^2}{\gamma^2}t \overset{(t\gg\tau)}{\approx} \frac{\sigma^2}{\gamma^2}t \overset{(II.18)}{=} \underbrace{2\frac{k_B T}{\gamma}}_{=:D} t, \tag{II.19}$$

which reproduces the diffusion law Einstein had derived before Langevin in his famous paper [5]. Einstein, however, employed a different approach we will present in the following chapter, where he used the probability distribution of the stochastic process $p(x,t)$. The Einstein relation $D = \frac{k_B T}{\gamma}$ shows that the diffusion coefficient D and the mobility of the particle are connected through the thermal energy (fluctuation-dissipation theorem).
To summarize, a general Langevin equation is a stochastic differential equation

$$\dot{X}_t = A(X_t,t) + B(X_t,t)\eta(t), \tag{II.20}$$

consisting of a deterministic part (A) and a stochastic part ($B\eta$), where the Gaussian noise $\eta(t)$ follows Equation II.11 and Equation II.12.
In this chapter, we used the Langevin equation to model a particle in a thermal bath (Brownian motion). We find that even if the particle inhabits a preferential direction in the beginning in the form of an initial velocity, this velocity decays rapidly so that on average the particle holds no preferential direction for movement (Equation II.14). Although no direction is preferred, the stochastic force (i.e. the incessant kicks from other particles in the bath) keeps the particle in motion (Equation II.16). The strength of the motion depends on the temperature of the bath and leads to diffusive behavior of the particle (Equation II.19).
The stochastic process with an initial velocity discussed here is also called an Ornstein-Uhlenbeck process, while the purely diffusive process (obtained in the over-damped limit $\tau \to 0$) is called a Wiener process.

II.2.2 FOKKER-PLANCK EQUATION

We will now turn from the description of stochastic processes with stochastic differential equations to the equivalent description using the probability density $p(x,t)$. Given a trajectory $\{(x_1,t_1),\ldots,(x_n,t_n)\}$ with time ordering $t_i < t_j$ for $i < j$, we denote the joint probability for the trajectory as $p(x_n,t_n;\ldots;x_1,t_1)$, which fully determines a (separable) stochastic process. We assume the Markov property

$$p(x_n,t_n;\ldots|x_i,t_i;\ldots;x_1,t_1) = p(x_n,t_n;\ldots|x_i,t_i) \quad \forall i \in \{1,\ldots,n-1\}, \tag{II.21}$$

which can be interpreted as "memorylessness", i.e. at each point in the trajectory the future always depends only on the present, but not on the past. This property is an idealization that can often be employed for macroscopic scales, while physical processes usually exhibit memory on microscopic scales. This memory, however, often decays over time so that the process is Markovian on an adequately chosen timescale. Using the Markov property, the joint probability can recursively be reduced to transition probabilities $p(x_j, t_j|x_i, t_i)$ and an initial probability $p(x_1, t_1)$, which fully describe the process:

$$\begin{aligned} p(x_n, t_n; \ldots; x_1, t_1) &= p(x_n, t_n|x_{n-1}, t_{n-1}; \ldots; x_1, t_1)\, p(x_{n-1}, t_{n-1}; \ldots; x_1, t_1) \\ &= \ldots \\ &\overset{(II.21)}{=} p(x_n, t_n|x_{n-1}, t_{n-1}) \ldots p(x_2, t_2|x_1, t_1) \cdot p(x_1, t_1). \end{aligned} \tag{II.22}$$

For these probabilities, two consistency equations for Markov processes can be derived. Integrating the definition of the conditional probability

$$p(x_j, t_j; x_i, t_i) = p(x_j, t_j|x_i, t_i) \cdot p(x_i, t_i) \tag{II.23}$$

over x_i gives

$$p(x_j, t_j) = \int p(x_j, t_j|x_i, t_i) \cdot p(x_i, t_i)\, dx_i, \tag{II.24}$$

which can be interpreted as the law of total probability for Markov processes.
To derive the second consistency equation, we first use the definition of conditional probability and the Markov property to derive the following expression:

$$\begin{aligned} p(x_3, t_3; x_2, t_2|x_1, t_1) &= \frac{p(x_3, t_3; x_2, t_2; x_1, t_1)}{p(x_1, t_1)} \\ &= \frac{p(x_3, t_3; x_2, t_2; x_1, t_1)}{p(x_2, t_2; x_1, t_1)} \cdot \frac{p(x_2, t_2; x_1, t_1)}{p(x_1, t_1)} \\ &= p(x_3, t_3|x_2, t_2; x_1, t_1) \cdot p(x_2, t_2|x_1, t_1) \\ &\overset{(II.21)}{=} p(x_3, t_3|x_2, t_2) \cdot p(x_2, t_2|x_1, t_1). \end{aligned} \tag{II.25}$$

Integrating Equation II.25 over x_2 yields the Chapman-Kolmogorov equation

$$p(x_3, t_3|x_1, t_1) = \int p(x_3, t_3|x_2, t_2) \cdot p(x_2, t_2|x_1, t_1)\, dx_2 \tag{II.26}$$

that relates all transition probabilities $p(x_j, t_j|x_i, t_i)$ of a Markov process to each other. It is both an integral equation describing the dynamics of a Markov process as well as a consistency equation we use later in this work to check whether our system under investigation can be adequately described with a Markov state model (see subsection II.3.5).

From the Chapman-Kolmogorov equation a partial differential equation for the time evolution of the probability $p(x,t)$ can be derived, which is the central equation for processes with diffusion, called the Fokker-Planck equation. The Fokker-Planck equation that is equivalent to the general Langevin equation of Equation II.20 can be derived to be

$$\frac{\partial}{\partial t}p(x,t) = -\frac{\partial}{\partial x}\left(A(x,t)p(x,t)\right) + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left(B(x,t)^2\,p(x,t)\right). \tag{II.27}$$

Coming back to the Brownian motion of subsection II.2.1, we consider the overdamped case ($\tau = \frac{m}{\gamma} \to 0 \Rightarrow \frac{m}{\gamma}\ddot{x}(t) \to 0$) of Equation II.10:

$$\begin{aligned}\dot{x}(t) &= \frac{\sigma}{\gamma}\eta(t)\\ &= \sqrt{2D}\,\eta(t),\end{aligned} \tag{II.28}$$

where in the last line Equation II.18 and the Einstein relation was used. Turning this Langevin equation into its respective Fokker-Planck equation (compare Equation II.20 and Equation II.27) yields the diffusion equation

$$\frac{\partial p(x,t)}{\partial t} = D\,\frac{\partial^2 p(x,t)}{\partial x^2}. \tag{II.29}$$

Assuming the particle is localized at $x_0 = x(0)$ in the beginning $p(x,0) = \delta(x - x_0)$, the solution of Equation II.29 is a Gaussian

$$p(x,t) = \frac{1}{\sqrt{4\pi D t}}e^{-\frac{(x-x_0)^2}{4Dt}} \tag{II.30}$$

with mean $\mu_x = x_0$ and variance $\sigma_x^2 = 2Dt$. By looking at the mean squared displacement, we recover the diffusion law of Equation II.19 that was originally derived by Einstein [5]:

$$\left\langle (x(t) - x(0))^2 \right\rangle = \left\langle (x(t) - \mu_x)^2 \right\rangle = \sigma_x^2 = 2Dt. \tag{II.31}$$

II.3 MARKOV STATE MODELING

In the previous chapters we showed, using the example of Brownian motion, how stochastic processes can be modeled using either Langevin equations or probability distributions (using the Chapman-Kolmogorov or the derived Fokker-Planck equation). Both approaches - implicitly or explicitly - assume the Markov property of "memorylessness" - that the process does only depend on the present and not on

the past. Physical processes usually do not exhibit memorylessness on microscopic scales, while the idealization often holds on appropriate macroscopic scales.
We now go a step further - and thereby make modeling more computationally feasible - by approximating the physical process with Markovian models discrete in time and states, Markov state models (MSMs). The challenge then is, to choose an appropriate timescale and state space such that the behavior is (approximately) Markovian *and* the states are (physically) meaningful. This requires a sophisticated modeling pipeline, the methods of which are described in this chapter.
Markov state models in conjunction with molecular dynamics simulations have been a long-standing research field spanning several decades [26–28, 41–43]. We give a short introduction here to provide enough information for understanding the results presented in chapter III.

II.3.1 MOTIVATION

The motivation to use MSMs for modeling molecular dynamics simulations is two-fold. Firstly, it helps tackling the sampling problem (slow barrier crossing and limited total simulation time) by making it possible to run a lot of short simulations from different starting points and combine them to estimate long term behavior that is otherwise not accessible through simulation alone. Secondly, the spectral decomposition of the transition matrix allows to extract the interesting dominant slow processes and their implicit timescales from data. We will elaborate on both points briefly.
In equilibrium molecular dynamics simulations the stationary distribution is given by the Boltzmann distribution

$$\mu(\mathbf{x}) = e^{-\beta E(\mathbf{x})}/Z \tag{II.32}$$

with the inverse temperature $\beta = 1/k_B T$, energy function $E(\mathbf{x})$ and partition function (i.e. the normalization over the state space Γ)

$$Z = \int_\Gamma e^{-\beta E(\mathbf{x})}\, d\mathbf{x}. \tag{II.33}$$

Interesting physical quantities are measured by averages

$$\langle A \rangle = \int_\Gamma A(\mathbf{x})\, \mu(\mathbf{x})\, d\mathbf{x}, \tag{II.34}$$

which involve integrals over a vast state space Γ. In general, these integrals can not be computed exactly, but they can be approximated by sampling the $x \in \Gamma$ with highest weights $\mu(\mathbf{x})$ which contribute the most to the integral. Performing this sampling is hard, especially because of metastable states and free energy barriers, which are hard to cross. Introducing a disjoint state space partitioning $\dot{\bigcup}_{i=1}^{k} S_i = \Gamma$,

however, can alleviate this problem when the partitions S_i do not have internal barriers [27]. Then the local stationary densities μ_i in set i can easily be sampled and are defined as

$$\mu_i(\mathbf{x}) = \begin{cases} \frac{\mu(\mathbf{x})}{\pi_i} & x \in S_i \\ 0 & x \notin S_i \end{cases} \tag{II.35}$$

with the stationary probability π_i to be in set i

$$\pi_i = \int_{S_i} \mu(\mathbf{x})\, d\mathbf{x}. \tag{II.36}$$

Estimating transition probabilities t_{ij} of transitioning from discrete state $i \to j$ in time τ then do not need information about the global equilibrium distribution $\mu(\mathbf{x})$, but only about the local equilibrium distribution $\mu_i(\mathbf{x})$

$$\begin{aligned} t_{ij}(\tau) &= \mathbb{P}[x(t+\tau) \in S_j \mid x(t) \in S_i] \\ &= \frac{\int_{x \in S_i} \int_{y \in S_j} p(\mathbf{x} \to \mathbf{y}; \tau)\, \mu_i(\mathbf{x})\, d\mathbf{x}\, d\mathbf{y}}{\int_{S_i} \mu_i(\mathbf{x})\, d\mathbf{x}}. \end{aligned} \tag{II.37}$$

Here, $p(\mathbf{x} \to \mathbf{y}; \tau)$ denotes the transition probability density from $\mathbf{x} \in \Gamma$ to $\mathbf{y} \in \Gamma$ after time τ. Following this approach allows for the full dynamical information of the discretized system to be obtained by simulating short trajectories from each of the partitions S_i. The starting points $\mathbf{x}$ of these simulations need to be drawn from the local equilibrium distributions $\mu_i(\mathbf{x})$. Estimating the transition matrix $\mathbf{T} = (t_{ij})$ allows to determine the stationary probabilities π_i for each state i as the discrete stationary distribution of $\mathbf{T}$ fulfilling

$$\boldsymbol{\pi} \cdot \mathbf{T} = \boldsymbol{\pi}. \tag{II.38}$$

This eventually allows to determine physical quantities

$$\begin{aligned} \langle A \rangle &= \int_\Gamma A(\mathbf{x})\, \mu(\mathbf{x})\, d\mathbf{x} \\ &= \sum_i \pi_i \int_{x \in S_i} A(\mathbf{x})\, \frac{\mu(\mathbf{x})}{\pi_i}\, d\mathbf{x} \\ &= \sum_i \pi_i \int_{x \in S_i} A(\mathbf{x})\, \mu_i(\mathbf{x})\, d\mathbf{x} \\ &= \sum_i \pi_i \langle A \rangle_i \end{aligned} \tag{II.39}$$

through local instead of global sampling, simultaneously offering a parallelization of the problem.

The transition matrix $\mathbf{T}(\tau)$ propagates the probabilities of being in different states i, (p_i), forward in time by lag time τ

$$\mathbf{p}(t_0 + \tau) = \mathbf{p}(t_0)\mathbf{T}(\tau). \tag{II.40}$$

We assume detailed balance $\pi_i t_{ij} = \pi_j t_{ji}$ to be fulfilled as it is equivalent to being in equilibrium [42]. Then the transition matrix is diagonizable and can be written as its spectral decomposition [44]

$$\mathbf{T}(\tau) = \mathbf{\Pi}^{-1} \sum_{i=1}^{k} \lambda_i \, \mathbf{l}_i \mathbf{l}_i^T \tag{II.41}$$

with the inverse of the discrete stationary distribution $\mathbf{\Pi}^{-1} = \mathrm{diag}(\pi_1^{-1}, \ldots, \pi_k^{-1})$, the eigenvectors $\mathbf{l}_i$ and their eigenvalues λ_i.
With iterated application of the transition matrix, a time evolution for $t = s\tau$ is given and as the matrix is diagonal, the exponential only acts on the diagonal of the decomposition

$$\begin{aligned}
\mathbf{p}(t) &= \mathbf{p}(s\tau) = \mathbf{p}(0)\mathbf{T}(s\tau) = \mathbf{p}(0)\mathbf{T}^s(\tau) \\
&= \mathbf{p}(0) \left(\mathbf{\Pi}^{-1} \sum_{i=1}^{k} \lambda_i \, \mathbf{l}_i \mathbf{l}_i^T \right)^s = \mathbf{p}(0) \, \mathbf{\Pi}^{-1} \sum_{i=1}^{k} \lambda_i^s \, \mathbf{l}_i \mathbf{l}_i^T \\
&= \mathbf{p}(0) \, \mathbf{\Pi}^{-1} \sum_{i=1}^{k} e^{\ln(\lambda_i) s} \, \mathbf{l}_i \mathbf{l}_i^T = \mathbf{p}(0) \, \mathbf{\Pi}^{-1} \sum_{i=1}^{k} e^{\underbrace{-\frac{t}{\left(-\frac{\tau}{\ln(\lambda_i)}\right)}}_{t_i}} \mathbf{l}_i \mathbf{l}_i^T.
\end{aligned} \tag{II.42}$$

So the time evolution for the probability distribution of the states $\mathbf{p}(t)$ can be decomposed into subprocesses $\mathbf{l}_i \mathbf{l}_i^T$ (almost) all of which decay exponentially on a timescale t_i. It can be shown that the largest eigenvalue of a stochastic matrix is equal to one, $\lambda_1 = 1$, and is associated to the equilibrium distribution $\mathbf{l}_1 = \boldsymbol{\pi}$, as it never decays. When arranging the eigenvalues in descending order, we therefore get a decomposition of the complete process into subprocesses of ascending timescales. This allows to extract the slowest timescales and its processes, which are often connected to slow transitions between metastable states and observable by experiment.

II.3.2 CLUSTERING/DISCRETIZATION

As Markov state models (MSMs) need a discrete state space, methods to turn a continuous trajectory into a discrete one are required. Clustering algorithms are

able to provide this discretization and are also used to group states of a finite state space together to reduce the amount of states, which is a form of coarse-graining.

II.3.2.1 K-means algorithm

The k-means algorithm [45] is used to group N_s samples/observations $\{\xi_1, \ldots, \xi_{N_s}\}$ into k clusters $\{S_1, \ldots, S_k\}$. The centers of these k clusters can then be seen as a discretization of the original data. The number of clusters k is the only input parameter for this algorithm, which has to be provided and can not be found automatically. To find a suitable k, there exist several heuristics and clustering evaluation metrics e.g. the elbow method. In Markov state modeling, the number of discrete clusters k is chosen such that it balances minimizing the discretization error (high k) and having enough transition statistics between these states to construct a reliable MSM (low k).
Starting with k cluster centers $\{c_1, \ldots, c_k\}$ the algorithm consists of alternating between two steps:

1. Assignment step:
   Assign all samples ξ_i to their nearest cluster centers c_j (i.e. $\xi_i \in S_j$) with

$$j = \underset{l \in \{1,\ldots,k\}}{\arg\min} \|\xi_i - c_l\|^2. \tag{II.43}$$

2. Update step:
   Recalculate all the cluster centers c_i as the mean of the assigned samples

$$c_i = \frac{1}{|S_i|} \sum_{\xi \in S_i} \xi. \tag{II.44}$$

It can be shown that this algorithm is minimizing the variance within the clusters and only converges to a local optimum. Although only a local optimum is guaranteed, in practice this algorithm is important as it solves a theoretically NP-hard problem heuristically in polynomial time and often converges quickly. The k-means++ algorithm [46] improves the original algorithm by choosing better starting cluster centers.

II.3.2.2 Fuzzy c-means

The fuzzy c-means (FCM) algorithm [47, 48] is a fuzzy clustering method in which every of the N_s samples/observations $\{\xi_1, \ldots, \xi_{N_s}\}$ can be assigned to several cluster centers $\{c_1, \ldots, c_k\}$ with membership degree $(m_{ij}) \in [0,1]^{N_s \times k}$. While k-means assigns every sample to exactly one cluster, fuzzy c-means can be thought of as a generalization thereof, which was developed for the case of imprecisely defined categories. The fuzziness of the clusters is controlled by an additional parameter

$f \in (1, \infty)$. In the limit of no fuzzyness ($f \to 1$) k-means is recovered. The FCM algorithm uses a weighted version to compute the cluster centers

$$c_j = \frac{\sum_{i=1}^{N_s} m_{ij}^f \xi_i}{\sum_{i=1}^{N_s} m_{ij}^f} \tag{II.45}$$

and minimizes a weighted version of the objective function of k-means (variance within clusters)

$$\sum_{i=1}^{N_s} \sum_{j=1}^{k} m_{ij}^f \|\xi_i - c_j\|^2 \tag{II.46}$$

with the memberships

$$m_{ij} = \left(\sum_{l=1}^{k} \left(\frac{\|\xi_i - c_j\|}{\|\xi_i - c_l\|} \right)^{\frac{2}{f-1}} \right)^{-1}. \tag{II.47}$$

The standard choice for the fuzziness is $f = 2$. Equivalently to k-means the algorithm only finds a local optimum and depends on the starting cluster centers. Although for fixed observations i, the memberships to be assigned to all of the cluster centers are normalized ($\sum_{j=1}^{k} m_{ij} = 1$), they can not be interpreted as probabilities, since the distribution is controlled by fuzziness f and therefore is somewhat arbitrary.

II.3.2.3 Robust Perron-Cluster Cluster Analysis (PCCA++)

When discretizing trajectories into k discrete states to be able to build a Markov state model (MSM), the goal is to model the transitions in the data with minimal error. Ultimately, the states we are interested in are the $N_c << k$ long-lived metastable states that could be observed in experiment.
Given a $k \times k$ transition matrix $\mathbf{T}$, the Robust Perron-Cluster Cluster Analysis (PCCA++) [49, 50] is a fuzzy clustering method that determines a membership matrix $(m_{ic}) \in \mathbb{R}^{k \times N_c}$ of discrete state k belonging to a given amount of (metastable) states N_c. The algorithm is best suited for this coarse-graining task as, with its fuzzy assignments, it is able to handle transition states and it exactly preserves the slow timescales while simultaneously aiming to maximize the 'crispness' of the clusters. It is important to mention that the resulting coarse process on the metastable states found is not guaranteed to be a Markov chain.
Thus coarse-graining introduces further errors so that calculation of system kinetics is better done on the more accurate discrete MSM with k states. Lastly, the method has been further developed in recent years to enable modeling of non-equilibrium systems under the name of generalized PCCA (G-PCCA) [51–53].

II.3.3 DIMENSIONALITY REDUCTION TECHNIQUES

As high-dimensional spaces are so vast that data often is sparse, dimensionality reduction techniques are applied to mitigate this problem by mapping the data into a lower-dimensional space, while retaining as much of the information as possible. Here, we present the dimensionality reduction methods that are used in this thesis and represent an important part of the modeling pipeline. The methods are categorized into classical linear methods and the non-linear modern approach using deep learning.

II.3.3.1 Curse of Dimensionality

The curse of dimensionality is a common term, referring to problems in machine learning associated with high-dimensional spaces. Assuming one needs n points to model a function on a 1-dimensional interval, n^d points are needed on its d-dimensional counterpart. This poses a problem in machine learning as algorithms need exponentially more data, the higher the dimension is, and this often leads to algorithms failing to extract meaningful information in high dimensions.
To circumvent this problem, dimensionality reduction techniques are employed to transform the data into a lower-dimensional space. Consequently, the quality of the data in the lower-dimensional space is dependent on the dimensionality reduction technique used. Since in high-dimensional spaces inherently more information can be stored, the dimensionality reduction technique decides, which part of the information is used and which is discarded. Ideally, only the information for the problem at hand is projected into the lower-dimensional space and noise in the data is discarded.
Naturally, this is not always feasible to do, with the exception of the data living on a submanifold and having a dimensionality equal or lower to the space the data is supposed to be projected into. Following these arguments - that machine learning algorithms may only have enough density of data in lower-dimensional spaces - dimensionality reduction techniques are often an important part of data processing pipelines.

II.3.3.2 Linear dimensionality reduction

PCA

Principal Component Analysis (PCA) is a method to find a linear transformation into a new orthogonal coordinate system, in which the variance of the data is maximized along the axes. It can then be used as a dimensionality reduction technique by projecting onto the first axes with maximum variance. This reduces the amount of dimensions while retaining the maximum amount of variation in the data.

The method is closely related to the rotation of rigid bodies in physics and finds the same principal axes as the principal axes of inertia for a set of point masses rotating around their center of mass. We follow the derivation in [54].
Let $\mathbf{x}(t)$ be a p-dimensional data vector out of $t \in \{1, \dots, n\}$ observations. Although for our purposes this will be time-series data, in general the data does not need to be ordered for PCA. We normalize the data in each of the p components:

$$(\tilde{\mathbf{x}}(t))_i = \frac{(\mathbf{x}(t))_i - \overline{(\mathbf{x}(t))_i}}{\sigma_i} \qquad \forall i \in \{1, \dots, p\} \tag{II.48}$$

with sample mean $\overline{(\mathbf{x}(t))_i}$ and sample standard deviation $\sigma_i = \sqrt{\sigma_i^2}$

$$\overline{(\mathbf{x}(t))_i} = \mathop{\mathbb{E}}_t \Big[(\mathbf{x}(t))_i \Big], \tag{II.49}$$

$$\sigma_i^2 = \sigma^2\Big((\mathbf{x}(t))_i \Big) = \frac{n}{n-1} \mathop{\mathbb{E}}_t \left[\left((\mathbf{x}(t))_i - \overline{(\mathbf{x}(t))_i} \right)^2 \right]. \tag{II.50}$$

While subtracting the mean is part of the PCA algorithm, scaling to unit variance leads to unitless dimensions in each entry of the data vector avoiding skew by high absolute values.
Now let Σ be the sample covariance matrix, which can be calculated using the data:

$$\Sigma = \frac{n}{n-1} \mathop{\mathbb{E}}_t \Big[\tilde{\mathbf{x}}(t) \, (\tilde{\mathbf{x}}(t))^\intercal \Big]. \tag{II.51}$$

The PCA problem can then be stated as a constrained optimization problem

$$\begin{aligned} \max_{\boldsymbol{\alpha}_0} \Big[\sigma^2\Big(\boldsymbol{\alpha}_0^\intercal \, \tilde{\mathbf{x}}(t) \Big) \Big] &= \max_{\boldsymbol{\alpha}_0} \left[\frac{n}{n-1} \mathop{\mathbb{E}}_t \Big[\boldsymbol{\alpha}_0^\intercal \, \tilde{\mathbf{x}}(t) \, (\tilde{\mathbf{x}}(t))^\intercal \, \boldsymbol{\alpha}_0 \Big] \right] \\ &= \max_{\boldsymbol{\alpha}_0} \Big[\boldsymbol{\alpha}_0^\intercal \, \Sigma \, \boldsymbol{\alpha}_0 \Big], \\ &\text{such that} \quad \boldsymbol{\alpha}_0^\intercal \boldsymbol{\alpha}_0 = 1. \end{aligned} \tag{II.52}$$

In words: Find a normalized vector $\boldsymbol{\alpha}_0$ along which the variance is maximized, when projected onto. This corresponds to the single dimension along which most of the variation in the data is retained.

Using the technique of Lagrange multipliers, this optimization problem turns into an eigenvalue problem:

$$\begin{aligned} \frac{d}{d\alpha_0}\left(\alpha_0^T \Sigma\, \alpha_0 - \lambda_0\left(\alpha_0^T \alpha_0 - 1\right)\right) &= 0 \\ \Sigma\, \alpha_0 - \lambda_0\, \alpha_0 &= 0 \\ \left(\Sigma - \lambda_0\, \mathbb{I}_p\right)\alpha_0 &= 0, \end{aligned} \tag{II.53}$$

with $\mathbb{I}_p$ being a p-dimensional identity matrix. Therefore α_0 is an eigenvector of Σ with eigenvalue λ_0. Since we are maximizing

$$\sigma^2\left(\alpha_0^T \tilde{x}(t)\right) = \alpha_0^T \Sigma\, \alpha_0 = \lambda_0\, \alpha_0^T \alpha_0 = \lambda_0, \tag{II.54}$$

we need to take the eigenvector which corresponds to the largest eigenvalue. To find the second axis α_1 with the second largest variance, we repeat the argument with the additional constraint to ensure orthogonality: $\alpha_0^T \alpha_1 = 0$.
Using Lagrange multipliers and following the same derivation yields another eigenvalue problem with α_1 being the eigenvector corresponding to the second largest eigenvalue λ_1. Repeating the argument for all dimensions of $\tilde{x}(t)$ gives rise to the following theorem:
If Σ is the covariance matrix defined above with ordered eigenvalues $\lambda_0 > \lambda_1 > \cdots > \lambda_{p-1}$ and corresponding eigenvectors $\alpha_0, \alpha_1, \ldots, \alpha_{p-1}$ then these eigenvectors solve the PCA problem of finding axes along which variance is maximized with the eigenvalues denoting the respective variances.
Numerically, this problem can be solved e.g. by singular value decomposition (SVD), which can be found in every linear algebra package.

TICA

Time-lagged independent component analysis (TICA) was first developed in [55] to separate independent signals and then explored independently in [56] and [57] as a method for analyzing molecular dynamics (MD) data. The method is similar to PCA explained in paragraph II.3.3.2. While PCA is used to find a new orthogonal coordinate system, in which the variance is maximized along its axes in descending order, TICA finds a new coordinate system, in which the autocorrelation of a time series is maximized along its axes.
While PCA finds the axes, along which the maximum of the variance in the data is retained, TICA finds the axes, along which a timeseries changes the slowest. Since TICA explicitly uses the autocorrelation function, it is in contrast to PCA only able to handle ordered data, namely time series. In MD simulation, we are usually interested in the slowest degrees of freedom, which correspond to conformational changes compared to fast degrees of freedom like rotations and vibrations.

The autocorrelation function $\mathrm{ACF}(\Delta t)$ of a 1-dimensional timeseries $\{Y_t\}_{t\in\{1,\dots,n\}}$ with lag time Δt is defined as

$$\mathrm{ACF}(Y_t, \Delta t) = \frac{\mathrm{COV}(Y_t, Y_{t+\Delta t})}{\mathrm{VAR}(Y_t)} = \frac{\mathop{\mathbb{E}}_t\left[(\delta Y_t)\cdot(\delta Y_{t+\Delta t})\right]}{\mathop{\mathbb{E}}_t\left[(\delta Y_t)^2\right]}, \tag{II.55}$$

with $\delta Y_t = Y_t - \mathbb{E}[Y_t]$ denoting the mean-free timeseries. We can formulate the TICA problem as an optimization problem similar to PCA. Let $|X_t\rangle$ be a p-dimensional data vector out of a timeseries of $t \in \{1,\dots,n\}$ observations. The TICA problem is then given by

$$\begin{aligned}
\max_{|\alpha_0\rangle}\left[\mathrm{ACF}\left(\langle\alpha_0|\,\delta X_t\rangle, \Delta t\right)\right] &= \max_{|\alpha_0\rangle}\left[\frac{\mathop{\mathbb{E}}_t\left[\langle\alpha_0|\,\delta X_t\rangle\langle\alpha_0|\,\delta X_{t+\Delta t}\rangle\right]}{\mathop{\mathbb{E}}_t\left[\langle\alpha_0|\,\delta X_t\rangle\langle\alpha_0|\,\delta X_t\rangle\right]}\right] \\
&= \max_{|\alpha_0\rangle}\left[\frac{\mathop{\mathbb{E}}_t\left[\langle\alpha_0|\,\delta X_t\rangle\langle\delta X_{t+\Delta t}|\,\alpha_0\rangle\right]}{\mathop{\mathbb{E}}_t\left[\langle\alpha_0|\,\delta X_t\rangle\langle\delta X_t|\,\alpha_0\rangle\right]}\right] \\
&= \max_{|\alpha_0\rangle}\left[\frac{\left\langle\alpha_0\middle|\,C^{(\Delta t)}\,\middle|\alpha_0\right\rangle}{\langle\alpha_0|\,\Sigma\,|\alpha_0\rangle}\right] \\
&= \max_{|\alpha_0\rangle}\left[\left\langle\alpha_0\middle|\,C^{(\Delta t)}\,\middle|\alpha_0\right\rangle\right], \\
&\text{such that}\quad \langle\alpha_0|\,\Sigma\,|\alpha_0\rangle = 1.
\end{aligned} \tag{II.56}$$

To simplify the expression we use the time-lagged correlation matrix

$$C^{(\Delta t)} = \mathop{\mathbb{E}}_t\left[|\delta X_t\rangle\langle\delta X_{t+\Delta t}|\right] \tag{II.57}$$

and the covariance matrix

$$\Sigma = \mathop{\mathbb{E}}_t\left[|\delta X_t\rangle\langle\delta X_t|\right], \tag{II.58}$$

which can be calculated from the data. Since the time-lag correlation matrix may not be exactly symmetric, it is symmetrized by adding the transpose and dividing by two. In contrast to PCA, we normalize the vector to have unit variance $\langle\alpha_0|\,\Sigma\,|\alpha_0\rangle = 1$ to achieve dimensionless units, when calculating distances along these projections.
An alternative normalization is scaling by the respective eigenvalues, which leads to distances in the projected space approximating kinetic distances [58]. Similar to the

PCA problem, we use Lagrange multipliers to turn the optimization problem into a (generalized) eigenvalue problem:

$$\left(C^{(\Delta t)} - \lambda_0 \Sigma\right) |\alpha_0\rangle = 0. \tag{II.59}$$

Since we are maximizing

$$\mathrm{ACF}\left(\langle \alpha_0 | \delta X_t \rangle, \Delta t\right) = \left\langle \alpha_0 \middle| C^{(\Delta t)} \middle| \alpha_0 \right\rangle = \lambda_0 \langle \alpha_0 | \Sigma | \alpha_0 \rangle = \lambda_0, \tag{II.60}$$

we need to take the eigenvector, which corresponds to the largest eigenvalue. To find the second axis $|\alpha_1\rangle$ with the second largest autocorrelation, we follow the same argument, while additionally ensuring the axes are uncorrelated: $\langle \alpha_0 | \Sigma | \alpha_1 \rangle = 0$. Repeating the derivation using Lagrange multipliers yields another eigenvalue problem with $|\alpha_1\rangle$ being the eigenvector corresponding to the second largest eigenvalue λ_1. Finally, repeating the argument for all dimensions of $|\delta X_t\rangle$ leads to the following statement: If $\lambda_0 > \lambda_1 > \cdots > \lambda_{p-1}$ are ordered eigenvalues with corresponding eigenvectors $|\alpha_0\rangle, |\alpha_1\rangle, \ldots, |\alpha_{p-1}\rangle$ of the generalized eigenvalue problem II.59, then these eigenvectors solve the TICA problem of finding axes, along which autocorrelations are maximized with the eigenvalues denoting the respective autocorrelations. The first eigenvectors correspond to the slowest degrees of freedom in the time series. The respective eigenvalues are connected to the implied timescales of these processes:

$$t_i = \frac{\Delta t}{\ln(\lambda_i)} \tag{II.61}$$

When analyzing a system, the hope usually is that there exists a clear spectral gap between fast and slow processes (i.e. timescale separation) and the description of the system can be simplified by projecting on the slow degrees of freedom and neglecting the fast ones. In reality, this gap is not always obvious or even existent.

II.3.3.3 Deep learning

Short introduction to Deep Learning

The deep learning revolution, fueled among others by achieving super-human performance on some tasks previously thought impossible, has made deep learning one of the most promising technologies of the decade. Here, we give a very short and rudimentary introduction to deep learning in the framework of statistical learning theory [59], with the aim to establish a general notion of it in the reader that will hopefully be able to carry him through the thesis. We will only concern ourselves with supervised deep learning, as it is the most prevalent one and although the deep learning method used in this thesis - EncoderMap - is unsupervised, it is eventually formulated internally as a supervised problem.

Supervised machine learning can be thought of as a basic fitting problem: Given tuples $(x, y) \in D \subset X \times Y$ of data D with inputs $x \in X$ and outputs (or labels) $y \in Y$, the aim is to find a mapping $f : X \mapsto Y$ such that f describes the data D in a specified optimal way: $f(x) \approx y \quad \forall (x, y) \in D$.
More specifically, this task can be formulated as an optimization problem: Chosen a class of parametric functions f_θ (also referred to as architecture in the deep learning context) with parameters/weights θ and a cost/error/loss function $L(Y, \hat{Y})$ that measures the error between a set of approximated output values $\hat{Y} \ni \hat{y} = f_\theta(x)$ and true output values $y \in Y$ of the data $(x, y) \in D$, the objective is to find the optimal parameters θ^* such that the function with this parameters f_{θ^*} best describes the data D according to the error function L:

$$\theta^* = \arg\min_\theta L(Y, f_\theta(X)) \qquad \text{with } (X, Y) = D \tag{II.62}$$

In practice, this is done by first choosing an architecture, which refers to the function f_θ with parameters θ. Usually, the architecture consists of chaining layers of 'artificial neurons', which are linear functions with a bias term and a subsequent non-linearity like $\mathrm{ReLU}(x) = \max(0, x)$.
The next step is to choose a loss function that depends on the learning task and which measures the error of the function on the data. Popular loss functions are cross-entropy for classification tasks and mean squared error and mean absolute error for regression tasks. Then training/learning consists of repeating the following steps until convergence (finding approximately good enough parameters $\hat{\theta}^*$):

1. Calculating the current error (forward pass):
   The current error $L(Y, f_{\theta_i}(X))$ is determined by first inserting the data $(X, Y) = D$ or a subset (e.g. training set) into the function f_{θ_i} with the current parameters θ_i and subsequently inserting the result into the error function L. Calculating the current output values $\hat{Y} = f_{\theta_i}(X)$ is called the forward pass in machine learning.

2. Improving the error (optimization):
   An optimization algorithm is applied to improve the parameters θ_i such that the error L is reduced and the function f_{θ_i} describes the data D better. There are a lot of optimization algorithms available. The ones used in the context of training artificial neural networks are usually of gradient descent type, using first-order derivatives with respect to the parameters. A generic gradient descent update rule would be

   $$\theta_{i+1} = \theta_i - \eta \nabla_{\theta_i} L(Y, f_{\theta_i}(X)) \tag{II.63}$$

   with learning rate η. This algorithm can be thought of as taking a step into the direction of steepest descent in the loss landscape $L(Y, f_{\theta_i}(X))$, aiming to

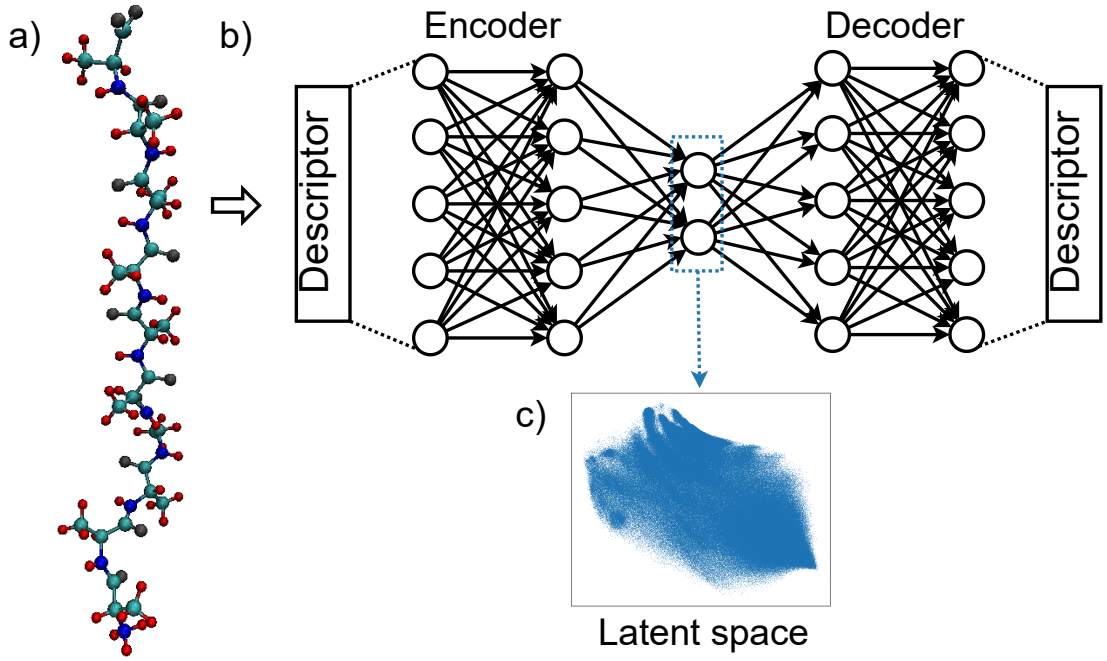


Figure 1: Exemplary usage of EncoderMap (as used in chapter III): (a) The coordinates of the atoms of a molecule are transformed into a descriptor, a representation respecting symmetries of the input data. (b) The descriptor is fed into EncoderMap, a deep neural network with a bottleneck in the middle. The network is trained to recover the input data and therefore learns to compress the data into a low-dimensional intermediate representation in the bottleneck. (c) The latent space of the compressed representation is taken as new collective variables for modeling. (Parts adapted from our publication [63].)

reduce the error L with the learning rate η controlling the step length. The derivative $\nabla_{\theta_i} L$ can be efficiently calculated with the famous back-propagation algorithm, in what is called the backward pass in machine learning.

EncoderMap

EncoderMap [60] is a non-linear dimensionality reduction algorithm combining a neural network autoencoder with the distance metric from sketch-map [61]. Sketch-map can be understood as a non-linear version of multi-dimensional scaling [62], which is an algorithm that tries to position data points in the best way in space, under the constraint of incorporating given distances between points.
In Figure 1(b) the architecture of EncoderMap is visualised. Data is fed into a neural network consisting of several layers, exhibiting a bottleneck layer in the middle. The output of the bottleneck layer neurons is considered to be the new coordinates of the low-dimensional latent space. The first half of the autoencoder representing the mapping from high-dimensional into latent space is called the encoder. The second half usually is a mirrored architecture of the encoder, called the decoder,

and is responsible for mapping the low-dimensional latent space back to the high-dimensional space.
Each neuron represents a linear function composed with a non-linear activation function. To train the model, a loss function consisting of several terms is minimized. One of the objectives encoded in the loss function is for the output of the neural network to reproduce the input as closely as possible after going through the information bottleneck. This is known as the autoencoder loss

$$L_{ae} = \frac{1}{N} \sum_{i=1}^{m} \|\hat{x}_i - x_i\|. \tag{II.64}$$

Here, $i \in \{1, \ldots, N\}$ is the index of the data, x_i is the respective data sample and $\hat{x}_i = f(x_i)$ denotes the output of the neural network f for this data. Minimizing this loss, the neural network "learns" to compress as much information as possible into the low-dimensional latent space, yielding a good latent space representation and therefore a good dimensionality reduction. EncoderMap combines the autoencoder with the sketch-map objective, which is to preserve the distances of points in the high-dimensional space also in the low-dimensional representation of the latent space as much as possible.
The idea is that low distances between data points only represent noise and fluctuations in baisins, while the high distances are only an artifact of distances in high dimensions. The relevant information lies in the intermediate distances, which represent the distances between energy baisins and therefore the connectivity between states. Sketch-map aims to preserve this connectivity through the dimensionality reduction resulting in good, informative latent variables. Sketch-map achieves this by using a separate sigmoid function as a filter for the distances in the high- and low dimensional spaces. These sigmoid functions

$$s_{\sigma, a, b}(r) = 1 - \left(1 + \left(2^{\frac{a}{b}} - 1\right) \left(\frac{r}{\sigma}\right)^{a}\right)^{-\frac{a}{b}} \tag{II.65}$$

contain free parameters, which have to be chosen for the respective problem at hand. The inflection point of the sigmoid is determined by σ and sets the length scale of intermediate distances, while a and b control how fast the function saturates towards $-\infty$ and $+\infty$ respectively. Since points are differentiated where the values are further apart, the slope of the sigmoid can be interpreted as a graph of the sensitivity regarding the different distance values. The sketch-map idea can be encoded in another loss function to be minimized:

$$L_{sm} = \frac{1}{m} \sum_{i \neq j} \left[G\left(R_{ij}\right) - g\left(r_{ij}\right)\right]^2 \tag{II.66}$$

Here, $G(R_{ij}) = s_{\sigma_h, a_h, b_h}(R_{ij})$ and $g(r_{ij}) = s_{\sigma_l, a_l, b_l}(r_{ij})$ are the sigmoids in the high- and low-dimensional space, while R_{ij} and r_{ij} are the distances between data sample i and j in high- and low-dimensional space respectively.
The loss is an average over the amount of pairwise distances m. Minimizing the sketch-map loss L_{sm} leads to the distance structure in the latent space resembling the distance structure between intermediate distanced points in the high-dimensional original dataset. The points far apart in high-dimensional space are therefore far apart in the latent space and originally close points are close. This property allows for the interpretation of the latent space as a map of states, in which structurally similar states are clustered together. The complete loss function to be minimized is then assembled to be

$$L = k_{ae} L_{ae} + k_{sm} L_{sm} + \text{REG}. \tag{II.67}$$

The prefactors k_{ae} and k_{sm} allow to set the relative priority of both of the minimization objectives, while REG is an additional regularization term to avoid overfitting. More information about the internal workings of the algorithm can be found in the original paper [60].

