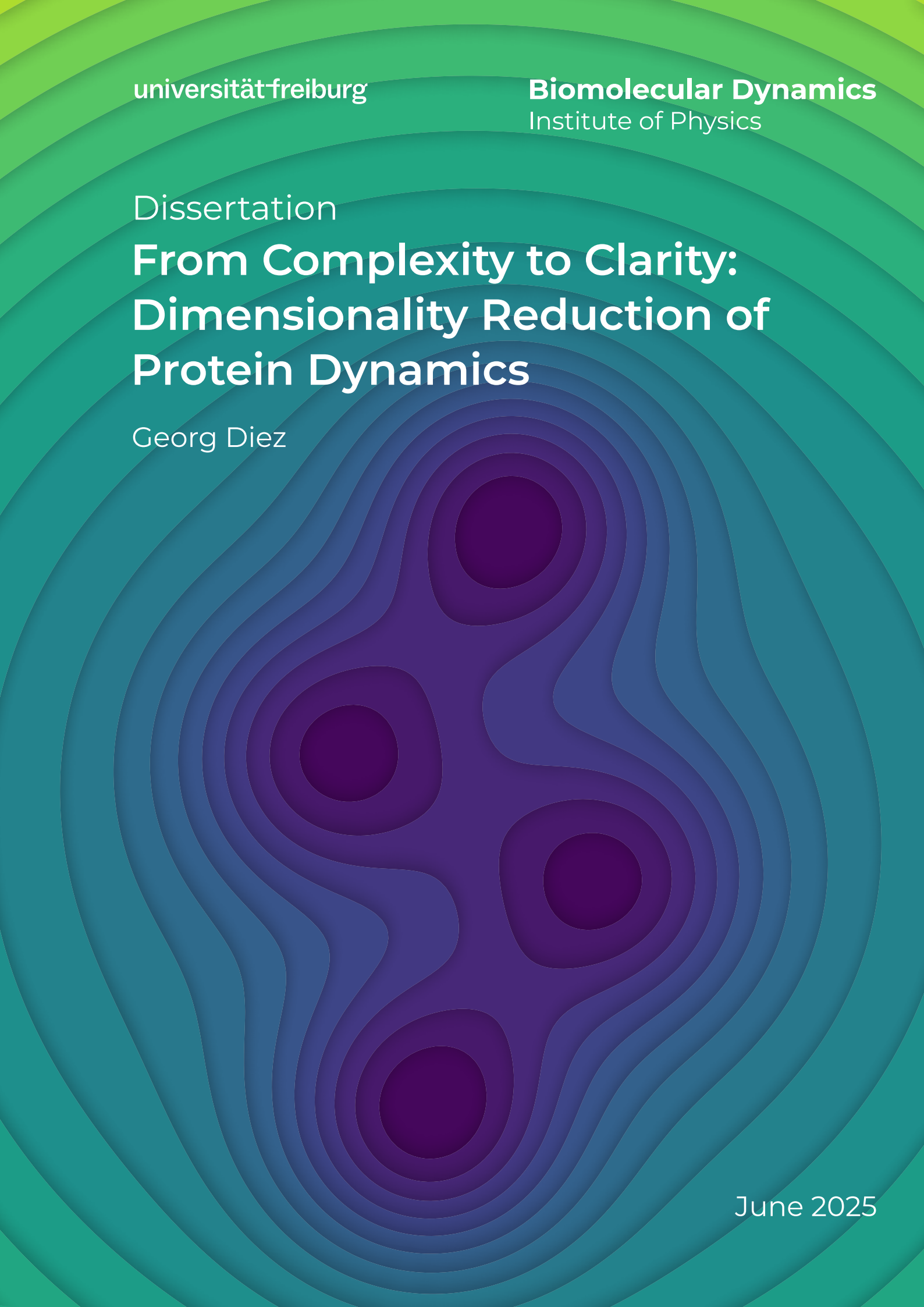
universität freiburg

Biomolecular Dynamics
Institute of Physics

Dissertation

From Complexity to Clarity: Dimensionality Reduction of Protein Dynamics

Georg Diez



June 2025

From Complexity to Clarity

Dimensionality Reduction of Protein Dynamics

DISSERTATION

zur

Erlangung des Doktorgrades

der

Fakultät für Mathematik und Physik

der

Albert-Ludwigs-Universität Freiburg

vorgelegt von

Georg Gabriel Diez

betreut durch

Prof. Dr. Gerhard Stock

June 2025

Institut

Dekan	Prof. Dr. Michael Růžička
Erstgutachter	Prof. Dr. Gerhard Stock
Zweitgutachter	Prof. Dr. Tanja Schilling
Theo. Prüfer	Prof. Dr. Joachim Dzubiella
Exp. Prüfer	Prof. Dr. Günter Reiter
Datum Disputation	09.09.2025

Cover

A complex potential energy landscape for overdamped Langevin dynamics simulations, serving as a testing ground for studying stochastic particle dynamics and dimensionality reduction methods.

Colophon

This document was typeset with the help of KOMA-Script and LaTeX using the kaobook class.
In order to ensure a consistent and modern visual appearance of the figures, prettypyplot[5] was used throughout this thesis, which provides an extended matplotlib[6] style with optimized presets. Most analysis were done using Python together with numpy,[7] scipy,[8] scikit-learn,[9] PyTorch,[10] and msmhelper.[11] All protein renderings presented here were created with PyMOL.[12] Parts of this thesis were linguistically refined with the assistance of (large) language models—specifically Perplexity's R1 1776, Claude 4.0 Sonnet and DeepL Write—for grammar, style, and clarity enhancement. All scientific content, analysis, and conclusions remain the original intellectual contribution of the author.

Acknowledgments

> *"Jude," he says, "there's not an expiration date on needing help, or needing people. You don't get to a certain age and it stops."*
>
> – Hanya Yanagihara (A Little Life)

First and foremost, I express my heartfelt gratitude to my doctoral supervisor, Gerhard, for his guidance, encouragement, and support. His open-mindedness and open-door policy allowed me to freely explore my research interests, while his mentorship and expertise helped transform these ideas into this thesis—thank you for everything!

Another big thank you goes to Daniel, with whom I had the pleasure to work with on several projects within the first half of my PhD. We formed a great team, and I thoroughly enjoyed sharing both our office and also time outside work—whether in Freiburg or at a conference at picturesque Costa d'Amalfi! In this spirit (and speaking of Italy), I also want to thank Sofia, who joined our office a little later. I truly enjoyed our post-lunch espresso sessions and discussions about life, work, and everything in between, and I will miss them dearly. My gratitude also extends to Fabian and Nadja, who joined the office for a shorter time, but made it count, nevertheless!

Beyond the office, many people contributed to the great atmosphere in Gerhard's group. Many thanks to Miriam, Camilla, and Nele, whose laughter and positive attitude resonated from their office, to Emanuel and Ahmed, and to Steffen—who not only helped with all kind of chemical queries—but extended our general knowledge by sharing his vast knowledge of random trivia during lunch. Many thanks also to Matthias, Benni, and Adnan, who were a great support during my initial steps in the group and were always eager to help; and to Victor for guiding Daniel and me through the Vosges! My thanks also go to Katharina for all her help in navigating the complex bureaucracy landscape of the university. Of course, there are many more people in the group I met throughout my PhD, and while I cannot mention everyone individually here, I am grateful to all of them for making this a great and memorable time!

Last but not least, I want to give special thanks to my good friend Marius—some of the best ideas here were born out of our discussions.

I am grateful to Daniel, Sofia, Emanuel, Fabian and Marius for proofreading this thesis and providing valuable feedback.

Georg Gabriel Diez

Abstract

Extracting meaningful insights from molecular dynamics simulations is challenging and typically involves employing dimensionality reduction techniques. In this thesis, we present novel theoretical and computational approaches that address two main pillars of dimensionality reduction, namely feature selection and feature extraction.

In the context of molecular dynamics, feature selection describes the identification of a small subset of coordinates that are most functionally relevant for the characterization of a specific process of interest. We introduce MoSAIC, a method that explores the correlation structure of the input data to distinguish functional dynamics from coordinates that reflect thermal noise. To address the limitations of the linear Pearson correlation coefficient, especially in the case of multidimensional data, we introduce a nonparametric mutual information estimator that captures all dependencies—no matter whether linear or nonlinear. A joint analysis employing both MoSAIC and the normalized mutual information estimator on T4 lysozyme reveals both the mechanistic network of highly correlated coordinates driving the allosteric transition and resulting global conformational changes.

Feature extraction, on the other hand, consists of projecting the original data into a lower-dimensional representation while preserving the essential topological structure of the data. To this end, we present two methods that combine the expressive power of neural networks with physical constraints, yielding interpretable and physically meaningful latent representations of protein dynamics that offer greater insights compared to conventional feature extraction methods.