II.3.4 ESTIMATION OF THE TRANSITION MATRIX

As data from molecular dynamics simulations is already time-discrete, discretizing the trajectory into k discrete states enables modeling the behavior of the molecule with a memoryless stochastic model that approximates the behavior of the original data. This Markov state model (MSM) is a $k \times k$ transition matrix $\mathbf{T} = (t_{ij})$ consisting of the probabilities

$$t_{ij} = \mathbb{P}(x_{t+\tau} = j \mid x_t = i) \tag{II.68}$$

to transition from discrete state i to discrete state j in time step τ. This model describes the system fully in a stochastic sense, when the 'memorylessness' or Markov property is fulfilled. This property is that the transition probability only depends on the present state and is independent of prior states

$$\mathbb{P}(x_{t+\tau} = j \mid x_t = i, x_{t-\tau} = k, \dots) = \mathbb{P}(x_{t+\tau} = j \mid x_t = i). \tag{II.69}$$

Given an observed trajectory of discrete states $x^{obs}(t) = (x_1^{obs}, x_2^{obs}, \dots, x_n^{obs})$ the number of transitions $\mathbf{C} = (c_{ij})$ from discrete state i to j in time step τ can be counted. From these quantities, maximum likelihood estimation can be used to estimate the transition matrix $\mathbf{T}$ from the data $x^{obs}(t)$. Likelihood is defined as being proportional to the probability density of the observed data given the model

$$
\begin{aligned}
L_{MSM}(\mathbf{T}) &\propto \mathbb{P}(\text{data} \mid \text{model}) = \mathbb{P}(\mathbf{C} \mid \mathbf{T}) \\
&= \mathbb{P}(x_1 = x_1^{obs}) \prod_{t=2}^{n} \mathbb{P}(x_t = x_t^{obs} \mid x_{t-1} = x_{t-1}^{obs}, \ldots, x_1 = x_1^{obs}) \\
&= \mathbb{P}(x_1 = x_1^{obs}) \prod_{t=2}^{n} \mathbb{P}(x_t = x_t^{obs} \mid x_{t-1} = x_{t-1}^{obs}) \\
&= \mathbb{P}(x_1 = x_1^{obs}) \prod_{i,j=1}^{k} \left(t_{ij}\right)^{c_{ij}} \\
&\propto \prod_{i,j=1}^{k} \left(t_{ij}\right)^{c_{ij}} .
\end{aligned}
\tag{II.70}
$$

Here, in the third row the Markov property (Equation II.69) and in the fourth row the definition of the transition probabilities was used (Equation II.68). As we want to maximize the likelihood, a constant factor can be discarded in the last row. Since maximizing a sum is easier than a product, we change to the equivalent formulation of log-likelihood

$$
\mathcal{L}_{MSM}(\mathbf{T}) = \log\left(L_{MSM}(\mathbf{T})\right) = \sum_{i,j=1}^{k} c_{ij} \log\left(t_{ij}\right), \tag{II.71}
$$

which needs to be maximized under the constraint

$$
\sum_{j=1}^{k} t_{ij} = 1 \quad \forall i \in \{1, \ldots, k\} \tag{II.72}
$$

that the probability over all possible end states j is normalized to one for each starting state i.

To maximize a function under constraints, the method of Lagrange multipliers can be applied and re-expresses the problem as setting all the partial derivatives of the Lagrangian function

$$
\Lambda(\mathbf{T}, \boldsymbol{\lambda}) = \mathcal{L}_{MSM}(\mathbf{T}) - \sum_{i}^{k} \lambda_i \left(\sum_{j=1}^{k} t_{ij} - 1 \right) \tag{II.73}
$$

to zero $\nabla\Lambda(\mathbf{T}, \boldsymbol{\lambda}) = 0$, with Lagrange multipliers $\lambda_1, \ldots, \lambda_k$. More specifically, this yields

$$\begin{aligned}\frac{\partial \Lambda}{\partial t_{ij}} &= \frac{c_{ij}}{t_{ij}} - \lambda_i \quad &= 0 \\ \frac{\partial \Lambda}{\partial \lambda_i} &= \sum_{j=1}^{k} t_{ij} - 1 \quad &= 0\end{aligned} \tag{II.74}$$

Solving the first equation for t_{ij} and inserting this expression into the second one gives an expression for the Langrange multipliers

$$\lambda_i = \sum_{j=1}^{k} c_{ij}. \tag{II.75}$$

From this follows the best (naive) estimator of the transition probabilities, given the data, as

$$\hat{t}_{ij} = \frac{c_{ij}}{\sum_{j=1}^{k} c_{ij}}. \tag{II.76}$$

This formula expresses the intuitive result that the transition probabilities between states can be estimated by the relative frequencies of the observed transitions in the data. The derivation shown here can act as an introduction to Markov state modeling and give a basic notion of what it is about, but is naive in the sense that the derived estimator has a lot of shortcomings and has already been improved upon significantly [42, 64, 65]. One of these shortcomings is that maximum likelihood estimation assumes a uniform prior, not using additional information we might have about transition matrices. Estimated transition matrices, for example, do not necessarily fulfill detailed balance $\pi_i \hat{t}_{ij} = \pi_j \hat{t}_{ji}$, which we know to hold true in equilibrium. Improved approaches incorporate Bayesian ideas and sample the posterior distribution of possible matrices compatible with the data, to be able to get error estimates on matrices and connected quantities.

II.3.5 CHAPMAN–KOLMOGOROV TEST

At the end of the modelling pipeline we want to validate that the Markov state model we built is consistent with the data we used to build it. This is done by conducting a Chapman-Kolmogorov test [42, 66] which amounts to checking that

$$[\mathbf{P}(\tau)]^s \approx \mathbf{P}(s\tau) \tag{II.77}$$

holds for as many $s \in \mathbb{N}$ as possible. Passing this test guarantees that transition probabilities $\mathbf{P}(s\tau)$ at lag times $s\tau$ are well described by iteratively applying the

transition matrix of the Markov model $\mathbf{T} = \mathbf{P}(\tau)$, meaning that the Markov model adequately captures the transitional behavior of the data. The matrices $\mathbf{P}(s\tau)$ are estimated by maximizing Equation II.70 or more sophisticated versions hereof respectively. As we are not interested in the correct modeling of all of the k discrete states, but rather of the N_c metastable states, the test is conducted on these states.
In practice, the time evolutions $p_{cc'}(s\tau)$ of starting in metastable state c with initial distribution $\boldsymbol{w}_c \in \mathbb{R}^k$ and ending up in metastable state c' are compared for the first s_{max} multiples of the lag time τ:

$$p_{cc'}^{MSM}(s\tau) \approx p_{cc'}^{MD}(s\tau), \quad s \in \{1, \ldots, s_{max}\}. \tag{II.78}$$

Let (m_{ic}) be the membership matrix calculated by the PCCA++ algorithm (see also II.3.2.3), then the initial distribution $\boldsymbol{w}_c$ for starting in metastable state c is chosen as the normalized stationary distribution $\boldsymbol{\pi}$ of the Markov model $\mathbf{P}(\tau)$ (i.e. $\boldsymbol{\pi} \cdot \mathbf{P} = 0$), weighted with the membership probabilities

$$(\boldsymbol{w}_c)_i = \frac{\pi_i m_{ic}}{\sum_{l=1}^{k} \pi_l m_{lc}} \tag{II.79}$$

The time evolutions are given by

$$p_{cc'}^{MSM}(s\tau) = \sum_{l=1}^{k} \left[\boldsymbol{w}_c^T \left[\mathbf{P}(\tau)\right]^s\right]_l m_{lc'} \tag{II.80}$$

$$p_{cc'}^{MD}(s\tau) = \sum_{l=1}^{k} \left[\boldsymbol{w}_c^T \mathbf{P}(s\tau)\right]_l m_{lc'} \tag{II.81}$$

calculated by propagating the initial distribution to time $s\tau$ and attributing the ratio of the resulting density according to the memberships of the final state j.
The one-sigma standard error of the transition probabilities, estimated from the MD data, is given by (SI of [66], [42])

$$\sigma_{MD}^{ij}(s\tau) = \sqrt{s \frac{p_{MD}^{ij}(s\tau) - \left[p_{MD}^{ij}(s\tau)\right]^2}{\sum_{l=1}^{k} m_{li} \sum_{j=1}^{k} c_{lj}(s\tau)}}, \tag{II.82}$$

while the error of the MSM can be estimated by Bayesian methods [42, 64, 65].

II.4 MULTI-ENSEMBLE METHODS

As the integration time step of molecular dynamics simulations is restricted to femtoseconds by the fastest degrees of freedom like bond vibrations, reaching timescales

of natural processes is often computationally not feasible. To bridge this timescale gap, a lot of different enhanced sampling techniques have been developed and the field keeps being an active area of research [24, 25, 67, 68]. Multi-ensemble methods allow to combine simulations conducted in different ensembles to estimate quantities of interest. Different ensembles are used to speed up exploration of the state space as - for example - free energy barriers are more easily crossed for higher temperatures. Multi-ensemble methods encompass enhanced sampling methods using biasing potentials like umbrella sampling [69–72], metadynamics [73, 74] as well as replica exchange molecular dynamics (REMD) [75, 76].

II.4.1 MULTISTATE BENNETT ACCEPTANCE RATIO (MBAR)

The Multistate Bennett Acceptance Ratio (MBAR) [77–80] allows to combine data from different ensembles to improve free energy estimates in a chosen ensemble. The potentials $u^{\alpha}(\mathbf{x})$ need to be known for each ensemble $\alpha \in \{1, \ldots, K\}$ out of K ensembles. A general reduced dimensionless potential is given by

$$u^{\alpha}(\mathbf{x}) = \beta^{\alpha} \left[U^{\alpha}(\mathbf{x}) + p^{\alpha} V(\mathbf{x}) + (\boldsymbol{\mu}^{\alpha})^{\intercal} \mathbf{n}(\mathbf{x}) \right] \tag{II.83}$$

with $x \in \Gamma$ being from a state space Γ. The other symbols correspond to the usual thermodynamic quantities: For ensemble α, $\beta^{\alpha} = 1/k_B T^{\alpha}$ is the inverse temperature, $U^{\alpha}(\mathbf{x})$ the potential energy function, p^{α} the external pressure, $V(\mathbf{x})$ the volume, $\boldsymbol{\mu}^{\alpha}$ the vector of chemical potentials in the system and $\mathbf{n}(\mathbf{x})$ the vector containing the number of molecules corresponding to the respective chemical potentials. Which terms in Equation II.83 are relevant depends on which thermodynamic quantities are held constant i.e., which thermodynamic potential we are in. Since our data was harvested in NPT ensembles with four different biasing forces $f^{\alpha} \in \{0, 5, 10, 15\}$, our dimensionless potentials reduce to the potential energy term

$$u^{\alpha}(\mathbf{x}) = u(\mathbf{x}) + b^{\alpha}(\mathbf{x}) \tag{II.84}$$

with dimensionless bias potentials

$$b^{\alpha}(\mathbf{x}) = \beta f^{\alpha} r(\mathbf{x}), \tag{II.85}$$

$\alpha \in \{1, \ldots, K\}$ denoting the ensemble with biasing force f^{α} and $r(\mathbf{x})$ the end-to-end distance of the respective configuration. The equilibrium distributions of the ensembles follow the Boltzmann distributions

$$p^{\alpha}(\mathbf{x}) = q^{\alpha}(\mathbf{x}) / Z^{\alpha} \tag{II.86}$$

with Boltzmann weights

$$q^\alpha(\mathbf{x}) = e^{-u^\alpha(\mathbf{x})}, \tag{II.87}$$

reduced dimensionless potentials $u^\alpha(\mathbf{x})$ and partition functions

$$Z^\alpha = \int_\Gamma q^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x} \tag{II.88}$$

that act as a normalization over the state space Γ. Differences in dimensionless free energies $\beta F^\alpha = -\ln(Z^\alpha)$ between ensembles then depend on the ratio of partition functions

$$\beta F^\beta - \beta F^\alpha = -\ln\left(\frac{Z^\beta}{Z^\alpha}\right), \tag{II.89}$$

$\beta = 1/k_B T$ being the inverse temperature and subscripts $\alpha, \beta \in \{1, \ldots, K\}$ denoting the ensembles. These free energies can be estimated, when configurations are sampled in the different ensembles with known potentials. To show that, let us first canonically define the equilibrium expectation of observable A in ensemble α by

$$\langle A\rangle^\alpha = \int_\Gamma A(\mathbf{x}) p^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x} = \frac{\int_\Gamma A(\mathbf{x}) q^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x}}{\int_\Gamma q^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x}}. \tag{II.90}$$

For arbitrary choice of functions $\alpha^{\alpha\beta}(\mathbf{x})$ it then holds by symmetry that

$$\begin{aligned} Z^\alpha \langle \alpha^{\alpha\beta}(\mathbf{x}) q^\beta(\mathbf{x})\rangle^\alpha &= \int_\Gamma q^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x} \cdot \frac{\int_\Gamma q^\alpha(\mathbf{x}) \alpha^{\alpha\beta}(\mathbf{x}) q^\beta(\mathbf{x})\,\mathrm{d}\mathbf{x}}{\int_\Gamma q^\alpha(\mathbf{x})\,\mathrm{d}\mathbf{x}} \\ &= \int_\Gamma q^\alpha(\mathbf{x}) \alpha^{\alpha\beta}(\mathbf{x}) q^\beta(\mathbf{x})\,\mathrm{d}\mathbf{x} \\ &= \int_\Gamma q^\beta(\mathbf{x})\,\mathrm{d}\mathbf{x} \cdot \frac{\int_\Gamma q^\alpha(\mathbf{x}) \alpha^{\alpha\beta}(\mathbf{x}) q^\beta(\mathbf{x})\,\mathrm{d}\mathbf{x}}{\int_\Gamma q^\beta(\mathbf{x})\,\mathrm{d}\mathbf{x}} \\ &= Z^\beta \langle \alpha^{\alpha\beta}(\mathbf{x}) q^\alpha(\mathbf{x})\rangle^\beta. \end{aligned} \tag{II.91}$$

The law of large number states that for a sufficient amount of samples, the ensemble average (Equation II.90) can be approximated by an average over the samples

$$\langle A\rangle^\alpha \approx \frac{1}{N^\alpha} \sum_{i=1}^{N^\alpha} A(\mathbf{x}_i^\alpha) =: \widehat{A}^\alpha, \tag{II.92}$$

with x_i^α being the i-th sample from ensemble α: $i \in \{1, \dots, N^\alpha\}$. Approximating the ensemble averages in this way in Equation II.91 and summing over β leads to the following K estimating equations with free parameter α:

$$\sum_{\beta=1}^{K} \frac{\hat{Z}^\alpha}{N^\alpha} \sum_{i=1}^{N^\alpha} \alpha^{\alpha\beta}\, q^\beta(x_i^\alpha) = \sum_{\beta=1}^{K} \frac{\hat{Z}^\beta}{N^\beta} \sum_{i=1}^{N^\beta} \alpha^{\alpha\beta}\, q^\alpha(x_i^\beta). \tag{II.93}$$

The quality of the estimation depends on the choice of $\alpha^{\alpha\beta}$ and

$$\alpha^{\alpha\beta}(x) = \frac{\frac{N^\beta}{\hat{Z}^\beta}}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma}\, q^\gamma(x)} \tag{II.94}$$

has been shown to be optimal by having the lowest variance for a large class of possible functions [81]. As it is not summed over α in Equation II.93, rearranging and inserting Equation II.94 yields an expression to estimate the partition function

$$\begin{aligned}
\hat{Z}^\alpha &= \frac{\sum_{\beta=1}^{K} \frac{\hat{Z}^\beta}{N^\beta} \sum_{i=1}^{N^\beta} \alpha^{\alpha\beta}\, q^\alpha(x_i^\beta)}{\sum_{\beta=1}^{K} \frac{1}{N^\alpha} \sum_{i=1}^{N^\alpha} \alpha^{\alpha\beta}\, q^\beta(x_i^\alpha)} \\
&= \frac{\sum_{\beta=1}^{K} \frac{\hat{Z}^\beta}{N^\beta} \sum_{i=1}^{N^\beta} \frac{\frac{N^\beta}{\hat{Z}^\beta}}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma} q^\gamma(x_i^\beta)}\, q^\alpha(x_i^\beta)}{\sum_{\beta=1}^{K} \frac{1}{N^\alpha} \sum_{i=1}^{N^\alpha} \frac{\frac{N^\beta}{\hat{Z}^\beta}}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma} q^\gamma(x_i^\alpha)}\, q^\beta(x_i^\alpha)} \\
&= \frac{\sum_{\beta=1}^{K} \sum_{i=1}^{N^\beta} \frac{q^\alpha(x_i^\beta)}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma} q^\gamma(x_i^\beta)}}{\frac{1}{N^\alpha} \sum_{i=1}^{N^\alpha} \frac{\sum_{\beta=1}^{K} \frac{N^\beta}{\hat{Z}^\beta} q^\beta(x_i^\alpha)}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma} q^\gamma(x_i^\alpha)}} \\
&= \sum_{\beta=1}^{K} \sum_{i=1}^{N^\beta} \frac{q^\alpha(x_i^\beta)}{\sum_{\gamma=1}^{K} \frac{N^\gamma}{\hat{Z}^\gamma} q^\gamma(x_i^\beta)},
\end{aligned} \tag{II.95}$$

which further leads to an estimator for free energies $\beta\hat{F}^\alpha = -\ln(\hat{Z}^\alpha)$. These estimating equations can be solved e.g. by self-consistent iteration or a Newton-Raphson solver and allow to combine samples from different ensembles to improve free energy estimates. It is important to note that the estimated partition function in Equation II.95 is only determined up to a multiplicative constant so that only free energy differences are meaningful. It can be shown that MBAR can also be derived by maximizing the likelihood [82]

$$L_{\mathrm{MBAR}} = \prod_{\alpha=1}^{K} \prod_{i=1}^{N^\alpha} p^\alpha(x_i^\alpha). \tag{II.96}$$

II.4.2 TRANSITION-BASED REWEIGHTING ANALYSIS METHOD (TRAM)

The general transition-based reweighting analysis method (TRAM) [83, 84] can be seen as an advancement of MBAR [77], combining reweighting of data from different ensembles with Markov state modeling. We use the TRAM method in this work as a multi-ensemble method to combine the information from different ensembles, in which trajectories were simulated. The TRAM method creates a multi-ensemble Markov state model (MEMM), which consists of a Markov model for each of the ensembles, while taking into account the information from all ensembles simultaneously.
In Markov state modeling (see also subsection II.3.4), describing the molecular dynamics in discrete states $\dot{\bigcup}_{i=1}^{m} S_i = \Gamma$ and discrete times $t = s\tau, s \in \mathbb{N}$ leads to the transition matrix $\mathbf{T}^{\alpha} = \left(t_{ij}^{\alpha}\right)$. The elements t_{ij}^{α} represent the probability in ensemble α to transition from discrete state $i \to j$ in time τ, while S_i represents the phase space volume belonging to discrete state i. The transition matrices are approximated by using a maximum likelihood approach: For number of transitions c_{ij} from discrete state i to j in time step τ observed in a fixed ensemble, the likelihood is given by (derived in Equation II.70)

$$L_{\mathrm{MSM}} = \prod_{i,j=1}^{m} \left(t_{ij}\right)^{c_{ij}} . \tag{II.97}$$

The equilibrium distributions of the ensembles follow the Boltzmann distributions $p^{\alpha}(\boldsymbol{x}) = q^{\alpha}(\boldsymbol{x})/Z^{\alpha}$, with Boltzmann weights $q^{\alpha}(\boldsymbol{x}) = e^{-u_{\alpha}(\boldsymbol{x})}$, reduced dimensionless potentials

$$u^{\alpha}(\boldsymbol{x}) = u(\boldsymbol{x}) + b^{\alpha}(\boldsymbol{x}) \tag{II.98}$$

and partition functions

$$Z^{\alpha} = \int_{\Gamma} q^{\alpha}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \tag{II.99}$$

that act as a normalization over the state space Γ. Compared to MBAR, which postulates global equilibrium for each of the ensembles, TRAM relaxes this requirement to only demanding local equilibrium within the discrete states i,

$$p_i^{\alpha}(\boldsymbol{x}) = \begin{cases} q^{\alpha}(\boldsymbol{x})/z_i^{\alpha} = p^{\alpha}(\boldsymbol{x})\frac{Z^{\alpha}}{Z_i^{\alpha}} & \boldsymbol{x} \in S_i \\ 0 & \text{else.} \end{cases} \tag{II.100}$$

This corresponds to renormalizing the distribution for each discrete state i separately with

$$Z_i^\alpha = \int_{S_i} q^\alpha(\mathbf{x}) d\mathbf{x}. \tag{II.101}$$

We can then formulate the local equilibrium likelihood analogously to the MBAR likelihood in Equation II.96:

$$\begin{aligned} L_{\text{LEQ}} &= \prod_{\alpha=1}^{K} \prod_{i=1}^{m} \prod_{\mathbf{x}\in X_i^\alpha} p_i^\alpha(\mathbf{x}) \\ &= \prod_{\alpha=1}^{K} \prod_{i=1}^{m} \prod_{\mathbf{x}\in X_i^\alpha} p(\mathbf{x}) e^{-b^\alpha(\mathbf{x})} e^{\Delta f_i^\alpha}, \end{aligned} \tag{II.102}$$

which is expressed in reference to the unbiased ensemble with distribution $p(\mathbf{x})$ and free energy differences $\Delta f_i^\alpha = f_i^\alpha - f$. The set X_i^α contains the samples belonging to discrete state i in ensemble α. Combining the likelihoods in Equation II.102 and Equation II.97 eventually gives the TRAM likelihood

$$L_{\text{TRAM}} = \prod_{\alpha=1}^{K} \left(\prod_{i,j=1}^{m} (t_{ij})^{c_{ij}} \right) \left(\prod_{i=1}^{m} \prod_{\mathbf{x}\in X_i^\alpha} p(\mathbf{x}) e^{-b^\alpha(\mathbf{x})} e^{\Delta f_i^\alpha} \right). \tag{II.103}$$

The TRAM likelihood is maximized, subject to the additional constraints

$$e^{-\Delta f_i^\alpha} p_{ij}^\alpha = e^{-\Delta f_j^\alpha} p_{ji}^\alpha \qquad \forall i, j, \alpha \tag{II.104}$$

$$\sum_{j=1} p_{ij}^\alpha = 1 \qquad \forall i, \alpha \tag{II.105}$$

$$\sum_{\mathbf{x}\in X} p(\mathbf{x}) = 1, \tag{II.106}$$

yielding the point densities $p(\mathbf{x})$ for all configurations $x \in X$, the transition probabilities p_{ij}^α to transition from discrete state $i \to j$ in time τ in ensemble α and the free energy differences Δf_i^α. The constraints correspond to normalizations and detailed balance, which enforces reversibility.

III

FOLDING OF DECA-ALANINE

Contents

III.1 Introduction/Overview 41
III.2 The modeling pipeline 43
III.3 The molecule 46
III.4 Molecular dynamics simulation 47
III.5 Descriptors 48
III.5.1 Pairwise distances with H-bonds 49
III.5.2 Torsion angles 49
III.5.3 H-bonds 50
III.6 Dimensionality reduction 50
III.6.1 Pairwise distances descriptor space 52
III.6.2 TICA 52
III.6.3 EncoderMap 52
III.6.4 TICA-4D-E2E 57
III.7 Excursus: H-bonds 60
III.7.1 Criteria for closed H-bonds 60
III.7.2 Distribution of closed H-bonds in latent space 63
III.8 Discretization 65
III.9 MSM modeling 67
III.10 Metastable states 69
III.11 Validation 71
III.11.1 Chapman-Kolmogorov test 71
III.11.2 Comparison of metastable sets 75

III.12 Summary & Conclusions 81

III.1 INTRODUCTION/OVERVIEW

Predicting the folded structure of a protein from its amino-acid sequence is a challenging task that has driven research in molecular simulations for decades [13, 20, 23, 85–88]. While deep learning architectures have recently been employed to achieve promising results such as, among others, Google's AlphaFold [34, 89, 90], investigating the folding mechanisms still necessitates studying the behavior of a protein in an aqueous environment with molecular dynamics situations. Molecular dynamics simulations, however, are computationally expensive, require specific choices of force fields [91–95] and are limited to the study of relatively small peptides with lengths up to approximately 100 residues [15–19].

Because of these limitations there have been long standing efforts to speed up simulations as well as improving the modeling of the harvested data. An elaborate modeling framework has been built around Markov state models (MSMs) with an involved modeling pipeline, consisting of numerous processing steps and algorithms [27–29, 41]. MSMs are memoryless discrete stochastic models approximating the behavior of a molecule as probabilistic transitions between metastable molecular conformations.

A significant step in the modeling process is dimensionality reduction, where a high-dimensional descriptor is transformed into a low-dimensional space, serving as collective variables (CVs). The MSMs are then built in the space of these CVs, so the modeling relies heavily on these CVs' ability to represent conformational transitions accurately. There exist various dimensionality reduction methods exploiting the temporal information present in molecular dynamics simulations data, such as, among others, TICA [56, 57], time-lagged autoencoders [96], and VAMPnet [97]. When there is no temporal information available (e.g., Monte Carlo simulations), or when the temporal information is not compatible (e.g., trajectories simulated under different external parameters), alternative methods for dimensionality reduction are needed. For that, approaches often involve constructing low-dimensional representations by considering the connectivity between data points, only incorporating structural (geometrical) information. Notable representatives of this approach include Isomap [98, 99], diffusion maps [100–102], and sketch-map [61], which was later integrated with an artificial neural network autoencoder architecture into EncoderMap [60].

In a prior investigation, we found that EncoderMap, utilizing solely structural data, produces representations with similar quality to those obtained by TICA (incorporating temporal information) for the self-assembly of two single-stranded DNA fragments into a ring-like structure [103]. Additionally, autoencoder architectures have demonstrated success in automated CV discovery [104–106] and accelerating simulations [107]. EncoderMap provides an added advantage through its sketch-map functionality, which preserves distances between configurations during dimensionality reduction and thus retains supplementary structural information.

We therefore formulate the following research question, which we explore in this thesis: Is it feasible to use deep learning (more specifically EncoderMap) for dimen-

sionality reduction to construct a descriptive, low-dimensional space from structural information alone, in which it is possible to model the behavior of molecules with MSMs and identify metastable sets?

We explore this question by following the three different paths in the modeling pipeline in Figure 2, representing three different methods and comparing their modeling capabilities. To control for variations in systems, we employ deca-alanine, an alpha-helix forming small peptide model system that has been extensively studied [108–115]. The first modeling path (dark yellow) skips dimensionality reduction and therefore retains more information as the other modeling paths. This is only possible for small systems and we use it here as baseline we compare the results of the other methods with. The second modeling path (red) uses TICA for dimensionality reduction, which is an established linear method that uses temporal information. The third modeling path (blue) represents our approach, using deep learning in the form of EncoderMap as the dimensionality reduction method, which only incorporates structural information. We compare our results with a fourth modeling path not visualized in the figure and published in [116]. This study was performed by Fabian Knoch, a former member of our research group, who originally harvested and analysed the data used in this work for his study.

The current main chapter starts out by shortly explaining the different steps of the modeling pipeline (Figure 2) in the subsequent section III.2 and then goes on to explore each step in a dedicated section afterwards. There, each step in the modeling pipeline is described, visualized and explored. Special emphasis is given to gaining insight into the low-dimensional latent spaces constructed by the dimensionality reduction methods TICA and EncoderMap as well as comparing the performance of MSMs built in these spaces. The performance of the models can be considered as a metric for the quality of the latent spaces constructed by these dimensionality reduction methods. Investigating these latent spaces leads us into an excursus about H-Bonds in section III.7 as the formation of the native H-Bonds of deca-alanine adds to the understanding of which structures are located where in each of the respective latent spaces. The outcomes of the modeling process are finally compared in section III.11 in two important aspects. These aspects consist of, firstly, the capability of the latent spaces to allow for accurate modeling of the transitions in the data with MSMs and, secondly, the capability to identify the same metastable sets in the latent spaces that are found in the modeling path without dimensionality reduction. Dimensionality reduction methods excelling in both of these aspects imply we are able to compress the important aspects of the data into lower dimensions, thereby vastly simplifying the modeling process.

This chapter represents a more extensive version of our results, which were published in [63], and it therefore contains significant overlap with this publication.

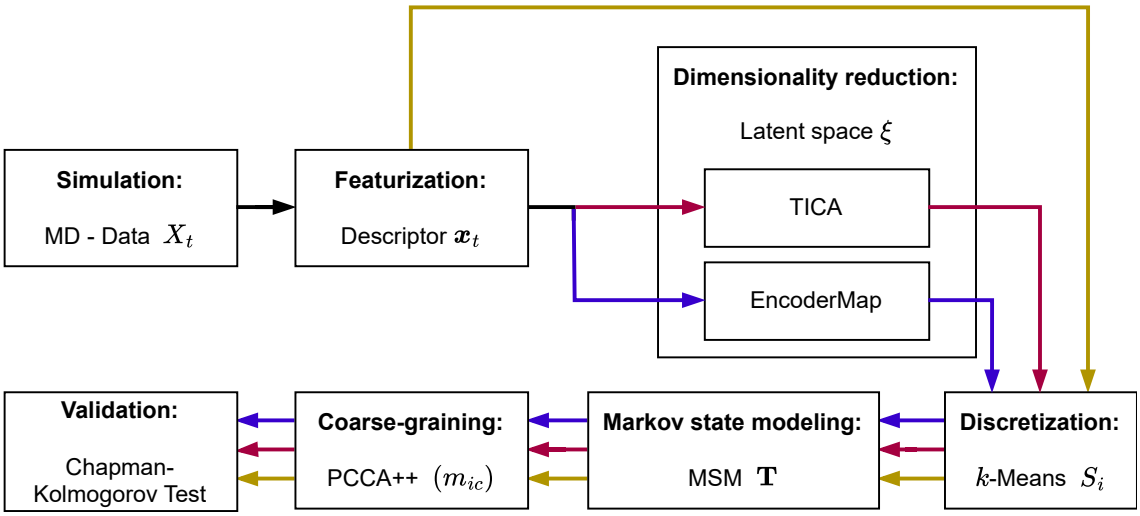


Figure 2: Modeling pipeline used in this work. The baseline approach involves omitting the dimensionality reduction step and discretizing the descriptor directly, which is represented as the dark yellow path. The established modeling pipeline follows the red path and utilizes the TICA method for dimensionality reduction, which leverages temporal information in the data. In this work, we propose and investigate the blue path as an alternative, using EncoderMap, a machine learning-based approach that relies solely on structure. (Adapted from our publication [63].)

III.2 THE MODELING PIPELINE

When modeling a molecule on the computer, we are usually not equally interested in all of the possible states (configurations), but rather in the longer-living states the molecule spends most of its (simulation) time in. These classes of similar states, which correspond to minima in the free energy, are called conformations and can be accessible to experiments. Several of these conformations can make up metastable sets representing big classes of different structures with slow transitions between. The notion of the behavior of molecules consisting of distinct metastable sets with transitions in between, leads to using Markov state models (MSMs) as a stochastic way to describe their behavior. Markov state models consist of a set of distinct states with transition probabilities. Approximating MSMs can help bridge the gap to longer timescales, which often are not accessible to simulation. A short introduction to MSMs can be found in section II.3, while more extensive introductions are offered by [26, 28]. Conducting a study on a molecule via the computer and buildig a MSM usually consists of the following steps:

1. Molecular dynamics (MD) simulation:
   The time-evolution of the molecule is simulated with a molecular dynamics simulation program. For this, an initial configuration and the force-fields describing the interactions have to be specified. Often a solvent is added, which is crucial for the behavior of the simulated molecule to be physical. The program

then evolves the configurations in discrete time steps by integrating Newtons equations of motion numerically. The described approach is classical, but there also exist a lot of different algorithms to simulate additional (quantum) effects. Two widely used programs are GROMACS [117] and LAMMPS [118]. The output of these simulations are trajectories consisting of snapshots of the system at different timestamps showing the evolution of the system.

2. Featurization:
   Since the output of molecular dynamics simulations are trajectories, usually containing positions and velocities of each atom in the simulation box at each time step, the data is naturally very high-dimensional already for small molecules. Ordinarily, one is not interested in the absolute values stored in the trajectory, but rather in relative values between the positions of the atoms, encoding the structure of the molecule. The absolute values furthermore do not exhibit any symmetries and change completely when the molecule is translated or rotated. To get a more suitable description, in which these symmetries are apparent, the original data is featurized - mapped to a new representation abstracting away these symmetries. Featurization can also be thought of as picking the observables. Two commonly used features are distances (e.g. pairwise distances) between atoms or angles (e.g. dihedrals or torsion angles) between backbone atoms, which both capture information about the structure of the molecule under investigation and obey translational and rotational invariance. Another desirable property of features is that the structure of the molecule can be reconstructed out of the featurization. While this is possible for dihedral angles, it is not unique for pairwise distances. As a welcome side-effect featurization can also offer a first form of dimensionality reduction. As data is sparse in high dimensions, dimensionality reduction is a necessary step to be able to use modeling algorithms (see also subsection II.3.3.1). Therefore, featurization acts as a first dimensionality reduction step, transforming the original data into a descriptor, while next in the modeling pipeline follows an explicit dimensionality reduction step to find good collective variables.

3. Dimensionality reduction:
   In this crucial step, the dimensionality is further reduced to find good collective variables, which describe the important processes. The goal is to find the least amount of coordinates, which still capture all of the relevant processes and describe them in an obvious way. Since this is no easy task to do by hand, machine learning algorithms are employed, which are able to do this automatically. Examples for linear algorithms are PCA (paragraph II.3.3.2) and TICA (paragraph II.3.3.2), whereas TICA is used as the standard in the modeling pipeline. In this work we explore the use of non-linear dimensionality reduction methods, namely EncoderMap (paragraph II.3.3.3), over the linear method of TICA. It can further be shown that TICA fulfills a variational principle called

variational approach for Markov processes (VAMP) [119]. There exist further non-linear dimensionality reduction methods built on this approach [97, 120].

4. Discretization:
   To be able to approximate a MSM, we need transitions between discrete states. In this step of the modeling pipeline, the continuous CVs are discretized. This can be done by using various binning strategies or applying clustering algorithms like k-means, which is the standard method. The k-means algorithm positions a given amount of cluster-centers in such a way that the error made by the discretization is minimal (see also subsection II.3.2). We aim for a discretization, which resolves the important processes.

5. Markov state modeling:
   The next step is to estimate the actual MSM and validate that it correctly captures the behavior of the molecule in the subsequent modeling steps. There exist several methods to build MSMs depending on the data and assumptions (see section II.3). Parameters must be chosen as well, one of them being the lag time, which represents the time after which transitions are counted and which therefore represents the temporal resolution of the MSM. When choosing the lag time, there exists a trade-off between resolving more processes (lower lag time) and ensuring that the implied timescales of the processes have converged (higher lag time).

6. Coarse-graining:
   The MSM is built on the k discrete states so that it adequately captures the transitional behavior on this discretized version of the original data. We are, however, interested in the long-lived metastable states and the slow transitions in between them that are accessible to experiment, so that a coarse-graining step is required. Clustering these k discrete states into N_c metastable sets is done in a fuzzy way by using the PCCA++ algorithm [49]. The amount of metastable states N_c is an input parameter for the PCCA++ algorithm, which is usually chosen as the amount of slow processes that are separated by a gap in the implied timescales from the fast processes of the model, implying a scale separation between slow and fast processes. The slow processes correspond to the slow transitions between metastable states we are interested in. After having identified these states, they can further be examined by looking at the average configurations to get an idea, which conformations these metastable states represent.

7. Validation:
   Since MSMs require Markovianity to be a good description of the process, the model has to be validated to exhibit this property for the data under investigation. This is done via the Chapman-Kolmogorow (CK) test, which tests to which degree the evolution of the MSM matches with the evolution in the data

(see also subsection II.3.5). This test is carried out on the metastable sets, which are a coarse-grained version of the discretized states defined in the discretization step. Usually, several of the k discrete states belong to each of the N_c metastable sets. Estimating and validating an MSM is an iterative process that involves almost all of the steps in the modeling pipeline (Figure 2). The parameters all over the pipeline are adjusted with the aim to build a model, which resolves the important processes and fulfills Markovianity.

8. MSM analysis:
   After having built a validated MSM, this model works as a substitute for the more complex system and can be used to extract knowledge about the system under investigation. Then stationary and kinetic properties can be extracted or experimental observables calculated. The contributions from individual transitions to the overall transition between two relevant states can furthermore be investigated with transition path theory (TPT) [121, 122].

Following this modeling pipeline, it is possible to study the metastable sets of a molecule simulated on the computer, the processes mediating between them as well as further stationary and kinetic properties of the system.

III.3 THE MOLECULE

The research community has long been interested in alanine-based peptides [123–132]. Alanine is one of the nonessential amino acids used to form proteins and exhibiting alpha helical behavior. Short alanine-based peptides were early adapted as toy models by the simulation community as they are short enough to be simulated in full-atom detail, while they exhibit non-trivial folding behavior. They can therefore be used to study folding mechanisms. Here we use deca-alanine, a homopeptide with ten alanine residues, which has been studied extensively in vacuo [102, 133–137] as well as in implicit and explicit solvent [108–115] numerically. The structural formula is depicted in (a) of Figure 3. The beginning on the left hand side is called N-terminus, while the end of the chain on the right hand side is called C-terminus. The repeating -N-C-C- structure in the middle is called the backbone of the peptide, while the middle carbon atoms in each of the ten residuals are called C_α atoms respectively. The backbone is visualized as a band in (c) of Figure 3 and emphasizes the configuration of the molecule. Deca-alanine folds into an alpha-helix which is stabilized by six native hydrogen bonds forming between the O of the i-th residue (acting as acceptor) and the hydrogen of the $(i+4)$th N (acting as donor).

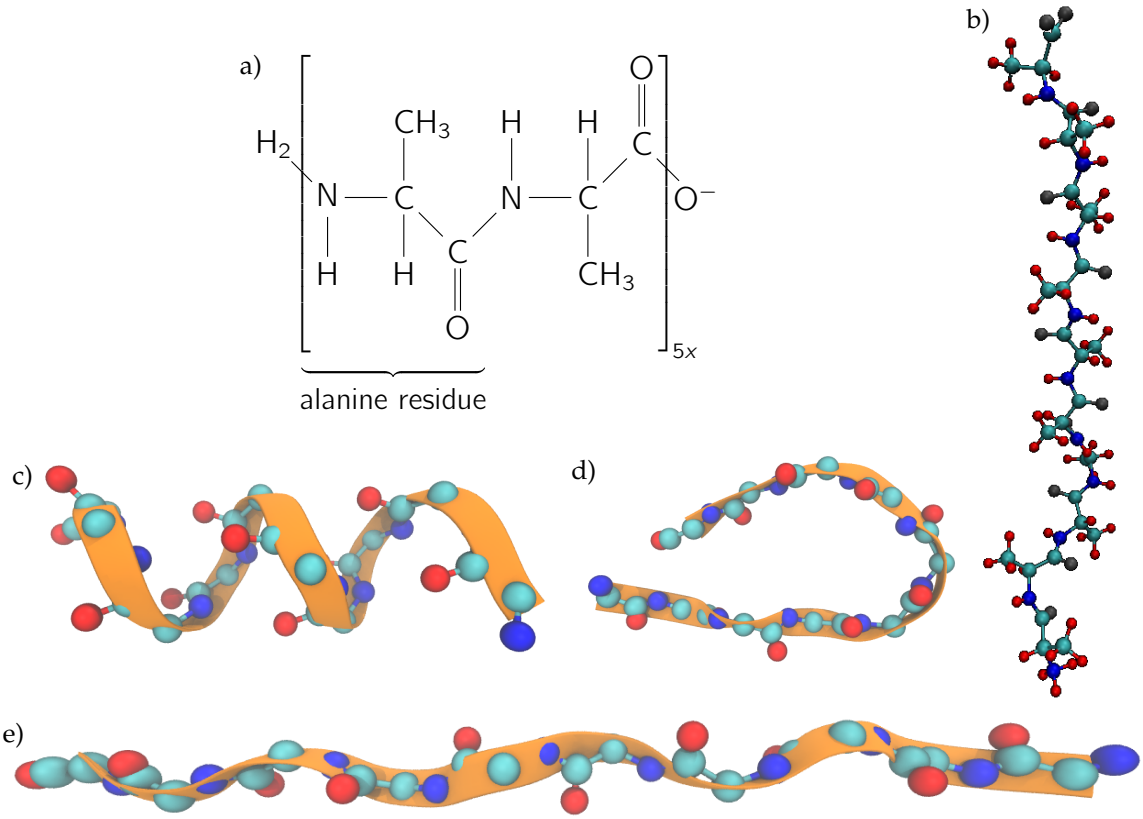


Figure 3: Various representations of deca-alanine and a selection of its molecule conformations. (a) Structural formula of deca-alanine, consisting of ten alanine residues with N-terminus to the left and C-terminus to the right. (b) Depiction of extended deca-alanine (in a similar style as encountered in [116]). (c-e) Depictions of selected configurations of deca-alanine with an orange band emphasizing the shape of the backbone: (c) Fully folded alpha-helix stabilized by native hydrogen bonds. (d) Misfolded beta-hairpin, stabilized by non-native hydrogen bonds. (e) Extended configuration belonging to the diffusive, unfolded conformation. ((c-e) adapted from our publication [63].)

III.4 MOLECULAR DYNAMICS SIMULATION

When running an all-atoms molecular dynamics simulation, Newton's equations are solved for N atoms numerically and the system is evolved for a small integration time step Δt repeatedly. In this way, high-dimensional data is produced in the form of 3N-dimensional snapshots. These snapshots, denoted as $X_t = \{\mathbf{r}_k(t)\}_{k=1}^{N}$, are comprised of the positions $\mathbf{r}_k(t)$ of all N atoms at time t.

To have an additional reference and baseline for the modeling, the data was taken from an earlier project in our group conducted by Fabian Knoch [116]. For completeness we replicate the simulation parameters here. The Gromacs 5.1.2 [117] software package was utilized for the molecular dynamics simulations of deca-alanine. An integration time step Δt of 2 fs was used alongside a temperature of $T = 300\,K$ in an NPT ensemble. To regulate the temperature, the velocity-rescaling thermostat [138] was employed with a τ_T value of 1 ps. The Parrinello–Rahman baro-

stat [139] was also used to adjust pressure with a τ_p value of 2 ps and a compressibility of $4.5 \times 10^{-5}\,\mathrm{bar}^{-1}$. The system's coordinates were recorded every 5000 integration time steps, equating to 10 ps per time step recorded in the data. Short range interactions were cut-off at a distance of 1.0 nm and covalent bonds that consisted of hydrogen atoms were controlled by means of the LINCS [140] algorithm. Long range electrostatic interactions were determined using the particle mesh Ewald summation technique, employing cubic interpolation and a Fourier grid spacing of 0.16 nm. The CHARMM22/CMAP39 force field [141] was utilized to model molecular interactions, with TIP3P40 water used for solvent interactions. Simulations were executed in a cubic container, subjected to periodic boundary conditions with a side-length of 7.2 nm, containing approximately 12,000 water molecules as solvent.
To cover a wider range of conformations, force-bias simulations were conducted using four distinct magnitudes of the pulling force, $f = \{0, 5, 10, 15\}\,\mathrm{kJ\,mol^{-1}\,nm^{-1}}$. The pulling force acted upon the center of mass of both the N-terminus and the C-terminus of the peptide, and the resulting distance defined the end-to-end distance. 80 independent simulations were conducted for $f = 0$ with 48 more simulations being run for each other applied force value. The simulations all originated from random configurations sampled from a 50 ns trajectory. We excluded the first 2 ns of each trajectory to guarantee equilibration, and were eventually left with $1.7 - 3.9\,\mu s$ of trajectory data per force, amounting to 11.3 µs in total.

III.5 DESCRIPTORS

The number of dimensions required to describe a molecule's atomistic configurations increases with the number of atoms it contains. High-dimensional data is often sparse, however, making it difficult to find and exploit correlations between different aspects of the dataset. This is known as the 'curse of dimensionality' (see subsection II.3.3.1). To address this issue, dimensionality reduction is an essential step in data processing pipelines. It involves projecting the data into a lower dimensional space that can be handled by algorithms, while retaining as much information as possible. Additionally, the physical symmetries of the molecular configuration, such as translation, rotation, and (potentially) permutations of indistinguishable atoms, are not inherent in the representation of the molecule in Cartesian coordinates.
To mitigate this, a first dimensionality reduction step is carried out by constructing structural descriptors, which are maps from the atomistic trajectory X_t to a time series $\mathbf{x}_t$ in a lower dimensional space $\mathbb{R}^n$ that respects the aforementioned symmetries, with $n \ll 3N$. The selection of a structural descriptor constitutes a decision by the modeler, often necessitating domain-specific knowledge. Common structural descriptors encompass pairwise distances, torsion angles and their combinations and variations.

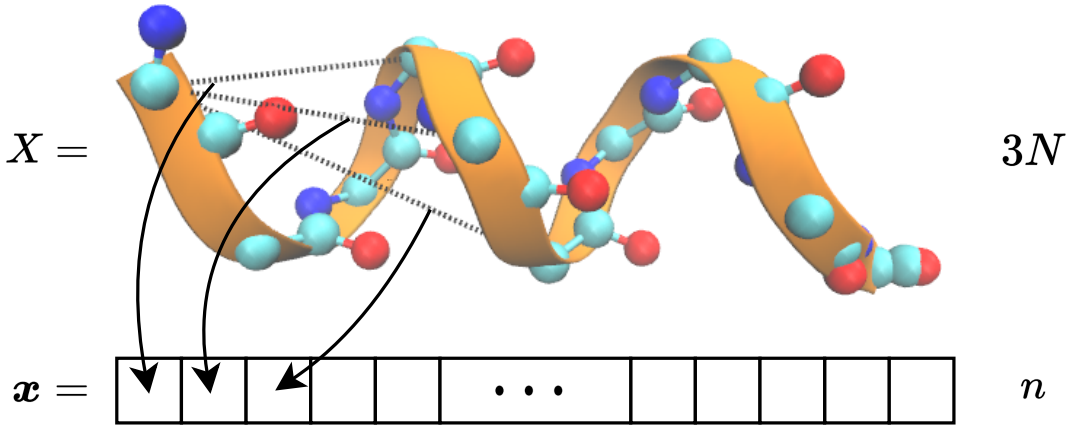


Figure 4: Depiction of how pairwise distances between atoms (of a configuration X) are used to build up a structural descriptor $\mathbf{x}$. (Adapted from our publication [63].)

In the course of our studies, we concern ourselves with three different descriptors – namely pairwise distances, torsion angles and H-bonds – which are described in the following. In the first half of our analysis, we investigate the low-dimensional latent spaces of all these descriptors to show that EncoderMap is able to construct sensible latent spaces for a wide range of different descriptors. In the second part of our analysis, we commit ourselves to the first of these (the pairwise distances descriptor with H-bonds) to be able to compare the performances between the different dimensionality reduction methods.

III.5.1 PAIRWISE DISTANCES WITH H-BONDS

In this descriptor, we consider all pair distances of C_α atoms in the backbone separated by at least three amino-acid residues, as the others are highly correlated. In order to incorporate the influence of the six native hydrogen bonds that contribute to the stabilization of alpha-helical structures, we additionally introduce their distances between each C=O and their corresponding N-H group four residues apart. This results in a total of $n = 6 + \sum_{i=1}^{10-3} i = 34$ entries (i.e. dimensions) for our structural descriptor $\mathbf{x}$, which captures the main features of protein backbone geometry. The resulting descriptor is illustrated in Figure 4.

III.5.2 TORSION ANGLES

Torsion angles are another translationally and rotationally invariant way to describe the geometry of a protein backbone. Let ABCD be four consecutive atoms in a protein backbone. As three points in space define a plane, we can form planes out of ABC and BCD. These two planes intersect with an angle which is called a torsion

angle as it describes how much the bond between $B - C$ is twisted. There are two different torsion angles in the backbone of peptide chains (also called Ramachandran angles): The torsion angle between $N - C_\alpha$ called Φ and the torsion angle between $C_\alpha - C$ called Ψ (compare the structural formula in (a) of Figure 3). These angles fully capture the geometry of the backbone. The resulting torsion angles descriptor consists of all the Φ/Ψ torsion angles of the backbone of deca-alanine, giving $n = 2(10 - 1) = 18$ dimensions.

III.5.3 H-BONDS

Since we know that stable structures tend to be stabilized by hydrogen bonds, we experimented with building a descriptor out of all pairwise distances between potential hydrogen bonds as an unsupervised approach. We find that these descriptors quickly grow to high dimensions with high correlation between entries, while only being sensitive to stabilized structures and not being able to distinguish between unstabilized structures well. Because of these reasons we abandon this approach, but we still present some of the results here for completeness. There exist several conventions for describing hydrogen bonds, for more information see section III.7. Here, we take all the possible distances between the O atoms (acceptor) and the hydrogen of the N atoms (donor) at least one residue apart. The H-bonds descriptor ends up with $n = 72$ dimensions.

III.6 DIMENSIONALITY REDUCTION

The dimensionality reduction achieved by utilizing a descriptor $\mathbf{x}$ is insufficient for bigger molecules or complex data and additional dimensionality reduction $\mathbf{x} \mapsto \xi \in \mathcal{L}$ is performed using dedicated algorithms, mapping from descriptor space into latent space $\mathcal{L}$. Numerous techniques have been proposed for this purpose. In this study, we compare TICA [55–57], a linear method which is widespread for its ability to optimally approximate the Markov transfer operator, with EncoderMap [60], a non-linear method using an autoencoder with an additional distance metric (see also subsection II.3.3). Another difference between these two methods is that TICA exploits the temporal information of the time series, while EncoderMap uses structural information only. This makes EncoderMap both more flexible and applicable on a larger class of data, when no such information is available.
We choose the latent space dimension as $d = 2$ for ease of visualization, but emphasize that the model can always be improved by considering more dimensions. To take all of the data, which was harvested in different ensembles, into account, we reweigh the data from the biased ensembles via the MBAR method (subsection II.4.1) to establish an unbiased estimate. To get insight into the latent spaces $\mathcal{L} = \mathbb{R}^d$ of

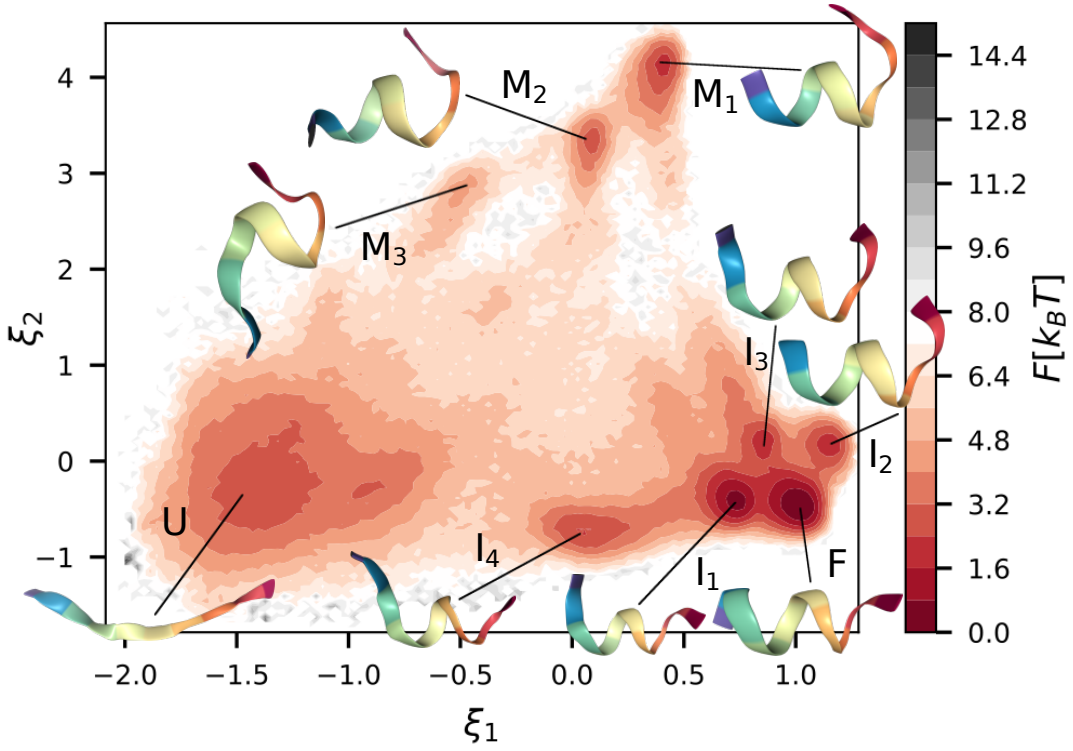


Figure 5: Free energy landscape $F(\xi)$ of the pairwise distances descriptor in TICA latent space $\mathcal{L}_{\text{TICA}}$ with selected average structures. The first two independent components of TICA display several local minima, with the global minimum corresponding to the helix conformation F. The slowest independent component ξ_1 describes the transition from an unfolded (U) to a folded (F) helical conformation, while the second slowest independent component ξ_2 allows to distinguish between intermediary (I) and misfolded (M) states. (Adapted from our publication [63].)

reduced dimension, we plot the free energy landscapes $F(\xi) = -k_B T \ln(\mu(\xi))$ with stationary probability distribution $\mu(\xi)$. This is done by binning the data in the latent space $\mathcal{L}$ into a 120×120 regular bin grid and calculating the free energy value for each of the bins. As only free energy differences are meaningful, we choose the global reference value $F_{\text{ref}} = 0$ such that the most probable bin has a value of zero.

Simulations can be thought of as the system exploring the free energy landscape. Free energy minima then correspond to more probable states, in which unbiased simulations spend more time and which therefore are candidates for metastable states. To understand which structures of the molecule belong to which part of the free energy landscape, average structures are determined for each bin. This is done by aligning all structures in a selected bin and then calculating the average position of each atom. As long as the structures in the bin are relatively homogeneous, the average structure is a good representation of the conformation belonging to the free energy minimum. This homogeneity is especially found in narrow and deep free energy minima, as structures are especially similar to each other in this case.

III.6.1 PAIRWISE DISTANCES DESCRIPTOR SPACE

We start by investigating the latent space generated by applying additional dimensionality reduction to the 34-dimensional pairwise distances descriptor with additional H-bonds described in subsection III.5.1. We will call this descriptor 'pairwise distances descriptor' for brevity. Although we show latent spaces of other descriptors in the following subchapters, we stay with this descriptor afterwards for the rest of this work, as it was also used in [116] and therefore gives us an additional reference, we can compare our results to.

III.6.2 TICA

We project our data into a 2-dimensional latent space $\mathcal{L}_{\text{TICA}} \subset \mathbb{R}^2$ created with the TICA method. TICA is one of the most widely used techniques for dimensionality reduction in this context. It is a linear method that finds the directions, in which the slowest processes occur (see paragraph II.3.3.2). We fit the TICA transformation only on the data from the unbiased ensemble $f = 0$, as the bias skews the timescales, but then project all the data from all ensembles into this latent space.
The resulting free energy landscape in TICA space is visualized in Figure 5 with selected average structures. The first coordinate ξ_1 represents the direction of the slowest process, which is the transition from the unfolded state U to the fully folded state F stabilized by native hydrogen bonds. As the unfolded state comprises a lot of different structures, the free energy minimum is spread out widely, implying a wide heterogeneity of states. In contrast, the free energy minima belonging to more folded states exhibit more narrow minima implying more similar configurations. The second coordinate ξ_2 describes the second slowest process and allows to differentiate between intermediary states I ($\xi_2 \approx 0$), giving rise to different folding pathways and misfolded states M ($\xi_2 > 2$), which disrupt the folding process by trapping the molecule in a stabilized metastable state with high free energy barriers.

III.6.3 ENCODERMAP

We continue with investigating the 2-dimensional latent space of EncoderMap $\mathcal{L}_{\text{EM}} \subset \mathbb{R}^2$. EncoderMap is a non-linear method using an autoencoder with a distance metric, such that close points in descriptor space (similar structures) are close to each other in the latent space (see paragraph II.3.3.3). The neural network is trained on all the data from all ensembles. This is possible since EncoderMap only takes into account structural information and does not make use of the time-series character of the data.

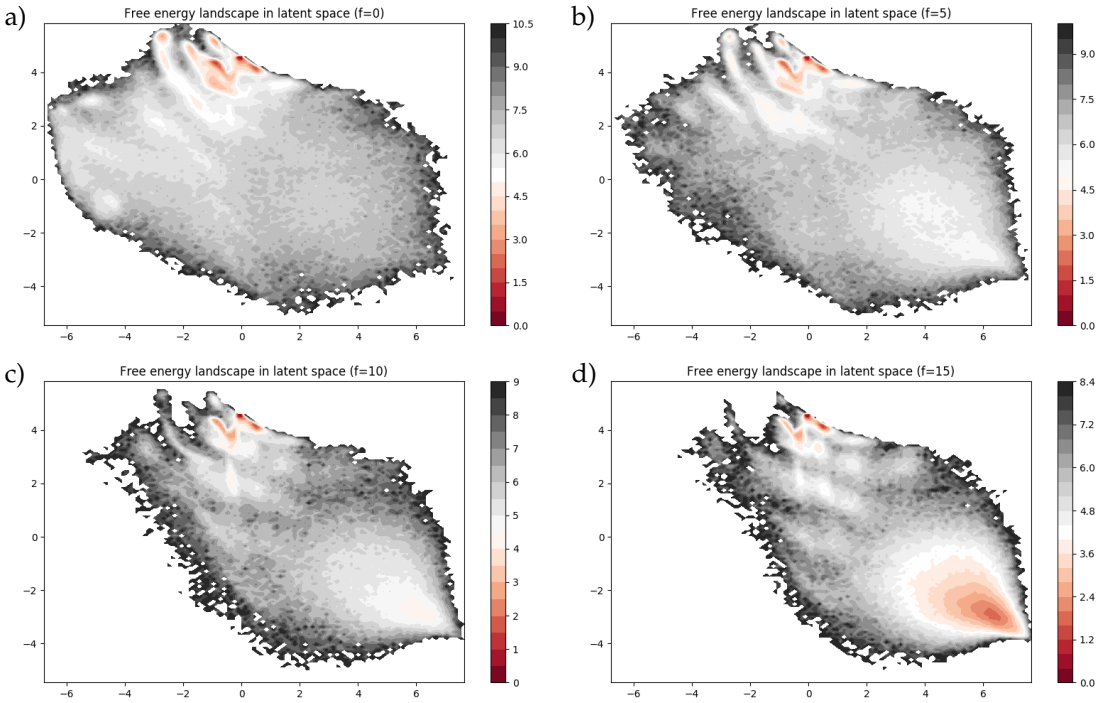


Figure 6: EncoderMap MBAR ensembles with different biasing forces f applied, pulling at the ends of the molecule: (a) $f = 0$ (unbiased), (b) $f = 5$, (c) $f = 10$, and (d) $f = 15$ with units of $\mathrm{kJ\,mol^{-1}\,nm^{-1}}$ respectively. With increasing pulling force f the folded states (areas of low free energy at the top center of the latent space) become less probable, while the extended states (lower right of the latent space) become more probable (compare also Figure 7). Some areas of rare misfolded states (left of the latent space) are not explored any more in ensembles of higher biasing forces.

We start by visualizing the free energy landscapes of each of the different ensembles separately *without* reweighting in Figure 6. In different ensembles different parts of the latent space are explored and the probability weight changes with increasing biasing force. When using MBAR (see also subsection II.4.1) to combine this data, it is important to have overlap between the ensemble of interest (the unbiased one in our case) with the ensembles used to improve the free energy estimates (the ones with biasing force $f > 0$), as estimates are only affected and improved in areas with overlap. The combined and reweighted unbiased free energy landscape is shown in Figure 7 with selected average structures. When comparing the free energies in the different ensembles (Figure 6) with the combined free energy and the average structures in Figure 7, we find as expected that for increasing biasing force strength f, which stretches the molecule, the folded states (in the upper part of the figure) become more and more improbable, while stretched unfolded states (in the bottom right part of the figure) become more frequent.

It is important to note that as for EncoderMap only distances are important, the absolute orientation and absolute values do not carry information in the latent space. More precisely, this means that repeating the dimensionality reduction with Enco-

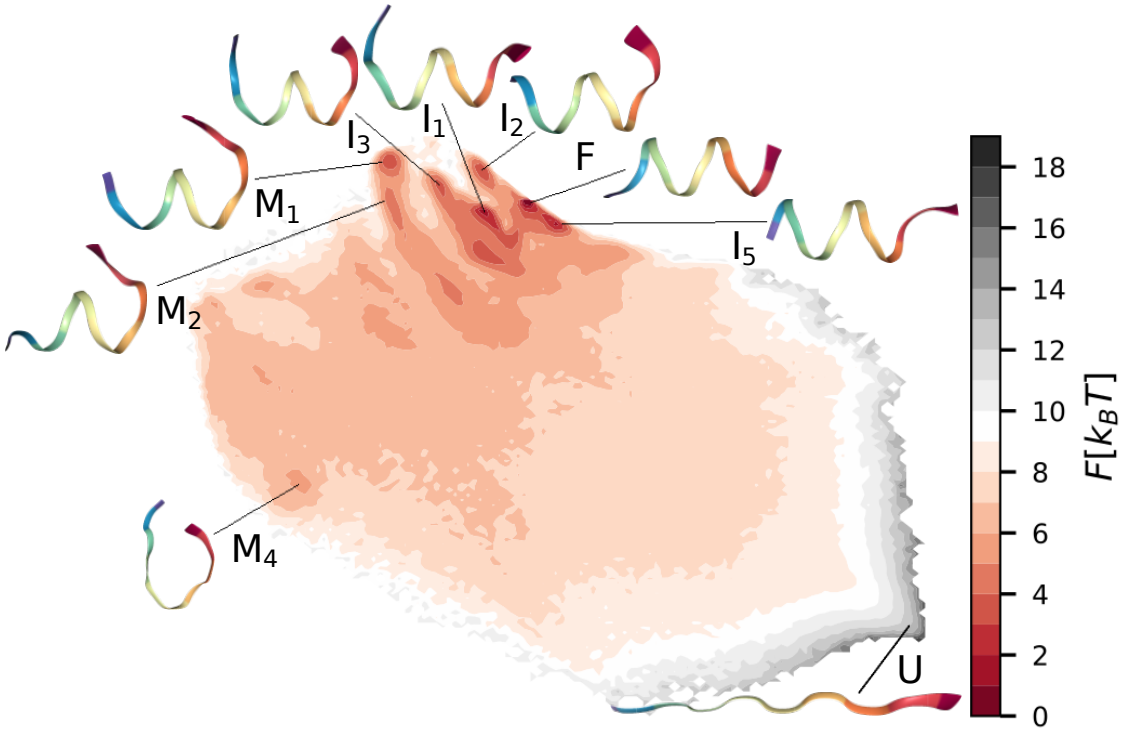


Figure 7: Free energy landscape $F(\xi)$ of the pairwise distances descriptor in EncoderMap latent space $\mathcal{L}_{\mathrm{EM}}$ with selected average structures. The distance metric enforces a latent space, in which similar structures are close to each other. As EncoderMap is only sensitive to relative distances, the orientation and absolute coordinates do not carry information and we omit the axes. To visualize the location of structures in the latent space, representative average structures of selected bins are shown, including the folded conformation (F) and an unfolded extended conformation (U).

derMap for serveral times leads to rotated and translated versions of the data in the latent space. Because of that the axes do not carry meaning and we leave out the axes in Figure 7 to reduce clutter. The unfolded states are found in the lower right of the latent space, while the increasingly folded states are found in the upper left of the space. Most of the misfolded M and intermediary states I are situated in the middle upper part of latent space, close to the fully folded state F. The plot indicates that the basins representing folded, intermediate, and misfolded conformations are found in the middle upper area of the plot, whereas the diffusive extended configurations occupy the lower right region. It is worth noting that there is no minimum in the free energy landscape for the extended unfolded configurations. Although we find most of the conformations from TICA (compare Figure 5), we do not find local minima corresponding to I_4 and M_3. Instead, two new minima emerge, which we designate as I_5 (near-helical conformation) and M_4 (beta-hairpin residing in a relatively flat minimum).

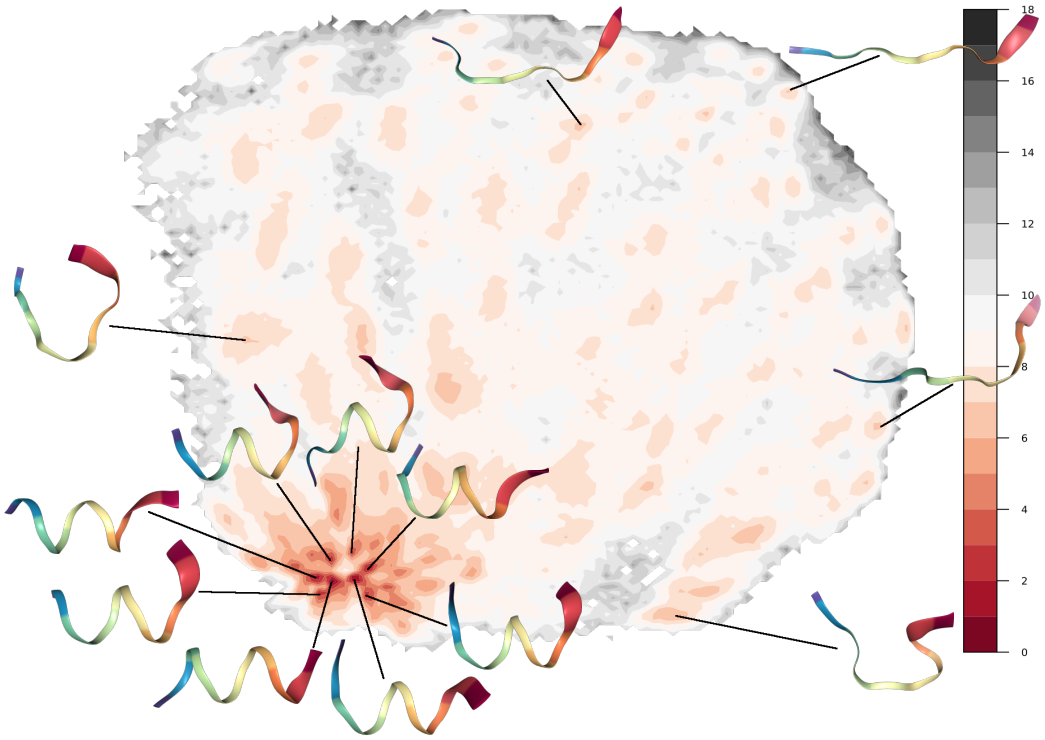


Figure 8: Free energy landscape $F(\xi)$ of the torsion angles descriptor in EncoderMap latent space with selected average structures. The distance metric enforces a latent space, in which similar structures are close to each other. As EncoderMap is only sensitive to relative distances, the orientation and absolute coordinates do not carry information and we omit the axes. To visualize the location of structures in the latent space, representative average structures of selected bins are shown. The conformations found are similar to the ones found for the pairwise distances descriptor in Figure 7. This shows that EncoderMap is able to build a meaningful latent space independent of the descriptor used, as long as the descriptor captures the structural information of the molecule.

III.6.3.1 Torsion angle descriptor

In Figure 8 the free energy landscape of the torsion angles descriptor (see subsection III.5.2) in 2-dimensional EncoderMap latent space is shown. The data of all ensembles was again used by reweighting with MBAR (subsection II.4.1). Compared to the pairwise distances descriptor in Figure 7, we find a lot of shallow minima which appear to belong to different unfolded configurations. The torsion angles descriptor therefore is better able to distinguish between unfolded configurations. The more folded configurations are situated in the bottom left of the latent space and exhibit similar conformations as the pairwise distances descriptor. Although we do not continue with this descriptor, the free energy landscape shows that EncoderMap is able to find a meaningful low-dimensional projection that distinguishes between important metastable states. It certainly would be interesting to repeat the analysis conducted in this work with the torsion angles descriptor and compare the performance of the two.

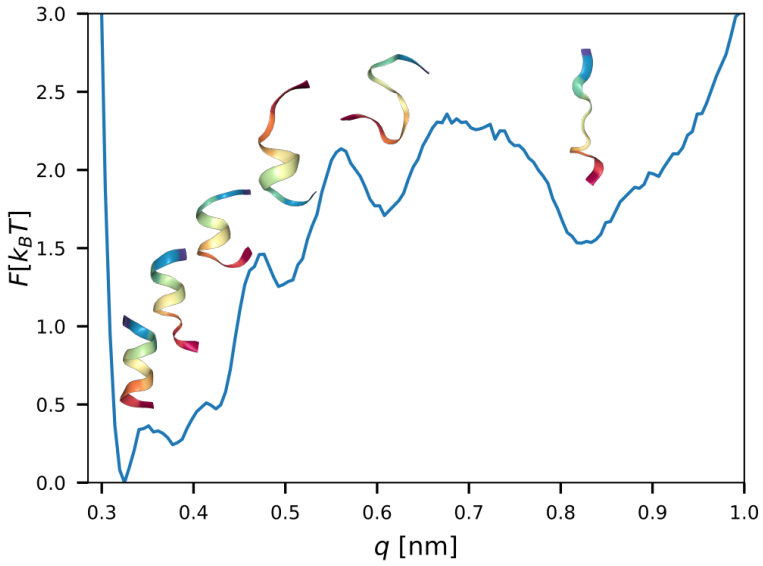


Figure 9: Free energy landscape of mean native hydrogen bond distances q with representative configurations of minima. Configurations are similar in bins with low q and become increasingly heterogeneous in the bins with high q. (Adapted from our publication [63].)

III.6.3.2 H-bond descriptors

As H-bonds stabilize deca-alanine and therefore are responsible for the metastable conformations, we shortly investigate some H-bond descriptors in this section, which we do not pursue further later on.

Scalar native H-bonds descriptor

We examine the mean distance

$$X \mapsto q = \frac{1}{6} \sum_{i=1}^{6} |\mathbf{x}_i^O - \mathbf{x}_{i+4}^N| \tag{III.1}$$

of the six native hydrogen bonds stabilizing the alpha helix of deca-alanine as a naive scalar order parameter (see also [142]). These hydrogen bonds occur between the acceptor oxygen at $\mathbf{x}_i^O(t)$ in residue i and the hydrogen of the donor nitrogen at $\mathbf{x}_{i+4}^N(t)$ four residues apart (compare the structural formula in Figure 3). As more native hydrogen bonds are formed, we expect a smaller value for $q(t)$ such that q is able to differentiate between different stages of folding.
Figure 9 displays the free energy landscape along q with representative structures corresponding to the respective free energy minima. As q is highly sensitive to most native hydrogen bonds closing, configurations in the bins corresponding to low-q free energy minima are very similar. The order parameter q thus distinguishes metastable states for low values of q. The absolute minimum of the free energy is

situated around $q_{\text{helix}} \approx 0.32\,\text{nm}$, which corresponds to the conformation of the fully folded alpha-helix with all native hydrogen bonds closed. This matches the literature value for mean donor-acceptor distances of closed hydrogen bonds of approximately $0.3\,\text{nm}$ [143].
As q is symmetric under permutation of native bond distances, it cannot distinguish between different native bonds. Consequently, structures with one broken native bond include configurations, where the helix is opened at either end. As q increases, the configurations in the bins become more heterogeneous and q is not able to distinguish between them. This is visualized in Figure 10(b), where we color configurations in TICA space based on their standard deviation of q values that provides a measure of configuration similarity within each bin. For the region containing folded, intermediate and misfolded conformations (high ξ_1), bins have similar configurations, while structures in bins belonging to diffusive conformations (low ξ_1) are strongly heterogeneous. Figure 10(a) shows that q does not distinguish between different minima of the free energy landscape apart from being minimal for the folded conformation. As such, q is a coarse order parameter that is sensitive to the folded structure but cannot further distinguish conformations of deca-alanine.

H-bonds descriptor

In Figure 11 the free energy landscape of the H-bonds descriptor (see subsection III.5.3) is visualized after projection down to $d = 2$ dimensions with EncoderMap. Although we abandoned this descriptor later on, we show its free energy landscape here to support the statement that EncoderMap is able to construct sensible latent spaces for different descriptors. As before, all ensembles are used and combined with MBAR to construct the free energy landscape and average structures are shown for selected free energy minima by averaging configurations over the respective bins. Most of the conformations found, again, are similar to the ones found in the pairwise descriptor case (compare Figure 7) and the torsion angle descriptor (compare Figure 8), implying that the important (stable) conformations are independent of the choice of descriptor, as is expected. By construction, the H-bonds descriptor is mainly sensitive to H-bonds so the different average structures reflect structures with different H-bonds being closed and broken. We investigate this further in the following section III.7.

III.6.4 TICA-4D-E2E

As we have a complete, published analysis of the data at our disposal [116], we compare the modeling in this previous publication with the modeling performed in our work. The publication was authored by Fabian Knoch, a former member of our research group, who for this publication also harvested the data used in this work. In his modeling, he uses the same pairwise distances descriptor with native H-bonds (as

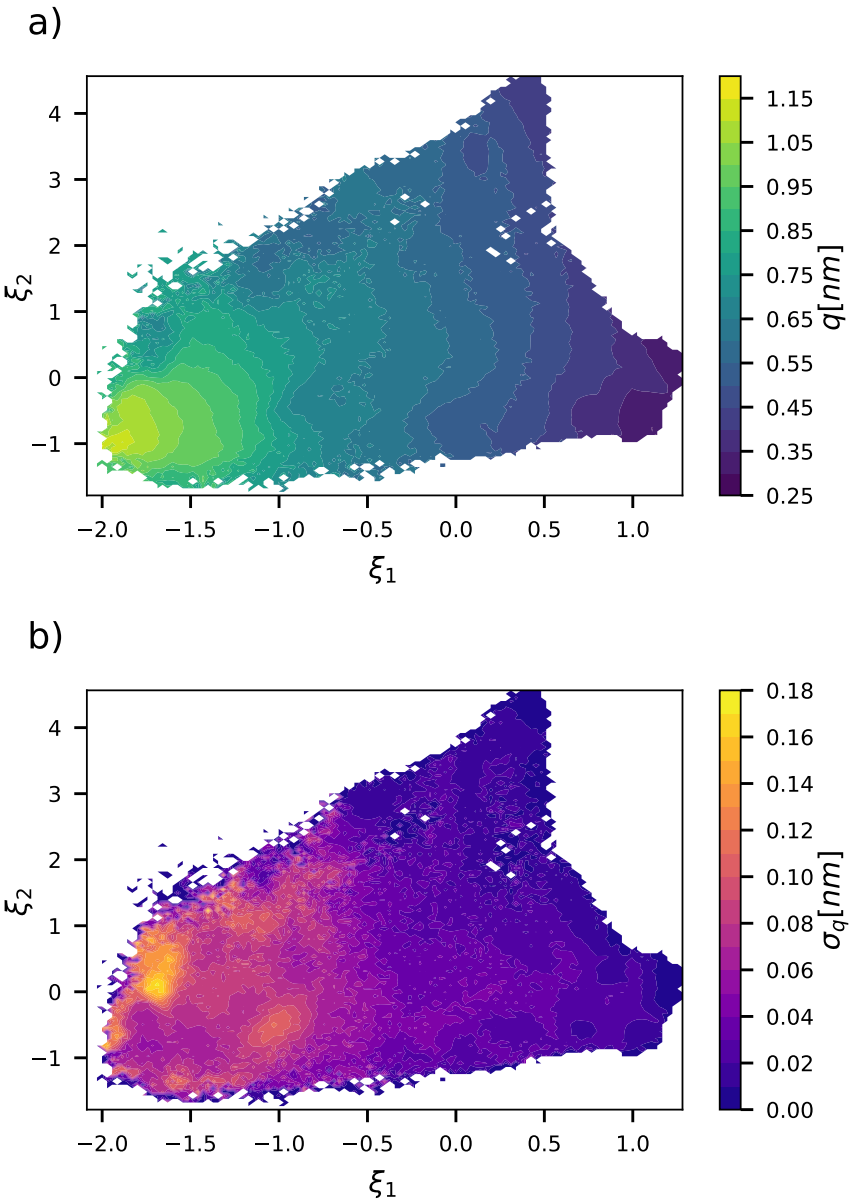


Figure 10: (a) TICA space colored by mean q values of the bins. The q values decrease along each TICA coordinate, indicating a transition towards partially folded conformations. (b) TICA space colored by standard deviations of q values in the bins. Partially folded regions (high ξ_1) exhibit similar configurations, while the bins in the unfolded regions (middle to low ξ_1) are highly heterogeneous. (Adapted from our publication [63].)

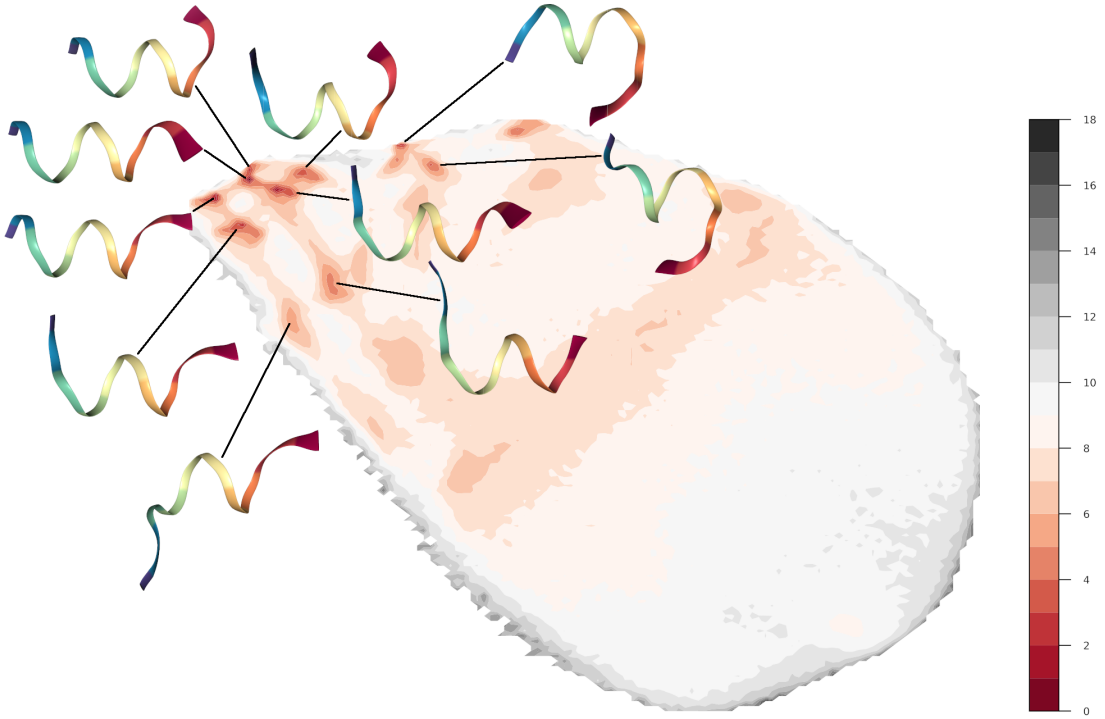


Figure 11: Free energy landscape $F(\xi)$ of the H-bonds descriptor in EncoderMap latent space with selected average structures. As before, due to the distance metric, similar structures are close to each other in the latent space and we omit the axes as they do not carry meaning. To visualize the location of structures in the latent space, representative average structures of selected bins are shown. The conformations found are similar to the ones found for the pairwise distances descriptor in Figure 7. This shows that EncoderMap is able to build a meaningful latent space independent of the descriptor used, as long as the descriptor captures the structural information of the molecule. Since the H-bonds descriptor is only sensitive to the H-bonds in this case, the latent space represents structures stabilized by H-bonds very well, while it is not able to distinguish other conformations.

described in subsection III.5.1) and a latent space of four TICA components with the end-to-end distances of the molecule as an additional dimension. The end-to-end distance contains additional information about the extension of the configurations, but generally has been found to be an insufficient coordinate for modeling. We call this 5-dimensional latent space – four TICA components plus end-to-end distance – the TICA-4D-E2E space. Our exposure to the data indicates that here the TICA algorithm was trained on the biased ensemble with $f = 10$, which only leads to small differences in the first two TICA coordinates compared to our TICA space. As the space is 5-dimensional, it can not be visualized completely, but the first two dimensions closely resemble Figure 5 (compare [116]).

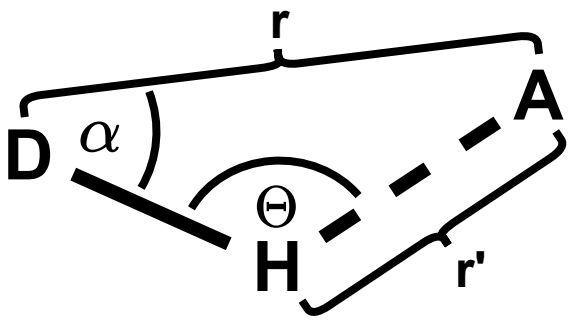


Figure 12: Visualization of a generic H-bond with donor atom (D), acceptor atom (A), hydrogen atom (H) and typical observables used to study bond formation.

III.7 EXCURSUS: H-BONDS

As the (native) H-bonds are the strongest stabilizers of the configurational structures, we investigate them in more detail. Determining which (native) hydrogen bonds are closed, can be a strong predictor of structure and gives qualitative insight into the distribution of configurations in the latent space.

III.7.1 CRITERIA FOR CLOSED H-BONDS

Because of the stabilizing nature of H-bonds, they have been investigated for a long time and represent a research field in its own right. There have been numerous studies to investigate the H-bond distributions in proteins [143, 144] and to determine suitable criteria for detecting H-bond formation.
The general structure of an H-bond is visualized in Figure 12: A hydrogen atom is in a covalent bond with a more electronegative donor atom (D). This shifts the densities of the covalent electrons to the donor atom and leads to a partial positive charge of the hydrogen. This partial positive charge can attract a negative partial charge of another more electronegative acceptor atom (A) with lone electrons forming a hydrogen bond. There have been indications that the interaction is not solely electrostatic in nature [145]. H-bonds occur in many scenarios with different variations, therefore the official definition is even more involved [146]. Donor and acceptor atoms are often atoms of the second period (e.g. N, O, F) because of their high electronegativity. The criteria for bond formation often involve the donor-acceptor-distance r or the bond length r' as well as the acceptor-donor-hydrogen angle α or the donor-hydrogen-acceptor angle θ.
Several criteria encompass

- General criteria:
  - GROMACS convention: $r \leqslant 0.35\text{nm}, \quad \alpha \leqslant 30°$
  - AMBER convention: $r \leqslant 0.3\text{nm}, \quad \theta \geqslant 135°$
  - Baker-Hubbard criterion [147]: $r' < 0.25\text{nm}, \quad \theta > 120°$
- More specific criteria applicable to helix formation in deca-alanine:
  - Study about folding of polyalanine [148, 149]:
    H-Bond fully intact $\Leftrightarrow r' < 0.18\text{nm}, \quad \theta \geqslant 120°$
  - $C=O\cdots N$ H-Bond from crystals [150]: $r \leqslant 0.33\text{nm}, \quad \theta \geqslant 135°$
  - Study about non-covalent interactions in the backbones of proteins [151, 152]: $r' < 0.244\text{nm}, \quad \theta \geqslant 90°$

Although several of the different criteria define bond formation in a comparable regime, the criteria differ from each other, which leads to different statements down the road about which hydrogen bonds are formed or not. Naturally, a certain loss of information is to be expected as these criteria turn a distribution into a binary statement. We therefore expect these criteria only to give a tendency and hence a qualitative picture of the hydrogen bonds formed.
In Figure 13 we visualize the distribution of donor-acceptor distances r of the six native hydrogen bonds in our data, in Figure 14 the distribution of hydrogen-acceptor distances r' and in Figure 15 the distribution of donor-hydrogen-acceptor angles θ. We show (a) the distributions of the full data and (b) the distributions in the bin with the lowest free energy belonging to the conformation of the fully folded helix (compare Figure 7) for r, r' and θ respectively. Examining these plots, two things stand out. Firstly, the means of the distributions of native H-bonds 5 and 6 differ from the other native H-bonds. This observation is found in both, the full data distribution and the folded helix distribution. Secondly, because of that, none of the above-mentioned criteria appear to generally describe the bond formation of the native hydrogen bonds correctly, with a particular discrepancy for native H-bonds 5 and 6. It is not clear whether this is physical or an artefact of the force field used. Although the applied force field CHARM22/CMAP was finetuned with the help of alanine dipeptide to improve the description of folding of small peptides and showed better agreement with NMR measurements for folded proteins [153], it is also known to be biased towards alpha helical conformations [154]. By now, more recent force fields exist for (folded) proteins, which have been validated over a wide range of parameters [155, 156].
To get a more appropriate picture of which native H-bonds are formed, we deduce individual criteria for each of the native H-bonds i from data by using the mean distances and angles $\mu_{r,i}/\mu_{\theta,i}$ and the standard deviations $\sigma_{r,i}/\sigma_{\theta,i}$ of the fully folded helix distributions (b) respectively: H-bond i formed $\Longleftrightarrow r_i < \mu_{r,i} + 2\sigma_{r,i}, \quad \theta_i > \mu_{\theta,i} - 2\sigma_{\theta,i}$. To sum up our criterion, we consider a native H-bond closed when the

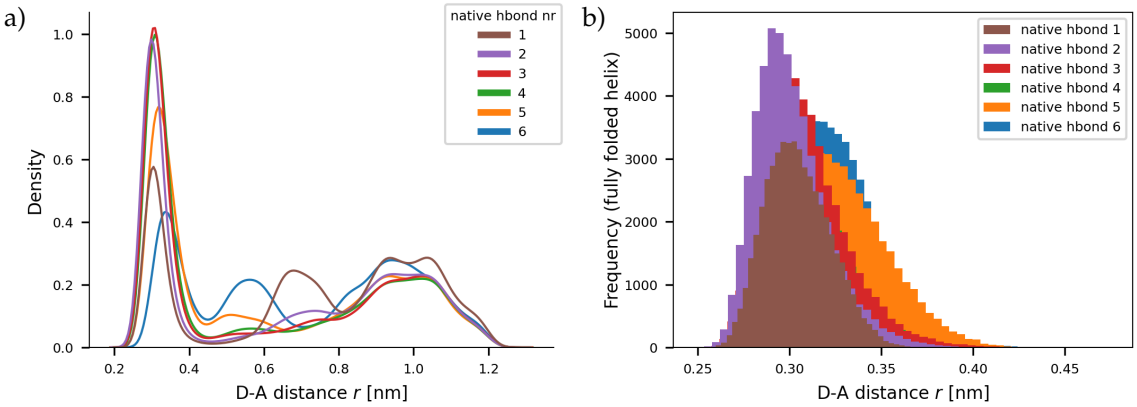


Figure 13: Distribution of donor-acceptor distances r of the six native hydrogen bonds for (a) the full dataset and (b) the bin with the fully folded helix structure.

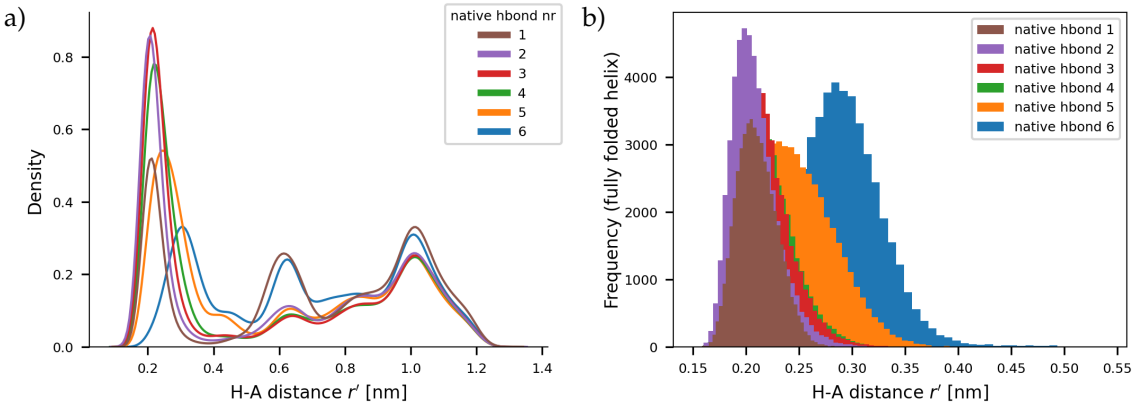


Figure 14: Distribution of hydrogen-acceptor distances r′ of the six native hydrogen bonds for (a) the full dataset and (b) the bin with the fully folded helix structure.