Table of Contents

List of Figures and Tables

Figures

Tables

List of Publications

The following publications were published as part of this thesis:

[1] G. Diez, D. Nagel, and G. Stock, "Correlation-based feature selection to identify functional dynamics in proteins," J. Chem. Theory Comput. **18**, 5079–5088 (2022),
Contributions: *G.D. and D.N. are co-first authors*; I contributed significantly to the conceptual idea and largely wrote a first version of the manuscript. I carried out the analysis and implementation of the Community Detection methods, applied it to the molecular system T4L, and contributed to the advertised software "MoSAIC". Furthermore, I extended the correlation analysis to further systems (e.g., Chapter 3.4.3) and to conceptually different areas of application (see path separation in Chapter 3.4.4).

[2] M. Post, B. Lickert, G. Diez, S. Wolf, and G. Stock, "Cooperative protein allosteric transition mediated by a fluctuating transmission network," J. Mol. Bio. **434**, 167679 (2022),
Contributions: I performed analysis (MoSAIC on preselected coordinates and free energy analysis) and wrote and revised parts of the manuscript.

[3] D. Nagel, G. Diez, and G. Stock, "Accurate estimation of the normalized mutual information of multidimensional data," J. Chem. Phys. **161**, 054108 (2024),
Contributions: I carried out the application of the normalized mutual information to the molecular system T4L, compared it with various correlation measures commonly used in the literature, and discussed the problems of Cartesian coordinate correlations (see Chapter III of the manuscript). I co-authored the draft of the manuscript.

[4] G. Diez, N. Dethloff, and G. Stock, "Recovering hidden degrees of freedom using Gaussian processes," J. Chem. Phys. **163**, 124105 (2025),
Contributions: I led the development of the conceptual framework, performed all analysis, implemented the GP-VAE software GP-TEMPEST, and wrote and revised a first version of the manuscript.

Decoding the Building Blocks of Life

1

O caos é uma ordem por decifrar.
– José Saramago (O Homem Duplicado)

Imagine a tiny molecular machine designed to autonomously perform specific chemical or physical tasks over and over again. To achieve this, it must gather energy and raw materials from its environment and convert them into desired products. To do so, the machine needs mobility and navigation, and after production, it must transport the product and dispose of any remaining waste. All of these different functions are complex and rely on the coordinated interaction of numerous components, which might fail over time. To guarantee long-term operation, it must detect and repair damaged parts, or perhaps even replicate itself, requiring memory, instructions, and the means to act upon them.[13] In nature, such machines exist as biological cells, and at their core lies an army of nanoscale engineers faithfully carrying out the above-mentioned tasks: proteins.[14,15]

The name "protein" can be derived from the Greek word "πρωτεῖος", translating as "fundamental" or "in the lead". Proteins are the quintessential building blocks of life, living up to their etymology by orchestrating almost every biological process in living organisms. Given that their architecture and function have been perfected through roughly four billion years of evolutionary pressure,[14] it might not come as a surprise that proteins are nature's most structurally complex and functionally sophisticated molecules. They perform a myriad of functions that sustain and regulate life, ranging from cellular functions such as transcription or metabolism to intercellular communication, including ion channel regulation and immune response. This functional diversity is reflected by the fact that proteins account for over 50% of the dry weight of cells.

Fulfilling such diverse functions is only made possible by the flexibility of proteins to adapt their structure depending on the task at hand: like a string of pearls, proteins are made up of different amino acids linked together by peptide bonds.[16] Each amino acid has a unique side chain with distinct physicochemical properties, and the potential interactions among all side chains result in an immense combinatorial space of possible three-dimensional conformations. However, only a small ensemble of energetically favorable conformations is realized in nature through the proteins' folding process. Yet, this folding process is highly complex and inherently error-prone. Interactions with other biomolecules can induce protein misfolding, culminating in the worst case in maladies ranging from cystic fibrosis to devastating neurodegenerative diseases—such as Alzheimer's, Parkinson's, or Huntington's.[17–20] A distinct class of disorders arises from prions[i] (proteinaceous infectious particles), where an infectious protein induces the host's protein to misfold as well, leading to diseases such as Creutzfeldt-Jakob or Mad Cow disease.[21] Given the

[i] The exact disease mechanism is still not fully understood, but it is assumed that the prion nucleates the misfolding of the host's protein into stable large-scale β−amyloids, which are toxic to the host.[13]

central role of proteins in health and disease, understanding their structure, dynamics, and functions is of paramount importance not only for the field of molecular biology but also for developing novel therapeutic strategies.[22,23]

Historically, the concept of proteins can be traced back to the early 19th century, when the Dutch chemist Gerardus Johannes Mulder analyzed various organic compounds and found that they share a common macromolecule, later coined "protein".[24] Almost a century later, in 1926, James B. Sumner made an important discovery that furthered our understanding of proteins. He was the first person to ever successfully crystallize an enzyme, urease, and thereby showed that enzymes represent chemical entities with a specific molecular structure.[25] Twenty years later, when he received the Chemistry Nobel Prize for this discovery, it had become clear that all enzymes are proteins. Another Chemistry Nobel Prize central to protein science was awarded in 1958 to Frederick Sanger for determining the sequence of insulin, thereby laying the foundations for modern protein sequencing.[26] The same year, the field achieved another breakthrough when John Kendrew[ii] unveiled the first three-dimensional structure of a protein by X-ray crystallography—a pioneering achievement that earned him the Nobel Prize in Chemistry together with Max Perutz just four years later.[30,31] This work set the stage for the field of structural biology, which aims at explaining biomolecular mechanisms through the lens of the structure of proteins and other biomolecules.[32]

These pioneering discoveries contributed to a central paradigm of structural biology, namely that the sequence of a protein determines its structure, which in turn determines its function. This idea was eventually formalized by Nobel laureate Christian Anfinsen (*"Anfinsen's dogma"*), who advocated that, under physiological conditions, the native structure of a protein is determined by its acid sequence alone.[33] However, this is only part of the story: Proteins are dynamic systems whose function often emerges from conformational changes between their metastable states. Moreover, intrinsically disordered proteins,[iii] which make up about 30% of the eukaryotic proteome,[36] completely lack a stable tertiary structure and yet perform crucial functions in cell signaling and regulation.[37,38]

These findings expanded our understanding of the structure-function relationship, leading to a paradigm shift that now recognizes protein dynamics as a crucial link between structure and function:[39]

$$\text{Sequence} \rightarrow \text{Structure} \rightarrow \textbf{Dynamics} \rightarrow \text{Function}$$

With the growing recognition of protein dynamics' importance, the requirement for new experimental and computational tools to study these motions has become apparent. On the one hand, several decades of technological advances have led to experimental techniques capable of obtaining detailed structural snapshots of protein through complementary techniques: X-ray crystallography[30,31] revealed the first protein structures, nuclear magnetic resonance spectroscopy[40–43] allowed for the study of protein dynamics in solution, and cryo-electron microscopy[44–46] enabled macromolecular structure determination without the need for crystallization. On the other hand, time-resolved techniques such as ultrafast transient absorption spectroscopy,[47] time-resolved X-ray spectroscopy,[48,49] or time-resolved infrared spectroscopy[50] track protein mo-

[ii] Looking at the structure of myoglobin, Kendrew *et al.* had directly recognized the great challenge protein folding imposes, a problem that still remains partially unsolved despite decades of progress,[27] especially in the field of artificial intelligence:[28,29] *"Perhaps the most remarkable features of the molecule are its complexity and its lack of symmetry. The arrangement seems to be almost totally lacking in the kind of regularities which one instinctively anticipates, and it is more complicated than has been predicated by any theory of protein structure. Though the detailed principles of construction do not yet emerge, we may hope that they will do so at a later stage of the analysis."*[30]

[iii] A particularly impressive example for intrinsically disordered proteins can be found in tardigrades (also known as water-bears). Tardigrades are microscopic animals that can endure extreme conditions, including desiccation, radiation and temperatures as low as $T = -273$°C, which even allows them to survive in space.[34] When exposed to desiccation, the tardigrade-specific IDP is expressed and forms a glass-like matrix that physically prevents protein denaturation, aggregation and membrane fusion.[35]

tions across different temporal regimes. Although all of these methods have significantly increased our understanding of protein structure and dynamics, each one faces fundamental trade-offs in spatial or temporal resolution.