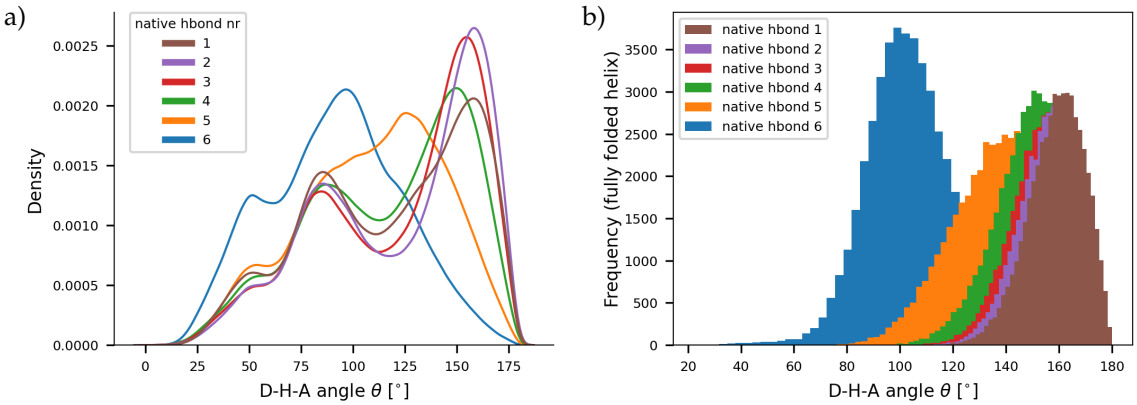


Figure 15: Distribution of donor-hydrogen-acceptor angles θ of the six native hydrogen bonds for (a) the full dataset and (b) the bin with the fully folded helix structure.

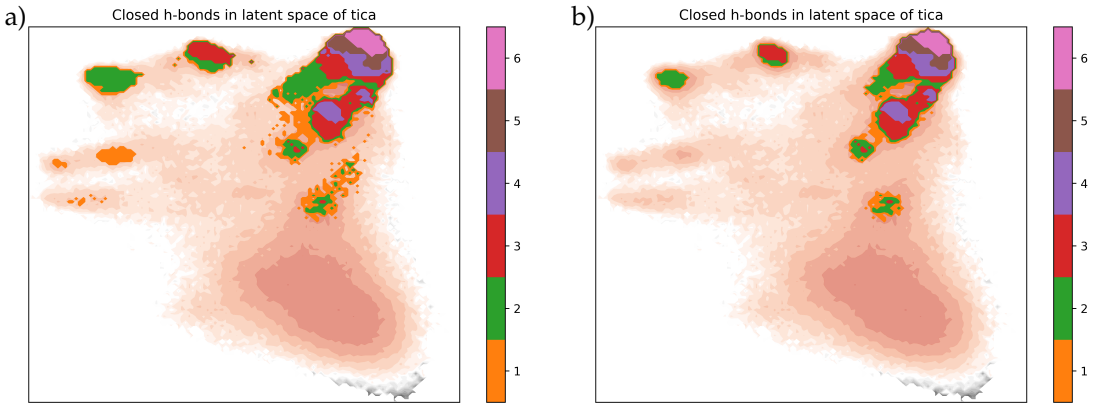


Figure 16: Sensitivity to parameters of distributions of closed H-bonds of pairwise descriptor in TICA latent space with (a) bin threshold of 30 and (b) bin threshold of 100. The colors represent the amount of native H-bonds closed of the structures corresponding to the respective bin in the latent space.

distance and angle are within 2 standard deviations of the fully folded helix structure distribution.

III.7.2 DISTRIBUTION OF CLOSED H-BONDS IN LATENT SPACE

Having established a sensible criterion for closed H-bonds from our data in the previous section, we can now explore the distribution of structures in the different latent spaces. For that, we take the free energy landscapes from section III.6, which were determined by binning the data into a regular 120×120 bin grid, and use these bins to look at the formed native H-bonds in each of the bins according to the aforedetermined criterion. To that end, we only take into account bins with at least 100 data points to ensure a minimum of statistics and consider a native H-bond closed for this bin when at least 60% of the configurations in this bin have this native H-bond closed according to the criterion. We acknowledge that these threshold values are somewhat arbitrary and have an effect on the details of the resulting distributions. The overall picture, however, of where which conformations are located in the latent spaces, stays the same so that these investigations can be used to get insight into the structure of the respective latent spaces.
To give an exemplary visualization of the sensitivity to these parameters, the resulting amount of native H-bonds closed in the TICA latent spaces are plotted in Figure 16 for (a) a bin threshold of 30 and (b) a bin threshold of 100. Lowering the bin threshold leads to bigger patches of closed H-bonds and new patches with low statistics appearing, as more bins are taken into account, while the overall picture of the distribution remains unchanged.

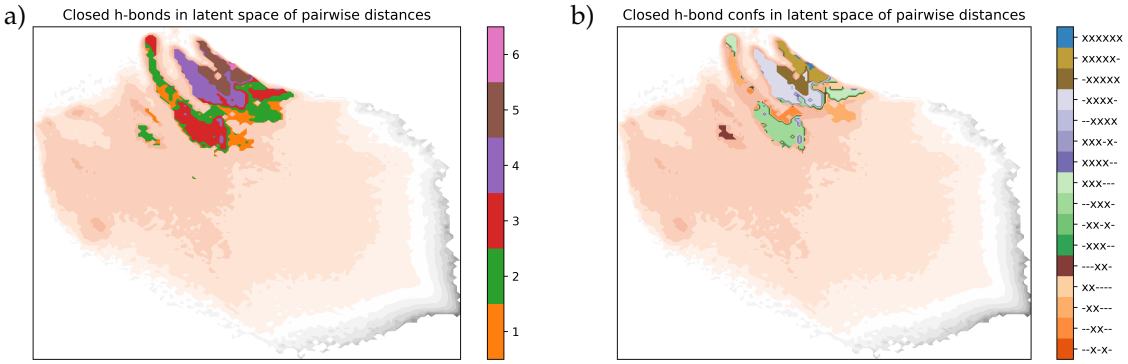


Figure 17: Distributions of closed native H-bonds of pairwise descriptor in EncoderMap latent space with (a) the amount of native H-bonds closed and (b) the specific native H-bonds closed ("x" closed, "-" open).

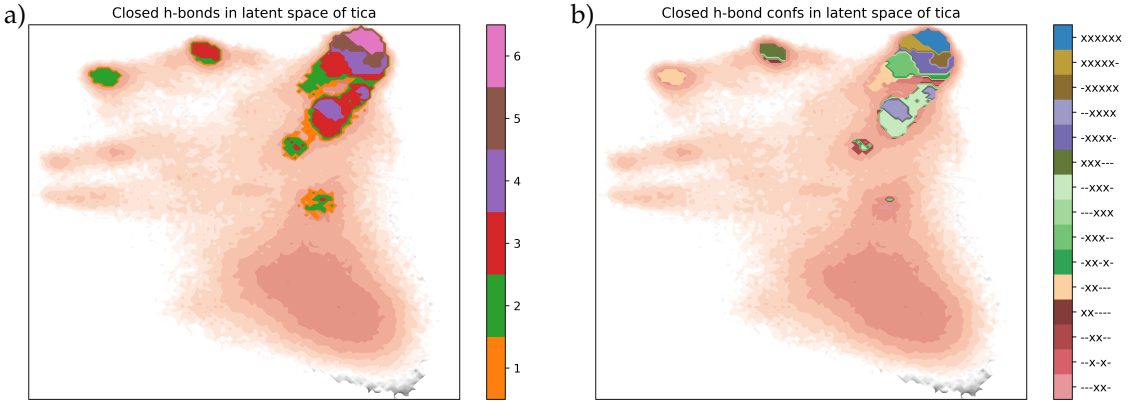


Figure 18: Distributions of closed native H-bonds of pairwise descriptor in TICA latent space with (a) the amount of native H-bonds closed and (b) the specific native H-bonds closed ("x" closed, "-" open).

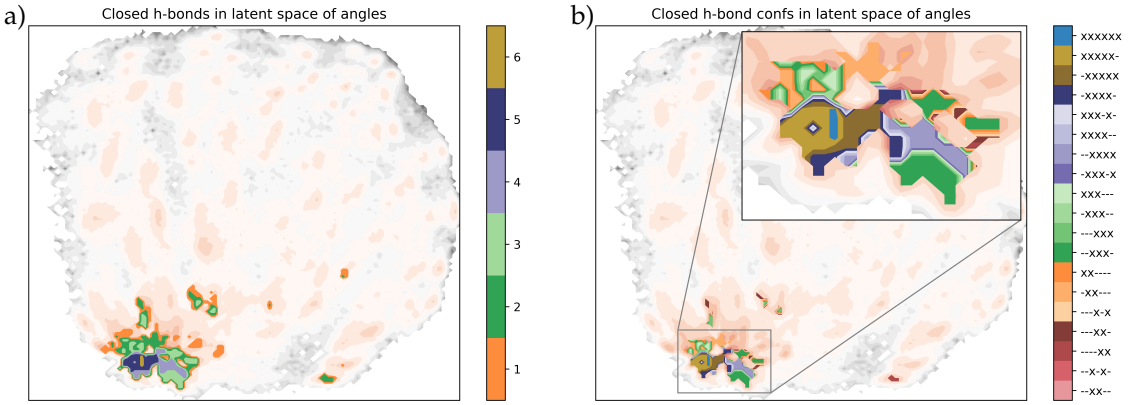


Figure 19: Distributions of closed native H-bonds of angles descriptor in EncoderMap latent space with (a) the amount of native H-bonds closed and (b) the specific native H-bonds closed ("x" closed, "-" open).

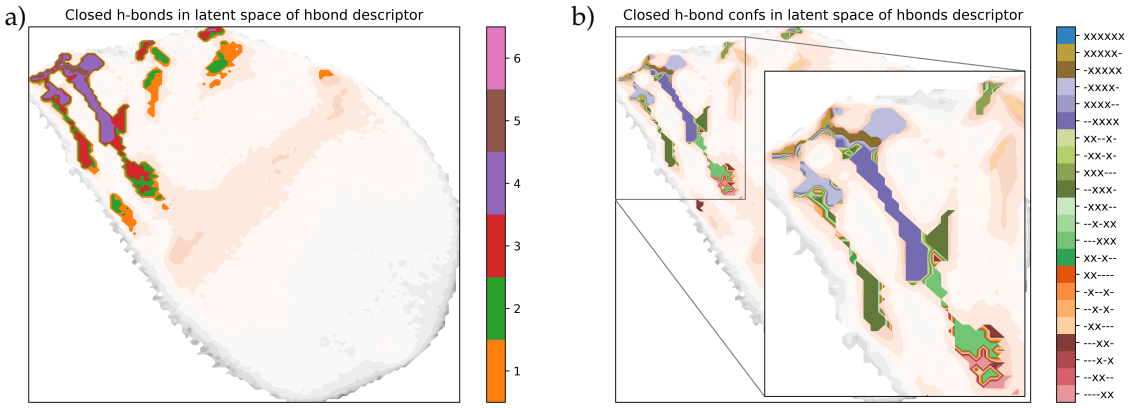


Figure 20: Distributions of closed native H-bonds of H-bonds descriptor in EncoderMap latent space with (a) the amount of native H-bonds closed and (b) the specific native H-bonds closed ("x" closed, "-" open).

To get an insight into how the structures are distributed in the respective latent spaces, we show where in the latent spaces (a) how many native H-bonds are closed and (b) which of the native H-bonds are closed for the pairwise distances descriptor latent space (Figure 17), the TICA latent space (Figure 18), the angles descriptor latent space (Figure 19) and the H-bonds descriptor latent space (Figure 20). We will not go into detail for each of the latent spaces, as they are shown here for completeness and as evidence that EncoderMap is able to construct sensible latent spaces with a wide variety of different descriptors. We mention a few general observations, however. Firstly, the H-bond information presented here matches and gives more context to the free energy landscapes with average structures presented in section III.6. Secondly, the configurations with closed native H-bonds are confined to a smaller region of the latent space, which coincide with several free energy minima that signify stabilized conformations. As H-bonds stabilize structures, this matches our expectations exactly. Thirdly, we observe artifacts in the more detailed (b) figures in the transition regions between bigger patches/domains of the same structures. Folding usually progresses by forming a 'helix nucleus' of three consecutive native H-bonds, which then 'zip' the alpha helix outwards [109, 126, 131, 157]. Configurations, which have single broken native H-bonds surrounded by closed H-bonds, are therefore unlikely and while these configurations appear in the figures as artefacts in transition regions, we do not find extended patches with these configurations in the latent spaces.

III.8 DISCRETIZATION

In order to assess and compare the modeling capabilities of the two dimensionality reduction techniques considered in this work, TICA and EncoderMap, we construct

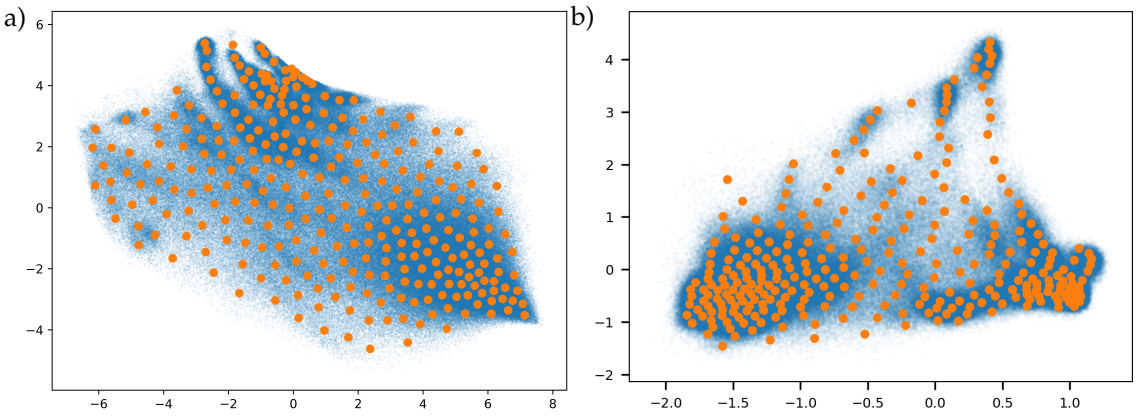


Figure 21: Discretized latent spaces with $k = 300$ cluster centers of (a) EncoderMap and (b) TICA (the respective axes labels ξ_i are omitted to avoid clutter).

Markov state models (MSMs) in their respective latent spaces. To be able to compare results between these methods, we constrain ourselves to the pairwise distances descriptor (subsection III.5.1) for the rest of this work. We follow the different modeling paths shown in Figure 2, which allows us to compare the ground truth of no dimensionality reduction with the performances of TICA and EncoderMap in two dimensions respectively.

It is necessary to discretize the latent spaces first, to be able to build MSMs. This can be accomplished by partitioning the latent spaces into discrete sets $S_i \subset \mathcal{L}$. In this work, we utilize the k-means algorithm [46] for the partitioning. This method optimizes the assignment of data points to a specified number of clusters k by minimizing the variance within each cluster (see also subsection II.3.2) leading to a Voronoi partition of the latent space. Each cluster then represents a finite region of the latent space, with higher densities of data points typically leading to higher densities of clusters and and therefore smaller attributed regions.

In the discretization step of our modeling pipeline (see Figure 2), each point is assigned to its nearest cluster center in order to partition the full space. These cluster centers will henceforth be referred to as discrete states. The number of discrete states k is chosen such as to strike a balance between minimizing discretization error (high k) and having enough transition statistics between these states to construct a reliable MSM (low k). Furthermore, k also needs to be chosen high enough such that transition states between metastable states are well represented to reduce errors in MSM construction [42].

In Figure 21 the final discretizations with $k = 300$ are visualized for (a) the EncoderMap and (b) the TICA latent space. The cluster centers are represented as big orange dots while all available data is projected into the respective latent spaces as small blue dots in the background. Areas of higher data density (dense blue regions) also lead to a higher density of cluster centers to reduce discretization error, as is expected. Discretizing the continuous trajectories in latent space $\xi_t \in \mathbb{R}^2$ by

assigning each point to its nearest cluster center leads to discretized trajectories $\xi_t^{disc} \in \{1, \ldots, 300\}$ only containing the numbers of the respective cluster centers for each time t.

III.9 MSM MODELING

Having turned the continuous trajectories into discretized trajectories in the previous section, we can now proceed to build Markov state models (MSMs) by counting transitions between the $k = 300$ discrete states in the discretized trajectories (see also section II.3). The discretized trajectories ξ_t^{disc} through the latent space are thus transformed into count matrices, resulting in the transition matrices $T_{ij} = P_{ij}(\tau)$, where $P_{ij}(\tau)$ is the probability of transitioning from state i to state j after a duration τ. For Markov state modeling the lag time τ is a crucial parameter that needs to be chosen well and represents the 'time-resolution' of the MSM. The choice of this parameter represents a tradeoff, as τ needs to be large enough that correlations have decayed sufficiently and the Markov property is fulfilled, while being small enough to allow for sufficient statistics and the resolution of important processes. The lag time represents the lower bound on timescales of processes, which can be resolved within a MSM and should therefore be as low as possible.
We use the PyEMMA package [158] for the whole modeling pipeline. The standard approach to choose the lag time τ_{MSM}, is to build MSMs for increasing lag times τ and plot the timescales of the slowest processes against them. We use Bayesian MSMs, which sample several transition matrices compatible with the data, to estimate errors. As Markovian dynamics implies timescales to be constant (but not the other way round) and true timescales are always underestimated [42], it is usual practice to aim to converge the timescales faster (allowing to pick lower τ_{MSM}) by adjusting parameters in all the possible previous steps in the modeling pipeline (Figure 2), which influence the MSM.
The resulting plots showing the convergence of timescales can be found in Figure 22. As the biased ensembles skew the timescales, only the data from the unbiased ensemble $f_0 = 0$ was used. We follow the three modeling paths visualized in Figure 2 of building MSMs in (a) descriptor space without additional dimensionality reduction and in the two-dimensional latent spaces after dimensionality reduction with (b) TICA and (c) EncoderMap. It should be emphasized that MSM building without additional dimensionality reduction is only feasible for small systems, as bigger systems have a lot of dimensions and suffer from the curse of dimensionality (see also subsection II.3.3.1). As deca-alanine is a small toy system, we make use of this here to establish a baseline by building an MSM directly in the descriptor space, in which no additional information is lost through dimensionality reduction. We com-

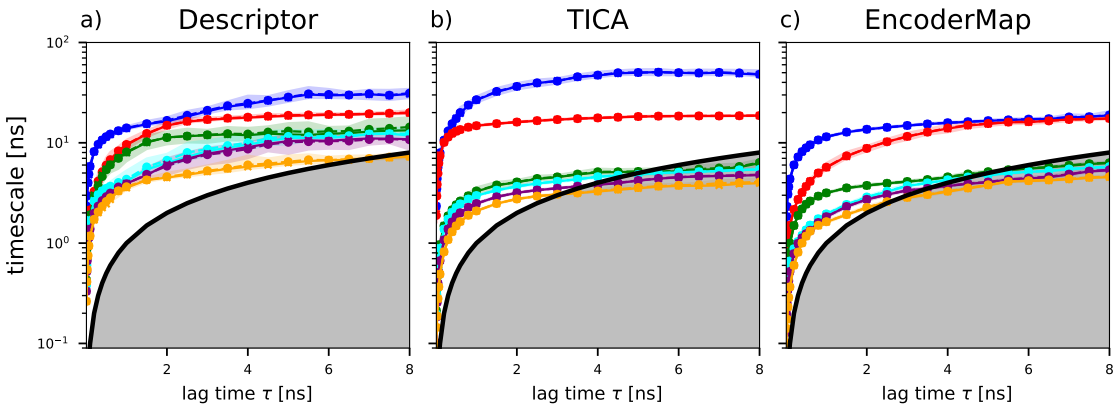


Figure 22: Convergence of the six slowest implied timescales of MSMs estimated in the unbiased ensemble $f_0 = 0$ as a function of the lag time τ for (a) the full descriptor without dimensionality reduction, (b) TICA and (c) EncoderMap. To ensure comparability, a lag time of $\tau_{MSM} = 4\,ns$ is chosen, for which the timescales have approximately converged for all modeling paths. (Adapted from our publication [63].)

pare the results of modeling with the dimensionality reduction methods TICA and EncoderMap against this baseline later on.

Figure 22 shows that the timescales have approximately converged for a choice of $\tau_{MSM} = 4\,ns$ for all three modeling paths. This choice of τ_{MSM} also matches Fabian Knochs publication [116], where the data was first used and therefore makes our results comparable to parts of this earlier analysis.

As the MSM built in descriptor space incorporates more information than after dimensionality reduction with TICA or EncoderMap, we use it to establish further parameters for the subsequent analysis. In Figure 23 the first ten eigenvalues and corresponding timescales for the MSMs built in descriptor space are shown with the previously determined $\tau_{MSM} = 4\,ns$. We find a small gap between the sixth and seventh eigenvalue and set the number of metastable sets to $N_c = 6$. It is usual practice to look for a timescale separation, which separates the slow interesting processes from the fast fluctuations and helps determine this cutoff. To make use of the full data including the biased ensembles, we then build multi-ensemble Markov models (MEMMs) with the TRAM algorithm, which is able to incorporate information from different ensembles (see also subsection II.4.2). TRAM uses all ensemble data to build a MSM for each of the ensembles and we continue analysis with the MSMs for the unbiased ensembles ($f = 0$) respectively, as we are interested in the unbiased behavior of the molecule.

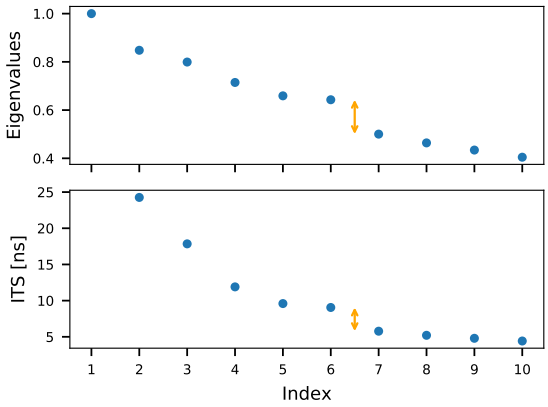


Figure 23: First ten eigenvalues and implied timescales (ITS) of an MSM built in the unbiased ensemble of descriptor space with lag time $\tau = 4\,\text{ns}$. The first eigenvalue is exactly one representing the stationary state. Between the sixth and seventh processes exists a small timescale separation. (Adapted from our publication [63].)

III.10 METASTABLE STATES

We now progress with analyzing these MSMs (one for each modeling path) and compare them in the subsequent chapters to derive statements about the capabilities of TICA and EncoderMap as dimensionality reduction methods for modeling. Our ultimate goal is to identify long-lived metastable sets, which correspond to molecular conformations and are represented as larger subdivisions of the latent space. The PCCA++ algorithm (Robust Peron Cluster Cluster Analysis) (see subsection II.3.2.3) can be employed in this context, as it is a fuzzy spectral clustering method that partitions the k discrete states into a specified number N_c of metastable sets, while preserving the slow timescales of the Markov chain. The algorithm produces a membership matrix $(m_{ic}) \in \mathbb{R}^{k \times N_c}$, with m_{ic} denoting the probability of state i belonging to set c, $S_i \subseteq S_c$. This allows PCCA++ to handle discrete transition states that do not have a clear affiliation with a metastable set, providing a more comprehensive analysis.

The membership matrix provided by PCCA++ contains probabilities for each discrete state to belong to each one of the six metastable sets. As it is not possible to visualize all of these distributions simultaneously, we depict this attribution in TICA space in Figure 24(b) by assigning each discrete state the set with the highest percentage. These sets represent contiguous regions, corresponding to localized kinetically accessible regions with similar configurations. Not all free-energy minima in the free energy landscape (see (a) of Figure 24) translate into metastable sets following this analysis. Most of the discrete states are assigned to the unfolded conformation $c = 1_T$ (orange) while the folded helix is combined with intermediate states I_1 and I_4 into partition $c = 2_T$ (green), showing that these states are kinetically accessible to each other. The two intermediate conformations I_2 and I_3 are kinetically separated

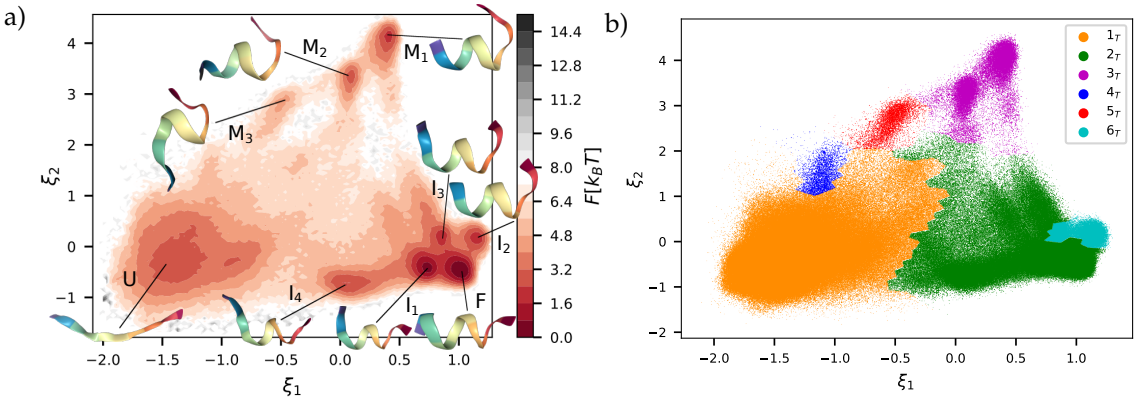


Figure 24: (a) Free energy landscape in TICA space with average structures (equivalent to Figure 5, reproduced here for ease of reading). (b) Metastable sets found in the TICA latent space by the PCCA++ algorithm for $N_c = 6$. (Adapted from our publication [63].)

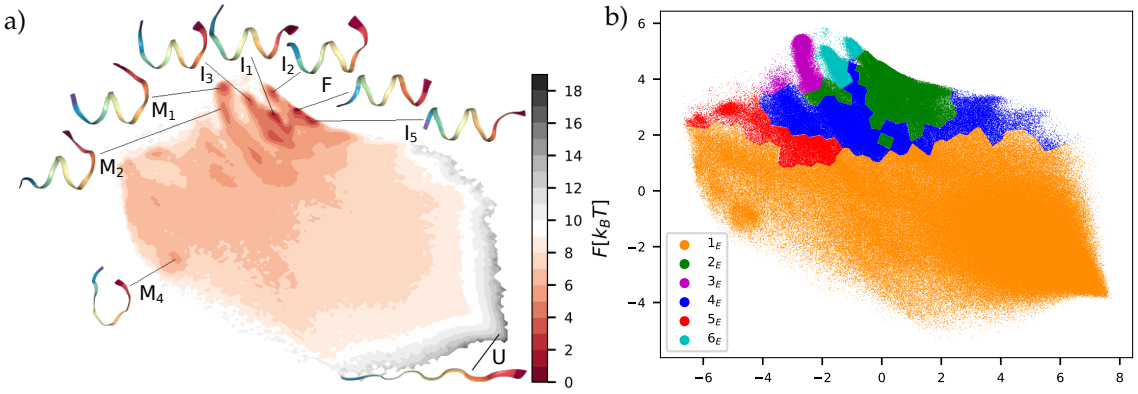


Figure 25: (a) Free energy landscape in EncoderMap space with average structures (equivalent to Figure 7, reproduced here for ease of reading). (b) Metastable sets found in the EncoderMap latent space by the PCCA++ algorithm for $N_c = 6$.

into another partition ($c = 6_T$, cyan) by the second coordinate ξ_2. Furthermore, three partitions are found at even larger values of ξ_2 ($c = 3_T$ contains M_1 and M_2 and $c = 5_T$ contains M_3). Interestingly, the $c = 4_T$ (blue) partition does not correspond to a minimum of the free energy and is not separated by a barrier.

For EncoderMap we proceed with the same methodology as before by mapping trajectories into the latent space and discretizing them using the k-means algorithm involving $k = 300$ discrete states. We then develop a MEMM incorporating trajectories from all ensembles and utilize the MSM of the unbiased one. Finally, we employ the PCCA++ algorithm to group the discrete states into six metastable sets in the EncoderMap latent space, as presented in Figure 25(b).

By analyzing the conformations within each set, we find qualitative overlap between sets 1_E and 1_T (unfolded configurations), sets 2_E and 2_T (folded and fast intermediates), sets 3_E and 3_T (misfolded), and set 6_E with 6_T (slower intermediates). Set 4_E however encompasses an extended region between unfolded and folded (containing

intermediate I_1) while 4_T is only linked to the unfolded region. In TICA space, 5_T is related to M_3 while 5_E occupies a flat region with respect to the free energy in the latent space of EncoderMap.
We compare the metastable sets found in the different latent spaces more quantitavely in subsection III.11.2, but validate the models first in the folllowing chapter.

III.11 VALIDATION

In this chapter, we validate the models built for each of the modeling paths on the basis of two important aspects. Firstly, we expect the models to describe the data sufficiently well and as we use MSMs, we validate that the models describe the transitions between metastable sets correctly. Secondly, we look at and compare the partitioning of the latent space into metastable states, which is different for each of the models. Ideally, we would expect that the same (physical) metastable sets are found and therefore the amount of correct metastable sets found can be viewed as an indicator for the quality of the latent space.

III.11.1 CHAPMAN–KOLMOGOROV TEST

At the end of the modeling pipeline, we need to ensure the consistency between the MSM we have constructed based on data and the behavior of this original data. To accomplish this task, we use the Chapman–Kolmogorov test (CK-Test) (see also subsection II.3.5). The CK-Test verifies that the transitions in the coarse-grained set of determined metastable sets accurately reflect the transitions in the continuous original data. For that, the CK-Test compares the transition statistics between all metastable sets in the MSM with the original data. There are two known weaknesses of the CK-Test. Firstly, a lot of the possible transitions can be rare so that there is not enough statistics for a certain amount of transitions. In practice, this is often dealt with by only conducting the test on the statistics of staying in the same states, i.e. transitions $i \rightarrow i$ for all of the N_c metastable states. Since metastable states decay slowly by definition, these transitions offer more statistics by default. In this work, however, we conduct the full test to show the full picture and to be able to compare the full information. Secondly, the CK-Test checks whether the MSM describes the transitional behavior in the data well with the chosen metastable sets, but it does not test that the metastable sets are sensible in the first place. Although unlikely, there is therefore always the possibility that pathological models are built that incorporate trivial or unphysical metastable states that work well with the data, but ultimately represent models with no physical value. To mitigate this, it is good practice to inspect the determined metastable sets or the distribution of structures in the free

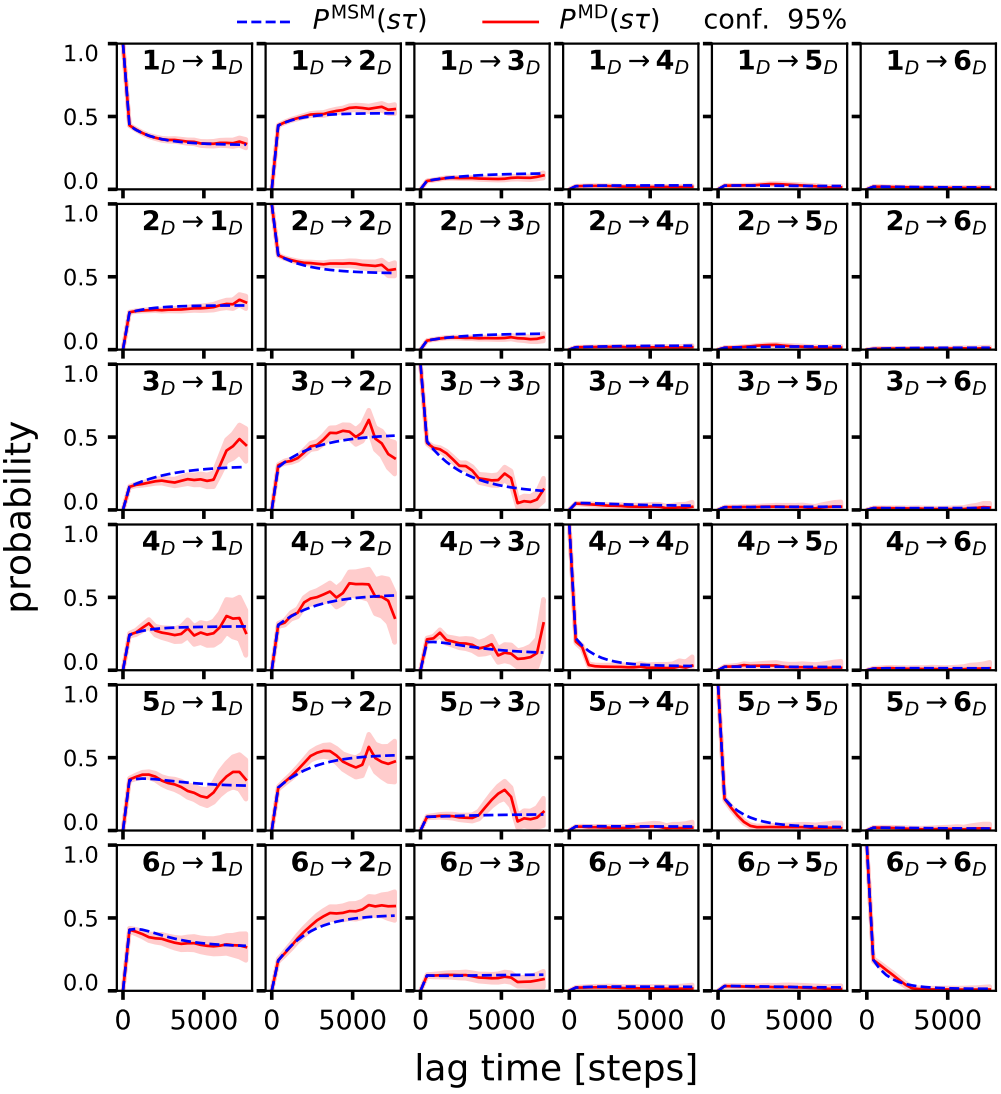


Figure 26: Descriptor CK-Test showing the probabilities of all the possible transitions over time between the metastable sets found in descriptor space. The blue dashed line represents the predictions of the Markov state model, while the red line represents the transitions observed in the data. (Adapted from our publication [63].)

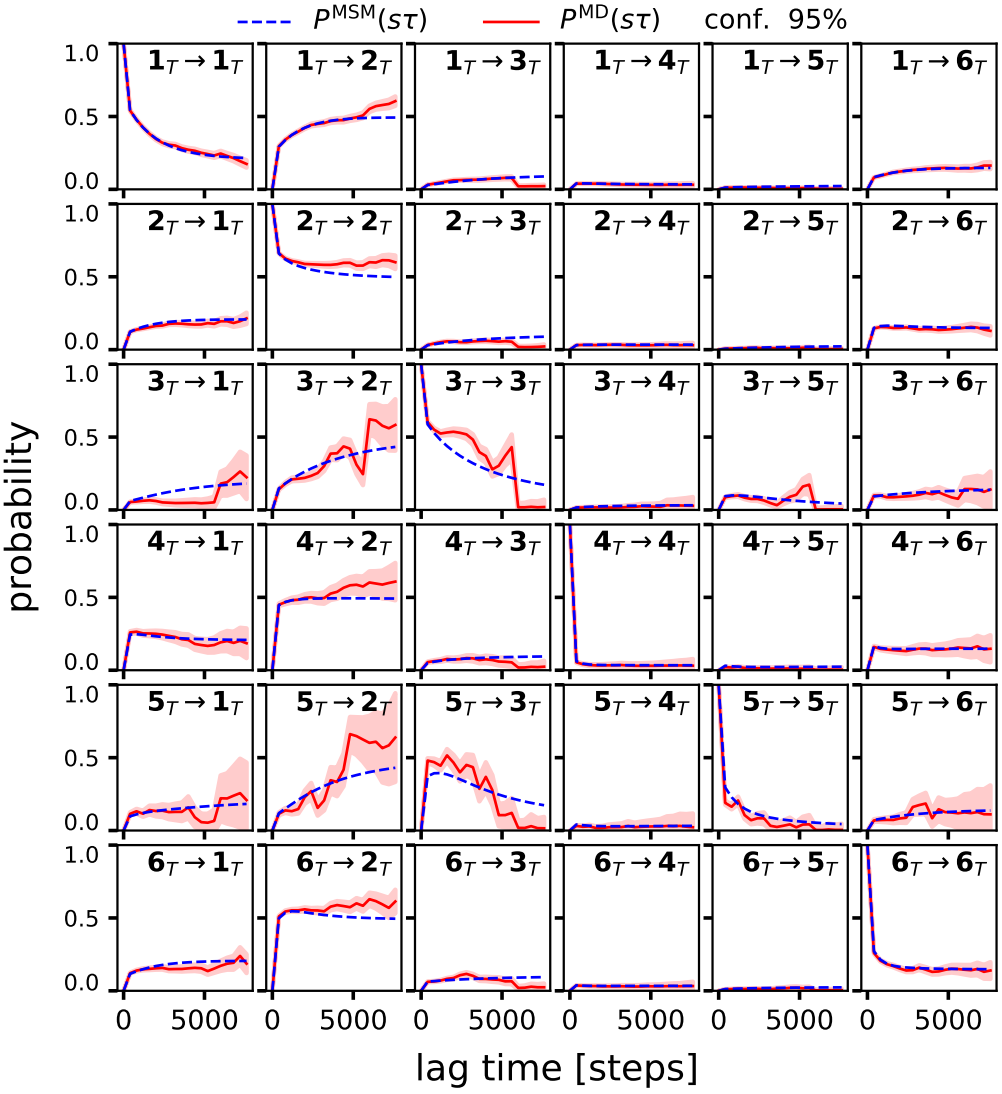


Figure 27: TICA CK-Test showing the probabilities of all the possible transitions over time between the metastable sets found in TICA space. The blue dashed line represents the predictions of the Markov state model, while the red line represents the transitions observed in the data.(Adapted from our publication [63].)

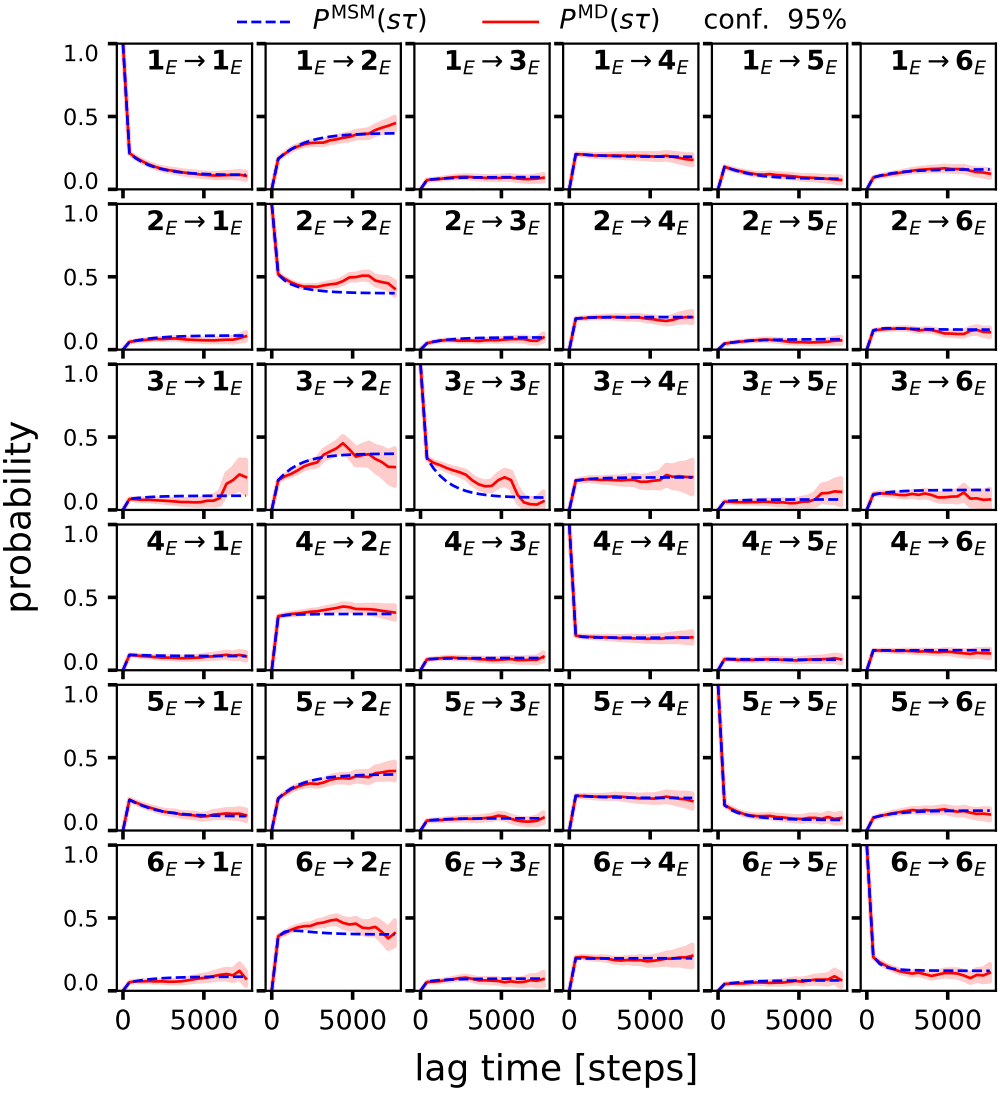


Figure 28: EncoderMap CK-Test showing the probabilities of all the possible transitions over time between the metastable sets found in EncoderMap space. The blue dashed line represents the predictions of the Markov state model, while the red line represents the transitions observed in the data.(Adapted from our publication [63].)

energy landscape visually. We will investigate and compare the metastable states found in the different latent spaces in the next chapter.

As a baseline model for comparison with TICA and EncoderMap, we first validate the MSM built without dimensionality reduction directly in the descriptor space. The CK-Test results presented in Figure 26 demonstrate that the iteratively applied transition matrix of the MSM (dashed blue line) accurately describes the behavior found in the original molecular dynamics (MD) data (red line). To this end, we use the methods outlined in [42] to calculate the CK-Test for the MD data and its error. The error bands shown represent $2\sigma_{MD}$. The CK-Test result affirms that for our particular system, modeling without dimensionality reduction is possible and can provide a suitable baseline. It is important to note, however, that the error on P^{MD} increases at long lag times. This is because the trajectories are of varying lengths, with only around a quarter of them exceeding 8000 time steps, leading to poor statistics. As such, the CK-Test results in this regime should be taken with caution.

To assess the quality of the TICA latent space for model building, we conduct a corresponding CK-Test for TICA space presented in Figure 27. The overall trend of the data is captured by the model, while there are discrepancies in the long-term behavior of several transitions (e.g., $3_T \rightarrow 3_T$). This again may be due to the fact that only around a quarter of the simulated trajectories exceed 8000 time steps, which could lead to poor statistics and make it difficult to observe certain transitions for high lag times. Another possible explanation is the linear nature of TICA as a model, which restricts its ability to capture non-linearities in few dimensions. To improve the performance of the model, increasing the dimensions of the latent space may be necessary.

To evaluate the extent to which the EncoderMap latent space accommodates Markovian dynamics between the metastable sets found, a CK-Test is performed EncoderMap space as well. The results of the CK-Test are presented in Figure 28 and indicate that the data can be accurately described by the model. When comparing the CK-Tests of the descriptor without dimensionality reduction (Figure 26) to the CK-Test in EncoderMap space, it stands out that the test in EncoderMap space is smoother than the one in descriptor space that contains more fluctuations. This is exactly what we expect, as it shows that EncoderMap distills the correct information for Markov state modeling from descriptor space, which contains additional information that can not be modeled by MSMs. Consequently, it can be concluded that EncoderMap produces a low-dimensional representation that is at least comparable in quality to the other methods.