Physics-based molecular dynamics simulations act as a powerful complementary tool to experimental approaches, serving as a "computational microscope" that bridges spatiotemporal resolution gaps by providing atomic-level descriptions of dynamic processes that might be inaccessible to conventional observation methods.[51–54] By numerically solving Newton's equations of motion for all atoms, molecular dynamics simulations can reveal mechanistic insights into protein function, such as conformational changes between metastable states, allosteric signaling pathways, ligand binding or unbinding events, and folding intermediates.[55,56] And the progress in molecular dynamics simulations over the past decades has been nothing short of remarkable: the first simulation of a protein—the relatively small bovine pancreatic trypsin inhibitor—in 1977 lasted just 8.8 picoseconds, did not include water, and involved less than 600 atoms.[51] Today, with special-purpose supercomputers like Anton3,[57] molecular dynamics simulations can routinely reach microseconds to even milliseconds timescales, capturing the dynamics of biologically relevant systems containing millions of atoms, including explicit solvent.[58–60] However, the evolution of molecular dynamics simulations has not been limited to hardware advancements. In parallel, significant processes in force fields,[61–63] enhanced sampling techniques,[64–68] and the integration of machine learning methods[69–72] have also led to remarkable improvements. Collectively, these advancements have doubled the accessible timescales reached by atomistic molecular dynamics simulations approximately every 1.3 years over the past four decades—outpacing Moore's law.[73,74]

Despite these advances—and despite the governing equations of protein dynamics being well understood—extracting meaningful insights from vast amounts of data continues to be challenging. Later Nobel laureate Philip W. Anderson famously captured these difficulties in his seminal article "More Is Different",[75] where he introduced the concept of *broken symmetry*: for larger and more complex aggregates of elementary particles (including proteins), it is not sufficient to understand the collective behavior in terms of the individual particles. Instead, entirely new properties of such systems emerge at each level of complexity, requiring completely new concepts and generalizations to understand the increasingly complex behavior of larger systems.

Additionally, the inherent stochasticity and high dimensionality of complex proteins render analytical derivations and first principle models impractical, driving a shift towards data-driven approaches—a paradigm fueled by the rapid advancements in machine learning and the growing availability of large-scale simulation datasets. A common strategy to this end involves projecting the high-dimensional data onto a lower-dimensional set of collective variables $\boldsymbol{x} = \{x_i\}$ that characterize the dominant biomolecular motion. These collective variables are designed to capture the most important conformational changes in the protein and are theoretically motivated through the *manifold hypothesis*[76, iv]—the assumption that high-dimensional biomolecular motions can be described by only a few intrinsic degrees of freedom.[78–81]

[iv] For proteins, the manifold hypothesis physically translates into nonlinear couplings in the protein giving rise to cooperative effects that effectively reduce the total number of degrees of freedom.[77]

If the collective variables effectively encode the dominant dynamics, the biomolecular process of interest can be mapped onto a free energy landscape $\Delta G(\boldsymbol{x}) \sim -\ln p(\boldsymbol{x})$, where $p(\boldsymbol{x})$ is the equilibrium probability distribution of the collective variables. Within this free energy landscape, metastable conformations of the protein correspond to local minima that are separated by kinetic barriers characterizing the transition pathway.[77] This perspective naturally motivates the application of Markov state models—a kinetic framework that discretizes the collective variables into a set of metastable states and consequently approximates the protein dynamics in terms of memoryless jumps between these states.[82–89] This approximation is only justified when a timescale separation between fast intrabasin fluctuations and slow interbasin transitions exists, which then reflects the Markov property: once the trajectory enters a metastable basin, its prior history becomes irrelevant for predicting the future dynamics. Crucially, Markov state models can leverage long-timescale dynamics from an ensemble of short molecular dynamics simulations[v], effectively bridging computationally accessible microsecond timescales with biologically relevant timescales of milliseconds to seconds.[90,91] As trajectories are only required to reach local minima, running parallelized simulations enables the exploration of rare events such as protein folding, allosteric transitions, or ligand binding events.[91–95]

[v] After discretization of the collective variables into a set of metastable states, Markov state models estimate a transition probability matrix $\boldsymbol{T}(\tau_{\text{lag}})$ over a lag time τ_{lag} and then propagate the dynamics via the Chapman-Kolmogorov equation $\boldsymbol{T}(n\tau_{\text{lag}}) = \boldsymbol{T}(\tau_{\text{lag}})^n$.

However, this idealized scenario, where the collective variables can be chosen such that a clear timescale separation between the slow functional dynamics and the fast intrabasin fluctuations exists, faces significant practical challenges when constructing the Markov state model. More precisely, in the context of molecular dynamics simulations of protein, such a workflow typically involves:[77,96]

1. The first step is the selection of a suitable **set of coordinates** that decouples the protein's internal motion from its global translations and rotations within the simulation box. Common choices include internal coordinates, such as interresidual distances and dihedral angles, but the ideal type of coordinate strongly depends on the process under study.[80,95,97,98]
2. **Feature Selection:** Within the chosen coordinate system, a small subset of relevant coordinates is selected that effectively describes the process of interest.[99–102, vi] This step seeks to enhance the signal-to-noise ratio and, therefore, significantly impacts the quality of the resulting model: *Garbage in, garbage out.*[103,104]
3. **Feature Extraction** then projects this selected subset into a lower-dimensional space of collective variables, mitigating the effects of the curse of dimensionality and enabling density-based estimates. This greatly facilitates subsequent analysis, such as constructing the free energy landscape $\Delta G(\boldsymbol{x})$. Common feature extraction techniques include principal component analysis,[80,105,106] time-lagged independent component analysis[107] or, popular nonlinear dimensionality reduction techniques such as, e.g., multidimensional scaling,[108] t-SNE,[109] or UMAP.[110] Recently, deep learning architectures have also been successfully applied to this task, the most popular being autoencoders[111–115] or Boltzmann generators.[116]
4. The trajectory, now expressed in terms of collective variables, is discretized by **clustering** into metastable microstates—ensembles of structurally similar configurations.[69] Geometrical clustering al-

[vi] For example, consider the coordinate system of inter-residual distances: For a protein with N amino acids, $N(N-1)/2$ distances can be computed. Many of these contain highly redundant information or are constant due to the structural stability of the protein (e.g. tertiary structure). Only a small fraction of these distances is typically relevant for the process of interest.[1,2]

gorithms such as $k-$means[117] or density-based approaches do the job.[115,118–120]

5. In the next step, **dynamical lumping** merges microstates dynamically into macrostates—groups of microstates that are kinetically strongly connected and share the same basin in the free energy landscape. This step facilitates human interpretation of the model while preserving the key characteristics of the model, such as the slowest timescales or important intermediates.[121–123]
6. Then, finally, the Markov state model is constructed by estimating the **transition probability matrix** $\boldsymbol{T}(\tau_{\text{lag}})$ for a specific lag time, approximating the protein dynamics as a memoryless Markov process between the obtained macrostates. To this end, more steps are often required in order to obtain a meaningful model. Examples include dynamical coring,[124] hidden Markov state models,[125] the inclusion of memory,[126–128] or better estimates for the transition matrix given a dynamical lumping.[129]

While the workflow outline above—from a high-dimensional molecular dynamics simulation towards a Markov state model—has provided profound insights into biomolecular dynamics, it is not without its limitations. The whole procedure, spanning from the selection of the coordinate system, feature selection, feature extraction, geometric clustering, and dynamical lumping into macrostates, inevitably involves many simplifications and approximations that lead to a loss of information.

Thus, the predictive performance of a Markov state model (or any other model) is fundamentally limited by the information content of its input features. Hence, feature selection must carefully balance the trade-off between relevance and sparsity: including too many irrelevant or redundant coordinates can obscure relevant dynamics, while overly aggressive filtering risks omitting important degrees of freedom, potentially violating Markovianity. An optimal feature selection not only preserves the essential protein dynamics but also improves the predictive power of the resulting model and helps to understand the biomolecular mechanisms in terms of a few, but very important coordinates. This challenge is amplified by the strong link to feature extraction: suboptimal coordinate choices propagate through subsequent analysis steps, leading to models not capable of describing the proteins' essential dynamics.

For instance, principal component analysis prioritizes geometric variance over kinetic relevance, potentially masking important slow degrees of freedom with irrelevant large amplitude motions like dangling termini.[77] Similarly, the focus of the time-lagged independent component analysis on the slowest timescales may struggle to distinguish between rare functional transitions and slow but non-functional motion like transitions between left- and right-handed helices, where one of them is hardly populated.[95] Yet, linear methods like principal component analysis or time-lagged independent component analysis remain the most popular choices for feature extraction due to their interpretability and computational efficiency.[77,106,107,130] Since these linear methods rely on orthogonal projections, their ability to resolve nonlinear conformational changes in the data is fundamentally limited. While this does not necessarily pose a problem for high-dimensional spaces, it can become a significant limitation when the desired latent representation must be limited to a few dimensions.[104,131] Nonlinear methods offer more variability in

this regard, but they are prone to overfitting, which is why they need to be properly regularized.[112,115,132–134]

Outline

In this thesis, we address two of the major tasks that have to be performed prior to constructing a dynamical model, namely the steps of *feature selection* and *feature extraction*. Modern molecular dynamics simulations routinely generate vast amounts of data by describing the motion of proteins along all $3N$ Cartesian coordinates of each N atoms. While arguably all of these atoms contributed to the function of the protein through the folding process, the proteins' functional dynamics can be described by only a small fraction of the degrees of freedom.