III.11.2 COMPARISON OF METASTABLE SETS

After having established in the previous chapter that the models built for all three modeling paths (from Figure 2) describe the transitions in the data sufficiently well,

we now concern ourselves with the metastable sets found in each of the latent spaces. The metastable sets are identified in an unsupervised manner and the partitioning is not guaranteed to be sensible, which could potentially lead to bad models. As we expect the descriptor space to contain more information than the modeling spaces of TICA and EncoderMap after dimensionality reduction, we take the metastable sets found in descriptor space as reference to compare to. We are aiming to investigate, which of the metastable sets found in the different latent spaces correspond to each other and which of the (ground truth) metastable states found in the descriptor space are found in the latent spaces of TICA and EncoderMap, too.
Measuring the correspondences of these metastable sets turns out to be a hard problem, because of several reasons. Firstly, the association of data points to different metastable sets is done by probabilities, which are determined by the PCCA algorithm. Comparing these associations therefore turns into comparisons of probability distributions. We find that often the same sets of metastable states are found in the data, but with different uncertainties. As we are only interested in which metastable sets are found, we abandon the probability distributions and turn them into hard attributions by assigning each data point the metastable set with the highest probability. Secondly, there exist transitory states between metastable sets so that the boundary between these sets is not unique and the boundary easily fluctuates without changing the underlying metastable conformation. To mitigate these problems, we measure the overlap between metastable sets by the fraction $\phi_{cc'} = N_{cc'}/N_s$ of configurations shared by the (baseline) set in descriptor space c and the sets in TICA or EncoderMap space c' respectively. This metric has the property that it is symmetric in the sets as well as that all fractions sum to unity ($\sum_{c,c'=1}^{N_c} \phi_{cc'} = 1$) since each configuration belongs to exactly one set. The fraction $\phi_{cc'} = N_{cc'}/N_s$ therefore represents what percentage of the whole data was assigned to metastable set c in descriptor space *and* to metastable set c' in TICA/EncoderMap space. As the metastable sets have different sizes in latent space, not the absolute size of the number is important, but the size of the number for a fixed c in relation to all the possible values for c', which we visualize by normalizing the colors for each row in the subsequent matrices. Note that this normalizing is not symmetric in the methods and we normalize here on the descriptor results, as we take them as baseline. As a side note, the colors can be seen as visualizing a non-symmetric variant of the Jaccard index, a common metric for measuring the overlap of sets.

III.11.2.1 TICA

In Figure 29 the metastable sets found in TICA space are compared with the metastable sets found in descriptor space. In (a) the free energy landscape in TICA space is shown with selected average structures and in (b) the metastable sets $1_T - 6_T$ found in TICA space are visualized. Both (a) and (b) are equivalent to Figure 24 and are reproduced here for ease of comparison. In (c) the baseline metastable sets

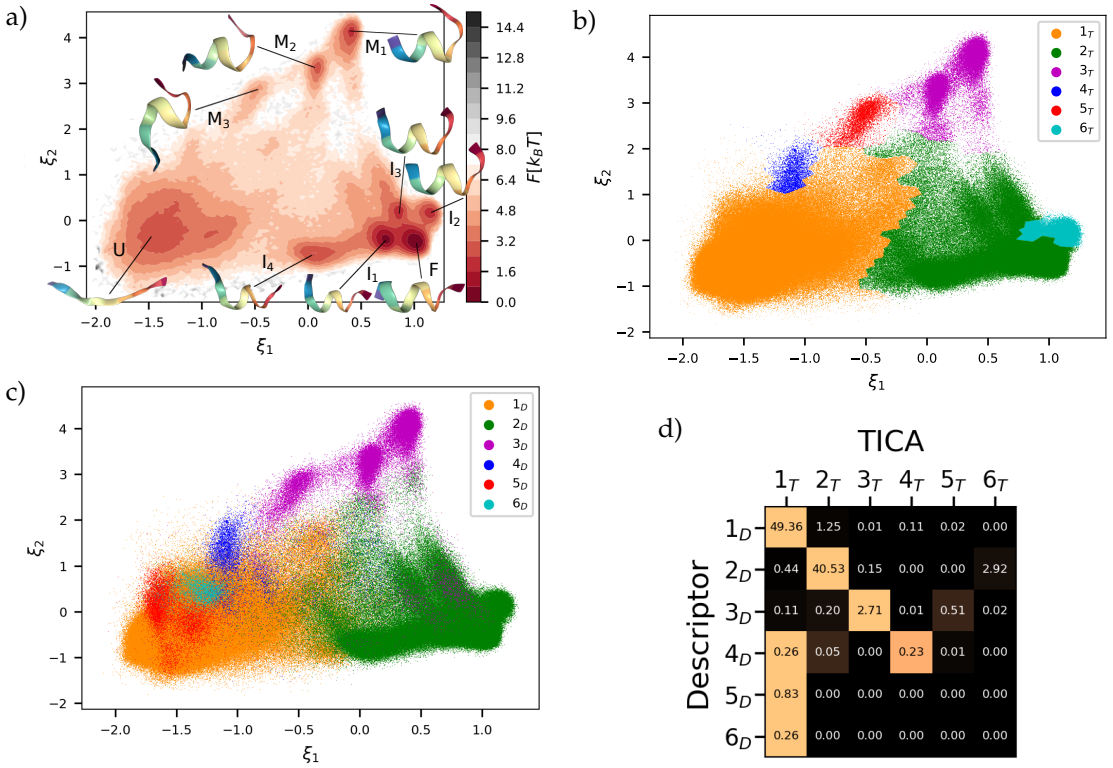


Figure 29: Comparison of TICA metastable sets with descriptor metastable sets. (a) Free energy landscape in TICA space, (b) Metastable sets found in TICA space ((a)+(b) equivalent to Figure 24, reproduced for ease of reading) (c) Metastable sets found in descriptor space (baseline) projected into TICA space, (d) Overlap matrix between metastable sets found in descriptor and TICA space. Counterparts to descriptor metastable sets $1_D - 4_D$ are found in TICA space as $1_T - 4_T$ while there are no counterparts for 5_D and 6_D.

$1_D - 6_D$ from descriptor space are projected into TICA space to allow for comparison with the TICA sets in (b). The amount of the baseline descriptor sets found in TICA space can be considered as a metric for the quality and the modeling capabilities of the TICA space. In (d) the previously described overlap matrix between these sets is shown, which illustrates the results of the overlaps for the full descriptor space with TICA's metastable sets. Each row and column sum represents the fraction of configurations assigned to the corresponding sets. The colors are normalized separately for each row to show which TICA conformations best match with the baseline conformations of the full descriptor. Non-diagonal entries indicate discrepancies in the partitionings generated by the different methods. While there may be some variation in attributing transition states to conformations across methods, these discrepancies are not significant if the same free energy basins are attributed. Analyzing the overlap matrix in (d), we find that the assignments in descriptor and TICA space agree on the three biggest metastable states, which together make up over 92% of the total data. As these three metastable states make up such a high amount of the data they appear to be the 'more obvious' metastable sets, which mainly consist of the unfolded conformation (U) for 1_D, the folded (F) and intermediate conformations (I_x) for 2_D

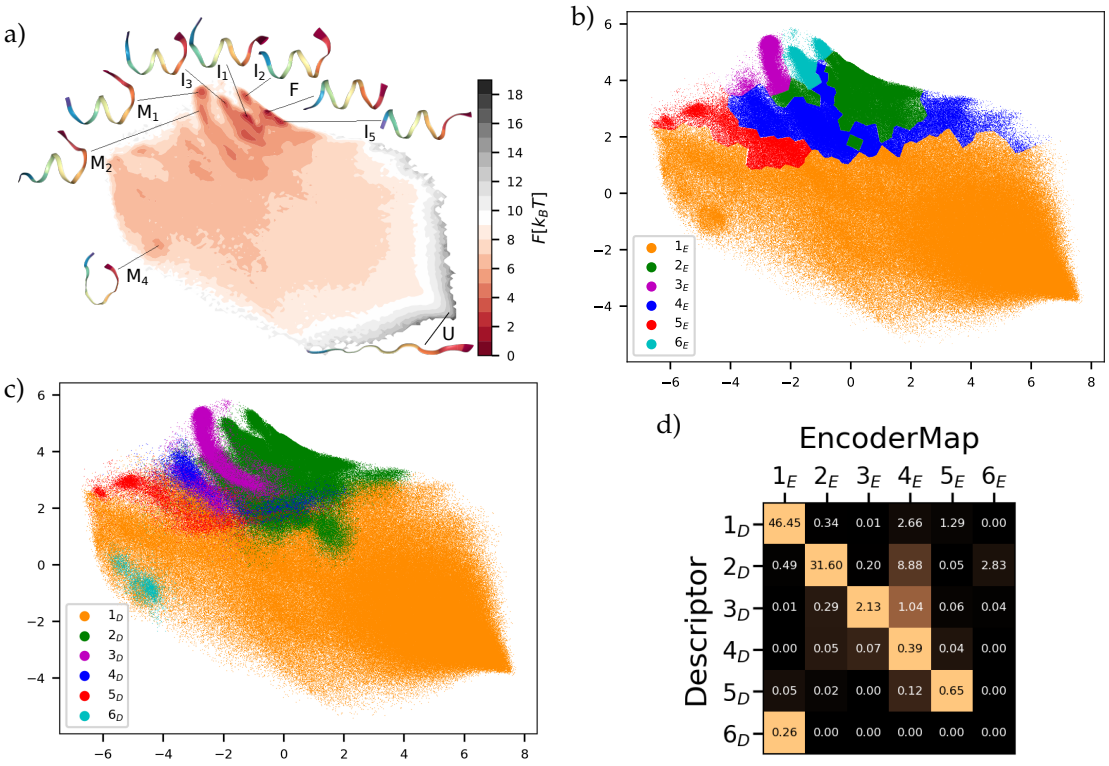


Figure 30: Comparison of EncoderMap metastable states with descriptor metastable states. (a) Free energy landscape in EncoderMap space, (b) Metastable sets found in EncoderMap space ((a)+(b) equivalent to Figure 25, reproduced for ease of reading) (c) Metastable sets found in descriptor space (baseline) projected into EncoderMap space, (d) Overlap matrix between metastable sets found in descriptor and EncoderMap space. Counterparts to descriptor metastable sets $1_D - 5_D$ are found in EncoderMap space as $1_E - 5_E$ (with pronounced differences) while no counterpart for 6_D is identified.

and the 'obvious' misfolded conformations (M_x) for 3_D. For the other metastable sets $4_D - 6_D$ there exist no 'obvious' pronounced free energy minima in the free energy landscape of TICA space (a). Furthermore, there is a discrepency between 3_D which encompasses free energy minima $M_1 - M_3$, while 3_T only contains $M_1 - M_2$ and in TICA space M_3 is identified as its own metastable set 4_T. Although the fourth metastable set of descriptor space 4_D also has significant overlap with 1_T (see row four in (d)), it is still identified in TICA space as 4_T, which is in accordance with visual inspection of (b) and (c). Metastable sets $5_D - 6_D$, however, completely overlap with 1_T in TICA space and are not found in TICA latent space. In contrast, another metastable state 6_T is suggested, which contains free energy minima $I_2 - I_3$ and belongs to the metastable set 2_D in descriptor space, containing folded and fast intermediate conformations.

III.11.2.2 *EncoderMap*

We compare the metastable sets found in EncoderMap space with the metastable sets found in descriptor space in Figure 30. In (a), we present a visual representation of the free energy landscape in EndoderMap space with selected average structures, and in (b) the metastable sets $1_E - 6_E$ found in EncoderMap space are displayed. These visualizations are equivalent to Figure 25 and are provided here for ease of comparison. In (c), we project the metastable sets $1_D - 6_D$ found in descriptor space and which serve as a baseline, into EncoderMap space to enable comparison with the EncoderMap sets displayed in (b).
The matrix in (d) shows the overlaps between the full descriptor space's and the EncoderMap's metastable sets. Each row and column sum represents the fraction of configurations assigned to the corresponding set. The colors are normalized separately for each row, highlighting which EncoderMap conformations most closely match the baseline conformations of the full descriptor. The diagonal entries reveal that five of the six descriptor metastable sets are found in EncoderMap space. The three biggest metastable sets again are easily identified with a total agreement of over 80% of the total data. This agreement is less than in TICA space as the states bleed into each other in EncoderMap space (compare (c)). Another way to see this are the non-diagonal entries in (d) indicating discrepancies in the partitionings generated in descriptor and EncoderMap space. Especially 4_E stands out, as it has significant overlap with $1_D - 4_D$ (compare column 4 in (d)). We still consider 4_E the counterpart to 4_D, as it visually follows the same area in latent space (compare dark blue in (b) and (c)). Comparing (c) with (a), we find analogously to TICA space that 1_D again describes the unfolded conformation (U), which has no free energy minimum in EncoderMap space, 2_D contains the folded (F) and intermediate conformations (I_x), and 3_D contains the 'obvious' misfolded conformations (in this case M_1 and M_2), which possess a pronounced free energy minimum in EncoderMap space. M_3 from TICA space is not found here, but potentially belongs to one of the more shallow free energy minima here. In EncoderMap latent space we uncover that 6_D which is neither identified in TICA nor in EncoderMap space as a metastable set, can be attributed to a free energy minimum M_4 in the EncoderMap free energy landscape, representing a misfolded beta-hairpin conformation. Interestingly, 6_E again contains the intermediate conformations $I_2 - I_3$ so that both, TICA and EncoderMap, postulate the same metastable set, which is not identified in the descriptor space.

III.11.2.3 *TICA-4D-E2E*

Finally, we compare our results with the 5-dimensional TICA-4D-E2E space (see subsection III.6.4) from [116] and the metastable sets found therein. The modeling in this space progresses analogously to the modeling pipeline (Figure 2) by discretizing with k-means to $k = 300$ discrete states, building a MEMM with TRAM to incorporate all ensembles, continuing with the unbiased MSM and coarse-graining

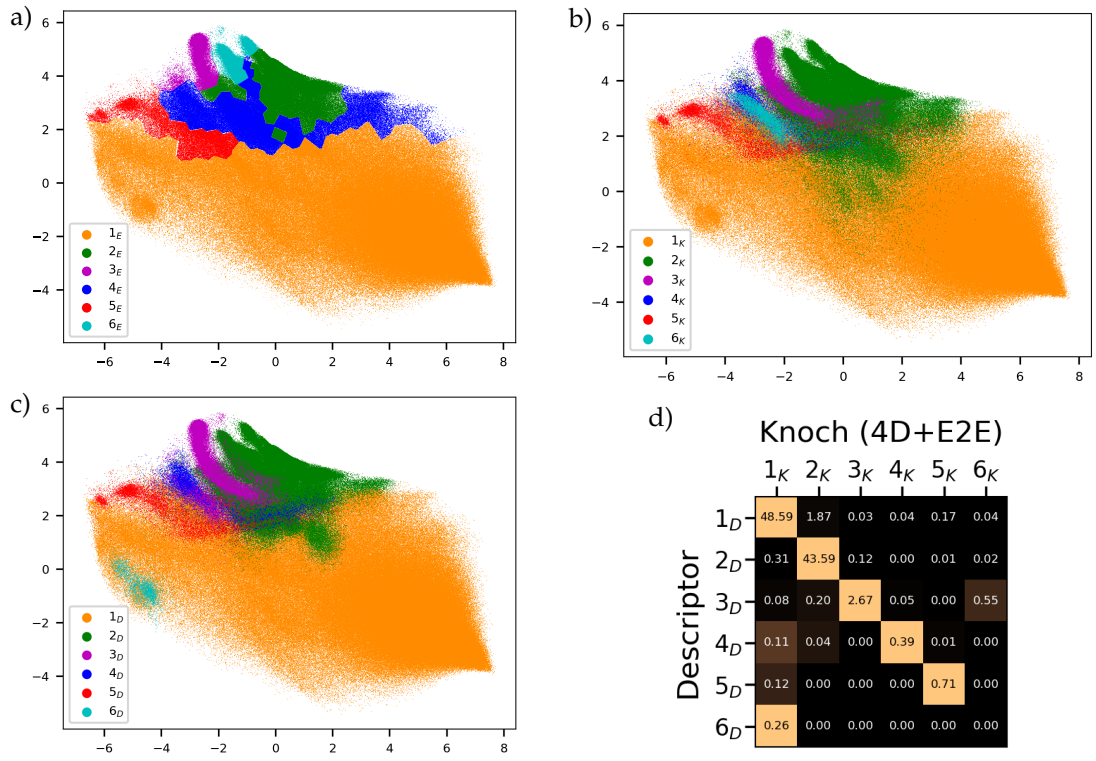


Figure 31: Comparison of metastable states in TICA-4D-E2E space identified in Knoch's publication [116] with descriptor and EncoderMap metastable states. (a) Metastable sets found in EncoderMap space ((a)+(c) as in Figure 30, reproduced for ease of comparison), (b) Metastable sets found in TICA-4D-E2E space projected into EncoderMap space, (c) Metastable sets found in descriptor space (baseline) projected into EncoderMap space, (d) Overlap matrix between metastable sets found in descriptor and TICA-4D-E2E space. Counterparts to descriptor metastable sets $1_D - 5_D$ are found in TICA-4D-E2E space as $1_K - 5_K$ while no counterpart for 6_D is identified.

the discrete states into $N_c = 6$ metastable sets with PCCA++. In Figure 31 the metastable sets found in descriptor space $1_D - 6_D$ are compared with the metastable sets $1_K - 6_K$, found in TICA-4D-E2E space from Knoch's publication [116]. As TICA-4D-E2E space is 5-dimensional, it is not possible to visualize the space directly and we project the metastable sets found into EncoderMap space for visualization, as we have studied this space in this work extensively. Since TICA-4D-E2E space mainly consists of a higher dimensional TICA space, we can consider this space as insight into the modeling capabilities of TICA when using more dimensions. We find in this space that metastable sets $1_D - 5_D$ from descriptor space are identified as $1_K - 5_K$, which closely resemble the descriptor metastable sets. The agreement and high overlap is expressed by the low non-diagonal entries in the overlap matrix in (d). A higher-dimensional TICA space thus manages to find one more of the descriptor metastable sets than 2-dimensional TICA space, but still fails to identify 6_D. Regarding metastable sets, it therefore has a similar performance as 2-dimensional EncoderMap space, albeit with better overlap of the descriptor sets found. That

6_D is not identified could be an artifact of the end-to-end distance incorporated in TICA-4D-E2E space, but that is highly unlikely as the additional end-to-end distance information should make it even easier to identify the beta hairpins in 6_D, as they usually distinguish themselves by their low end-to-end distance.

III.12 SUMMARY & CONCLUSIONS

In summary, we have demonstrated that the low-dimensional latent space of an autoencoder neural network with an additional distance metric (EncoderMap) can be utilized as suitable coordinates for Markov state modeling. This implies that the sketch-map metric is capable of preserving kinetic connectivity between adjacent structures in latent space. Unlike TICA, this method does not require the time-series character of trajectories and instead constructs the latent space from structural data alone.

To further investigate the latent space representation, we performed similar analyses using the full descriptor space without additional dimensionality reduction, then using TICA to create a low-dimensional representation that includes temporal information and thirdly, we looked at the results of the already available analysis of Fabian Knoch, which uses a higher-dimensional TICA space. The results of the Chapman-Kolmogorov tests show that the EncoderMap latent space performed at least as well as for TICA (in low and higher dimensions) and the full descriptor space. This suggests that the kinetics of such peptides is strongly influenced by their structure, especially through the conformations assumed by the molecule. The metastable sets identified through building Markov state models agree quite well with all three methods: EncoderMap, TICA, and the full descriptor space. EncoderMap, however, was able to discover five of the six descriptor conformations, while TICA could only identify four in the same amount of dimensions. For higher-dimensional TICA space (TICA-4D-E2E space) the same five descriptor conformations were identified, but exhibit higher overlap with the descriptor conformations, implying that TICA improves in performance with more dimensions. Moreover, without dimensionality reduction, beta-hairpins were found to be a metastable set that was missed by both TICA and EncoderMap.

There are advantages to using only structural information as with EncoderMap in this work. Firstly, as demonstrated here, data from different ensembles can be combined straightforwardly. Secondly, it can be used to identify long-lived conformations in unordered data harvested with different methods (e.g., Monte Carlo simulations [159–162]) with the possibility of using bias potentials and virtual moves to speed up the exploration of configuration space. Finally, EncoderMap is a nonlinear method, which could potentially capture more relationships in the data compared to TICA, especially in fewer dimensions. This is exactly what we find: more dimensions are

needed for TICA to identify the same amount of metastable sets as EncoderMap is already able to find in low dimensions. This means that especially for bigger molecules leading to descriptors with more dimensions, EncoderMap should be able to compress more information into less dimensions than TICA, making modeling and visualization easier.

IV

SOCIAL COMPLEXITY

Contents

IV.1 Structure of the data **84**
IV.2 Connection to molecular dynamics **87**
IV.3 Comparison of different methods **88**
IV.3.1 Fraction of variance explained 88
IV.3.2 Crossvalidation on imputed datasets 89
IV.3.3 Performance in different dimensions 91
IV.3.4 PCA vs. EncoderMap (1D) 93
IV.3.5 PCA vs. EncoderMap (2D) 96
IV.4 Exploratory data analysis in the latent space **98**
IV.4.1 Independence of the data from region 98
IV.4.2 Time series behavior 99
IV.4.3 Interpreting the latent space 107
IV.5 Summary **121**

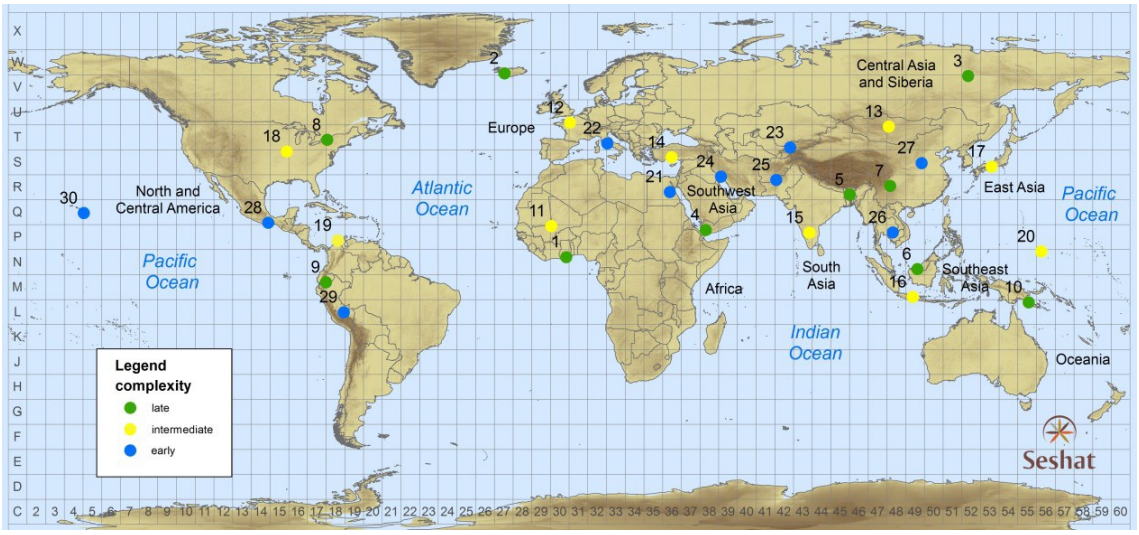


Figure 32: Visualization of the 30 natural geographic areas (NGAs) in the data and their locations on the globe, three belonging to a particular region respectively. (Reproduced from Turchin et al. [163], licensed under CC BY-NC-ND 4.0.)

The "Seshat: Global History Databank" is a massive database of quantitative historical information incorporating 414 human societies from 30 regions over a timespan of 10.000 years. It has been shown that from this data a single principal component can be extracted, named "Social Complexity" which captures almost three quarters of the observed variation in the dataset [163]. This finding hints at structural similarities in human organization and the way they evolve.
Since this setting exhibits similarities to the dimensionality reduction step in the modeling pipeline of molecular dynamics, we explore the dataset as a second project in this work to show the capability of EncoderMap to find reasonable latent representations. We aim to improve on the result presented in the aforementioned paper by comparing the performance of EncoderMap with principal component analysis (PCA) used therein.

IV.1 STRUCTURE OF THE DATA

The unit of analysis in the data is called a polity, an independent political unit, which can range from small villages to whole states. There are 414 polities in the dataset, distributed over 30 natural geographic areas (NGAs) on the globe, three belonging to a particular region respectively. The regions and corresponding NGAs are visualized in Figure 32, namely:

- Africa: Ghanaian Coast (1), Niger Inland Delta (11), Upper Egypt (21)
- Europe: Iceland (2), Paris Basin (12), Latium (22)

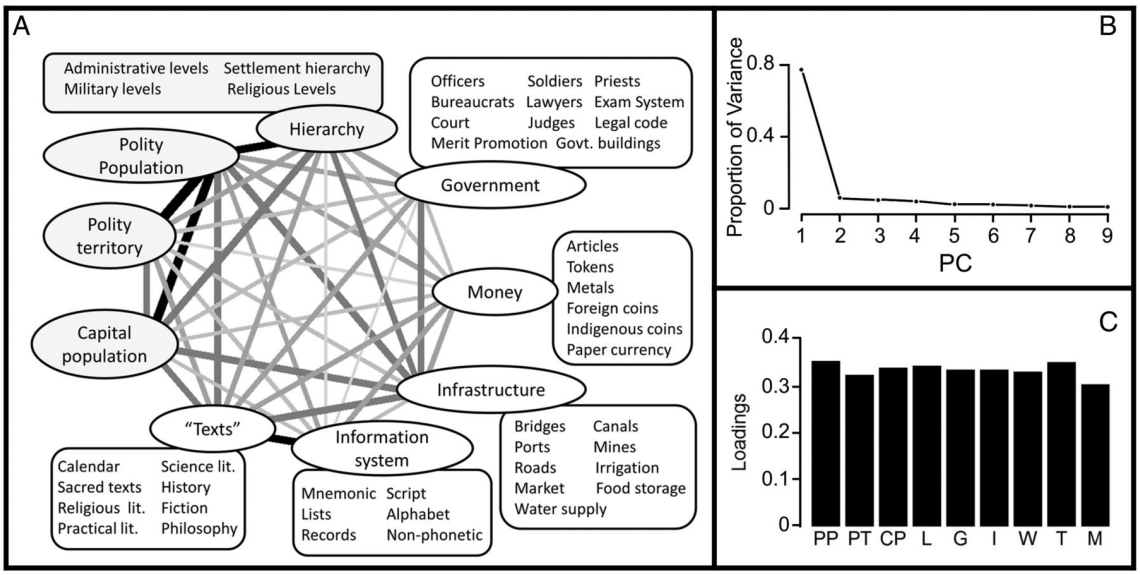


Figure 33: (A) Complexity characteristics and factors contributing to them respectively. (B) Proportion of variance in the data that can be explained by the principal components (PC). Approximately three quarters of the variation can be captured by a single principal component. (C) Loadings of the different complexity characteristics on the first principal component. Comparable loadings for all of them indicates that they contribute equally. (Figure reproduced from Turchin et al. [163], licensed under CC BY-NC-ND 4.0.)

- Central Eurasia: Lena River Valley (3), Orkhon Valley (13), Sogdiana (23)
- Southwest Asia: Yemeni Coastal Plain (4), Konya Plain (14), Susiana (24)
- South Asia: Garo Hills (5), Deccan (15), Kachi Plain (25)
- Southeast Asia: Kapuasi Basin (6), Central Java (16), Cambodian Basin (26)
- East Asia: Southern China Hills (7), Kansai (17), Middle Yellow River Valley (27)
- North America: Finger Lakes (8), Cahokia (18), Valley of Oaxaca (28)
- South America: Lowland Andes (9), North Colombia (19), Cuzco (29)
- Oceania-Australia: Oro PNG (10), Chuuk Islands (20), Big Island Hawaii (30)

For the analysis, 51 variables are used, which could be reliably coded across different polities, whereby the temporal sampling rate is one hundred years. These 51 variables are then compressed into 9 different complexity characteristics (CCs), which reflect different aspects of the polity (see Figure 33):

- Scale of societies
  - Polity population: Total population of the polity (log-transformed).

  - Polity territory: Extent of the territory the polity controls (log-transformed).
  - Capital population: Size of the largest urban center or capital respectively (log-transformed).

- Measures of hierarchical or vertical complexity
  - Levels: Number of control/decision levels in the administrative, religious, military and settlement hierarchies (averaged).

- Organization and specialized positions
  - Government: Presence of professional soldiers and officers, priests, bureaucrats, and judges as well as characteristics of the bureaucracy and of the judicial system (averaged binary values).

- Infrastructure
  - Infrastructure: Variety of public goods and public works involved in the functioning of the polity (averaged binary values).

- Informational complexity
  - Writing: Characteristics of writing and record-keeping systems (averaged binary values).
  - Texts: Presence of specialized literature, including history, philosophy, and fiction (averaged binary values).

- Complexity of cash economy
  - Money: "Most sophisticated" monetary instrument present (aggregated value between 0 and 6).

Since our knowledge of the past is patchy at best, there are not all variables available for all times in the dataset. For the final dataset the threshold was set to 30%, meaning polities were only included if at least thirty percent of all 51 variables were coded. To avoid bias from these lacking values and to reflect the uncertainty present in the data, 20 imputed datasets of complexity characteristics were created and published alongside the original paper [163]. We use dataset S1 in our work, containing 20 imputed versions of the CC time series. Imputation procedures replace missing values with plausible values. When only ranges or expert disagreement was present, values were probabilistically sampled from these options. This procedure was repeated 20 times to create 20 datasets representing the inherent uncertainty in the data, which we use for the analyses in the upcoming chapters.

IV.2 CONNECTION TO MOLECULAR DYNAMICS

In the modeling pipeline of molecular dynamics (see section III.2), we are usually confronted with the output of molecular dynamics simulations: High-dimensional time series, consisting of positions and velocities of each atom at each time step. Subjected to the curse of dimensionality (see also subsection II.3.3.1), it is not possible to extract meaningful information directly from such a high-dimensional space. Dimensionality reduction then takes place as a two step process:

1. Creating features using a (structural) descriptor:
   A descriptor is used to abstract away symmetries which are inherent in the data, but are not explicit. This could be the rotational symmetry of the structures under investigation for example, which is not explicit in the coordinates of the atoms.

2. Dimensionality reduction algorithms:
   These are algorithms designed to compress the information in the data into fewer dimensions, while throwing away mostly irrelevant variation (e.g. fluctuations). Notable representatives are e.g. PCA, TICA and autoencoders.

Applying these steps, one ends up with new coordinates describing the properties of interest, which are amenable to investigation. If chosen well, analyses can be conducted in the space of these newly found collective variables that were not possible in the higher-dimensional space. There, different metastable sets are identified and Markov state models are built to model the dynamics between these sets (compare the analysis in chapter III).
In the data presented here, forming the 9 complexity characteristics (CCs) plays the role of a structural descriptor. Principal components analysis (PCA) was used in the original publication [163] to construct a single variable containing $77.2 \pm 0.4\%$ of the original variance, describing the social complexity of the polity. We aim to improve upon this result by going from a linear method (PCA) to a non-linear method (EncoderMap). Hereby the goal is not only to improve on the amount of variance explained, but also to find novel insights in these improved coordinates. To push the analogy between molecular dynamics and this setting further, we think of societies described by these coordinates as - analogously to molecules - being in a metastable state and undergoing transitions to other metastable states.

IV.3 COMPARISON OF DIFFERENT METHODS

IV.3.1 FRACTION OF VARIANCE EXPLAINED

To be able to compare different methods against each other, we need an appropriate metric, which is able to capture the performance of the particular method involved. Here, we use the fraction of variance explained (FVE)

$$\mathrm{FVE} = 1 - \frac{\mathrm{MSE}(\hat{Y})}{\mathrm{VAR}(Y)}, \tag{IV.1}$$

where Y is the data, $\hat{Y}$ are the values predicted by the algorithm, MSE is the mean squared error and VAR is the sample variance, both defined below. The concept of FVE is closely related to the coefficient of determination R^2. For ease of reading, we will mostly use percentages later on, with percentage of variance explained being defined as $\mathrm{PVE} = 100 \cdot \mathrm{FVE}$.
For a dataset, comprised of points $\{Y_i\}_{i\in\{1,\dots,n\}}$, and predictions for these points $\{\hat{Y}_i\}_{i\in\{1,\dots,n\}}$ made by an algorithm, the mean squared error (MSE) is defined as

$$\mathrm{MSE}(\hat{Y}) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2. \tag{IV.2}$$

The sample variance is defined as

$$\mathrm{VAR}(\hat{Y}) = \sigma^2(\hat{Y}) = \frac{1}{n-1}\sum_{i=1}^{n}\left(Y_i - \overline{Y}\right)^2, \tag{IV.3}$$

measuring the variation around the sample mean $\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$.
The fraction of variance unexplained $\mathrm{FVU} = \mathrm{MSE}(\hat{Y})/\mathrm{VAR}(Y)$ measures, which fraction of the variation in the data is due to the prediction errors made by the prediction algorithm. We can investigate the edge cases:

- Worst prediction:
  An algorithm always predicting the mean of the dataset, $\hat{Y} = \overline{Y}$, is considered as the worst prediction possible, since the algorithm is not sensitive to the variation in the data. It follows then that the fraction of variance explained vanishes.

  $$\hat{Y} = \overline{Y} \;\Rightarrow\; \mathrm{MSE}(\hat{Y}) = \mathrm{VAR}(Y) \;\Rightarrow\; \mathrm{FVE} = 0 \tag{IV.4}$$

  It should be mentioned that in general, the fraction of variance explained can even turn negative ($\mathrm{FVE} < 0$), if the algorithm does not explain any of the

existing variance in the original data, but introduces additional variance into the prediction instead ($\mathrm{MSE}(\hat{Y}) > \mathrm{VAR}(Y)$).

- Best prediction:
  The best prediction possible means predicting every value correctly $\hat{Y} = Y$. The prediction algorithm is then able to explain all the variation in the data.

$$\hat{Y} = Y \;\Rightarrow\; \mathrm{MSE}(\hat{Y}) = 0 \;\Rightarrow\; \mathrm{FVE} = 1 \tag{IV.5}$$

A dimensionality reduction algorithm f is a mapping from a high-dimensional space Y to a low-dimensional latent space X, $f : Y \mapsto X$. To measure the information retained by such an algorithm we use an (approximate) inverse $f^{-1} : X \mapsto Y$ to transform the data into the latent space and back into the original space $\hat{Y} = f^{-1}(f(Y))$. With this predictions we can assess, which fraction of the variance is explained by the latent representation. There is a caveat to this methodology however: The algorithm could be learning the data "by heart", a problem widely known as "overfitting". For example, trivial mappings exist, which reproduce each data point perfectly: Just map every data point to an integer on the number line. These algorithms and representations, however, do not generalize well on unseen data. This exactly is the reason, why the performance on unseen data (also called out-of-sample data) is used to assess the degree of overfitting and is the ultimate test of the usefulness of the algorithm.

IV.3.2 CROSSVALIDATION ON IMPUTED DATASETS

To avoid overfitting, we make use of the 20 imputed datasets mentioned in section IV.1 to validate the performance on unseen data. For this, we train the algorithm on one of the datasets and measure the performance on all the others respectively. In this manner, we are able to get a handle on the variation in performance and how well the model generalizes. We then repeat this for each dataset. If the model generalizes well and learns the important aspects of the data, the results should be comparable between different imputed datasets. By using the imputed datasets, we have therefore $n_d = 20$ versions of the same data. To get a measure of the overall performance, we aggregate the performance on the individual datasets $\{X_i\}_{i\in\{1,\dots,n_d\}}$ into one big dataset $X := \bigcup_{i=1}^{n_d} X_i$, on which we measure mean and variance. It can be shown that these quantities can be calculated from the individual means $\{\overline{X}_i\}_{i\in\{1,\dots,n_d\}}$ and variances $\{\sigma_i^2\}_{i\in\{1,\dots,n_d\}}$:

$$\overline{X} = \overline{\bigcup_{i=1}^{n_d} X_i} = \frac{1}{\sum_{i=1}^{n_d} n_i} \sum_{i=1}^{n_d} \sum_{j=1}^{n_i} x_{i,j} = \frac{1}{\sum_{i=1}^{n_d} n_i} \sum_{i=1}^{n_d} n_i \overline{X}_i, \tag{IV.6}$$

$$
\begin{aligned}
\sigma^2(\overline{X}) &= \frac{1}{\left(\sum_{i=1}^{n_d} n_i\right) - 1} \sum_{i=1}^{n_d} \sum_{j=1}^{n_i} \left(x_{i,j} - \overline{X}\right)^2 \\
&= \frac{1}{\left(\sum_{i=1}^{n_d} n_i\right) - 1} \sum_{i=1}^{n_d} \sum_{j=1}^{n_i} \left(x_{i,j} - \overline{X}_i + \overline{X}_i - \overline{X}\right)^2 \\
&= \frac{1}{\left(\sum_{i=1}^{n_d} n_i\right) - 1} \sum_{i=1}^{n_d} \sum_{j=1}^{n_i} \left[\left(x_{i,j} - \overline{X}_i\right)^2 + 2\left(x_{i,j} - \overline{X}_i\right)\left(\overline{X}_i - \overline{X}\right) + \left(\overline{X}_i - \overline{X}\right)^2\right] \\
&= \frac{1}{\left(\sum_{i=1}^{n_d} n_i\right) - 1} \sum_{i=1}^{n_d} \\
&\quad \underbrace{\sum_{j=1}^{n_i} \left(x_{i,j} - \overline{X}_i\right)^2}_{(n_i-1)\sigma_i^2} + 2\left(\overline{X}_i - \overline{X}\right) \underbrace{\sum_{j=1}^{n_i} \left(x_{i,j} - \overline{X}_i\right)}_{\cancel{n_i\left(\overline{X}_i - \overline{X}_i\right)}} + \underbrace{\sum_{j=1}^{n_i} \left(\overline{X}_i - \overline{X}\right)^2}_{n_i\left(\overline{X}_i - \overline{X}\right)^2} \\
&= \frac{1}{\left(\sum_{i=1}^{n_d} n_i\right) - 1} \sum_{i=1}^{n_d} \left[\left(n_i - 1\right)\sigma_i^2 + n_i\left(\overline{X}_i - \overline{X}\right)^2\right],
\end{aligned}
\tag{IV.7}
$$

with $x_{i,j} \in X_i$ denoting the j-th element in the i-th dataset and $n_i = |X_i|$ the number of elements in X_i.

It is important to mention that since the performance on the training dataset can contain an arbitrary amount of overfitting, it is highly likely that these numbers overestimate the accuracy of the algorithms. We therefore only trust the performance on the test sets as a viable metric. The errors given represent 2σ, which amounts to a confidence interval of roughly $\approx 95.4\%$, given the data is distributed normally.

It should be remarked that another common way to combine measurements is by using a weighted sum with the inverse variances as the weights:

$$
\overline{X}_w = \frac{\sum_i \overline{X}_i/\sigma_i^2}{\sum_i 1/\sigma_i^2}, \tag{IV.8}
$$

with the variance on the quantity resulting in

$$
\sigma^2(\overline{X}_w) := \frac{1}{\sum_i 1/\sigma_i^2}. \tag{IV.9}
$$

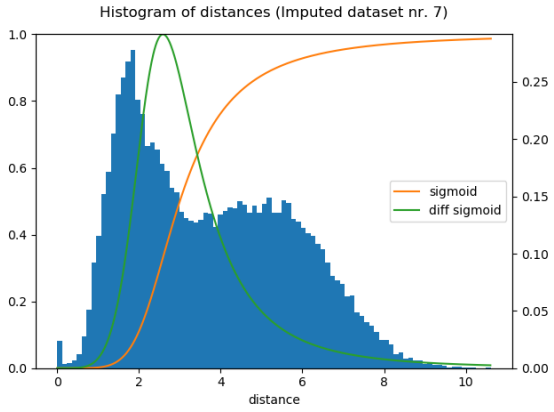


Figure 34: Histogram of distances in high-dimensional space of complexity characteristics using the example of imputed dataset 7. The graphs of the sigmoid and the differentiated sigmoid are overlayed and show the sensitivity to the distances in the EncoderMap algorithm.

This estimator maximizes the likelihood assuming all the measurements are samples from Gaussian distributions with the same mean $\overline{X}_w$ and different standard deviations σ_i with $i \in \{1, \ldots, n_d\}$:

$$L(\overline{X}_w) = \prod_i f(\overline{X}_i | \overline{X}_w, \sigma_i) = \prod_i \frac{1}{\sqrt{2\pi\sigma_i}} \exp\left(-\frac{1}{2}\frac{(\overline{X}_i - \overline{X}_w)^2}{\sigma_i^2}\right) \tag{IV.10}$$

Therefore using this approach implicitly assumes a Gaussian distribution and we choose the former more general one.

IV.3.3 PERFORMANCE IN DIFFERENT DIMENSIONS

To train EncoderMap, parameters for the sigmoid function must be chosen which define the distances in the high-dimensional space of complexity characteristics EncoderMap is sensitive to (see also paragraph II.3.3.3). In Figure 34 the distances between data points in the high-dimensional space are visualized as a normalized histogram for imputed dataset 7. All the other imputed datasets lead to a comparable plot. Furthermore, the sigmoid and the differentiated sigmoid that are used in the high-dimensional space are plotted into the same diagram to show which distances the EncoderMap algorithm is sensitive to. The sigmoid for the high-dimensional distances uses the parameters $\sigma_h = 3$, $a_h = 6$ and $b_h = 3$. The value of σ_h determines the inflection point of the sigmoid, as can be seen in the plot, while a_h and b_h determine how quickly the function falls off towards $-\infty$ and $+\infty$ respectively. As the algorithm is most sensitive where the change in the sigmoid is largest, the graph of the differentiated sigmoid (green) gives a direct measure of the sensitivity for different distances. Following the original idea of Sketchmap that small and big

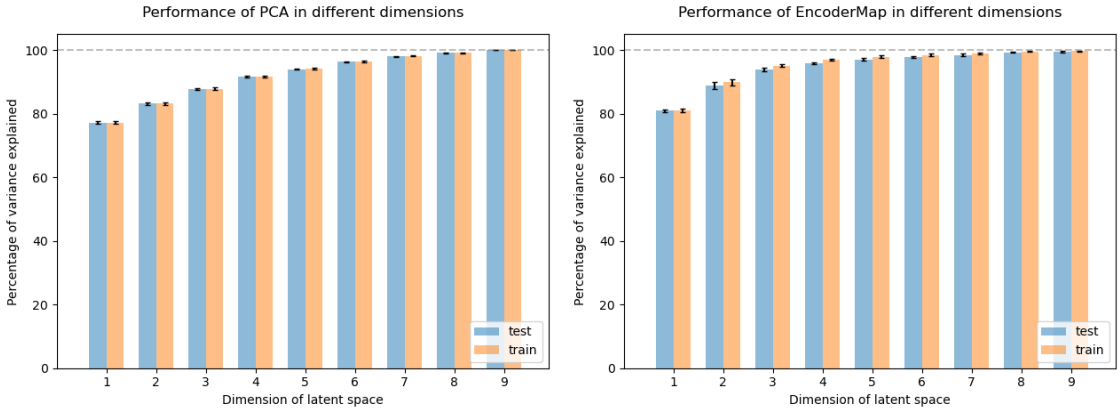


Figure 35: Comparison of performance of PCA (left) and EncoderMap (right) algorithm against dimension of latent space. More latent dimensions allow for richer representations, while having diminishing returns in performance for each latent dimension added.

distances are irrelevant in high-dimensional spaces, the parameters are chosen such that EncoderMap is most sensitive for intermediate distances, which is between the two local maxima for the distance histogram presented here.
The sigmoid for the low-dimensional distances (not shown) uses parameters $\sigma_l = 1$, $a_h = 2$ and $b_h = 6$. These parameters are not as crucial as the ones for high-dimensional distances as they usually only scale, contract and spread out the data in latent space. The EncoderMap algorithm is trained on the data for $n = 5000$ steps, setting the first two layers to 128 neurons each and the third layer as the latent space with the amount of neurons set to the desired number of latent dimensions, while leaving the standard values for all the other parameters.
To compare the performance of the PCA and EncoderMap algorithm in different latent space dimensions, we train the algorithms on each imputed dataset and cross-validate on all the others as described in the previous subsection IV.3.2, calculating the percentage of variance explained. We repeat this process for each of the possible latent dimensions $n_{dim} \in \{1, \ldots, 9\}$ to get an estimate of the performance in every dimension.
The results are shown in Figure 35 with PCA to the left, EncoderMap to the right, and a summary in Figure 36 for both algorithms. Both algorithms gain the most performance in the first few dimensions and experience diminishing returns per dimension for higher latent dimensions. This is to be expected for PCA by construction, whereas EncoderMap does not exhibit a guarantee for this property. More notably, EncoderMap gains the most performance until a latent dimension of three, after which the rate of improvement per additional dimension changes (note the different slope for more than three dimensions). Furthermore, EncoderMap reaches the highest uncertainty in the performance for a latent dimension of two, as depicted by the error bars. The errors on performance are always comparable between train and test datasets, while in contrast to PCA the mean values differ for EncoderMap. For

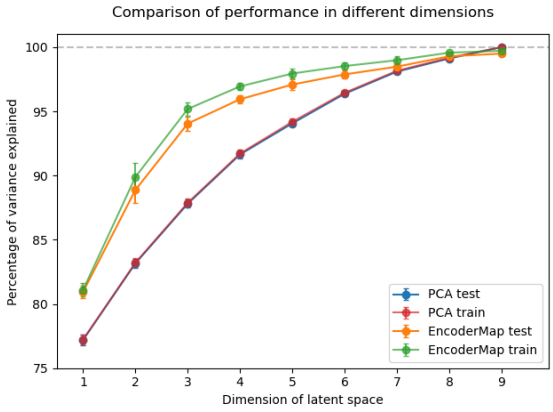


Figure 36: Comparison of cross-validated performance in all dimensions between PCA and EncoderMap. To allow for easier comparison of the differences the y-axis does not start at the origin.