To extract the relevant information from the data, we introduce a feature selection method—called MoSAIC—in the chapter 3 that systematically identifies this small subpart of the coordinates that perform collective motion and separates it from noisy coordinates describing thermal fluctuations and non-functional movements. We identify these functional groups of coordinates by block-diagonalization of a similarity matrix between internal coordinates. Balancing computational efficiency while accurately capturing statistical dependencies, the Pearson correlation coefficient represents an excellent choice for the similarity measure.

However, the Pearson correlation coefficient suffers from a range of drawbacks in certain scenarios—especially in the context of multidimensional coordinates. Therefore, we propose a new nonparametric estimator for normalized mutual information in the chapter 4 that captures nonlinear statistical dependencies beyond the limitations of Pearson correlation and can be readily employed for multidimensional coordinates.

In chapter 5, we demonstrate the effectiveness of these two approaches by conducting a comprehensive analysis of the open↔closed allosteric transition of the protein T4 lysozyme. Both correlation-based approaches systematically reveal different aspects: the local interaction patterns identified by MoSAIC explain the molecular mechanisms underlying the transition, while the resulting global conformational changes are captured by normalized mutual information.

Moving beyond instantaneous correlations, we turn to temporal dynamics and propose two complementary feature extraction techniques in the chapter 6, which combine the expressive power of neural networks with physical constraints and hence enable the construction of physically meaningful models. First, using graph neural networks, we introduce an autoencoder network that directly operates on the proteins' graph structure and thus mimicking a local propagation of perturbations throughout the protein. This results in robust, physically meaningful, low-dimensional representations of the data. Second, considering the sequential nature of molecular dynamics simulations, we employ Gaussian processes to model its temporal relationships. By using the Gaussian processes as a prior in a Bayesian framework for representation learning, we obtain low-dimensional embeddings that preserve the Markovian properties in the input data.

Theory & Methods 2

2.1 Proteins

A protein molecule is a chain of amino acids that are linked together by covalent peptide bonds, which is why proteins are also called polypeptides. As noted in the introduction, these biomolecules exhibit remarkable functional diversity due to variations in their size and shape.

To adapt to their unique shape and function, proteins can rely on 20 different proteinogenic amino acids acting as fundamental building blocks. Structurally, all amino acids share a common structure, namely

- an amino group (NH_2),
- a central carbon atom, referred to as C_α atom, to which the side chain R is attached,
- and a carboxyl group (COOH).

By sharing electrons between the carbon atom of the carboxyl group from one amino acid and the nitrogen atom of the NH_2 group of another amino acid, the proteins' *primary structure* (amino acid sequence) is established. This condensation reaction, where a water molecule is eliminated, is called a *peptide bond* (see Fig. 2.1) and leads to the formation of the *protein backbone*, a repeating sequence of N-C_α-C atoms (see Fig. 2.2). The asymmetry of amino acids introduces directionality to the amino acid chain: the end with the free amino group is referred to as the N-terminus, and the other end carrying the free carboxyl group is named C-terminus.[14]

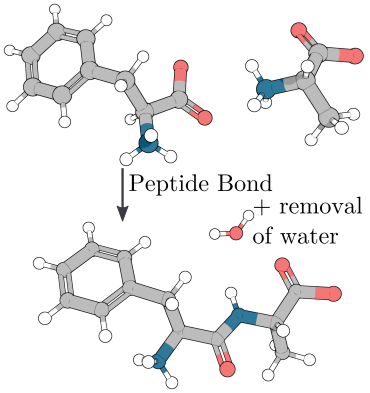


Figure 2.1 | Proteins are chains of amino acids linked together by covalent peptide bonds. By the removal of water, the carbon atom of the carboxyl group (shown here: phenylalanine) shares electrons with the nitrogen atom of the amino group of another amino acid (here: alanine) to form a peptide bond.

The *secondary structure* of proteins refers to regular, repetitive structural motifs stabilized by hydrogen bonds that form between NH and CO groups in the backbone.[135] Among the most common secondary structures are:

- α-helices, where the NH group of the i-th amino acid hydrogen bonds to the CO group of the $(i+4)$-th amino acid, and
- β-sheets, in which several adjacent segments of the backbone are connected by hydrogen bonds between NH and CO groups.

At the *tertiary structure* level, the overall three-dimensional fold of a protein is primarily stabilized by interactions among the side chains R of different amino acids. The unique chemical characteristics of each amino acid side chain—including polarity, charge, size and hydrophobicity—result in a complex network of interactions that determine the final shape of the protein. For example, hydrophobic amino acids like leucine or valine try to avoid contact with surrounding water molecules by clustering together in the interior of the folded protein, while polar side chains like e.g. lysine form hydrogen bonds with water molecules on the protein surface.[14]

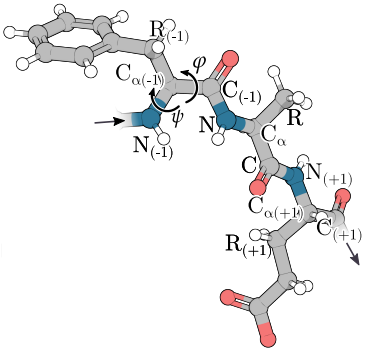


Figure 2.2 | Structure of a protein consisting of amino acids linked along the backbone N-C_α-C and side chains R. The section of the protein shown here consists of the amino acids with their different side chains: phenylalanine ($R_{(-1)}$), alanine (R), and glutamine ($R_{(+1)}$). We also introduce the two dihedral angles φ and ψ, defined by the bond between the $C(-1)$-N-C_α-C atoms and N-C_α-C-$N(+1)$ atoms, respectively.

These interactions result in a vast number of possible conformations, but the final number of (meta-)stable folded structures is relatively small due to energetic constraints: the native fold of the protein minimizes the protein's free energy ΔG, defined through a balance of stabilizing interactions such as hydrogen bonds or hydrophobic side chains and destabilizing factors such as conformational entropy.[39]

As far as this thesis is concerned, the first three levels of protein structure fully describe the proteins studied here as they function as single units. However, for the sake of completeness, it is worth mentioning the *quaternary structure* of proteins, which describes the structural arrangement in (large) proteins consisting of at least two smaller protein chains. Since this requires interactions among multiple individually folded and structurally stable protein chains, quaternary structure is typically a feature more common in larger proteins (in contrast to the first three levels of protein structure, which every protein possesses).

2.2 Molecular Dynamics Simulations

As outlined above, the interactions between all the atoms constituting the protein, as well as the interactions between them and their surrounding solvent, ultimately determine the dynamics of the protein. While the underlying physical equations that govern the atomistic motions are well understood (i.e., the Schrödinger equation), their application to systems like proteins is computationally (still) not feasible due to their size and structural complexity.[13]

Instead, molecular dynamics (MD) simulations offer the best current alternative to approximate the dynamics of proteins in high spatiotemporal resolution. MD simulations rely on a number of approximations, such as neglecting relativistic effects, the decoupling of electronic and nuclear motion via the Born-Oppenheimer approximation, and the treatment of the atom nuclei as classical point particles. Effectively, these assumptions allow us to approximate the dynamics of a protein classically through Newton's equations of motion describing the conservative forces acting on all N atoms

$$m_i \frac{\mathrm{d}^2 \boldsymbol{r}_i}{\mathrm{d}t^2} = -\nabla_{\boldsymbol{r}_i} U(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N). \tag{2.1}$$

Here, $\boldsymbol{r}_i$ denotes the position of atom i with mass m_i and $\nabla_{\boldsymbol{r}_i} U$ is the gradient of the potential with respect to $\boldsymbol{r}_i$. Numerical integration of Eq. (2.1) for all atoms using an appropriate step size, typically in the femtosecond range,[i] yields the time evolution or *trajectory* of the system.

[i] This is due to fast vibrational motion in the C-H bonds of the protein.

The potential energy U, also called force field, in Eq. (2.1) contains empirical terms modeling the interactions between atoms in the protein and its surrounding solvent. As shown in Fig. 2.3, two classes of interactions are typically distinguished: bonded and nonbonded interactions

$$U = U_{\text{bonded}} + U_{\text{nonbonded}}. \tag{2.2}$$

Here, the bonded term includes interactions between neighboring atoms that are covalently bound, such as bond stretching, bond bending, and

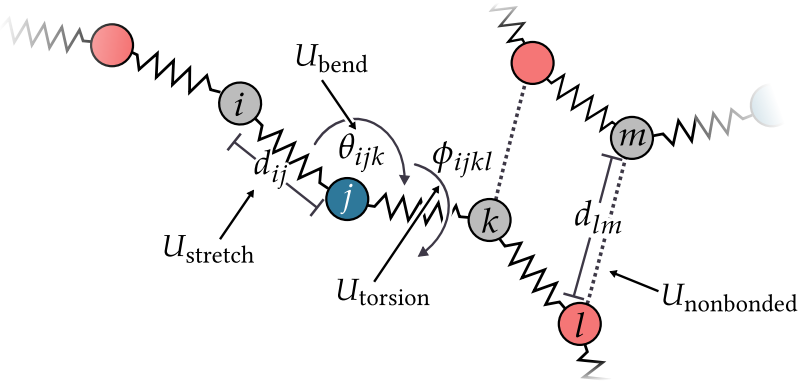


Figure 2.3 | Force field model of the dynamics of a protein. The energy components of a typical force field: In a spring-like picture, bonded interactions are modeled by harmonic potentials and several metastable conformations are allowed by the torsional potential. Nonbonded interactions between non-covalently bound atoms are modeled by Lennard-Jones potentials and Coulombic interactions.

bond torsion. The harmonic potentials governing bond stretching and bond bending mimic Hooke's law for springs, where the restoring force is linear in the displacement from the equilibrium position $\left(d_{ij} - d_{ij}^{\text{eq}}\right)^2$ or $\left(\theta_{ijk} - \theta_{ijk}^{\text{eq}}\right)^2$

$$U_{\text{stretch}} = \sum_{i,j} K_{ij} \left(d_{ij} - d_{ij}^{\text{eq}}\right)^2, \tag{2.3}$$

$$U_{\text{bend}} = \sum_{i,j,k} K_{ijk} \left(\theta_{ijk} - \theta_{ijk}^{\text{eq}}\right)^2. \tag{2.4}$$

Here, K_{ij} and K_{ijk} (as well as K_{ijkl} later) denote the force constants and are parameterized for all kinds of different combinations of atoms i, j, k, l. While this is straightforward for bond stretching, the bond bending term can be thought of as an angular spring. For proteins, in general, bond stretching is much stiffer than bond bending $K_{ij} \gg K_{ijk}$. Unlike the two harmonic terms, the torsional potential has multiple minima and maxima along the torsion angle ϕ_{ijkl}—also called the *dihedral* angle—and is given by

$$U_{\text{torsion}} = \sum_{i,j,k,l} K_{ijkl} \left[1 + \cos\left(n\phi_{ijkl} - \phi_{ijkl}^{\text{eq}}\right)\right]. \tag{2.5}$$

This means that these resulting torsional degrees of freedom (d.o.f.) can readily adopt different conformations (given that the energy barriers are sufficiently low), resulting in them being the primary source of conformational changes within the protein backbone.[13]

In contrast to the bonded terms, which describe local interactions between neighboring atoms, the nonbonded terms in Eq. (2.1) model interactions between non-covalently bound atoms. To this end, only atom pairs l and m that are at least separated by three or four intervening bonds[ii] are included. Interactions between uncharged atoms are modeled by the Lennard-Jones potential [first two terms in Eq. (2.6)], while Coulomb's law additionally accounts for the electrostatic interactions (last term)

$$U_{\text{nonbonded}} = \sum_{l<m} \left[\left(\frac{a_{lm}}{d_{lm}^{12}}\right) - \left(\frac{b_{lm}}{d_{lm}^{6}}\right) + \frac{q_l q_m}{\epsilon d_{lm}}\right]. \tag{2.6}$$

[ii] Here, we use l and m instead of i and j for the atom indices to emphasize that they are not neighboring atoms as shown in Fig. 2.3. d_{lm} denotes the distance between atoms l and m.

a_{lm} specifies the Lennard-Jones repulsive strength, while b_{lm} is the attraction coefficient, and d_{lm} the interatomic distance. The Coulombic term consists of the charges of atoms l and m with q_l and q_m, respectively, and ϵ is the effective dielectric constant.

All the parameters above have been parameterized through classical laboratory experimental data or quantum mechanical calculations to faithfully approximate the dynamics of proteins and other biomolecules at computational scales far below the quantum mechanical level.[53] These parameters, in combination with the functional form of the potential, are called *force fields*. Popular force fields for MD simulations are e.g. AMBER,[63,136] GROMOS,[137,138] or CHARMM.[62]

2.3 From Complexity to Clarity: Dimensionality Reduction

In MD simulations, the configuration of a protein is fully described at any time step t by its $3N$-dimensional Cartesian coordinates $\boldsymbol{r} = ((r_1)_x, \dots, (r_N)_z)$ and the corresponding momenta $\boldsymbol{p_r}$ (neglecting the solvent d.o.f.). In classical mechanics, both variables $(\boldsymbol{r}, \boldsymbol{p_r})$ together define the state of the system in the $6N$-dimensional phase space. However, as the autocorrelation functions of the velocities decay rapidly in the picosecond time range while conformational changes typically occur on the microsecond to millisecond timescale, the momenta are commonly neglected in the analysis of the protein dynamics.[39,139,140]

For a typical protein consisting of $N \approx 10^3$–10^5 atoms, this still leaves us with such a high dimensionality that it is neither possible nor desirable to follow the dynamics of each atom individually in full detail. Furthermore, given that a typical MD simulation of a protein usually outputs between 10^4 and 10^6 time steps (despite being computed over significantly more time steps due to the femtosecond time step), we are faced with a configuration space where most of the regions are empty—a phenomenon well known and feared as the *curse of dimensionality*.[131,141] Therefore, the analysis of protein dynamics in full space, e.g., by characterizing a free energy landscape, is forbidden because it is basically impossible to estimate a probability density function in such empty spaces faithfully.[142,143] Fortunately, the atomic coordinates of a protein are not randomly or even uniformly distributed in the configuration space, but due to high correlations among the atomic coordinates of biomolecular systems, they rather live on a low-dimensional manifold in $\mathbb{R}^{3N}$.[iii]

[iii] In fact, this is true for most natural high-dimensional datasets, which is formulated by the *manifold hypothesis*.[144] This motivates the use of dimensionality reduction techniques and is the basis for success in whole fields of machine learning and statistics, which is why some even call this effect the *blessing of dimensionality*.[145]

This motivates the search for a low-dimensional set of collective variables (CVs) that capture the most relevant dynamics of the biomolecular process of interest while discarding irrelevant d.o.f., such as fast vibrational motion or coordinates that remain constant throughout the simulation—a procedure coined *dimensionality reduction*.[77] Generally speaking, the dimensionality reduction workflow for MD simulations consists of several major steps that ultimately aim to construct interpretable models of the protein dynamics. Typically, this process involves:

1. Choice of a suitable coordinate system

2. Feature selection
3. Feature extraction
4. Clustering of metastable states in the reduced dimensional space
5. Construction of kinetic models

While some of these steps are more specific to the analysis of MD simulation data and require domain-specific knowledge, such as the choice of a suitable coordinate system or the model-building step, others are commonly used in all fields of machine learning, especially feature selection and feature extraction.[141] Clustering is not necessarily required for all applications but is crucial in the context of Markov state models (see Sec. 2.4).

2.3.1 Choice of Suitable Coordinate Systems

Cartesian coordinates are generally not well suited for dimensionality reduction due to their inevitable mixing of internal motion with global translation and rotation of the protein within the solvent.[97]

A common attempt to circumvent this problem is to apply a rotational and translation fit,[146] which removes global rotation and translation by aligning the trajectory to a reference structure through the minimization of the atomic least square distances. However, this approach fails to address the core issue: due to its flexibility, a protein can still exhibit relative rotational motion between different parts, even under zero-total angular momentum conditions.[147]

Instead, internal coordinates, such as the dihedral angles φ/ψ or inter-residual distances d_{ij}, are not plagued by this problem because they decouple internal and global motion by definition. Since the force field is given in terms of such internal coordinates (compare Sec. 2.2), they represent a natural choice for further analysis of the MD simulation data.

Dihedral Angles

Dihedral angles (φ, ψ, see Fig. 2.2) are particularly useful for describing the formation of secondary structures, such as α-helices or β-sheets.[148,149] However, their periodicity, $(\varphi, \psi) \in [-\pi, \pi]$, requires an appropriate treatment in order to prevent discontinuities in the data. Accounting for the circular statistics, this can be achieved by either gap-shifting the data such that the maximal gap is shifted to the periodic boundary[150, iv] or transforming the angles into the (linear) metric coordinate space:[151]

$$\varphi \mapsto \begin{cases} x = \cos(\varphi) \\ y = \sin(\varphi). \end{cases} \tag{2.7}$$

While they scale linearly with the size of the protein, dihedral angles only indirectly capture the formation of the important tertiary structure of a protein, which significantly reduces their applicability for describing global conformational dynamics of proteins.[95]

[iv] In the (ϕ, ψ)-conformational space, most residues only populate small areas, leaving natural gaps that can be placed at the periodic boundary.[150]

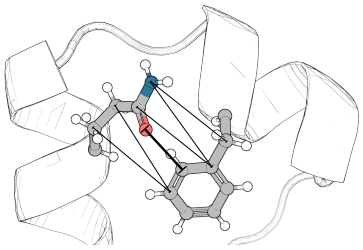


Figure 2.4 | Calculating the minimal distance between two residues requires the computation of all pairwise distances between the heavy atoms of both residues.