EncoderMap, the mean values of performance are systematically higher on the training dataset than on the test dataset, hinting to a small but systematic overfitting in every dimension, which PCA as a linear method is not plagued by. This is a common danger of the higher flexibility of non-linear models on the one hand, while on the other hand they are able to provide superior performance for the same reason when validated carefully.
This is the case in Figure 36. Although the EncoderMap algorithm slightly overfits in every dimension (note the green graph always being above the orange one), the performance of EncoderMap, as measured by this metric, is superior to the performance of PCA in every amount of dimensions except for the full original 9 dimensions of the dataset (i.e. the CCs). The usual strategy is to find a trade-off between the amount of dimensions and the percentage of variance these dimensions can represent. Ideally, we want the least amount of dimensions with a maximum amount of information captured from the high-dimensional dataset. Additionally, a big advantage of few dimensions (especially one or two) lies in the fact that they can easily be visualized and investigated with human eyes. In the following chapters, we compare the performance of PCA and EncoderMap with a latent space of one and two dimensions. A one-dimensional latent space is used in the original publication [163] and going to two dimensions gives the highest improvement in performance when adding additional dimensions, while still being low-dimensional enough to be visualized easily.

IV.3.4 PCA VS. ENCODERMAP (1D)

As discussed in subsection IV.3.2, the PCA and EncoderMap models are trained on one imputed dataset and then validated on all the others. We start by investigating the performance of both dimensionality reduction algorithms in a one-dimensional

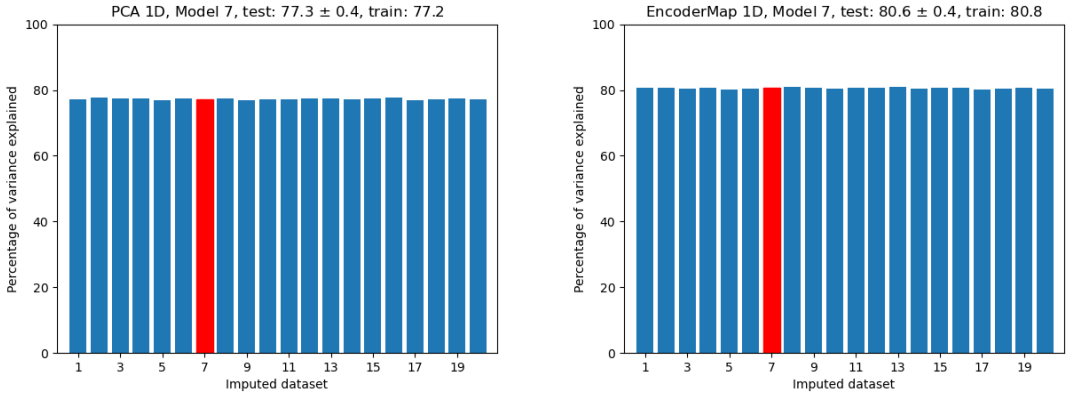


Figure 37: Comparison of performance of PCA (left) and EncoderMap (right) algorithm in one dimension trained on imputed dataset number 7 (red bar) with the performance on the remaining out-of-sample datasets (blue bars).

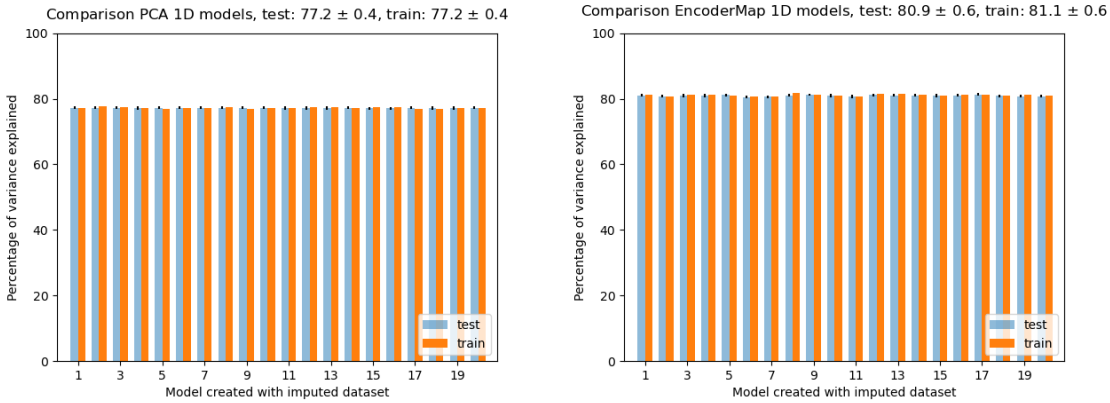


Figure 38: Comparison of performance of PCA (left) and EncoderMap (right) algorithm in one dimension on all the imputed datasets.

latent space. The performance is comparable between being trained on different datasets, so we choose the model trained on imputed dataset $i = 7$ as an arbitrary representative. In Figure 37 the performance on the datasets is shown for the PCA algorithm on the left hand side and for EncoderMap on the right. The performance on the dataset, on which the model was trained ($i = 7$ here), is shown in red, while the test datasets are shown in blue. The out-of-sample performance on the test datasets is comparable to the performance on the training dataset, which shows that the model is generalizing very well with no overfitting occurring. When comparing the overall performance between the algorithms, EncoderMap is around 3% better than PCA at compressing the information into one dimension. The likely reason for that is that EncoderMap is non-linear and more expressive, while PCA is a linear method. Combining the performance data of Figure 37 with the data from the models trained on all of the other 19 datasets, we get a summary of the performance of all the models depicted in Figure 38. There, the performance for every model is presented

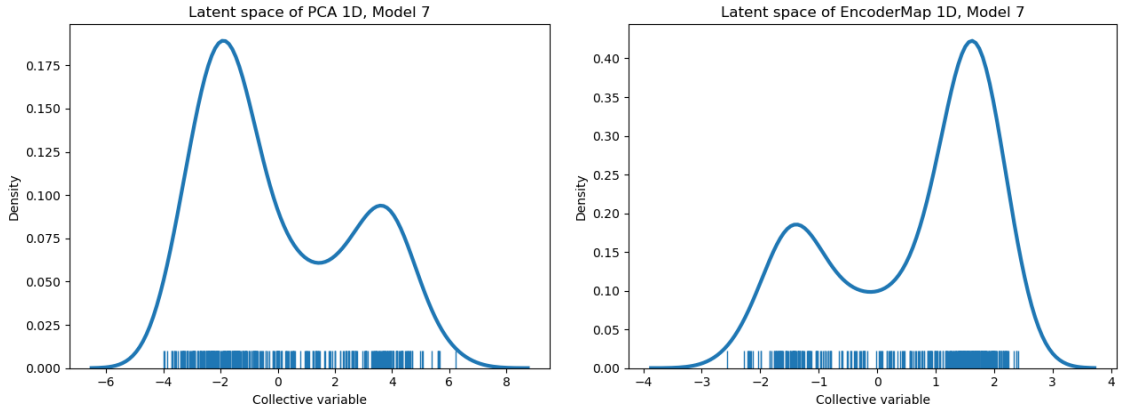


Figure 39: Comparison of the latent space of model 7 trained with the PCA (left) and EncoderMap (right) algorithm in one dimension.

along with the performance and the 2σ interval on the respective test datasets. The combined overall performance of the respective algorithm, denoted in the title, is determined as described in subsection IV.3.2. The train value for the PCA algorithm of 77.2 ± 0.4 is the reproduced value from the original publication [163]. Therein, to our knowledge, the test value is not determined, which represents a more accurate measure of the performance, since it reflects the performance on unseen data. The overall performance of the EncoderMap algorithm is on average 3.7 percentage points better, while having a comparable error. This likely is again due to the non-linear nature of EncoderMap. The performance on the test datasets is comparable to the performance on the training datasets for both algorithms. Consequently, this excludes overfitting and gives reason to believe that trustworthy models are produced by both algorithms.

In Figure 39, the latent space of model 7 trained with the PCA algorithm is plotted on the left hand side, while the EncoderMap model is shown on the right hand side. The plots contain the actual data points as blue ticks above the x-axis, while also showing a kernel density estimate (KDE) on the y-axis. The kernel density estimate depicts a bimodal distribution, where the data is composed of two differently populated clusters. The distributions of the two different algorithms look qualitatively the same, while they are mirrored and extend differently on the x-axis. Both these differences are negligible, since the absolute values do not carry any meaning and only the relative placement is important for the collective variables. The solution of the eigenvalue problem PCA is based upon is only determined up to a sign and EncoderMap only takes into account distances. Consequently, both algorithms give a similar result in terms of latent space up to scaling and rotation.

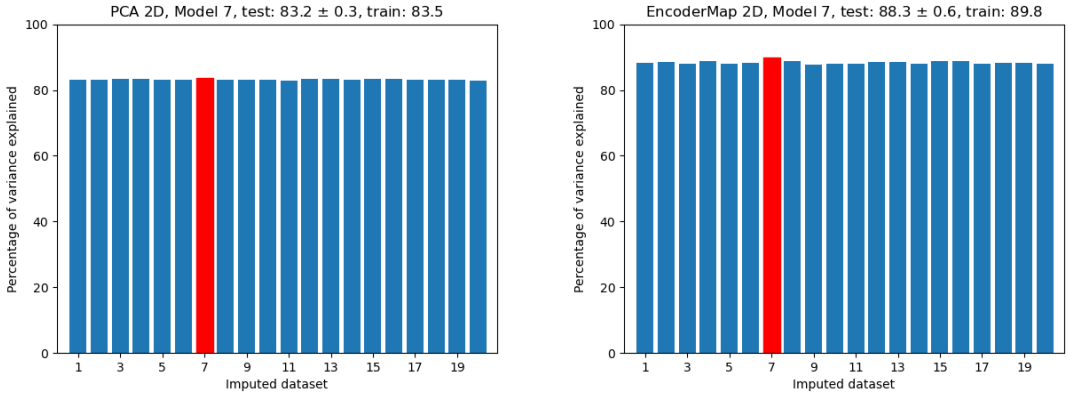


Figure 40: Comparison of performance of PCA (left) and EncoderMap (right) algorithm in two dimensions trained on imputed dataset number 7 (red bar) with the performance on the remaining out-of-sample datasets (blue bars).

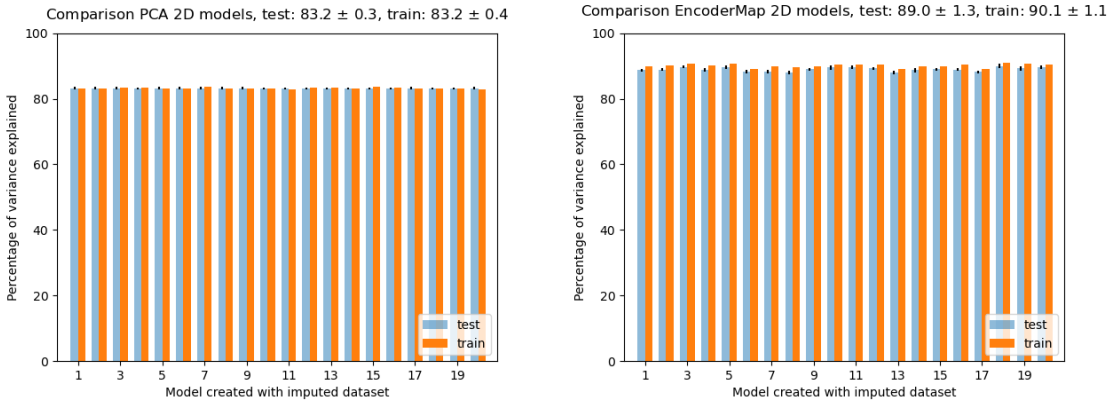


Figure 41: Comparison of performance of PCA (left) and EncoderMap (right) algorithm in two dimensions on all the imputed datasets.

IV.3.5 PCA VS. ENCODERMAP (2D)

In our quest to find a better measure for social complexity, we add another dimension to be able to retain more information of the original dataset and find a better representation.

In Figure 40 the equivalent of Figure 37 from the last chapter is visualized, here for the 2-dimensional versions of the algorithms. The red bar again denotes the performance on the imputed dataset, on which the model was trained, while the other blue bars represent the out-of-sample performances on each of the other imputed datasets. The performance on the training dataset is comparable to the performance on the test datasets for the PCA algorithm. For EncoderMap this seems not to be the case in Figure 40, as the training value lies outside of the 2σ confidence interval of the performance on the test sets.

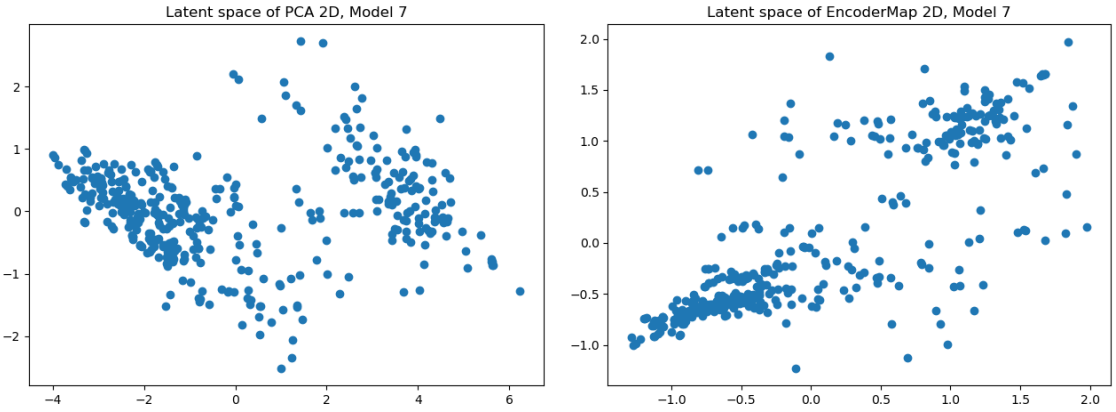


Figure 42: Comparison of the latent space of model 7 trained with the PCA (left) and EncoderMap (right) algorithm in two dimensions.

This gives the suspicion that the EncoderMap algorithm is slightly overfitting in two dimensions, which is confirmed in Figure 41 showing the comparison of performance of all the models. On the left hand side, we show the comparison of the performance for the PCA models in two dimensions, giving an overall percentage of variance explained of 83.2 ± 0.3 on the test datasets. This is comparable to the value on the training data of 83.2 ± 0.4, which reproduces the value for two dimensions of the original paper (SI of [163]). Therefore the model generalizes well on unseen data and appears to learn reasonable, general structures in the data.
The comparison of the performance of EncoderMap in two dimensions on the right hand side tells a slightly different story: Here, we find that the performance on the training data is always better, for each model, compared to the test data, and that the variation on the performance is almost twice as high as for the 1-dimensional case. This hints to a slight overfitting - the model learning some non-linear relationships in the training dataset, which do not generalize to the unseen test data. The overall performance on the test data, which is the most trustworthy value, is with 89.0 ± 1.3 on average still 6 percentage points higher than the PCA algorithm, though. Since all the models perform similarly well, we stay with model 7 in two dimensions to investigate further.
In Figure 42, the 2-dimensional latent space of model 7 is plotted, for the PCA algorithm on the left hand side and for EncoderMap on the right hand side. The latent space of the 2-dimensional PCA algorithm looks like two ellipsoid clusters situated parallel to each other. This pattern is qualitatively the same for all PCA models in 2D, up to scaling and rotation. The EncoderMap latent space in two dimensions on the other hand consists of two densely populated clusters, which seem to be located around a straight line, connecting the two with a sparsely populated gap in between. Outside the densely populated clusters, there are randomly distributed outliers, which spread out perpendicular to the line outside the clusters and the gap. This pattern is qualitatively the same for all EncoderMap models in 2D, up

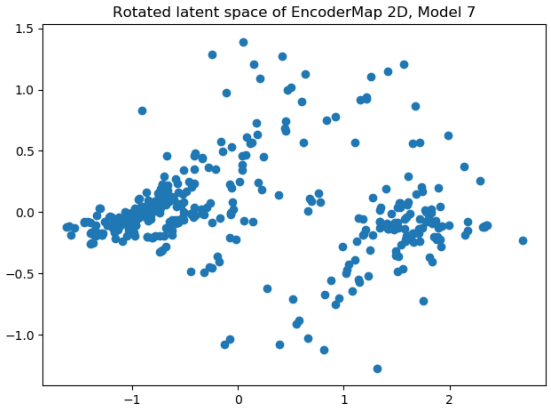


Figure 43: Rotated latent space of EncoderMap model 7 (Figure 42 right), aligned along the axes with PCA.

to scaling and rotation. It should be mentioned that since PCA is a linear model, the first axis in the 2-dimensional model is the same as in the 1-dimensional one, while EncoderMap in two dimensions has no such connection to the 1-dimensional case. Therefore, the first axis in the PCA model still represents the most variation in the data and the second axis the second-most. In contrast, the orientation in the EncoderMap latent space is arbitrary and the axes do not carry any meaning. It is possible to align the data along the axes using PCA, used here to find an optimal rotation and not as a dimensionality reduction algorithm. This rotated version of the EncoderMap latent space of model 7 in two dimensions is visualized in Figure 43. We will use both versions of the EncoderMap latent space interchangeably on the following pages, depending on suitability.

IV.4 EXPLORATORY DATA ANALYSIS IN THE LATENT SPACE

In the following section, we concern ourselves with exploring the dataset in the 2-dimensional latent spaces, created by the PCA and EncoderMap algorithms. We examine these spaces to verify that these dimensionality reduction methods extract meaningful information from the high-dimensional spaces, as well as to visualize and gain insight into the dataset.

IV.4.1 INDEPENDENCE OF THE DATA FROM REGION

To be sure that the complexity measure is independent of the region, we plot the latent spaces of both of the algorithms and color the data points according to the

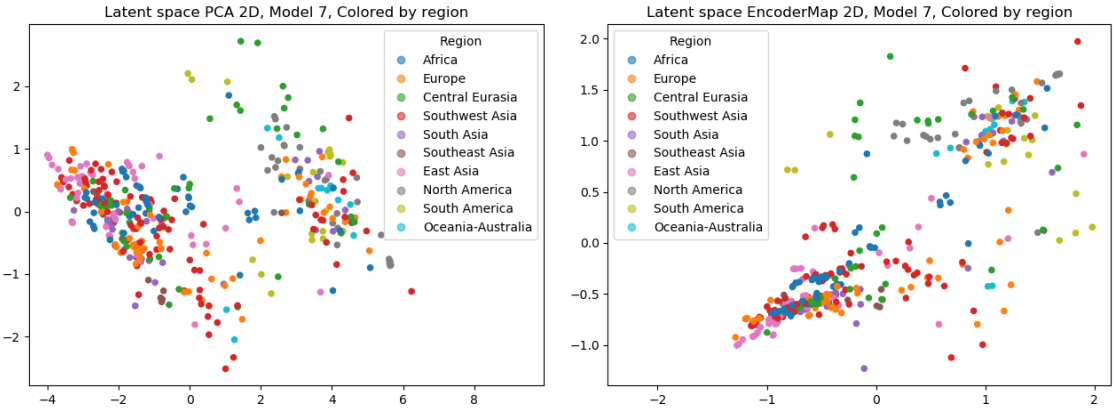


Figure 44: Comparison of the latent spaces of model 7, trained with the PCA (left) and EncoderMap (right) algorithm in two dimensions, colored by region. The latent spaces do not separate the data by region, which shows that the latent spaces represent general properties of societal development.

regions, which were introduced in section IV.1. If the models find relationships in the development of societies which are general and are not connected to particular regions, we expect that the regions are not clustered together in the latent space, but rather randomly distributed. The respective plots in Figure 44 exhibit exactly this property with the PCA latent space to the left and the EncoderMap one to the right.

IV.4.2 TIME SERIES BEHAVIOR

The previous analyses did not take into account the ordered structure of the data, namely that we are dealing with time series. Acknowledging the time-series character of the data, we end up with 30 time series, one for each NGA defined in section IV.1. The distribution of the lengths of the time series is plotted in Figure 45. More than half of the time series (16 out of 30) span less than 2,000 years, with several very short ones only covering a few centuries. The longer time series vary greatly, with lengths of up to over 11,000 years.

IV.4.2.1 Influence of time on the latent space

To get an idea of the behavior of the data, we plot the latent space colored by absolute time in the standard Gregorian calendar. In Figure 46 the data is visualized in the respective latent spaces, PCA to the left and EncoderMap to the right, with the color encoding the year of the data points. The brighter (the more yellow) the points are, the more recent is the data, while the darker colors represent older data points. Taking into consideration the colors, going from black to blue to green to yellow, the evolution of the data over time can be reconstructed in the latent space. We observe

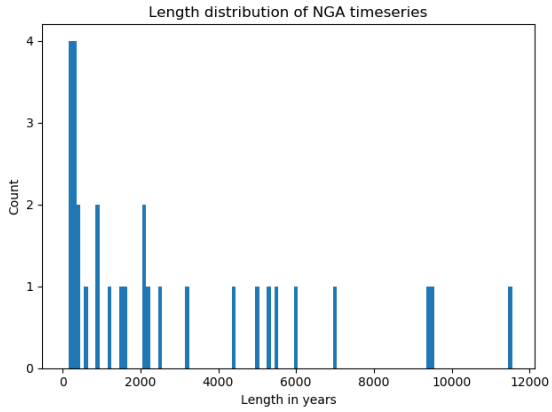


Figure 45: Distribution of length of the 30 time series in imputed dataset 7, each from one natural geographic area (NGA). The distribution is leaning towards shorter time series of a few thousand years.

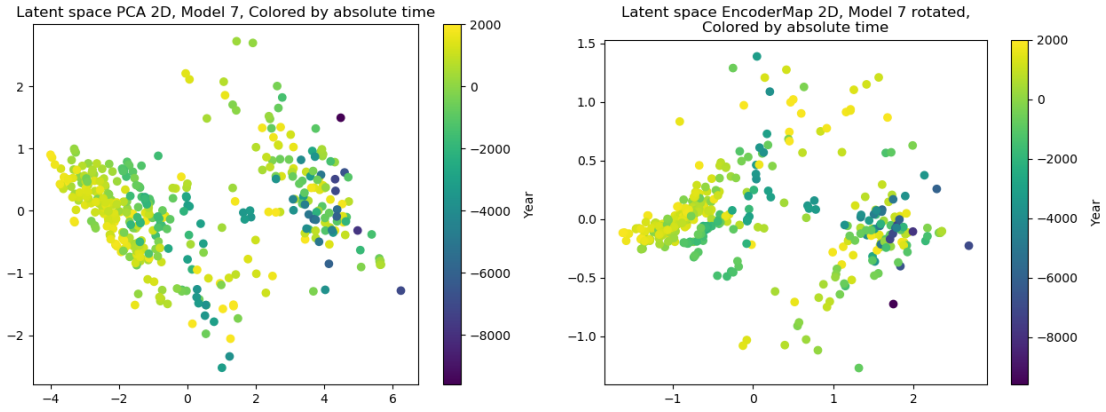


Figure 46: Comparison of the latent space of model 7 trained with the PCA (left) and EncoderMap (right) algorithm in two dimensions, colored by absolute time in the Gregorian calendar.

that most of the recent brightest points lie on the very left of the left cluster, while most of the darkest old points belong to the right cluster or lie to the far right of the left cluster. The overall temporal pattern – going from mostly older points to mostly newer points, when going from right to left – suggests that social complexity increases in this direction, while there are still a lot of more recent points found in the area of lower complexity.

Another perspective on the data is visualized in Figure 47, where the color encodes the relative position in their respective time series as the fraction passed. We observe that most of the 30 time series either start somewhere in the right cluster or far to the right of the left cluster, while most of the time series end to the left of the left cluster, which we identify as the location of higher complexity. This implies that most time series either start and end in the left cluster or they start in the right cluster and end in the left cluster, while having an unknown amount of transitions

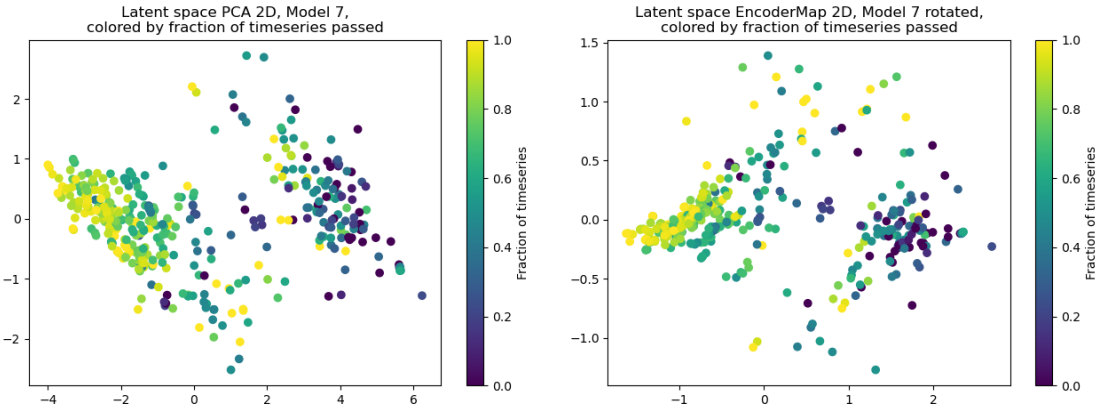


Figure 47: Comparison of the latent space of model 7 trained with the PCA (left) and EncoderMap (right) algorithm in two dimensions, colored by the fraction of the respective time series passed.

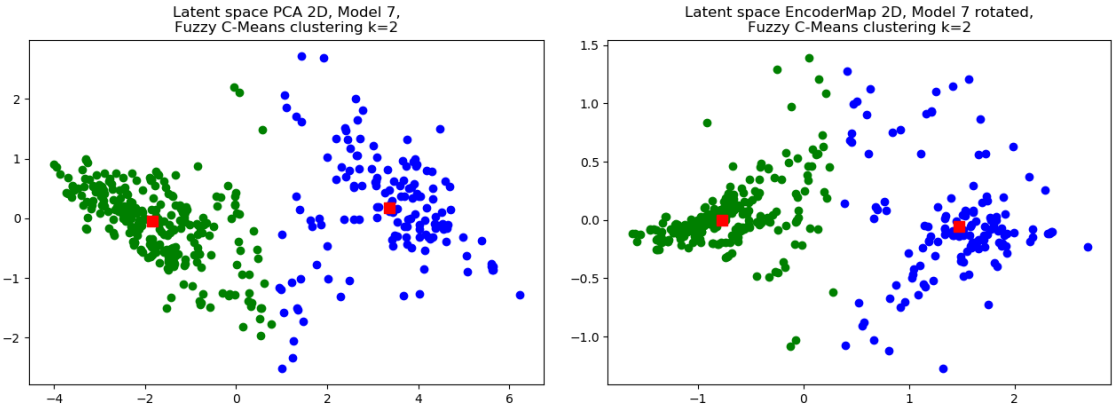


Figure 48: Comparison of the latent space of model 7 trained with the PCA (left) and the EncoderMap (right) algorithm, both clustered with c-means.

in between. To investigate this behavior further, we look at transitions between the clusters in the following chapter.

IV.4.2.2 Transitions between clusters

To investigate the transitions between the clusters, we first need a model to be able to associate the points to its respective clusters. There exists a plethora of different clustering methods, the most prominent being the k-means clustering algorithm. The k-means algorithm only gives a hard clustering, however, separating the data into distinct (Voronoi) cells. We use a different, soft clustering approach, which gives a measure of membership to each of the clusters for all points, more specifically the fuzzy c-means algorithm. This algorithm is a soft clustering extension of the k-means algorithm (see also subsection II.3.2.2).
In Figure 48 the results of clustering the data into $k = 2$ distinct clusters is depicted with the PCA latent space on the left and EncoderMap on the right. The cluster

Start \ End	Blue	Green
Blue	11	16
Green	1	2

(a) PCA 2D

Start \ End	Blue	Green
Blue	11	16
Green	1	2

(b) EncoderMap 2D

Table 1: Comparison of the start and end clusters of the 30 NGA time series in the latent space of PCA(left) and EncoderMap(right). In the respective latent spaces the blue cluster (right) is associated with lower complexity while the green cluster (left) is associated with higher complexity.

centers are shown as red squares. In both cases the algorithm captures the clusters well and the centers are located in the dense regions. In between the dense cluster regions there exists a transition region, which is populated less densely with points, while these points are distributed on a wider range along the second axis. Although the fuzzy c-means algorithm is a soft clustering algorithm, the memberships are depicted as hard associations, attributing each point to the cluster with the highest certainty. This is comparable to using the k-means algorithm, but we will make use of the uncertain nature of the assignments later on. Having established the clusters, we can now analyze the transitions between them.

To verify the implications of Figure 47 that the majority of the time series start in the right hand cluster and end in the left hand one, we look at the statistics of the start and end clusters of the time series. The results of this analysis can be found in Table 1. It immediately stands out that the statistics of both of the different latent spaces match. This does not mean, however, that the mapping of the points to the clusters is the same and it will turn out not to be. It can be a hint though that the assignment may be similar. Inspecting the table, we find that 27 out of 30 time series start in the blue cluster of lower complexity, while only a slight majority of 18 out of 30 time series end up in the green cluster of higher complexity.

Since Table 1 does not make a statement about the amount of transitions in each time series, we provide this information in Figure 49. Here, we find that the cluster assignments are indeed not identical and that transitions between the clusters appear to be rare in general. Slightly less than half of the time series (13 out of 30) do not contain a transition at all, while about half of the time series contain only one transition and a minority of a few time series contain three transitions. Combining Figure 49 with Table 1 leads to the conclusion that all of the 13 time series which start and end in the same cluster (the diagonal of Table 1) do not contain a transition at all. Furthermore, most of the time series that contain a transition only transition once from the blue to the green cluster.

The only difference between the different latent spaces in Figure 49, is, that in the PCA latent space there exists one more time series with three transitions, which in contrast appears to have only one transition in the EncoderMap latent space. It was

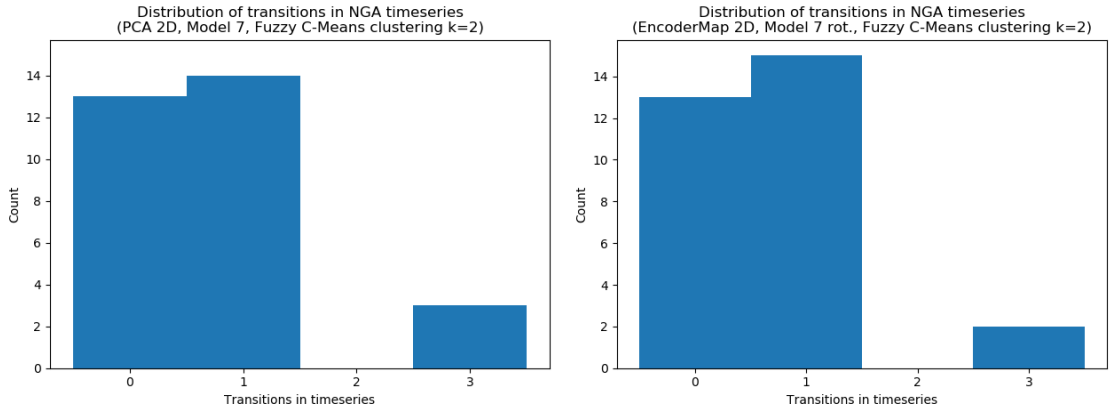


Figure 49: Comparison of the distribution of transitions between clusters in the 30 NGA time series for the latent space of PCA (left) and EncoderMap (right). Most time series exhibit a single transition or no transition at all. The similar distributions suggest that both latent spaces capture similar trends.

stated before that the less densely populated region between the cluster centers can be interpreted as a transition region between the dense clusters. This could be the reason for the difference of transitions between the latent spaces. Since the decision boundary – which point is assigned to which cluster – is exactly in the middle between the cluster centers, this boundary is somewhat arbitrary. It is not apparent to which cluster points in the transition region belong and whether they belong to a cluster at all. To consider this argument, we introduce a transition region between the clusters and examine the impact of the transition region on the transition statistics.

As we use a soft clustering algorithm, the algorithm provides us for each point with a measure of certainty that the point belongs to each of the clusters. To use a consistent criterion for declaring points as transitory, we take a fixed percentage of all the points (here 15%), for which the uncertainty is highest. More specifically, we declare a fixed fraction of points as transitory, which exhibit the property that the absolute difference between the measure of certainty for both of the cluster assignments is minimal. This naturally gives points in the region between the clusters, where the algorithm is least certain about the cluster assignment. A fixed percentage is chosen so that the transition regions between PCA and EncoderMap can be compared.

The respective latent spaces with the clustering and the transition region are shown in Figure 50, with the PCA latent space on the left and the EncoderMap on the right. The criterion for a transition state, defined above, defines a reasonable transition region as expected, although the margin now becomes a new parameter which has to be chosen and is somewhat arbitrary.

In Table 2 the statistics of the start and end clusters of each of the 30 NGA time series is shown, with the transition state as an additional cluster the points can be assigned to. Naturally, a fraction of the previously hard clustered time series now changes to the transitory state. In contrast to the hard clustering in Table 1, the statistics do not agree any more between the methods. The tables share common

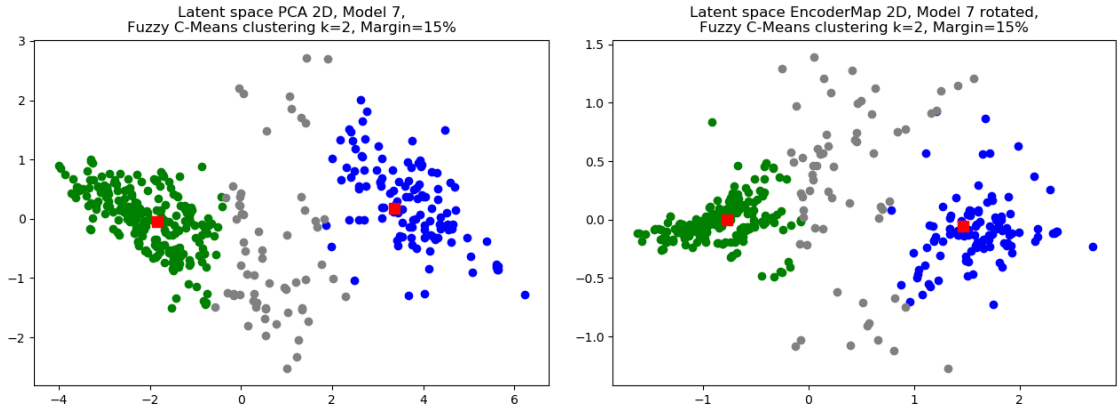


Figure 50: Comparison of the latent space of model 7 trained with the PCA (left) and the EncoderMap (right) algorithm, both clustered with c-means and a declared transition margin of 15%.

Start \ End	Blue	Green	Gray
Blue	7	12	6
Green	0	2	0
Gray	0	0	3

(a) PCA 2D

Start \ End	Blue	Green	Gray
Blue	8	14	5
Green	0	2	0
Gray	0	0	1

(b) EncoderMap 2D

Table 2: Comparison of the start and end clusters of the 30 NGA time series in the latent space of PCA (left) and EncoderMap (right). In the respective latent spaces the blue cluster (right) is associated with lower complexity while the green cluster (left) is associated with higher complexity. In between the clusters is a transition region which belongs to none of the clusters (gray).

ground in the statement, however, that most of the time series start in the blue cluster of low complexity (25 for PCA, 27 for EncoderMap) and approximately half of them end up in the green cluster of high complexity (12 for PCA, 14 for EncoderMap). Apparently, for a fraction of the time series which started in the blue cluster, the end cluster was not as certain as implied in the previous analysis and they are now categorized as ending in the transition region (6 for PCA, 5 for EncoderMap).
Another important point the statistics in both latent spaces agree on, is, that none of the 30 time series end in a cluster with lower complexity than they started in. Of course the complexity may fluctuate, may drop within one cluster and even transitions might occur, but the general trend appears to be towards higher social complexity in the long run. This confirms the general trend implied in Figure 46 and Figure 47, where the temporal evolution of the data in the latent spaces is shown. Naturally, we expect this trend to be reflected in the data, as we all know the development of the human species over the last centuries. Furthermore, the statistics in both latent spaces appear to agree on two time series starting and ending in the green cluster of higher complexity, while disagreeing about the amount of time series starting and

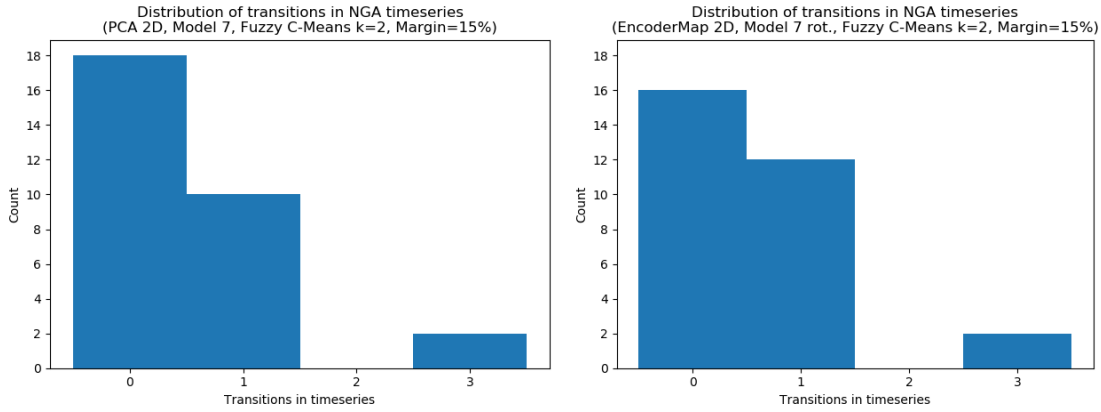


Figure 51: Comparison of the distribution of transitions between clusters in the 30 NGA time series for the latent space of PCA (left) and EncoderMap (right), with a declared transition margin of 15% of the data. Most time series exhibit a single transition or no transition at all.

also ending in the transition region (3 for PCA, 1 for EncoderMap). Since not all of the time series span several millenia, the two time series which start and end in the green cluster, suggest that they are among the more recent ones.
In Figure 51, the distribution of the amount of transitions in the 30 NGA time series is visualized, with the PCA latent space on the left and EncoderMap on the right. Here we only count the transition when it is from one cluster to the next, surpassing the transition region. In both latent spaces there exist two time series with three transitions and around 10 with one transition (10 for PCA, 12 for EncoderMap). Since the amount of time series in Table 2, which start in the blue cluster of low complexity and end up in the green cluster of high complexity, exactly matches the amount of time series in Figure 51 which exhibit transitions, we conclude that these are the same time series. All other time series do not contain transitions between the clusters, albeit containing transitions into the transitory region.

IV.4.2.3 Region time series in latent space

As described in section IV.1, the data consists of 30 time series from natural geographic areas (NGAs), while always three belong to a particular region respectively. In this chapter, we visualize these three time series respectively for several regions in the respective latent spaces to give some exemplary insight into the behavior of a selection of different NGA time series. The rest of the data is slightly implied in the background to give context to the trajectories of the time series. The points in the time series are color-coded by the year in the Gregorian calendar.
In Figure 52 the three time series for the Europe region are shown in the latent space of PCA to the left and for EncoderMap to the right. The time series for Latium and the Paris Basin contain a single transition from the right cluster of lower complexity to the left cluster of higher complexity at around year zero, probably related to the

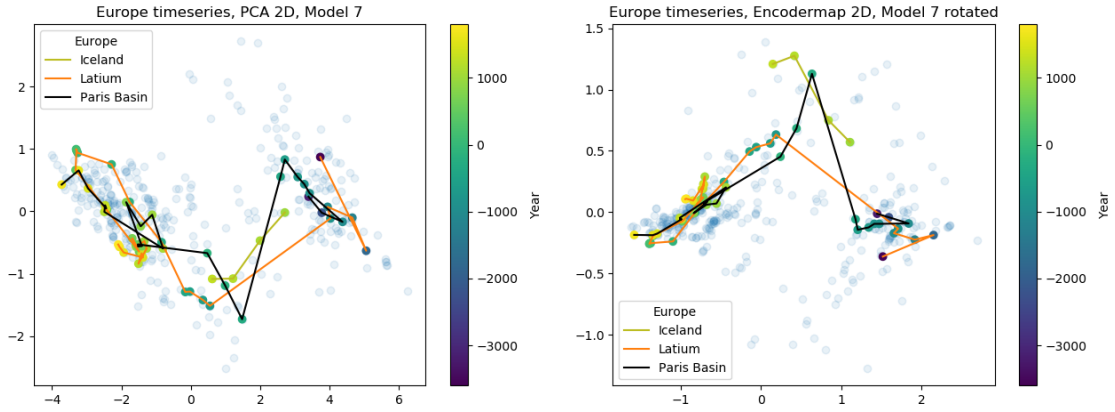


Figure 52: Comparison of the time series in Europe in the latent space of PCA (left) and EncoderMap (right) in two dimensions colored by region.

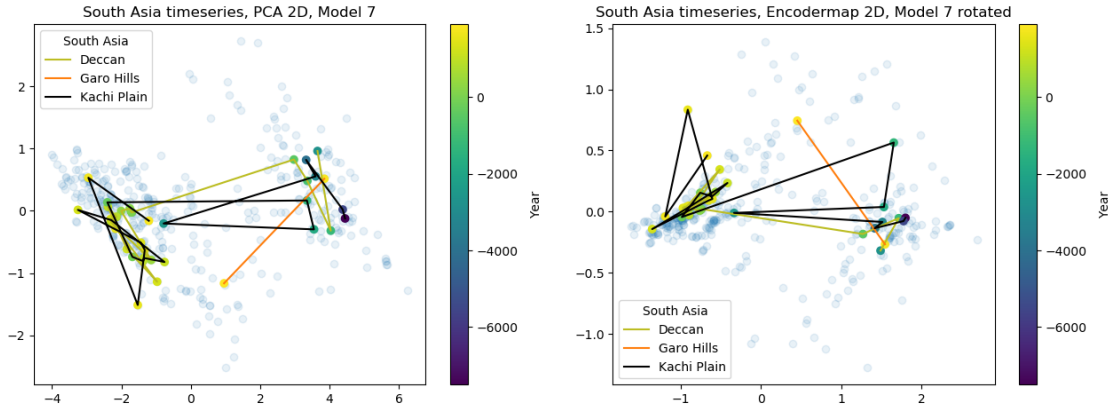


Figure 53: Comparison of the time series in South Asia in the latent space of PCA (left) and EncoderMap (right) in two dimensions colored by region.

Roman Empire. In contrast, the time series for Iceland is way more recent, starting around year 1000 in the right cluster of lower complexity and only manages to progress into the transition region.

In Figure 53 the three time series for the South Asia region are visualized in the latent space of PCA to the left and for EncoderMap to the right. The time series for the Kachi Plain is one of the time series exhibiting three transitions in both latent spaces. The transitions are probably related to the data gathered from the Neolithic archeological site Mehrgarh, which is situated in the Kachi Plain and today belongs to Pakistan. The evidence dates back to starting between years -7000 and -5500 and the city appears to have been abandoned between years -2600 and -2000, which probably constitutes the back-transition in the data.