Inter-residual Distances

In contrast, inter-residual minimal distances d_{ij} (as shown in Fig. 2.4) between the closest heavy (i.e., non-hydrogen) atoms of two residues i and j have been shown to faithfully capture the structural changes in the tertiary structure of a protein (only a few representatives shown here).[80,95] and is defined by

$$d_{ij}(t) = \min_{n,m} |\boldsymbol{r}_{i,n}(t) - \boldsymbol{r}_{j,m}(t)|. \tag{2.8}$$

Here, n and m are the indices of the heavy atoms of residues i and j. Of course, this is not limited to heavy atoms but can also be applied to the C_α atoms of the residues, which then focuses more on the backbone of the protein. However, since we are interested in the physicochemical processes that govern structural changes, minimal distances between heavy atoms come in handy as they directly report on interactions between the side chains, such as bonds. Focusing on bonds or *contacts* also helps to circumvent the largest drawback of inter-residual distances, which is their quadratic scaling $\mathcal{O}(N^2)$ with respect to the number of residues N: either focusing on native contacts[152,v] or on distances that form a contact during the course of the simulation,[80,95] can significantly reduce complexity while simultaneously highlighting the most relevant interactions. Commonly, two residues i and j are considered to be in contact if their minimum distance d_{ij} is below a certain threshold, typically set to $d_{ij} \leq 0.45\,\text{nm}$.[80]

[v] The use of only native contacts is motivated by the "*principle of minimal frustration*", which hypothesizes that evolution has led to strong correlations between native contacts and protein folding landscapes in order to guarantee efficient folding via smooth, funneled pathways without misfolding traps.[153]

2.3.2 Feature Selection

The process of restricting analysis to only a subset of the available coordinates—such as e.g. only focusing on contact distances—is referred to as *feature selection*.[154,155] Yet, it might be beneficial to even further narrow down the coordinate selection as many biologically relevant structural changes are spatially confined. Hence, only a small subset of the internal coordinates is typically involved in a specific biomolecular process, while the remaining coordinates are often uncorrelated or describe noise.[1,2] Examples of this include coordinates that remain stable during the functional process (e.g., stable contacts), coordinates that exhibit random but large-scale motion (e.g., wildly dangling terminal ends), or coordinates that describe slow but nonfunctional transitions (for example, rare transitions between left- and right-handed helices, where the right-handed one is hardly populated).[95,156]

The advantages of excluding such irrelevant d.o.f. are manifold: first, popular feature extraction techniques like principal component analysis (PCA)[105] or time-lagged component analysis (TICA)[107] aim to maximize variance or autocorrelation (see Sec. 2.3.3), respectively, which could lead to a misclassification of the above-mentioned processes as the most relevant ones. For meaningful subsequent modeling, it is therefore crucial to exclude large amplitude/slow autocorrelation coordinates that are not (co)related to the process of interest. Second, internal coordinates, especially inter-residual distances, extensively carry redundant information due to overlapping spatial relationships, while PCA works best with only a few weakly correlated input coordinates.[157] Lastly, feature selection

can directly shed light onto the most relevant processes, sometimes resulting in a few interpretation-ready coordinates that may explain the biomolecular process even without further modeling.[158]

Being one of the main pillars in the field of machine learning, countless methods have been developed over the last decades. Fundamentally, feature selection techniques can be categorized into *supervised* and *unsupervised* methods, and in the realm of MD simulation analysis, we can distinguish between physics-based methods and those that are data-driven.

Physics-based feature selection methods aim to identify key coordinates leveraging physical principles,[vi] as done by e.g. the force distribution analysis,[159] which constructs an underlying network of residues based on pairwise mechanical force and strain differences among residues. Another notable example is functional mode analysis, which constructs a linear collective variable that is maximally correlated with a user-defined target quantity $f(t)$.[99] This target quantity $f(t)$—such as solvent-accessible surface area or cleft volume—is typically physically motivated and requires a priori biological knowledge, which is why this approach is supervised. Finally, as a final example of physics-based feature selection, we want to mention committor-based techniques that seek to determine key coordinates by focusing on transition probabilities between metastable states.[160] Given two metastable states A, and B, the committor p_B represents the probability of reaching B before returning to A. Hence, when expressed in terms of a few key coordinates, the comittor can reveal the governing coordinates that drive the transition $A \to B$.[161]

[vi] Because physics-based feature selection techniques rely on prior knowledge of physically meaningful quantities—such as regions of interest in a protein or metastable states—to guide the identification of key coordinates, they are predominantly supervised.

In contrast, data-driven approaches exploit statistical relationships within the data and consult machine learning algorithms to extract important coordinates. For example, similar to the committor framework, Brandt *et al.* use a set of predefined metastable states and an XGBoost architecture to rank internal coordinates based on their importance for classifying a set of metastable states.[101,162,163] oASIS[164] and spectral oASIS[165] characterize datasets through a kernel matrix $\boldsymbol{C} \in \mathbb{R}^{N\times N}$,[vii] measuring the similarity among data points in the set. By relying on the Nymström method,[166] an optimal feature subset k is then selected by approximating the kernel matrix $\boldsymbol{C}$ by a sparse, reconstructed kernel matrix $\tilde{\boldsymbol{C}} = \boldsymbol{C}_k \boldsymbol{W}_k^{\dagger} \boldsymbol{C}_k^T$, where $\boldsymbol{C}_k$ denotes a subset of k columns of $\boldsymbol{C}$ and $\boldsymbol{W}_k$ denotes the kernel matrix of this subset. Last but not least, another class of data-driven feature selection methods computes the full pairwise similarity (such as Pearson correlation or mutual information) between all coordinates in the data set, and subsequent clustering groups these coordinates into clusters of coordinates with similar behavior.[1,167] We will discuss this in detail in Chapter 3.

[vii] N still denotes the number of data points here.

2.3.3 Feature Extraction

After having performed feature selection, we now represent the dynamics of the protein in terms of a few key coordinates $\boldsymbol{x}$—those that are most relevant for describing the process of interest—often referred to as CVs. Despite feature selection, the resulting feature space often remains high-dimensional (typically 10^1–10^3 dimensions), which does not

necessarily allow for straightforward analysis and raises the following questions:[2,77,95]

- Can the effective dimensionality be further reduced?
- How can we extract the most relevant information from the subset of selected coordinates?

Concerning the former question, studies show that the intrinsic dimension for MD simulation dynamics is in the order of $d \lesssim 10$.[78–81,150]

Assuming the protein system to be in thermal equilibrium, a common modeling practice is to characterize the state of the protein by its *free energy landscape*[168]

$$\Delta G(\boldsymbol{x}) = -k_{\mathrm{B}} T \ln p(\boldsymbol{x}), \tag{2.9}$$

where k_{B}, T, and $p(\boldsymbol{x})$ denote Boltzmann's constant, temperature, and the local probability density at $\boldsymbol{x}$, respectively. Regions with high local densities $p(\boldsymbol{x})$ represent metastable states, while the barriers between them govern transition kinetics.

However, the construction of free energy landscapes that accurately capture the dynamics of the protein depends critically on a reasonable density estimate $p(\boldsymbol{x})$. Because high-dimensional data often suffers from sparsity and noise, the density estimation in such spaces is notoriously difficult.[131,169,170] As a remedy, *feature extraction* techniques can be employed, which project the high-dimensional data $\boldsymbol{x} \in \mathbb{R}^D$ into a lower dimensional *latent space* $\boldsymbol{z} \in \mathbb{R}^d$, where $d \ll D$.[viii] Such a transformation $\boldsymbol{x} \mapsto \boldsymbol{z}$ significantly increases the probability density in the latent space $p(\boldsymbol{z})$ compared to the one in the original space $p(\boldsymbol{x})$, hence greatly facilitating subsequent analysis.

[viii] Unlike *feature selection*, in which only a subset of already existing features are chosen, *feature extraction* creates entirely new features by extracting the most relevant information from the input data.

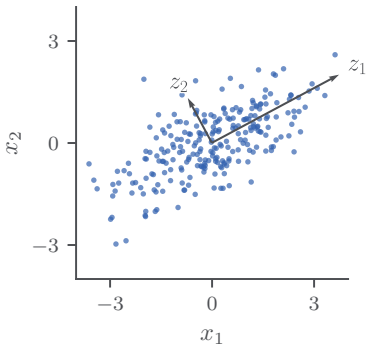


Figure 2.5 | PCA identifies directions of maximum variance which can be used for a low dimensional representation of the input data.