In Figure 54 the three time series for the Southeast Asia region are visualized in the latent space of PCA to the left and for EncoderMap to the right. It stands out that all of the time series in this region contain relatively few data points and that none of the time series contain a transition between the clusters. In fact, the time series

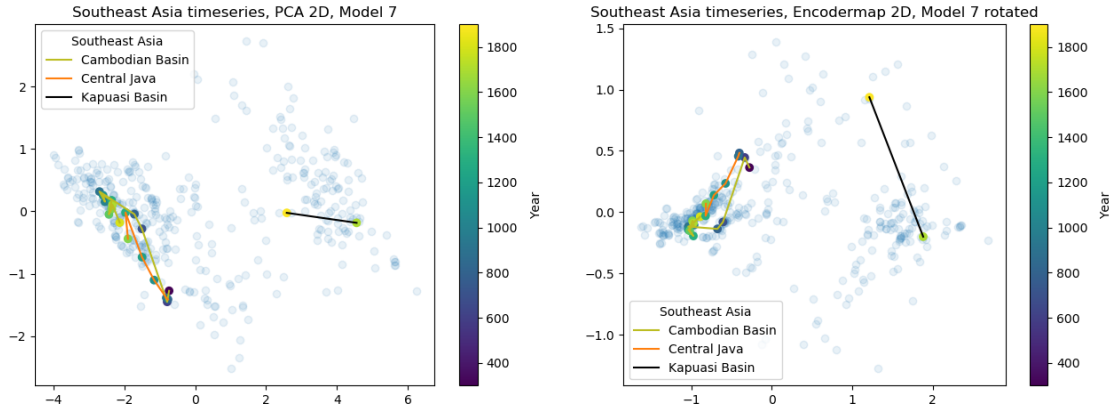


Figure 54: Comparison of the time series in Southeast Asia in the latent space of PCA (left) and EncoderMap (right) in two dimensions colored by region.

for the Cambodian Basin and the Central Java natural geographic regions are the only two time series, which start and end in the green left cluster in Table 2. The color scale shows that all of the time series span only comparatively recent times, starting around year zero.
When comparing the time series in the latent spaces of PCA and EncoderMap, we observe qualitative agreement for a lot of the visualized trajectories (while the directions along the y-axes in latent space appear to be mirrored). This implies that both of the algorithms pick up on similar relationships in the data.

IV.4.3 INTERPRETING THE LATENT SPACE

In this section we want to investigate the effect the input variables in the high-dimensional space (i.e. the CCs) have on the latent variables in the low-dimensional spaces and explore if certain input variables drive the latent variables predominantly.

IV.4.3.1 Correlations between complexity characteristics

One thing to look at first is the correlation in the original data between the complexity characteristics defined in section IV.1. Let X_i be the data of the $i \in \{1, \ldots, 9\}$ respective complexity characteristics. Then the (Pearson) correlation coefficient between two complexity characteristics $i, j \in \{1, \ldots, 9\}$ is defined as

$$\rho_{X_i, X_j} = \frac{\mathbb{E}\left[(X_i - \mu_i)\,(X_j - \mu_j)\right]}{\sigma_{X_i} \cdot \sigma_{X_j}}, \tag{IV.11}$$

with μ_i, μ_j being the means and σ_i, σ_j being the standard deviations of the respective CCs. The correlation matrix in Figure 55 shows all possible correlations between the different CCs. The values in the matrix shown are determined by calculating the

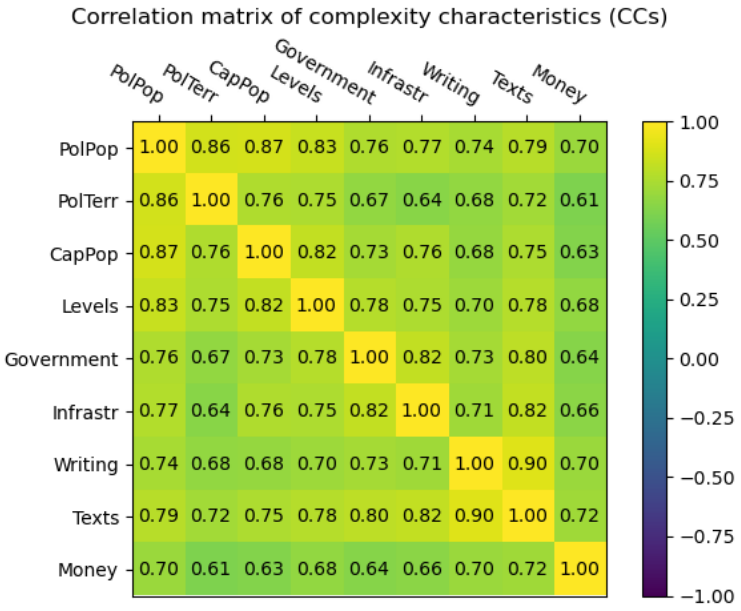


Figure 55: Correlation between complexity characteristics (CCs). The overall high correlation between the CCs suggests that the information can be compressed into fewer variables.

correlation matrix for each of the 20 imputed datasets and then taking the average. We find the behavior to be very consistent over all of the 20 imputed datasets with the maximum standard deviation being $\sigma_{max} = 0.013$ and we therefore do not show the respective standard deviations.

The high correlation values imply that each of the complexity characteristics contain overlapping information. This is often a prerequisite for dimensionality reduction since this shared information can then be described by fewer variables. As the Pearson correlation only captures linear relationships, the correlations shown in the correlation matrix can be exploited by both, linear and non-linear dimensionality reduction techniques, namely PCA and EncoderMap that we use in this scenario. There may exist further non-linear relationships, however, which only non-linear techniques like EncoderMap can take advantage of. On the one hand high correlation in the high-dimensional variables is important for dimensionality reduction techniques to be able to find good low-dimensional representations, on the other hand it complicates interpreting the resulting mappings, as we will see later. Sensitivity analyses are often distorted by (already approximate) colinearities in the input variables, since it is unclear in which manner effects on the target values are supposed to be attributed to the colinear input variables.

IV.4.3.2 PCA specific analysis methods

As PCA is a linear method, we have certain methods for interpreting the dimensionality reduction mapping available, for which no equivalent exists for non-linear

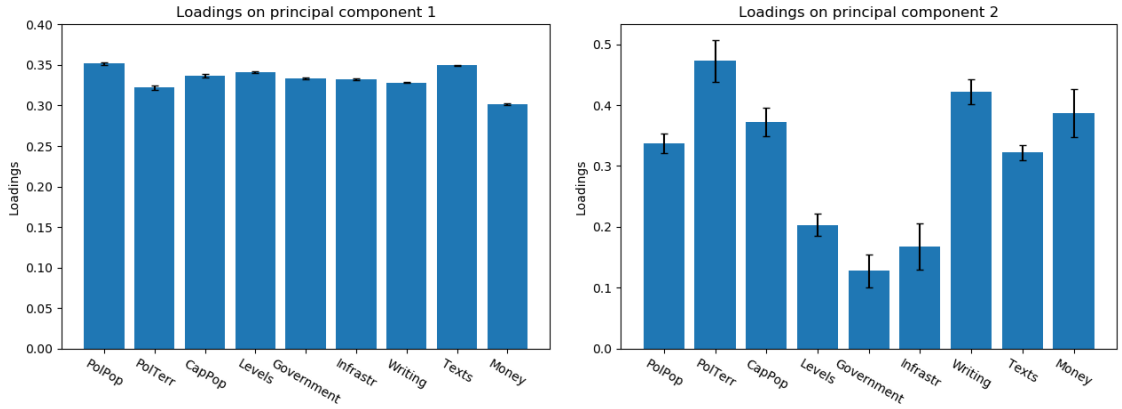


Figure 56: Absolute loadings of the complexity characteristics on the first principal component (left) and on the second principal component (right). The absolute loadings shown here can be interpreted as a form of feature importance.

methods. In Figure 56 the loadings of the nine complexity characteristics (defined in section IV.1) on the first two principal components are shown. The loadings can be understood as the impact the CCs have on the position along the respective axis and therefore represent a form of feature importance. To get the average loadings over the 20 imputed datasets, the PCA algorithm was applied to each of the imputed datasets and the average of the absolute loadings was determined. The error bars are the standard deviations belonging to the respective values. The left hand side of Figure 56 shows that each of the complexity characteristics load equally strongly on the first principal component, which can be interpreted as each of the complexity characteristics equally impacting this 1-dimensional measure of social complexity. Hereby, the loadings are very consistent over all the imputed datasets, as they have a vanishingly small error bar. This plot is basically reproducing the plot in the lower right corner of Figure 33, which was reproduced from the original publication [163]. The loadings on the second principal components are varying more on both, between the complexity characteristics as well as between the different imputed datasets, as is implied by having a larger error bar on the values. The complexity characteristics of levels, government and infrastructure seem to load less on the second principal component, while the others appear to load approximately the same on it.
Another way to visualize the loadings is a PCA biplot, which is shown in Figure 57. Here, the directions are shown in which the complexity characteristics drive the values in the latent space. The vectors are scaled versions of the values of Figure 56 and were shifted away from the origin to increase visibility. By imagining the projection of the vectors on the first axis, one can see that all of the complexity characteristics load on the first principal component equally. All of the complexity variables drive the points in the latent space to the left, which is the direction of higher social complexity. Imagining the projections on the second axis, we reproduce that levels, government and infrastructure load less on the second principal component than the

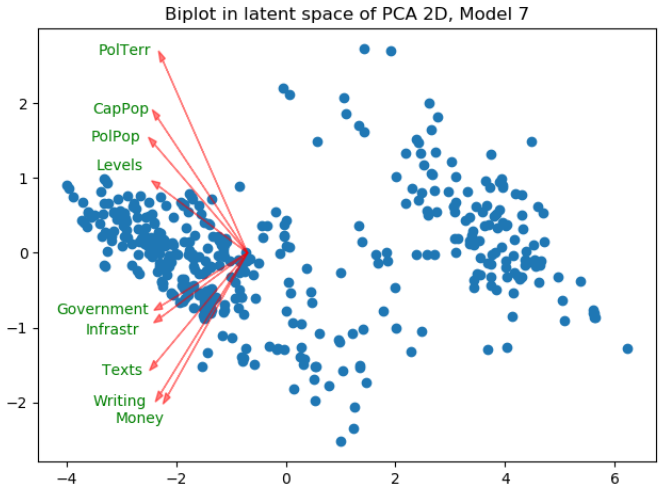


Figure 57: PCA biplot showing the direction of the high-dimensional complexity characteristics (input variables) in the latent space. The arrows show the directions in which a point is driven, when the respective complexity characteristics (CCs) are increased. Higher CCs all drive the points in latent space to the left (higher social complexity), while CCs of scale drive the points up and non-scale CCs drive the points down.

other complexity characteristics. In the original paper [163], it was hypothesized that social complexity is driven by two distinct groups of complexity characteristics:

1. Complexity characteristics of scale:
   Encompassing variables of size (polity population, polity terrain, capital population), combined with the variable describing hierarchical organization (levels).

2. Non-scale complexity characteristics:
   All the other variables, representing organisation and technology, e.g. amongst others, specialization of roles and products.

These two groups of variables appear to drive the second principal component in Figure 57. The complexity characteristics of scale drive the points in the latent space in the direction of the second principal component (up), while the non-scale complexity characteristics drive the points in the opposite direction (down). The value of the second principal component can thus be interpreted as a form of imbalance between complexity characteristics of scale and non-scale.

IV.4.3.3 Correlation with latent space variables

Another way to investigate feature importance is to determine the correlation between each of the input variables (CCs) with each of the latent space variables.
The correlation of all the complexity characteristics with the principal components of PCA is shown in Figure 58, with the correlations with the first principal component being on the left and the correlations with the second principal component on the right. Notably, the left figure shows a strong negative correlation of all the complexity

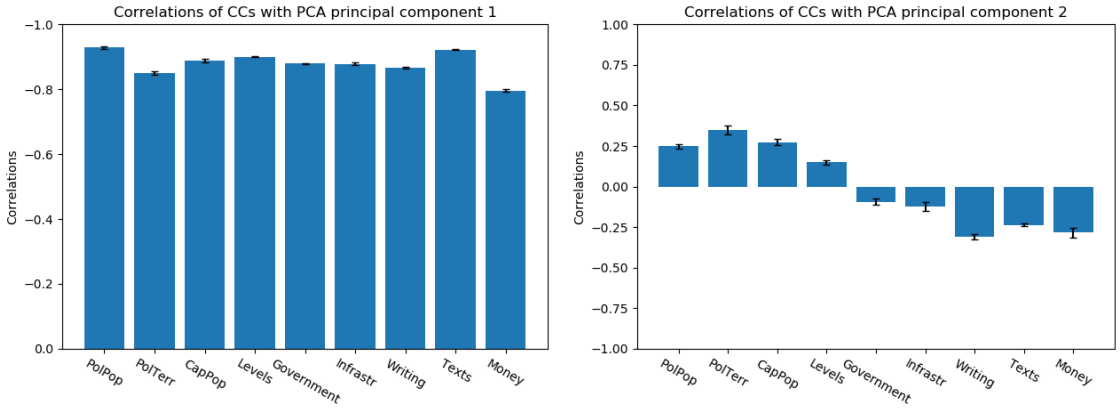


Figure 58: Correlations of the complexity characteristics with the first principal component (left) and the second principal component (right) of PCA. All correlations with the first principal component are negative (left) as increases in all CCs drive the respective point in the latent space towards negative x-values, which represent higher complexity (note all arrows pointing to the left in Figure 57). There exist two groups of CCs - distinguished by positive and negative correlation with the second principal component (right) - which drive the respective point in the latent space along or against the y-axis, when increased (note the groups of arrows pointing up and down in Figure 57).

characteristics with the first principal component, showing again that higher values in all of the complexity characteristics drive values in the PCA latent space to the left. The small error bar in the correlations implies that this behavior is consistent over all the models built from different imputed datasets.

Since the sign of the direction of the principal components is not unique, some of the axes of the second principal components were flipped. To make the correlations comparable, the dominating directions of the models over the 20 imputed datasets were chosen, which are the directions visualized in Figure 57. The correlations with the second principal component (on the right hand side of Figure 58) are notably weaker than with the first one, but are still relatively consistent over the 20 datasets, which hints at them not only being random noise. The first four complexity characteristics are positively correlated with the second principal component, while the other five are negatively correlated with it, which is equivalent to what was already found in Figure 57. In fact, it can be shown that the correlations in PCA are equivalent to the loadings scaled by the square root of the respective eigenvalue, which explains the similarity to Figure 56.

In Figure 59 the correlations of the complexity characteristics with the axes in latent space of EncoderMap 2D are shown, with the correlations with the first axis on the left and the correlations with the second axis on the right. The correlations with the first axis appear to be similar to the first axis of PCA in Figure 58. As the Pearson correlation coefficient only measures linear correlation, this suggests the conclusion that the first axis of EncoderMap captures at least the same linear relationships as the first principal component of PCA with possible further non-linear relationships

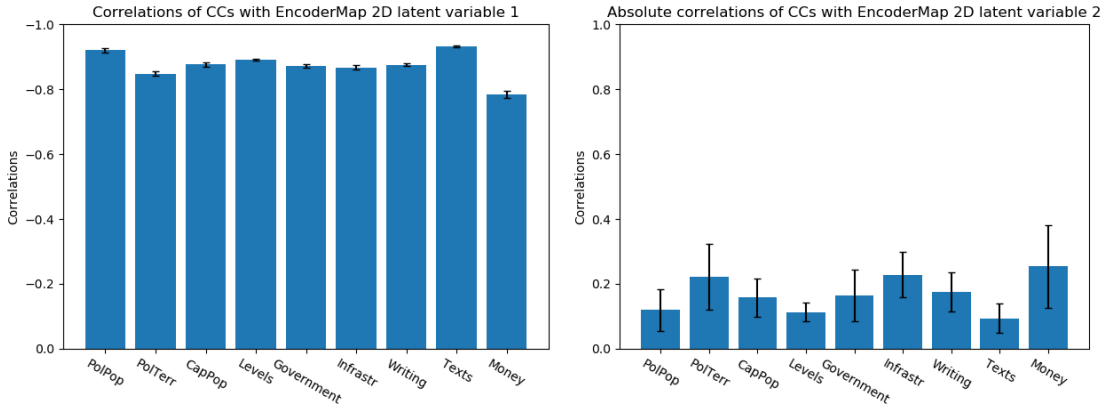


Figure 59: Correlations of the complexity characteristics with the first latent variable (left) and absolute correlations with the second latent variable (right) in the EncoderMap 2D latent space of model 7. Similar to PCA, an increase in all CCs drives the respective point in EncoderMap latent space towards negative x-values in the direction of higher complexity (expressed by the negative correlation of all CCs along latent variable 1, left figure). Along the second latent variable the impact of the CCs differs between the imputed datasets as indicated by the error bars.

that are not represented in the correlation coefficient. For the correlations of the CCs with the second axis of EncoderMap 2D, the sign structure varies greatly over the 20 imputed datasets so that we take the absolute value to be able to at least compare the strength of the correlations between different models. The comparatively big error bars on the correlations in the visualization on the right side of Figure 59 only leave the conclusion that the correlations with the second axis of EncoderMap 2D models are not consistent between different imputed datasets.
This can be due to several reasons. First of all, PCA guarantees by definition that the axes are independent (linearly uncorrelated), while EncoderMap does not. As all the EncoderMap 2D models are rotated to have the most variance on the first axes, the behavior on the first axis appears to be similar, but there is no such comparability between the second axes. Since there exist different sign structures for the correlations between the different imputed datasets, the models also appear to catch on to different (sometimes probably also spurious) relationships depending on the model. Another fact may be that non-linear relationships are used by EncoderMap that are not captured by the linear Pearson correlation coefficient. As the performance comparison in Figure 41 hints at EncoderMap 2D being superior to PCA in terms of performance, all these reasons probably have a say and the correlation analysis for EncoderMap 2D should be taken with a grain of salt.

IV.4.3.4 Model-Agnostic interpretation methods

With the advent of large scale machine learning and deep learning the last decades, models have become significantly more complex with hundred thousands of parameters that often do not allow for easy interpretation of their behavior. The more these

models are deployed and their decisions impact our everyday life, the more the need for interpretability grows. This is one of the main challenges the emerging fields of interpretable machine learning and explainable artificial intelligence aim to solve. This chapter is based on information from [164], which is an excellent free book on the topic.
Since EncoderMap is already considered a deep neural network, it suffers from the same lack of interpretability and we are forced to use methods from this field to get a glimpse of what it is doing with the data. In this section, we will apply the method of Shapley values on both the PCA and EncoderMap mappings to examine influence of the input variables (CCs) on the latent spaces.

Shapley values

Shapley values are a concept from cooperate game theory. They were introduced in 1951 by Lloyd Shapley [165], who was awarded the nobel price in economics for them in 2012.
Given a set $P = \{1, \dots, n\}$ of n players and a value function $\nu : 2^P \mapsto \mathbb{R}$, which measures the value $\nu(S)$ a subset of players $S \in 2^P$ create together, the Shapley values are defined as

$$\phi_i = \sum_{S \subseteq P \setminus \{i\}} \frac{|S|!\,(n - |S| - 1)!}{n!} \left(\nu(S \cup \{i\}) - \nu(S)\right), \tag{IV.12}$$

while the additional condition needs to be fulfilled that no players create no value, $\nu(\varnothing) = 0$. For a given coalition of players S, the term $\nu(S \cup \{i\}) - \nu(S)$ describes the value a player i is bringing to the coalition. The Shapley value ϕ_i for player i then represents the average value, which player i contributes over all possible coalitions. The factor in the formula is the inverse of a multinomial coefficient counting the number of ways the players in S can be ordered, making the expression independent of ordering of players. In other words, the Shapley value ϕ_i measures the average value a player i contributes to a given coalition $S \subseteq P \setminus \{i\}$.
Considering the players P collaborate and create the value $\nu(P)$ together, it can be shown that the only "fair" payout method is giving every player i their respective Shapley value ϕ_i. "Fair" in this context means that Shapley values can be shown to be the only attribution method fulfilling the properties of efficiency, symmetry, dummy and additivity:

- Efficiency:
  The value is completely distributed between the players: $\sum_{i \in P} \phi_i = \nu(P)$.
- Symmetry:
  If two players $i, j \in P$ contribute equally in the sense that they create the same value $\nu(S \cup \{i\}) = \nu(S \cup \{j\})$ for all possible coalitions $S \subseteq P \setminus \{i, j\}$ then they also receive the same payout $\phi_i = \phi_j$.

- Dummy:
  A dummy player k, who does not contribute to any coalition, i.e. $v(S \cup \{k\}) = 0$ $\forall S \subseteq P \setminus \{k\}$, does not get any payout $\phi_k = 0$.

- Additivity:
  If the players P cooperate in two games creating value $v_1(P) + v_2(P)$ then the payout for each player i is $\phi_i^1 + \phi_i^2$.

Interpreting models with Shapley values

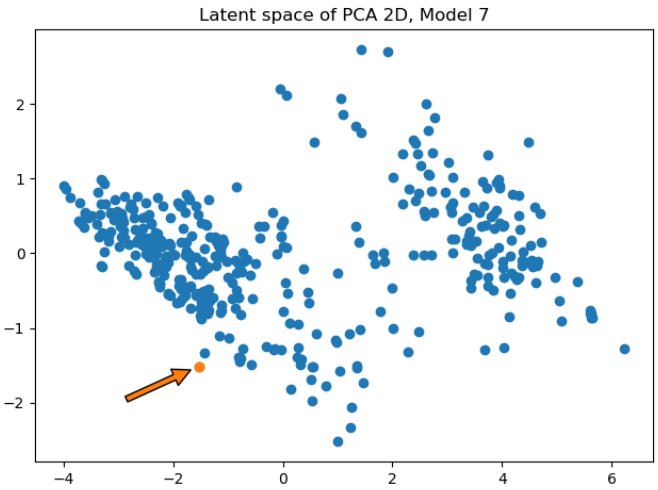
To make the connection back to interpreting machine learning models, the features (input variables) of a model can be considered as playing a game with the prediction being the value function. Shapley values of the features in this scenario then describe the fair attribution of the share in the prediction to each of the individual input variables. With this method, theoretically, we can get an interpretation for every individual prediction, in the sense that we get a decomposition of a prediction value into Shapley values of the features, which represent the share or influence of the features in this particular prediction.
In reality this idea suffers from two bottlenecks:

1. For the calculation of Shapley values the average over all subsets of features needs to be calculated. The number of subsets grows exponentially with the number of features and quickly becomes infeasible, when more features are present.

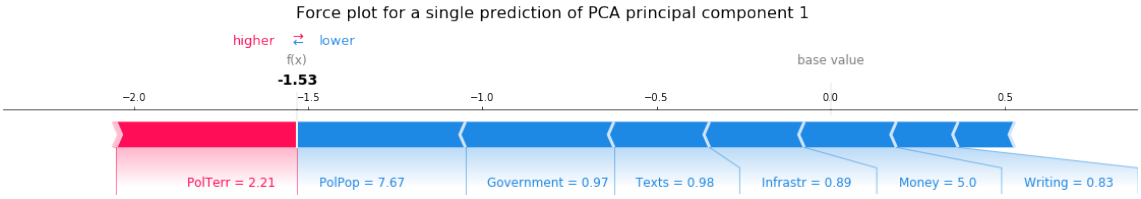
2. It is not defined in which way a feature can be removed from a prediction, since the model usually has a fixed amount of input variables. Schemes for removing a feature often involves sampling, creating and using data the model was never trained on, giving non-viable predictions. This is especially true when features are correlated, since the sampling procedures do not take correlation into account.

It follows from the Symmetry property of Shapley values that if we replace a feature with two of the exact same features, the previous Shapley value of the feature is split up between the two new equal features. This gives a hint as to what happens with correlated features: The more correlated features are, the more information they share and so the attribution for this shared information is fairly split between the two correlated features. Naturally, this complicates interpretation, since this shared information could have been an isolated feature on its own to which the value would have been attributed to in a more interpretable manner. So correlation between features leads to a smearing out of the contribution over all the correlated variables.

(a)



(b)



(c)

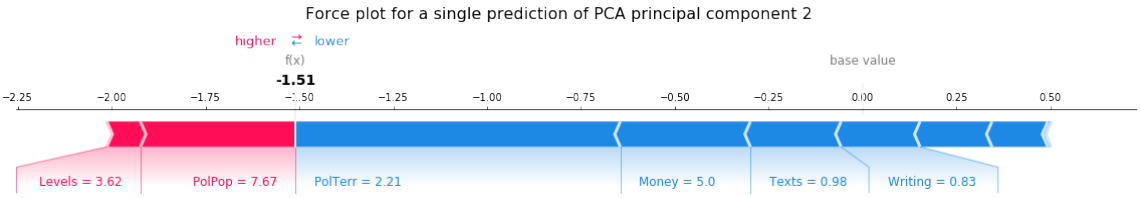


Figure 60: Interpretation with Shapley values for a single point in PCA 2D latent space of model 7, emphasized with an arrow in (a). The decomposition in Shapley values is shown in the force plots below for principal component 1 (b) and principal component 2 (c).

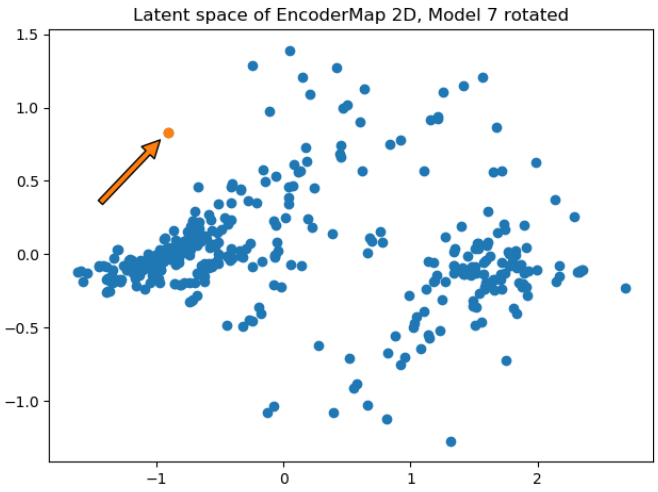
Results of Shapley analysis

To approximate the Shapley values, we use the SHAP (SHapley Additive exPlanations) package [166].

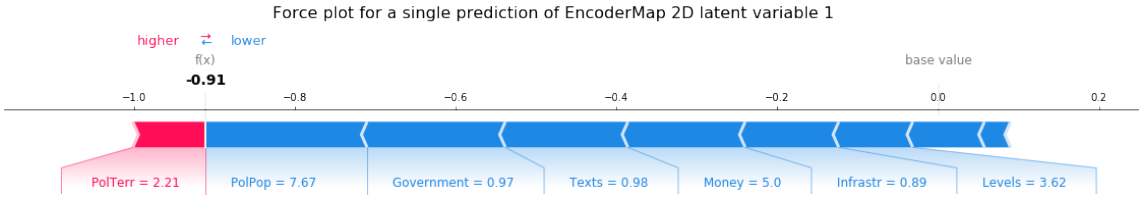
LOCAL INTERPRETATION WITH SHAPLEY VALUES

To demonstrate how interpretation with Shapley values works, we show the Shapley values for the same exemplary data point in PCA 2D latent space (Figure 60) and in EncoderMap 2D latent space (Figure 61). The interpretation is done for each of the axes independently, since a scalar value function is needed for Shapley values. The point chosen belongs to the left cluster of higher complexity (as defined in Figure 50),

(a)



(b)



(c)

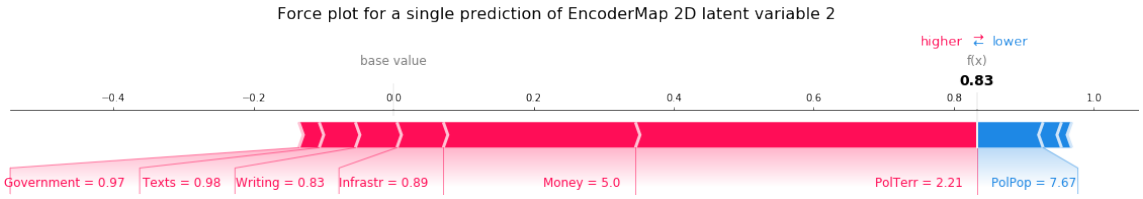


Figure 61: Interpretation with Shapley values for a single point in EncoderMap 2D rotated latent space of model 7, emphasized with an arrow in (a). The decomposition in Shapley values is shown in the force plots below for principal component 1 (b) and principal component 2 (c).

while being an outlier of varying degree along the second axis in both of the latent spaces.

The force plots in (b) and (c) show the decomposition of the position in the respective latent spaces into Shapley values for both of the axes. Since the data is centered in both latent spaces, the base value equals zero and the coordinates of each data point correspond to the sum of its respective Shapley values. As both latent spaces have different scales, the absolute Shapley values can not be compared directly, while the relative impact of the complexity characteristics on the respective predictions can be checked against each other qualitatively.

Comparing the force plot (b) in Figure 60 and Figure 61, which show the impact of the complexity characteristics on the position along the respective first axis for this particular data point, we find that the explanations are qualitatively the same. In

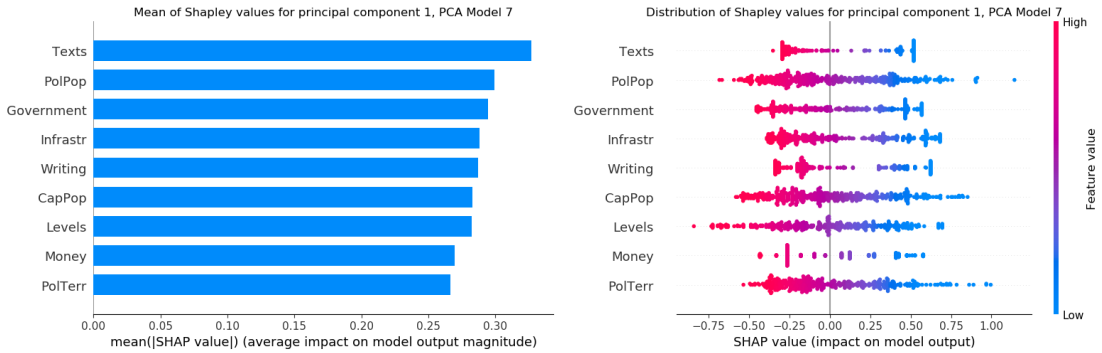


Figure 62: Mean of Shapley values over the whole dataset for the first principal component in PCA latent space of model 7 (left) and distribution of Shapley values for each complexity characteristic (right).

both plots the position in latent space is driven to the left towards higher complexity by all of the complexity characteristics except for the polity territory (PolTerr), albeit in slightly different proportions (compare (b) of Figure 60 and Figure 61). The value for polity territory appears to deviate from the usual values expected from the values of the other complexity characteristics, driving the point in latent space a bit back to the right towards lower complexity. When comparing the force plots for the second latent axis (plot (c) in Figure 60 and Figure 61 respectively) the plots appear to agree on the outlier character of the point along this axis, mainly being driven by the values for the complexity characteristics of polity territory (PolTerr) and money. Although both latent spaces exhibit a different structure and the impact of different complexity characteristics may vary between the latent spaces, Shapley values still allow to qualitatively compare the main reasons for the latent space placings.

GLOBAL INTERPRETATION WITH SHAPLEY VALUES

To obtain a global view on the influences of the complexity characteristics on the placements in the respective latent spaces, the decomposition into Shapley values analogously to Figure 60 and Figure 61, as described in the previous paragraph, is carried out for each of the data points. The results for the first principal component of the PCA latent space are shown in Figure 62. These decompositions over the complete dataset give a distribution of contributions for each of the complexity characteristics, showing a global picture of their effects on the output (right). The mean of the absolute Shapley values for each of the complexity characteristics represents a measure of the mean impact on the output and can be interpreted as a form of feature importance (left). It can be qualitatively compared to the loadings in Figure 56, which also represent a variant of feature importance. Here, the two most important (Texts, PolPop) and two less important (Money, PolTerr) features are the same while the other features are in a different order, but are still qualitatively recognized as being almost equally important.

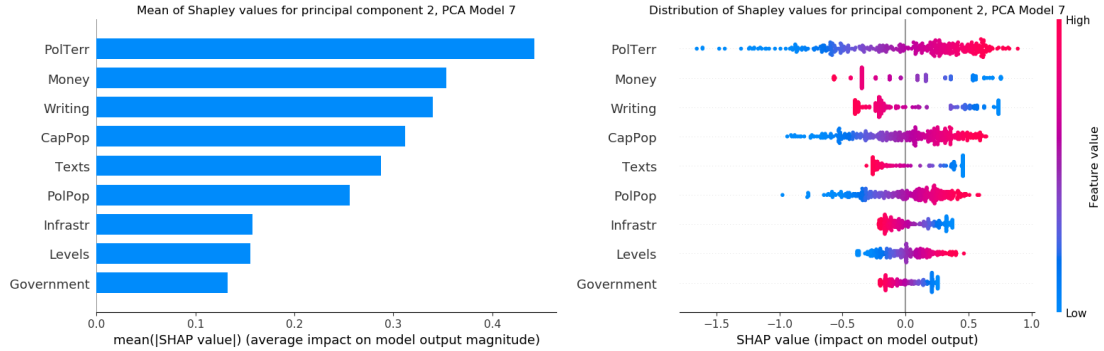


Figure 63: Mean of Shapley values over the whole dataset for the second principal component in PCA latent space of model 7 (left) and distribution of Shapley values for each complexity characteristic (right).

On the right hand side of Figure 62 the distribution of Shapley values is shown for each of the complexity characteristics. Higher feature values are represented with red color, while low feature values are in blue. That the red points of the distributions are positioned on the left, at negative Shapley values, restates the fact that higher values in all of the complexity characteristics drive points in the latent space to the left towards higher complexity, while low values do the opposite. Furthermore, all the distributions extend similarly with often having some places with more density at the edges. This already hints to all of them being more or less similarly important. The text feature being the most important probably stems from its density being concentrated at the edges. The distribution for the money feature stands out as having a discrete character. Lastly, the distributions are not symmetric around zero, which is due to the data in latent space extending further to the right than to the left.

In Figure 63 the same plots are depicted for the second principal component of PCA with the distributions of Shapley values for each complexity characteristic to the right and the mean abolute Shapley values to the left. In contrast to the distributions in the first principal component (Figure 62), the distributions in Figure 63 vary more greatly and (a) shows, for example, that the polity territory (PolTerr) feature has more impact on the second principal component than the government feature. Looking at the impact of higher feature values (red) in (b), we find that higher feature values in polity territory (PolTerr), capital population (CapPop), polity population (PolPop) and levels drive the prediction up in varying degrees in latent space along the second principal component axis, while higher feature values in the other complexity characteristics drive the prediction down. These are the same directions already found in the biplot in Figure 57. The plot showing the mean of absolute Shapley values (Figure 63 left) is best compared to the loadings plot in Figure 56 to the right. Although the order is sometimes swapped for complexity characteristics with similar importance, the qualitative feature importance found is the same. One reason for the differences could be fluctuations, as the Shapley values are only calculated for

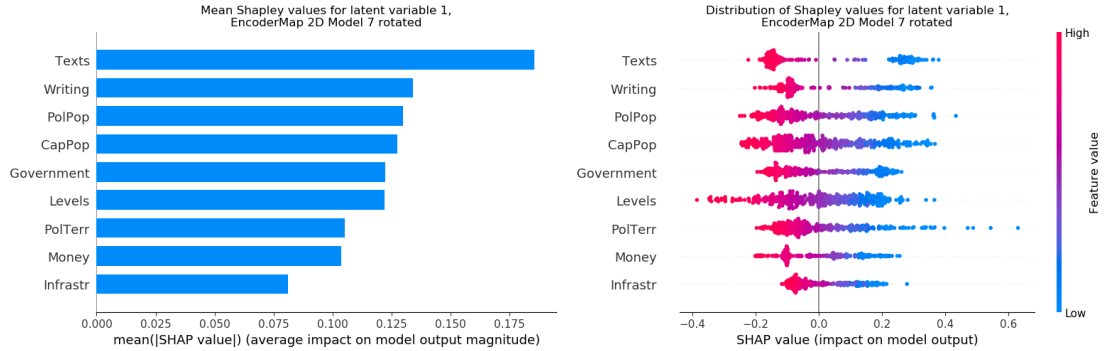


Figure 64: Mean of Shapley values over the whole dataset for the first latent variable in EncoderMap 2D latent space of model 7 (left) and distribution of Shapley values for each complexity characteristic (right).

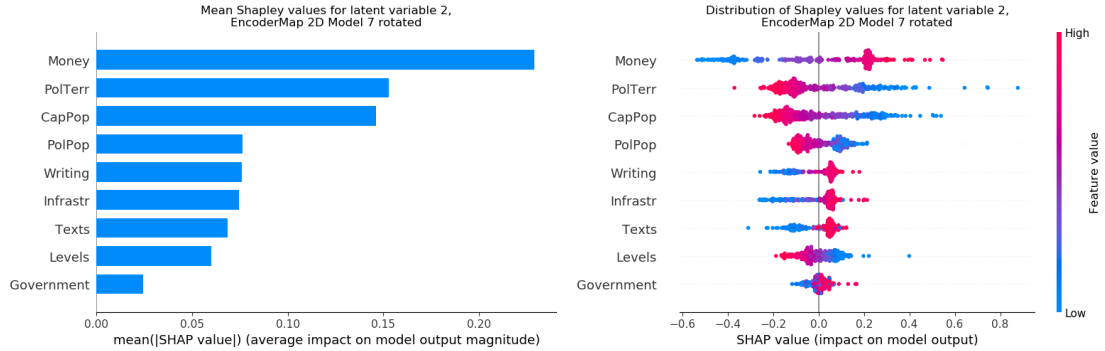


Figure 65: Mean of Shapley values over the whole dataset for the second latent variable in EncoderMap 2D latent space of model 7 (left) and distribution of Shapley values for each complexity characteristic (right).

one particular model, while the loadings are averaged over all of the models. These findings validate that the interpretation with Shapley values is consistent with other interpretation methods like loadings, which are available for PCA.

Although these particular interpretation methods are not available for other algorithms, Shapley values are model-agnostic and can also be used for neural networks like EncoderMap. The respective plots are shown in Figure 64 for the first latent variable and in Figure 65 for the second of the EncoderMap 2D latent space. Comparing the plots for the first principal component of PCA (Figure 62) with the plots for the first latent variable in EncoderMap 2D latent space (Figure 64), the plots draw qualitatively the same picture, with a few exceptions. In both latent spaces higher feature values drive the prediction to the left in all complexity characteristics. Furthermore, the text feature appears to have the biggest effect on the prediction, while polity territory (PolTerr) and money are less important. The biggest difference along these axes is the infrastructure feature, which scores last in importance for the first latent variable in EncoderMap latent space, but is in the middle region for PCA latent space.

As before with the correlations in subsection IV.4.3.3, the latent spaces differ more greatly along the second axis between PCA in Figure 63 and EncoderMap 2D in Figure 65. While the polity territory (PolTerr) scores highest in feature importance for the second principal component of PCA and money comes in second, the situation is reversed for the second latent variable in EncoderMap 2D latent space: The money feature appears to be the most important by far, with polity territory being second. Although both algorithms identify the levels and government feature as being least important for the second axis of both latent spaces, the overall importance structure for both algorithms is quite different. This is expected, since in contrast to PCA, EncoderMap is a non-linear method with no guarantee for independent axes.
Differences can also be found in the shape of the distributions (compare Figure 63 and Figure 65 respectively), with the distributions of Shapley values for the second latent variable of EncoderMap 2D appearing to be more evenly spread out with one or two places of higher density. When looking at where the high (red) and low (blue) feature values are situated in the distributions in Figure 65, we can infer the influence of different complexity characteristics on the position on the second axis in EncoderMap 2D latent space. Thus higher values in the complexity charactersitics of money, writing, infrastructure, texts and government drive the position up along the second axis in latent space, while high values in the other complexity characteristics drive the prediction down.
When comparing the directions, in which the different complexity characteristics drive the position in the latent spaces, we find that these directions are opposite to each other along the second axis in the latent spaces of PCA and EncoderMap for model number 7. This information can be read off by comparing the direction of the color gradient in Figure 63 (PCA) for each complexity characteristic with its counterpart in Figure 65 (EncoderMap). The time series in subsection IV.4.2.3 reflect this insight, as the time series mirror the directions along the second axis between the two methods. While for PCA the directions between models only differ by a global minus, for EncoderMap the sign structure can vary independently for each complexity characteristic. This leads to the structure of different latent spaces not being comparable between models, as was already implied in Figure 59. These directions can easily be read of from the global distributions of Shapley values and thus give a hint as to where each input feature drives the prediction in latent space. Finally, each of the distributions in the Shapley analyses show a well defined behaviour of a continuous transition from blue to red values or vice versa. This is expected as a prerequisite for a sensible latent space, where each change in the input variable has a smooth effect in the latent space.

IV.5 SUMMARY

In the present chapter, we have systematically compared the deep learning architecture EncoderMap against the established linear method of PCA in their capacities to find expressive low-dimensional representations of human history data describing "Social Complexity". We find that EncoderMap outperforms PCA in all dimensionality reduction tasks. More specifically, we find that EncoderMap is able to explain 80.9 ± 0.6 percent of the variance in one dimension compared to 77.2 ± 0.4 percent for PCA and 89.0 ± 1.3 percent (EncoderMap) vs. 83.2 ± 0.3 (PCA) in two dimensions.
As the improvement in performance with non-linear methods over linear methods is often paid for in interpretability, we looked at methods to interpret both of the latent spaces found by the respective methods. Here, we uncover that EncoderMap picks up on the same main relationships in the data as PCA, namely the two groups of complexity characteristics of scale and non-scale, driving the placements in the latent spaces. We think that this coherency between linear models and their non-linear advancements is a crucial baseline to verify for rooting out non-sensical models that can potentially occur in deep learning due to the high flexibility of the models. We showed that using a non-linear method (EncoderMap) leads to a measurably better latent space, while loosing some of the interpretability of the linear method (PCA). Therefore, we advocate for a symbiosis of linear and non-linear methods: Linear methods will stay relevant to identify and interpret the big driving factors in the data, while non-linear methods allow to incorporate non-linear effects and therefore improve the overall performance of the methods, as demonstrated in this chapter.
A big obstacle in this case study is the lack of sufficient data. Naturally, historical data is very hard to acquire, as it takes a lot of historians, archaeologists and research to even produce single data points that furthermore have large error-bars attached. Because of that, the data is coarse and we can only conclude rather simple insights. There appear to be only two big 'metastable basins' in the data, a basin of low complexity and a basin of high complexity with rare transitions between. There are too few transitions and states to reasonably apply Markov state models for further analysis. The most common transition is the transition from a state of low complexity to one of high complexity, a rather obvious observation, when looking back at history. The most profound insight was already stated in the original paper [163], namely that the aspects of human social organization appear to co-evolve in similar ways, but there are still a lot of interesting open questions left:

- Recent data:
  Recent data including the industrial revolution and globalization is not present in the data. As globalization erodes the notion of a locally isolated polity it is not clear this data can even be looked at in this framework. Nonetheless, it would be interesting to include this information as it probably would establish a new third cluster of even higher complexity.

- Amount of latent dimensions:
  The majority of the variation in the data is unambiguously captured in one dimension, independent of the dimensionality reduction method applied. When adding additional latent dimensions, however, EncoderMap and PCA differ in their statements (compare Figure 36). While the fraction of variance explained gradually increases with additional dimensions for PCA, EncoderMap paints a different picture. For EncoderMap, most fraction of variance explained is gained until a latent space with three dimensions, while the rate of improvement drops after that. An interesting question to investigate therefore would be whether a latent space of dimension three is optimal and which relationships in the data are caught in these dimensions.

- Non-linear effects: More generally, it would be interesting to conduct further research into identifying the non-linear effects, which are captured by EncoderMap and that improve the performance of the representation over that of PCA.

V

PERSPECTIVES

In the present thesis we showed that deep learning, in the form of an autoencoder architecture with a distant metric (EncoderMap), is able to construct descriptive low-dimensional spaces in an unsupervised manner that are fit for modeling. Our two case studies show that EncoderMap is a promising tool for dimensionality reduction in two vastly different circumstances and is potentially applicable in even more contexts.
Our main result implies that our approach enables the modeling of molecules in-silico in at least the same or even fewer dimensions than established methods, while only using structural information. This result adds another modeling tool to the already vast arsenal that is potentially able to improve the modeling process in various scenarios. Naturally, as it is important to look at things from different angles, there does not exist a one-size-fits-all modeling tool. It is important to use different modeling tools depending on the task at hand and aim to retain as much of the interesting information as possible throughout the modeling process, which usually is mediated through the methods applied.
We believe that our approach has the capability to be applied to scenarios, where no time series are available and only unordered structural data exists. This data is often cheaper to generate (with e.g. Monte Carlo Simulations, which can be sped up by trial moves), while time series need more computationally expensive molecular dynamics simulations. It is important to emphasize that ordered data, like time series, inherently contains more information than unordered data, as the order can always be discarded and therefore ordered data can always be turned into unordered data. Because of that, it is possible to apply algorithms that work on unordered data to ordered data, but not the other way round. The drawback is that the information inherent in the ordering is lost that way and not made use of. Both kinds of algorithms, for unordered and for ordered data, will therefore stay relevant and should be applied depending on the task.
This thesis successfully lays the groundwork for further research into the applicability of EncoderMap-type deep learning methods by showing feasibility on two particular

(model) problems. As feasibility does not mean practicability nor generalizability, this work is not exhaustive and further research is needed. For the modeling of molecules on the computer, potentially interesting directions for further research include:

- Larger systems:
  As we only did a feasibility study on a small system, the next step would be to scale the approach up to a larger system and investigate the efficacy there. It was already shown in [104] that the approach works for larger systems with an autoencoder architecture without distance metric. This suggests that our slightly different approach should also work for larger systems.

- Impact of the additional distance metric:
  As mentioned in the previous point, the approach was already found to be working without the additional distance metric that EncoderMap employs [104]. It would therefore be interesting to investigate the effect the additional distance metric in our approach, using EncoderMap, has on the quality of the latent space compared to using an autoencoder alone.

- Encoding of dynamical information:
  Our work using EncoderMap [63] and the previously mentioned work using autoencoders alone [104] strongly suggest that modeling is possible with structural data alone. This poses the question, where the dynamic information is encoded that allows these methods to find good collective variables. A hint might give diffusion maps [100–102], which also use structural data alone to extract dynamical quantities. Following the theory of diffusion maps, the dynamics is encoded in the connectivity between data points, which ultimately is encoded (locally) in the distances between (neighbouring) points. If this applies as well for our approach, several conclusions worth investigating come to mind: First of all, it would hint in favor of using EncoderMap, which also reproduces distances, over using an autoencoder alone as in [104], an answer to the previous point made in this list. Secondly, it would mean that the Johnson–Lindenstrauss lemma applies, which gives an upper bound for the amount of dimensions needed to embed data points, while retaining distances between them. It would be interesting to investigate with different systems, whether this could be a good heuristic for the amount of dimensions needed to construct a good latent space for modeling. Thirdly, it would be interesting to investigate, whether and in which manner distances between points potentially need to be rescaled to preserve correct dynamics when combining different biased simulations.

BIBLIOGRAPHY

[1] D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, 2023.

[2] K. Binder and D.W. Heermann. *Monte Carlo Simulation in Statistical Physics: An Introduction*. Graduate Texts in Physics. Springer Berlin Heidelberg, 2010.

[3] M D Haw. *Colloidal Suspensions, Brownian Motion, Molecular Reality: A Short History*. In: *J. Phys.: Condens. Matter* 14.33 (Aug. 2002), pp. 7769–7779.

[4] Stephen G. Brush. *A History of Random Processes*. In: *Archive for History of Exact Sciences* 5.1 (Jan. 1968), pp. 1–36.

[5] A. Einstein. *Über Die von Der Molekularkinetischen Theorie Der Wärme Geforderte Bewegung von in Ruhenden Flüssigkeiten Suspendierten Teilchen*. In: *Annalen der Physik* 322.8 (1905), pp. 549–560.

[6] M. von Smoluchowski. *Zur Kinetischen Theorie Der Brownschen Molekularbewegung Und Der Suspensionen*. In: *Annalen der Physik* 326.14 (1906), pp. 756–780.

[7] Don S. Lemons and Anthony Gythiel. *Paul Langevin's 1908 Paper "on the Theory of Brownian Motion" ["Sur La Théorie Du Mouvement Brownien," C. R. Acad. Sci. (Paris) 146, 530–533 (1908)]*. In: *American Journal of Physics* 65.11 (Nov. 1997), pp. 1079–1081.

[8] Jean Perrin. *Mouvement Brownien et Réalité Moléculaire*. In: (1909).

[9] Nicholas Metropolis, Arianna W. Rosenbluth, et al. *Equation of State Calculations by Fast Computing Machines*. In: *The Journal of Chemical Physics* 21.6 (June 1953), pp. 1087–1092.

[10] B. J. Alder and T. E. Wainwright. *Phase Transition for a Hard Sphere System*. In: *The Journal of Chemical Physics* 27.5 (Nov. 1957), pp. 1208–1209.

[11] Ulf Hashagen. *The Computation of Nature, or: Does the Computer Drive Science and Technology?* In: *The Nature of Computation. Logic, Algorithms, Applications*. Ed. by Paola Bonizzoni, Vasco Brattka, and Benedikt Löwe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 263–270.

[12] Eric Winsberg. *Science in the Age of Computer Simulation*. University of Chicago Press, 2019.

[13] Scott A. Hollingsworth and Ron O. Dror. *Molecular Dynamics Simulation for All*. In: *Neuron* 99.6 (Sept. 2018), pp. 1129–1143.

[14] G. Ciccotti, C. Dellago, et al. *Molecular Simulations: Past, Present, and Future (a Topical Issue in EPJB)*. In: *The European Physical Journal B* 95.1 (Jan. 2022), p. 3.

[15] Qiang Shao and Weiliang Zhu. *How Well Can Implicit Solvent Simulations Explore Folding Pathways? A Quantitative Analysis of α-Helix Bundle Proteins*. In: *J. Chem. Theory Comput.* 13.12 (Dec. 2017), pp. 6177–6190.

[16] Anne Gershenson, Shachi Gosavi, Pietro Faccioli, and Patrick L. Wintrode. *Successes and Challenges in Simulating the Folding of Large Proteins*. In: *J. Biol. Chem.* 295.1 (Jan. 2020), pp. 15–33.

[17] Hao Geng, Fangfang Chen, Jing Ye, and Fan Jiang. *Applications of Molecular Dynamics Simulation in Structure Prediction of Peptides and Proteins*. In: *Comput. Struct. Biotechnol. J.* 17 (2019), pp. 1162–1170.

[18] Lili Duan, Xiaona Guo, et al. *Accelerated Molecular Dynamics Simulation for Helical Proteins Folding in Explicit Water*. In: *Front. Chem.* 7 (Aug. 2019), p. 540.

[19] Jianlin Chen, Xiaorong Liu, and Jianhan Chen. *Atomistic Peptide Folding Simulations Reveal Interplay of Entropy and Long-Range Interactions in Folding Cooperativity*. In: *Sci Rep* 8.1 (Sept. 2018), p. 13668.

[20] James C. Phillips, David J. Hardy, et al. *Scalable Molecular Dynamics on CPU and GPU Architectures with NAMD*. In: *J. Chem. Phys.* 153.4 (July 2020), p. 044130.

[21] Michael Gecht, Marc Siggel, et al. *MDBenchmark: A Toolkit to Optimize the Performance of Molecular Dynamics Simulations*. In: *The Journal of Chemical Physics* 153.14 (Oct. 2020), p. 144105.

[22] D C Rapaport. *GPU Molecular Dynamics: Algorithms and Performance*. In: *J. Phys.: Conf. Ser.* 2241.1 (Mar. 2022), p. 012007.

[23] David E. Shaw, Peter J. Adams, et al. *Anton 3: Twenty Microseconds of Molecular Dynamics Simulation before Lunch*. In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. St. Louis Missouri: ACM, Nov. 2021, pp. 1–11.

[24] Yi Isaac Yang, Qiang Shao, et al. *Enhanced Sampling in Molecular Dynamics*. In: *J. Chem. Phys.* 151.7 (Aug. 2019), p. 070902.

[25] Jérôme Hénin, Tony Lelièvre, et al. *Enhanced Sampling Methods for Molecular Dynamics Simulations [Article v1.0]*. In: *LiveCoMS* 4.1 (2022).

[26] Vijay S. Pande, Kyle Beauchamp, and Gregory R. Bowman. *Everything You Wanted to Know about Markov State Models but Were Afraid to Ask*. In: *Methods* 52.1 (Sept. 2010), pp. 99–105.

[27] Gregory R. Bowman, Vijay S. Pande, and Frank Noé, eds. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*. Vol. 797. Advances in Experimental Medicine and Biology. Dordrecht: Springer Netherlands, 2014.

[28] Brooke E. Husic and Vijay S. Pande. *Markov State Models: From an Art to a Science*. In: *J. Am. Chem. Soc.* 140.7 (Feb. 2018), pp. 2386–2396.

[29] Kirill A. Konovalov, Ilona Christy Unarta, et al. *Markov State Models to Study the Functional Dynamics of Proteins in the Wake of Machine Learning*. In: *JACS Au* 1.9 (Sept. 2021), pp. 1330–1341.

[30] Mario Krenn, Robert Pollice, et al. *On Scientific Understanding with Artificial Intelligence*. In: *Nat Rev Phys* 4.12 (Oct. 2022), pp. 761–769.

[31] Partha Pratim Ray. *ChatGPT: A Comprehensive Review on Background, Applications, Key Challenges, Bias, Ethics, Limitations and Future Scope*. In: *Internet of Things and Cyber-Physical Systems* 3 (2023), pp. 121–154.

[32] Kamal Choudhary, Brian DeCost, et al. *Recent Advances and Applications of Deep Learning Methods in Materials Science*. In: *npj Computational Materials* 8.1 (Apr. 2022), p. 59.

[33] Peter Sadowski and Pierre Baldi. *Deep Learning in the Natural Sciences: Applications to Physics*. In: *Braverman Readings in Machine Learning. Key Ideas from Inception to Current State: International Conference Commemorating the 40th Anniversary of Emmanuil Braverman's Decease, Boston, MA, USA, April 28-30, 2017, Invited Talks*. Ed. by Lev Rozonoer, Boris Mirkin, and Ilya Muchnik. Cham: Springer International Publishing, 2018, pp. 269–297.

[34] John Jumper, Richard Evans, et al. *Highly Accurate Protein Structure Prediction with AlphaFold*. In: *Nature* 596.7873 (Aug. 2021), pp. 583–589.

[35] F. Reif. *Fundamentals of Statistical and Thermal Physics*. Fundamentals of Physics Series. McGraw-Hill, 1965.

[36] Matthias Bartelmann, Björn Feuerbacher, et al. *Theoretische Physik 4 | Thermodynamik Und Statistische Physik*. Jan. 2018.

[37] W. K. Hastings. *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*. In: *Biometrika* 57.1 (Apr. 1970), pp. 97–109.

[38] C.W. Gardiner. *Handbook of Stochastic Methods for Physics, Chemistry, and the Natural Sciences*. Proceedings in Life Sciences. Springer-Verlag, 1985.

[39] N.G. Van Kampen. *Stochastic Processes in Physics and Chemistry*. North-Holland Personal Library. North Holland, 2011.

[40] H. Risken and T. Frank. *The Fokker-Planck Equation: Methods of Solution and Applications*. Springer Series in Synergetics. Springer Berlin Heidelberg, 1996.

[41] John D Chodera and Frank Noé. *Markov State Models of Biomolecular Conformational Dynamics*. In: *Current Opinion in Structural Biology* 25 (Apr. 2014), pp. 135–144.

[42] Jan-Hendrik Prinz, Hao Wu, et al. *Markov Models of Molecular Kinetics: Generation and Validation*. In: *J. Chem. Phys.* 134.17 (May 2011), p. 174105.

[43] Ch Schütte, A Fischer, W Huisinga, and P Deuflhard. *A Direct Approach to Conformational Dynamics Based on Hybrid Monte Carlo*. In: *Journal of Computational Physics* 151.1 (May 1999), pp. 146–168.

[44] Jan-Hendrik Prinz, Bettina Keller, and Frank Noé. *Probing Molecular Kinetics with Markov Models: Metastable States, Transition Pathways and Spectroscopic Observables*. In: *Phys. Chem. Chem. Phys.* 13.38 (2011), p. 16912.

[45] S. Lloyd. *Least Squares Quantization in PCM*. In: *IEEE Trans. Inform. Theory* 28.2 (Mar. 1982), pp. 129–137.

[46] David Arthur and Sergei Vassilvitskii. *K-Means++: The Advantages of Careful Seeding*. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[47] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Boston, MA: Springer US, 1981.

[48] J. C. Dunn. *A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters*. In: *Journal of Cybernetics* 3.3 (Jan. 1973), pp. 32–57.

[49] Susanna Röblitz and Marcus Weber. *Fuzzy Spectral Clustering by PCCA+: Application to Markov State Models and Data Classification*. In: *Adv Data Anal Classif* 7.2 (June 2013), pp. 147–179.

[50] Peter Deuflhard and Marcus Weber. *Robust Perron Cluster Analysis in Conformation Dynamics*. In: *Linear Algebra and its Applications* 398 (Mar. 2005), pp. 161–184.

[51] Anna-Simone Frank, Alexander Sikorski, and Susanna Röblitz. *Spectral Clustering of Markov Chain Transition Matrices with Complex Eigenvalues*. Aug. 2022. arXiv: 2206.14537 [stat].

[52] Bernhard Reuter, Konstantin Fackeldey, and Marcus Weber. *Generalized Markov Modeling of Nonreversible Molecular Kinetics*. In: *The Journal of Chemical Physics* 150.17 (May 2019), p. 174103.

[53] Bernhard Reuter, Marcus Weber, et al. *Generalized Markov State Modeling Method for Nonequilibrium Biomolecular Dynamics: Exemplified on Amyloid β Conformational Dynamics Driven by an Oscillating Electric Field*. In: *J. Chem. Theory Comput.* 14.7 (July 2018), pp. 3579–3594.

[54] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[55] L. Molgedey and H. G. Schuster. *Separation of a Mixture of Independent Signals Using Time Delayed Correlations*. In: *Phys. Rev. Lett.* 72.23 (June 1994), pp. 3634–3637.

[56] Christian R. Schwantes and Vijay S. Pande. *Improvements in Markov State Model Construction Reveal Many Non-Native Interactions in the Folding of NTL9*. In: *J. Chem. Theory Comput.* 9.4 (Apr. 2013), pp. 2000–2009.

[57] Guillermo Pérez-Hernández, Fabian Paul, et al. *Identification of Slow Molecular Order Parameters for Markov Model Construction*. In: *J. Chem. Phys.* 139.1 (July 2013), p. 015102.

[58] Frank Noé and Cecilia Clementi. *Kinetic Distance and Kinetic Maps from Molecular Dynamics Simulation*. In: *J. Chem. Theory Comput.* 11.10 (Oct. 2015), pp. 5002–5011.

[59] Ulrike von Luxburg and Bernhard Schölkopf. *Statistical Learning Theory: Models, Concepts, and Results*. In: *Handbook of the History of Logic*. Ed. by Dov M. Gabbay, Stephan Hartmann, and John Woods. Vol. 10. North-Holland, Jan. 2011, pp. 651–706.

[60] Tobias Lemke and Christine Peter. *EncoderMap: Dimensionality Reduction and Generation of Molecule Conformations*. In: *J. Chem. Theory Comput.* 15.2 (Feb. 2019), pp. 1209–1215.

[61] Michele Ceriotti, Gareth A. Tribello, and Michele Parrinello. *Simplifying the Representation of Complex Free-Energy Landscapes Using Sketch-Map*. In: *Proc. Natl. Acad. Sci. U.S.A.* 108.32 (Aug. 2011), pp. 13023–13028.

[62] J. B. Kruskal. *Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis*. In: *Psychometrika* 29.1 (Mar. 1964), pp. 1–27.

[63] Simon Lemcke, Jörn H. Appeldorn, Michael Wand, and Thomas Speck. *Toward a Structural Identification of Metastable Molecular Conformations*. In: *The Journal of Chemical Physics* 159.11 (Sept. 2023), p. 114105.

[64] Benjamin Trendelkamp-Schroer, Hao Wu, Fabian Paul, and Frank Noé. *Estimation and Uncertainty of Reversible Markov Models*. In: *J. Chem. Phys* 143.17 (Nov. 2015), p. 174101.

[65] Philipp Metzner, Frank Noé, and Christof Schütte. *Estimating the Sampling Error: Distribution of Transition Matrices and Functions of Transition Matrices for given Trajectory Data*. In: *Phys. Rev. E* 80.2 (Aug. 2009), p. 021106.

[66] Frank Noé, Christof Schütte, et al. *Constructing the Equilibrium Ensemble of Folding Pathways from Short Off-Equilibrium Simulations*. In: *Proc. Natl. Acad. Sci. U.S.A.* 106.45 (Nov. 2009), pp. 19011–19016.

[67] Rafael C. Bernardi, Marcelo C.R. Melo, and Klaus Schulten. *Enhanced Sampling Techniques in Molecular Dynamics Simulations of Biological Systems*. In: *Biochimica et Biophysica Acta (BBA) - General Subjects* 1850.5 (May 2015), pp. 872–877.

[68] Vojtech Spiwok, Zoran Sucur, and Petr Hosek. *Enhanced Sampling Techniques in Biomolecular Simulations*. In: *Biotechnology Advances* 33.6 (Nov. 2015), pp. 1130–1140.

[69] G.M. Torrie and J.P. Valleau. *Nonphysical Sampling Distributions in Monte Carlo Free-Energy Estimation: Umbrella Sampling*. In: *Journal of Computational Physics* 23.2 (Feb. 1977), pp. 187–199.

[70] Johannes Kästner. *Umbrella Sampling: Umbrella Sampling*. In: *WIREs Comput Mol Sci* 1.6 (Nov. 2011), pp. 932–942.

[71] Wanli You, Zhiye Tang, and Chia-en A. Chang. *Potential Mean Force from Umbrella Sampling Simulations: What Can We Learn and What Is Missed?* In: *J. Chem. Theory Comput.* 15.4 (Apr. 2019), pp. 2433–2443.

[72] Vivek Govind Kumar, Adithya Polasa, et al. *Binding Affinity Estimation from Restrained Umbrella Sampling Simulations*. In: *Nat Comput Sci* 3.1 (Dec. 2022), pp. 59–70.

[73] Alessandro Laio and Michele Parrinello. *Escaping Free-Energy Minima*. In: *Proc. Natl. Acad. Sci. U.S.A.* 99.20 (Oct. 2002), pp. 12562–12566.

[74] Omar Valsson, Pratyush Tiwary, and Michele Parrinello. *Enhancing Important Fluctuations: Rare Events and Metadynamics from a Conceptual Viewpoint*. In: *Annu. Rev. Phys. Chem.* 67.1 (May 2016), pp. 159–184.

[75] Yuji Sugita and Yuko Okamoto. *Replica-Exchange Molecular Dynamics Method for Protein Folding*. In: *Chem. Phys. Lett.* 314.1-2 (Nov. 1999), pp. 141–151.

[76] Lukas S. Stelzl and Gerhard Hummer. *Kinetics from Replica Exchange Molecular Dynamics Simulations*. In: *J. Chem. Theory Comput.* 13.8 (Aug. 2017), pp. 3927–3935.

[77] Michael R. Shirts and John D. Chodera. *Statistically Optimal Analysis of Samples from Multiple Equilibrium States*. In: *J. Chem. Phys.* 129.12 (Sept. 2008), p. 124105.

[78] Ahmet Yildirim, Tsjerk A. Wassenaar, and David van der Spoel. *Statistical Efficiency of Methods for Computing Free Energy of Hydration*. In: *J. Chem. Phys.* 149.14 (Oct. 2018), p. 144111.

[79] Timothy J. Giese, Şölen Ekesan, and Darrin M. York. *Extension of the Variational Free Energy Profile and Multistate Bennett Acceptance Ratio Methods for High-Dimensional Potential of Mean Force Profile Analysis*. In: *J. Phys. Chem. A* 125.19 (May 2021), pp. 4216–4232.

[80] Yasuhiro Matsunaga, Motoshi Kamiya, et al. *Use of Multistate Bennett Acceptance Ratio Method for Free-Energy Calculations from Enhanced Sampling and Free-Energy Perturbation*. In: *Biophys Rev* 14.6 (Dec. 2022), pp. 1503–1512.

[81] Zhiqiang Tan. *On a Likelihood Approach for Monte Carlo Integration*. In: *J. Am. Stat. Assoc.* 99.468 (Dec. 2004), pp. 1027–1036.

[82] A. Kong, P. McCullagh, et al. *A Theory of Statistical Models for Monte Carlo Integration:* Statistical Models for Monte Carlo Integration. In: *J. R. Stat. Soc., B: Stat. Methodol.* 65.3 (Aug. 2003), pp. 585–604.

[83] Hao Wu, Fabian Paul, Christoph Wehmeyer, and Frank Noé. *Multiensemble Markov Models of Molecular Thermodynamics and Kinetics*. In: *Proc. Natl. Acad. Sci. U.S.A.* 113.23 (June 2016).

[84] Ziwei He, Fabian Paul, and Benoît Roux. *A Critical Perspective on Markov State Model Treatments of Protein–Protein Association Using Coarse-Grained Simulations*. In: *J. Chem. Phys.* 154.8 (Feb. 2021), p. 084101.

[85] Jaewoon Jung, Chigusa Kobayashi, et al. *New Parallel Computing Algorithm of Molecular Dynamics for Extremely Huge Scale Biological Systems*. In: *J Comput Chem* 42.4 (Feb. 2021), pp. 231–241.

[86] Juan R Perilla, Boon Chong Goh, et al. *Molecular Dynamics Simulations of Large Macromolecular Complexes*. In: *Curr Opin Struct Biol* 31 (Apr. 2015), pp. 64–74.

[87] Thomas J Lane, Diwakar Shukla, Kyle A Beauchamp, and Vijay S Pande. *To Milliseconds and beyond: Challenges in the Simulation of Protein Folding*. In: *Curr Opin Struct Biol* 23.1 (Feb. 2013), pp. 58–65.

[88] Adam L. Beberg, Daniel L. Ensign, et al. *Folding@home: Lessons from Eight Years of Volunteer Distributed Computing*. In: *2009 IEEE International Symposium on Parallel & Distributed Processing*. 2009, pp. 1–8.

[89] Alexander Rives, Joshua Meier, et al. *Biological Structure and Function Emerge from Scaling Unsupervised Learning to 250 Million Protein Sequences*. In: *Proc. Natl. Acad. Sci. U.S.A.* 118.15 (Apr. 2021), e2016239118.

[90] Zeming Lin, Halil Akin, et al. *Evolutionary-Scale Prediction of Atomic-Level Protein Structure with a Language Model*. In: *Science* 379.6637 (Mar. 2023), pp. 1123–1130.

[91] Thorben Fröhlking, Mattia Bernetti, Nicola Calonaci, and Giovanni Bussi. *Toward Empirical Force Fields That Match Experimental Observables*. In: *J. Chem. Phys.* 152.23 (June 2020), p. 230902.

[92] Wendy D. Cornell, Piotr Cieplak, et al. *A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules*. In: *J. Am. Chem. Soc.* 117.19 (May 1995), pp. 5179–5197.

[93] Alexander D. MacKerell, Joanna Wiorkiewicz-Kuczera, and Martin Karplus. *An All-Atom Empirical Energy Function for the Simulation of Nucleic Acids*. In: *J. Am. Chem. Soc.* 117.48 (Dec. 1995), pp. 11946–11975.

[94] William L. Jorgensen and Julian Tirado-Rives. *The OPLS [Optimized Potentials for Liquid Simulations] Potential Functions for Proteins, Energy Minimizations for Crystals of Cyclic Peptides and Crambin*. In: *J. Am. Chem. Soc.* 110.6 (Mar. 1988), pp. 1657–1666.

[95] Chris Oostenbrink, Alessandra Villa, Alan E. Mark, and Wilfred F. Van Gunsteren. *A Biomolecular Force Field Based on the Free Enthalpy of Hydration and Solvation: The GROMOS Force-Field Parameter Sets 53A5 and 53A6*. In: *J. Comput. Chem.* 25.13 (Oct. 2004), pp. 1656–1676.

[96] Christoph Wehmeyer and Frank Noé. *Time-Lagged Autoencoders: Deep Learning of Slow Collective Variables for Molecular Kinetics*. In: *J. Chem. Phys.* 148.24 (June 2018), p. 241703.

[97] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. *VAMPnets for Deep Learning of Molecular Kinetics*. In: *Nat Commun* 9.1 (Jan. 2018), p. 5.

[98] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. *A Global Geometric Framework for Nonlinear Dimensionality Reduction*. In: *Science* 290.5500 (Dec. 2000), pp. 2319–2323.

[99] Payel Das, Mark Moll, et al. *Low-Dimensional, Free-Energy Landscapes of Protein-Folding Reactions by Nonlinear Dimensionality Reduction*. In: *Proc. Natl. Acad. Sci. U.S.A.* 103.26 (June 2006), pp. 9885–9890.

[100] R. R. Coifman, S. Lafon, et al. *Geometric Diffusions as a Tool for Harmonic Analysis and Structure Definition of Data: Diffusion Maps*. In: *Proc. Natl. Acad. Sci. U.S.A.* 102.21 (May 2005), pp. 7426–7431.

[101] Andrew L. Ferguson, Athanassios Z. Panagiotopoulos, Ioannis G. Kevrekidis, and Pablo G. Debenedetti. *Nonlinear Dimensionality Reduction in Molecular Simulation: The Diffusion Map Approach*. In: *Chem. Phys. Lett.* 509.1-3 (June 2011), pp. 1–11.

[102] Z. Trstanova, B. Leimkuhler, and T. Lelièvre. *Local and Global Perspectives on Diffusion Maps in the Analysis of Molecular Systems*. In: *Proc. R. Soc. A.* 476.2233 (Jan. 2020), p. 20190036.

[103] Jörn H. Appeldorn, Simon Lemcke, Thomas Speck, and Arash Nikoubashman. *Employing Artificial Neural Networks to Identify Reaction Coordinates and Pathways for Self-Assembly*. In: *J. Phys. Chem. B* 126.27 (July 2022), pp. 5007–5016.

[104] Satyabrata Bandyopadhyay and Jagannath Mondal. *A Deep Autoencoder Framework for Discovery of Metastable Ensembles in Biomacromolecules*. In: *J. Chem. Phys.* 155.11 (Sept. 2021), p. 114106.

[105] Wei Chen and Andrew L. Ferguson. *Molecular Enhanced Sampling with Autoencoders: On-the-fly Collective Variable Discovery and Accelerated Free Energy Landscape Exploration*. In: *J. Comput. Chem.* 39.25 (Sept. 2018), pp. 2079–2102.

[106] Nawavi Naleem, Charlles R. A. Abreu, et al. *An Exploration of Machine Learning Models for the Determination of Reaction Coordinates Associated with Conformational Transitions*. In: *J. Chem. Phys.* 159.3 (July 2023), p. 034102.

[107] Hythem Sidky, Wei Chen, and Andrew L. Ferguson. *Molecular Latent Space Simulators*. In: *Chem. Sci.* 11.35 (2020), pp. 9459–9467.

[108] Gungor Ozer, Stephen Quirk, and Rigoberto Hernandez. *Thermodynamics of Decaalanine Stretching in Water Obtained by Adaptive Steered Molecular Dynamics Simulations*. In: *J. Chem. Theory Comput.* 8.11 (Nov. 2012), pp. 4837–4844.

[109] Anthony Hazel, Christophe Chipot, and James C. Gumbart. *Thermodynamics of Deca-alanine Folding in Water*. In: *J. Chem. Theory Comput.* 10.7 (July 2014), pp. 2836–2844.

[110] Feliks Nüske, Bettina G. Keller, et al. *Variational Approach to Molecular Kinetics*. In: *J. Chem. Theory Comput.* 10.4 (Apr. 2014), pp. 1739–1752.

[111] F. Vitalini, A. S. J. S. Mey, F. Noé, and B. G. Keller. *Dynamic Properties of Force Fields*. In: *J. Chem. Phys.* 142.8 (Feb. 2015), p. 084101.

[112] Hailey R. Bureau, Dale R. Merz, et al. *Constrained Unfolding of a Helical Peptide: Implicit versus Explicit Solvents*. In: *PLoS ONE* 10.5 (May 2015). Ed. by Claudio M Soares, e0127034.

[113] Dheeraj S. Tomar, Valéry Weber, B. Montgomery Pettitt, and D. Asthagiri. *Importance of Hydrophilic Hydration and Intramolecular Interactions in the Thermodynamics of Helix–Coil Transition and Helix–Helix Assembly in a Deca-Alanine Peptide*. In: *J. Phys. Chem. B* 120.1 (Jan. 2016), pp. 69–76.

[114] Feliks Nüske, Reinhold Schneider, Francesca Vitalini, and Frank Noé. *Variational Tensor Approach for Approximating the Rare-Event Kinetics of Macromolecular Systems*. In: *J. Chem. Phys.* 144.5 (Feb. 2016), p. 054105.

[115] Stefan Klus, Feliks Nüske, et al. *Data-Driven Model Reduction and Transfer Operator Approximation*. In: *J Nonlinear Sci* 28.3 (June 2018), pp. 985–1010.

[116] Fabian Knoch and Thomas Speck. *Unfolding Dynamics of Small Peptides Biased by Constant Mechanical Forces*. In: *Mol. Syst. Des. Eng.* 3.1 (2018), pp. 204–213.

[117] Mark James Abraham, Teemu Murtola, et al. *GROMACS: High Performance Molecular Simulations through Multi-Level Parallelism from Laptops to Supercomputers*. In: *SoftwareX* 1–2 (Sept. 2015), pp. 19–25.

[118] Aidan P. Thompson, H. Metin Aktulga, et al. *LAMMPS - a Flexible Simulation Tool for Particle-Based Materials Modeling at the Atomic, Meso, and Continuum Scales*. In: *Computer Physics Communications* 271 (Feb. 2022), p. 108171.

[119] Hao Wu and Frank Noé. *Variational Approach for Learning Markov Processes from Time Series Data*. In: *J Nonlinear Sci* 30.1 (Feb. 2020), pp. 23–66.

[120] Andreas Mardt, Luca Pasquali, Frank Noé, and Hao Wu. *Deep Learning Markov and Koopman Models with Physical Constraints*. In: *Proceedings of the First Mathematical and Scientific Machine Learning Conference*. Ed. by Jianfeng Lu and Rachel Ward. Vol. 107. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 451–475.

[121] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. *Transition Path Theory for Markov Jump Processes*. In: *Multiscale Model. Simul.* 7.3 (Jan. 2009), pp. 1192–1219.

[122] Weinan E and Eric Vanden-Eijnden. *Transition-Path Theory and Path-Finding Algorithms for the Study of Rare Events*. In: *Annu. Rev. Phys. Chem.* 61.1 (Mar. 2010), pp. 391–420.

[123] S Marqusee, V H Robbins, and R L Baldwin. *Unusually Stable Helix Formation in Short Alanine-Based Peptides.* In: *Proc. Natl. Acad. Sci. U.S.A.* 86.14 (July 1989), pp. 5286–5290.

[124] Victor Muñoz and Luis Serrano. *Elucidating the Folding Problem of Helical Peptides Using Empirical Parameters. III>Temperature and pH Dependence*. In: *Journal of Molecular Biology* 245.3 (Jan. 1995), pp. 297–308.

[125] Xiongwu Wu and Shaomeng Wang. *Helix Folding of an Alanine-Based Peptide in Explicit Water*. In: *J. Phys. Chem. B* 105.11 (Mar. 2001), pp. 2227–2235.

[126] Alexander N. Morozov and Sheng Hsien Lin. *Modeling of Folding and Unfolding Mechanisms in Alanine-Based α-Helical Polypeptides*. In: *J. Phys. Chem. B* 110.41 (Oct. 2006), pp. 20555–20561.

[127] Rainer Hegger, Alexandros Altis, Phuong H. Nguyen, and Gerhard Stock. *How Complex Is the Dynamics of Peptide Folding?* In: *Phys. Rev. Lett.* 98.2 (Jan. 2007), p. 028102.

[128] I. A. Solov'yov, A. V. Yakubovich, A. V. Solov'yov, and W. Greiner. *α-Helix↔random Coil Phase Transition: Analysis of Ab Initio Theorypredictions*. In: *Eur. Phys. J. D* 46.2 (Feb. 2008), pp. 227–240.

[129] Peter Palenčár and Tomáš Bleha. *Molecular Dynamics Simulations of the Folding of Poly(Alanine) Peptides*. In: *J Mol Model* 17.9 (Sept. 2011), pp. 2367–2374.

[130] Zhaoxi Sun and Xiaohui Wang. *Thermodynamics of Helix Formation in Small Peptides of Varying Length in Vacuo, Implicit Solvent and Explicit Solvent: Comparison between AMBER Force Fields*. In: *J. Theor. Comput. Chem.* 18.03 (May 2019), p. 1950015.

[131] Krzysztof Kuczera, Robert Szoszkiewicz, Jinyan He, and Gouri S. Jas. *Length Dependent Folding Kinetics of Alanine-Based Helical Peptides from Optimal Dimensionality Reduction*. In: *Life* 11.5 (Apr. 2021), p. 385.

[132] Jonathan Hungerland, Anders Frederiksen, Luca Gerhards, and Ilia A. Solov'yov. *Studying Folding $$\leftrightarrow $$ Unfolding Dynamics of Solvated Alanine Polypeptides Using Molecular Dynamics*. In: *Eur. Phys. J. D* 76.8 (Aug. 2022), p. 154.

[133] Akio Kitao and Nobuhiro Gō. *Conformational Dynamics of Polypeptides and Proteins in the Dihedral Angle Space and in the Cartesian Coordinate Space: Normal Mode Analysis of Deca-alanine*. In: *J Comput Chem* 12.3 (Apr. 1991), pp. 359–368.

[134] Sanghyun Park, Fatemeh Khalili-Araghi, Emad Tajkhorshid, and Klaus Schulten. *Free Energy Calculation from Steered Molecular Dynamics Simulations Using Jarzynski's Equality*. In: *J. Chem. Phys.* 119.6 (Aug. 2003), pp. 3559–3566.

[135] Jérôme Hénin and Christophe Chipot. *Overcoming Free Energy Barriers Using Unconstrained Molecular Dynamics Simulations*. In: *The Journal of Chemical Physics* 121.7 (Aug. 2004), pp. 2904–2914.

[136] Christophe Chipot and Jérôme Hénin. *Exploring the Free-Energy Landscape of a Short Peptide Using an Average Force*. In: *J. Chem. Phys.* 123.24 (Dec. 2005), p. 244906.

[137] Sunhwan Jo, Donghyuk Suh, et al. *Leveraging the Information from Markov State Models To Improve the Convergence of Umbrella Sampling Simulations*. In: *J. Phys. Chem. B* 120.33 (Aug. 2016), pp. 8733–8742.

[138] Giovanni Bussi, Davide Donadio, and Michele Parrinello. *Canonical Sampling through Velocity Rescaling*. In: *J. Chem. Phys.* 126.1 (Jan. 2007), p. 014101.

[139] M. Parrinello and A. Rahman. *Polymorphic Transitions in Single Crystals: A New Molecular Dynamics Method*. In: *J. Appl. Phys.* 52.12 (Dec. 1981), pp. 7182–7190.

[140] Berk Hess, Henk Bekker, Herman J. C. Berendsen, and Johannes G. E. M. Fraaije. *LINCS: A Linear Constraint Solver for Molecular Simulations*. In: *J. Comput. Chem.* 18.12 (Sept. 1997), pp. 1463–1472.

[141] Alexander D. Mackerell, Michael Feig, and Charles L. Brooks. *Extending the Treatment of Backbone Energetics in Protein Force Fields: Limitations of Gas-Phase Quantum Mechanics in Reproducing Protein Conformational Distributions in Molecular Dynamics Simulations*. In: *J. Comput. Chem.* 25.11 (Aug. 2004), pp. 1400–1415.

[142] Cihan Ayaz, Lucas Tepper, et al. *Non-Markovian Modeling of Protein Folding*. In: *Proc. Natl. Acad. Sci. U.S.A.* 118.31 (Aug. 2021), e2023856118.

[143] Kuan Pern Tan, Khushboo Singh, Anirban Hazra, and M.S. Madhusudhan. *Peptide Bond Planarity Constrains Hydrogen Bond Geometry and Influences Secondary Structure Conformations*. In: *Curr Res Struct Biol* 3 (2021), pp. 1–8.

[144] Matthew Merski, Jakub Skrzeczkowski, Jennifer K. Roth, and Maria W. Górna. *A Geometric Definition of Short to Medium Range Hydrogen-Mediated Interactions in Proteins*. In: *Molecules* 25.22 (Nov. 2020), p. 5326.

[145] G.A. Jeffrey. *An Introduction to Hydrogen Bonding*. Topics in Physical Chemistry - Oxford University Press. Oxford University Press, 1997.

[146] Elangannan Arunan, Gautam R. Desiraju, et al. *Definition of the Hydrogen Bond (IUPAC Recommendations 2011)*. In: *Pure and Applied Chemistry* 83.8 (2011), pp. 1637–1641.

[147] E.N. Baker and R.E. Hubbard. *Hydrogen Bonding in Globular Proteins*. In: *Progress in Biophysics and Molecular Biology* 44.2 (1984), pp. 97–179.

[148] Milo M. Lin, Dmitry Shorokhov, and Ahmed H. Zewail. *Dominance of Misfolded Intermediates in the Dynamics of α-Helix Folding*. In: *Proc. Natl. Acad. Sci. U.S.A.* 111.40 (Oct. 2014), pp. 14424–14429.

[149] Ivan Y. Torshin, Irene T. Weber, and Robert W. Harrison. *Geometric Criteria of Hydrogen Bonds in Proteins and Identification of 'bifurcated' Hydrogen Bonds*. In: *Protein Engineering, Design and Selection* 15.5 (May 2002), pp. 359–363.

[150] J. E. J. Mills and P. M. Dean. *Three-Dimensional Hydrogen-Bond Geometry and Probability Information from a Crystal Survey*. In: *J Computer-Aided Mol Des* 10.6 (Dec. 1996), pp. 607–622.

[151] Gail J. Bartlett and Derek N. Woolfson. *On the Satisfaction of Backbone-carbonyl Lone Pairs of Electrons in Protein Structures*. In: *Protein Science* 25.4 (Apr. 2016), pp. 887–897.

[152] Roger A. Klein. *Modified van Der Waals Atomic Radii for Hydrogen Bonding Based on Electron Density Topology*. In: *Chemical Physics Letters* 425.1-3 (July 2006), pp. 128–133.

[153] Xiao Zhu, Pedro E. M. Lopes, and Alexander D. MacKerell. *Recent Developments and Applications of the CHARMM Force Fields*. In: *WIREs Comput Mol Sci* 2.1 (Jan. 2012), pp. 167–185.

[154] Robert B. Best, Nicolae-Viorel Buchete, and Gerhard Hummer. *Are Current Molecular Dynamics Force Fields Too Helical?* In: *Biophysical Journal* 95.1 (July 2008), pp. L07–L09.

[155] You Xu and Jing Huang. *Validating the CHARMM36m Protein Force Field with LJ-PME Reveals Altered Hydrogen Bonding Dynamics under Elevated Pressures*. In: *Commun Chem* 4.1 (June 2021), p. 99.

[156] Jing Huang, Sarah Rauscher, et al. *CHARMM36m: An Improved Force Field for Folded and Intrinsically Disordered Proteins*. In: *Nat Methods* 14.1 (Jan. 2017), pp. 71–73.

[157] B. H. Zimm and J. K. Bragg. *Theory of the Phase Transition between Helix and Random Coil in Polypeptide Chains*. In: *The Journal of Chemical Physics* 31.2 (Aug. 1959), pp. 526–535.

[158] Martin K. Scherer, Benjamin Trendelkamp-Schroer, et al. *PyEMMA 2: A Software Package for Estimation, Validation, and Analysis of Markov Models*. In: *J. Chem. Theory Comput.* 11.11 (Nov. 2015), pp. 5525–5542.

[159] Nana Heilmann, Moritz Wolf, et al. *Sampling of the Conformational Landscape of Small Proteins with Monte Carlo Methods*. In: *Sci Rep* 10.1 (Oct. 2020), p. 18211.

[160] Michio Iwaoka, Toshiki Suzuki, et al. *Development of SAAP3D Force Field and the Application to Replica-Exchange Monte Carlo Simulation for Chignolin and C-peptide*. In: *J Comput Aided Mol Des* 31.12 (Dec. 2017), pp. 1039–1052.

[161] Israel Cabeza de Vaca, Yue Qian, et al. *Enhanced Monte Carlo Methods for Modeling Proteins Including Computation of Absolute Free Energies of Binding*. In: *J. Chem. Theory Comput.* 14.6 (June 2018), pp. 3279–3288.

[162] Montserrat Penaloza-Amion, Elaheh Sedghamiz, et al. *Monte-Carlo Simulations of Soft Matter Using SIMONA: A Review of Recent Applications*. In: *Front. Phys.* 9 (Mar. 2021), p. 635959.

[163] Peter Turchin, Thomas E. Currie, et al. *Quantitative Historical Analysis Uncovers a Single Dimension of Complexity That Structures Global Variation in Human Social Organization*. In: *Proc. Natl. Acad. Sci. U.S.A.* 115.2 (Jan. 2018).

[164] Christoph Molnar. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. 2nd ed. 2022.

[165] L. S. Shapley. *17. A Value for n-Person Games*. In: *Contributions to the Theory of Games (AM-28), Volume II*. Ed. by Harold William Kuhn and Albert William Tucker. Princeton University Press, 2016, pp. 307–318.

[166] Scott M Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, et al. Vol. 30. Curran Associates, Inc., 2017.

A

APPENDIX

Contents

A.1 List of publications 140
A.2 Acknowledgements 141
A.3 Curriculum Vitae 142

A.1 LIST OF PUBLICATIONS

This thesis is in part a recapitulation and extension of the following peer-reviewed publications that originated in the PhD program:

- Simon Lemcke, Jörn H. Appeldorn, Michael Wand, and Thomas Speck. *Toward a Structural Identification of Metastable Molecular Conformations*. In: *The Journal of Chemical Physics* 159.11 (Sept. 2023), p. 114105

  Contribution: I carried out all the analyses of the data, created all the figures and wrote a complete first draft of the manuscript, which was improved by T. Speck.

- Jörn H. Appeldorn, Simon Lemcke, Thomas Speck, and Arash Nikoubashman. *Employing Artificial Neural Networks to Identify Reaction Coordinates and Pathways for Self-Assembly*. In: *J. Phys. Chem. B* 126.27 (July 2022), pp. 5007–5016

  Contribution: As J. Appeldorn and myself both used EncoderMap, we had several discussions, in which we shared insights and code snippets that influenced our analyses and manuscripts. Furthermore, I proof-read the manuscript and supplemental information.

A.2 ACKNOWLEDGEMENTS

> "There is a theory which states that if ever anyone discovers exactly what the Universe is for and why it is here, it will instantly disappear and be replaced by something even more bizarre and inexplicable. There is another theory which states that this has already happened."
>
> *The Restaurant at the End of the Universe*
> *Douglas Adams*

Pursuing (and finishing) a PhD probably has been the biggest project I ever completed, spanning decades when including all my time in education institutions. After finishing either my life looses all meaning or the "real" life starts, I'm curious to find out. Independent of the outcome I am thankful for all the people I met along the way, for the uncountable interesting conversations and discussions I had and for all the people who supported me along the way.

(further paragraphs deleted from the electronic version of this dissertation)

A.3 CURRICULUM VITAE

(deleted from the electronic version of this dissertation)