Principal Component Analysis

Due to its simplicity and computational efficiency, principal component analysis (PCA) is one of the most common techniques for feature extraction.[131,171,172] Its goal is to find an optimal low-dimensional representation $\boldsymbol{z} \in \mathbb{R}^d$ of the high-dimensional input data $\boldsymbol{x} \in \mathbb{R}^D$ by relying solely on linear orthogonal projections $\boldsymbol{W}$, as depicted in Fig. 2.5. Optimal in this context means that when we first encode (project) the input data $\boldsymbol{z} = f_{\mathrm{E}}(\boldsymbol{x}) = \boldsymbol{W}^\top \boldsymbol{x}$ and then decode it back into the original space $\hat{\boldsymbol{x}} = f_{\mathrm{D}}(\boldsymbol{z}) = \boldsymbol{W}\boldsymbol{z}$, the *reconstruction* $\hat{\boldsymbol{x}}$ should be as close as possible to the original input $\boldsymbol{x}$. This can be achieved by minimizing the loss function $\mathcal{L}$ defined by

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{N} \sum_{n=1}^{N} \| \boldsymbol{x}_n - \underbrace{f_{\mathrm{D}}\left(f_{\mathrm{E}}\left(\boldsymbol{x}_n; \boldsymbol{W}\right); \boldsymbol{W}\right)}_{\hat{\boldsymbol{x}}} \|^2. \tag{2.10}$$

We will now show that minimizing this loss in Eq. (2.10) leads to $\boldsymbol{W} = \boldsymbol{U}_d$, where $\boldsymbol{U}_d$ contains the d eigenvectors with the largest corresponding eigenvalues of the covariance matrix $\boldsymbol{\Sigma} = \frac{1}{N} \boldsymbol{X}_c^\top \boldsymbol{X}_c$.[131, ix] To this end, we consider $\boldsymbol{x}$ to be mean free and want to express $\boldsymbol{x}$ as a linear combination of our latent representation $\boldsymbol{z}$, $\boldsymbol{x}_n = \sum_{i=1}^{d} z_{ni} \boldsymbol{w}_i$. To simplify the derivation, we only consider a one-dimensional latent space, meaning

[ix] $\boldsymbol{X}_c$ is the mean free $N \times D$ data matrix.

that we seek a single projection direction $\boldsymbol{w}_1 \in \mathbb{R}^D$ such that $\boldsymbol{x}_n = z_{n1}\boldsymbol{w}_1$, $\boldsymbol{z}_1 = [z_{11}, \dots, z_{N1}]$ is the one-dimensional latent representation.[x]

[x] Higher order projections $\boldsymbol{w}_2$, $\boldsymbol{w}_3$ etc. can be derived through induction (see e.g. chapter 20.1.2.3 in Ref. 131).

Our goal is to minimize $\mathcal{L}$

$$\mathcal{L}(\boldsymbol{w}_1, \boldsymbol{z}_1) = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - z_{n1}\boldsymbol{w}_1\|^2 \tag{2.11}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \left(\boldsymbol{x}_n^\top \boldsymbol{x}_n - 2z_{n1}\boldsymbol{w}^\top \boldsymbol{x}_n + z_{n1}^2 \boldsymbol{w}_1^\top \boldsymbol{w}_1\right). \tag{2.12}$$

Assuming orthonormality $\boldsymbol{w}_i^\top \boldsymbol{w}_j = \delta_{ij}$, we take the derivative w.r.t. z_{n1} and set it to zero

$$0 = \frac{\partial}{\partial z_{n1}}\mathcal{L}(\boldsymbol{w}_1, \boldsymbol{z}_1) = \frac{1}{N}\left(-2\boldsymbol{w}_1^\top \boldsymbol{x}_n + 2z_{n1}\right) \Rightarrow z_{n1} = \boldsymbol{w}_1^\top \boldsymbol{x}_n. \tag{2.13}$$

Substituting back into Eq. (2.12) and ignoring the constant term $\boldsymbol{x}_n^\top \boldsymbol{x}_n$ yields

$$\mathcal{L}(\boldsymbol{w}_1) = -\frac{1}{N}\sum_{n=1}^{N} z_{n1}^2 = -\frac{1}{N}\sum_{n=1}^{N} \boldsymbol{w}_1^\top \boldsymbol{x}_n \boldsymbol{x}_n^\top \boldsymbol{w}_1 = -\boldsymbol{w}_1^\top \boldsymbol{\Sigma} \boldsymbol{w}_1, \tag{2.14}$$

where $\boldsymbol{\Sigma}$ denotes the covariance matrix (because $\boldsymbol{x}$ is mean free). To avoid trivial optimization by $\|\boldsymbol{w}_1\| \to \infty$, we constrain the solution to $\boldsymbol{w}_1^\top \boldsymbol{w}_1 = 1$ via Lagrange multiplier λ_1

$$\tilde{\mathcal{L}}(\boldsymbol{w}_1) = \boldsymbol{w}_1^\top \boldsymbol{\Sigma} \boldsymbol{w}_1 + \lambda_1\left(\boldsymbol{w}_1^\top \boldsymbol{w}_1 - 1\right), \tag{2.15}$$

Derivation w.r.t. $\boldsymbol{w}_1$ yields

$$\frac{\partial}{\partial \boldsymbol{w}_1}\tilde{\mathcal{L}}(\boldsymbol{w}_1) = 2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\lambda_1 \boldsymbol{w}_1 = 0 \Rightarrow \boldsymbol{\Sigma}\boldsymbol{w}_1 = \lambda_1 \boldsymbol{w}_1. \tag{2.16}$$

This shows that the optimal (linear) projection direction [in the sense of Eq. (2.10)] is an eigenvector of the covariance matrix $\boldsymbol{\Sigma}$. Furthermore, since we want to maximize Eq. (2.16), we left-multiply $\boldsymbol{w}_1^\top$ and find $\boldsymbol{w}_1^\top \boldsymbol{\Sigma} \boldsymbol{w}_1 = \lambda_1$, meaning that we select the eigenvector with the largest corresponding eigenvalue λ_1. To sum this up, we directly highlight the key feature of PCA, namely the link between loss minimization and variance $\mathbb{V}$ maximization: taking the expectation value $\langle \dots \rangle$

$$\langle z_{n1} \rangle = \langle \boldsymbol{x}_n^\top \boldsymbol{w}_1 \rangle = \langle \boldsymbol{x}_n \rangle^\top \boldsymbol{w}_1 = 0, \tag{2.17}$$

establishes the connection

$$\mathbb{V}[z_1] = \langle z_1^2 \rangle - \langle z_1 \rangle^2 = \frac{1}{N}\sum_{n=1}^{N} z_{n1}^2 - 0 = -\mathcal{L}(\boldsymbol{w}_1) + \text{const.} \tag{2.18}$$

While this is the standard formulation of PCA, it is often better to use the correlation matrix instead of the covariance matrix to mitigate the impact of different scales of the input features. This is particularly relevant for proteins since irrelevant large-amplitude motion, such as wildly dangling terminal residues, would easily overshadow important functional but small-scale motion.[95,98]

Autoencoder

In the last section, we learned how PCA can be used to perform an "optimal" projection of high dimensional input data $\boldsymbol{x} \in \mathbb{R}^D$ onto a low(er) dimensional *bottleneck* (latent space) $\boldsymbol{z} \in \mathbb{R}^d$, where $d \ll D$, using a linear and orthogonal mapping. This information bottleneck approach can be generalized by replacing the linear functions with nonlinear transformations, which are realized using neural networks. The result is what is called an *autoencoder* (AE) consisting of an encoder and decoder part (see Fig. 2.7).[111,173,174] Fewer dimensions in the bottleneck force the network to efficiently learn a compact yet meaningful representation of the input data and requires simultaneously discarding irrelevant noise. Analogous to Eq. (2.10), the standard reconstruction loss for an autoencoder is given by

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{N} \sum_{n=1}^{N} \left\| \boldsymbol{x}_n - f_{\mathrm{D},\boldsymbol{\phi}} \left[f_{\mathrm{E},\boldsymbol{\theta}} \left(\boldsymbol{x}_n \right) \right] \right\|^2, \tag{2.19}$$

where f_E and f_D are the encoder and decoder functions parameterized by the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, respectively. These learnable parameters are optimized using backpropagation.[175,176] Restricting $f_{\mathrm{E},\boldsymbol{\theta}}$ and $f_{\mathrm{D},\boldsymbol{\phi}}$ to linear functions and assuming the loss corresponds to the mean squared loss [as in Eq. (2.19)] recovers the latent space identified by PCA, rendering PCA as a special case of autoencoders.[177,178]

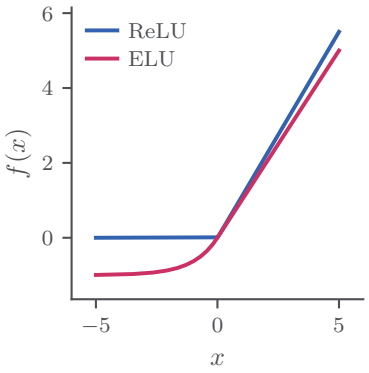


Figure 2.6 | Commonly used activation functions for neural networks.

However, in general, both the encoder and decoder are recursively decomposed into a composition of simple but nonlinear activation functions $f_{\boldsymbol{\theta}_l}^{(l)}$ acting at each layer

$$f_{E,\boldsymbol{\theta}}(\boldsymbol{x}) = \left(f_{\mathrm{E},\boldsymbol{\theta}^L}^{(L)} \circ f_{\mathrm{E},\boldsymbol{\theta}^{L-1}}^{(L-1)} \circ \cdots \circ f_{\mathrm{E},\boldsymbol{\theta}^1}^{(1)} \right) (\boldsymbol{x}), \tag{2.20}$$

where $\circ$ is the composition operator and $\boldsymbol{\theta}^{(l)}$ denotes the learnable parameters of layer $l \in [1, \ldots, L]$. Two commonly used activation functions are shown in Fig. 2.6.

Traditional autoencoders use fully connected layers (see Fig. 2.7), in which the activation $a_k^{(l)}$ of each neuron k is computed by applying an activation function to the weighted sum of all activations from the previous layer

$$\boldsymbol{a}_k^{(l)} = f_{\boldsymbol{\theta}^{(l)}}^{(l)} \left(\boldsymbol{W}^{(l)} \boldsymbol{a}^{(l-1)} + \boldsymbol{b}^{(l)} \right), \tag{2.21}$$

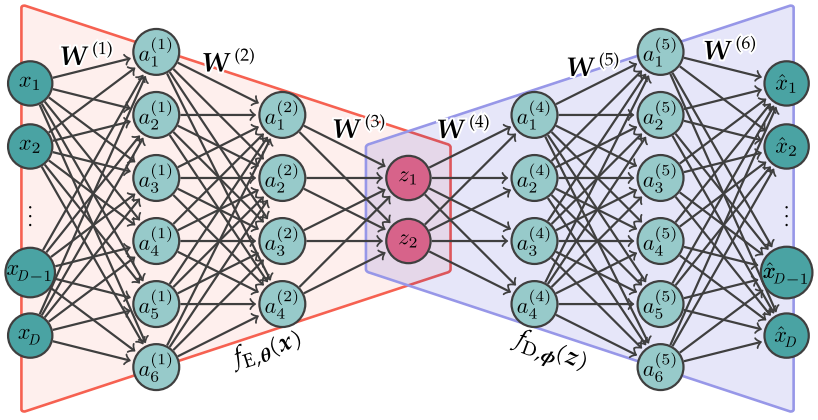


Figure 2.7 | An autoencoder, consisting of an encoder f_E (red) and a decoder f_D (blue), represented by a fully connected feed-forward neural network with inputs $x_1, \ldots, x_n$, latent representation z_1, z_2, and reconstructed outputs $\hat{x}_1, \ldots, \hat{x}_n$.

where $\boldsymbol{\theta}^{(l)} = \{\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)}, \dots\}$ includes all learnable parameters for layer l, such as e.g. the weight matrix $\boldsymbol{W}^{(l)}$ and biases $\boldsymbol{b}^{(l)}$. Despite the simplicity of the activation functions, introducing these simple nonlinearities allows neural networks to approximate very complex functions and to identify important patterns in the data—given that they are *deep* enough. This makes them a powerful tool not only for feature extraction.

While Eq. (2.19) is the standard formulation of the autoencoder loss function, the great flexibility of neural networks allows one to adjust it depending on the problem. For example, Lemke and Peter[113] extended the loss with a sketch-map[179] cost function that preserves proximity information among data points in the high-dimensional and low-dimensional space. A comparison between PCA and autoencoders for the Swiss roll dataset can be found in Fig. 2.8, demonstrating that the autoencoder is able to capture the nonlinear structure of the Swiss roll but fails to account for the constant density along the surface of the roll.

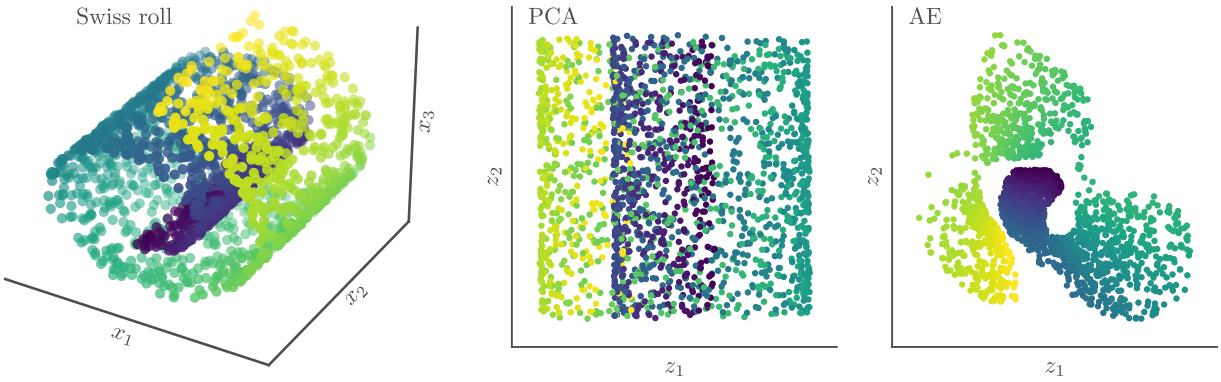


Figure 2.8 | Comparison of feature extraction via PCA and autoencoder applied to the Swiss roll dataset (on the left). The original three-dimensional structure is projected onto two dimensions using PCA (center) and a nonlinear autoencoder (right). Compared to the PCA, the autoencoder does a better job of capturing the complex, nonlinear relationship of the Swiss roll, but it fails to account for the constant density along the surface of the roll.

Latent Variable Models

Both PCA and autoencoders operate in a purely deterministic framework; that is, they lack the ability to probabilistically capture continuous probability densities resulting from iteratively solving Newton's equation of motion (see Sec. 2.2). In practice, this limitation is less problematic for PCA due to its restriction to linear mappings, which maintain proportional relationships between data points in the high- and low-dimensional space. However, the nonlinearity of autoencoders can result in ruptures in the probability density that alter the topology and disrupt the continuity of the latent space (see Fig. 2.8 for an example). This poses a significant challenge when modeling protein dynamics in terms of a free energy landscape since, e.g., transition pathways need to pass through regions with continuous probability density.

Latent variable models address these challenges by introducing a *prior distribution* $p(\boldsymbol{z})$ over the latent space, transforming deterministic mappings

into Bayesian frameworks as depicted in Fig. 2.9.[180,181] This prior regularizes the latent space by encoding our assumptions about the latent structure before we have observed any data—for instance, enforcing smoothness and connectivity through a simple Gaussian $p(\boldsymbol{z}) \sim \mathcal{N}(0, \mathbb{1})$.[112, xi]

[xi] Such a simple (and non-learnable) prior obviously oversimplifies the typically rich structural versatility of MD simulations, which is why more sophisticated priors have been developed in order to better capture more complex latent embeddings. For example, a Gaussian mixture model can reflect several metastable states by a multimodal distribution[115] and data-dependent priors, such as the VampPrior, adapt to the observed data during training.[132] But physics-informed priors are also investigated, e.g., Tiwary and coworkers constructed a prior leveraging the overdamped Langevin equation to naturally describe the temporal evolution of the system.[134]

In a Bayesian model, the latent variables $\boldsymbol{z}$ are probabilistically linked to the observations $\boldsymbol{x}$ via their joint probability density function $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$, where $p(\boldsymbol{x}|\boldsymbol{z})$ is the conditional probability of $\boldsymbol{x}$ given $\boldsymbol{z}$.[180] Crucially for the analysis of MD simulation data, Bayes' theorem inverts this relationship

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})}, \tag{2.22}$$

rendering the *posterior probability* $p(\boldsymbol{z}|\boldsymbol{x})$ the central quantity for the identification of CVs and thus, the construction of meaningful free energy landscapes $\Delta G(\boldsymbol{z}) \sim -\langle \ln p(\boldsymbol{z}|\boldsymbol{x}) \rangle_{\boldsymbol{x}}$.

The observations $\boldsymbol{x}$ are modeled to be probabilistically caused by the underlying latent variable $\boldsymbol{z}$ through two steps:[182]

1. $\boldsymbol{z} \sim p(\boldsymbol{z})$:[xii] Sampling according to *a priori* assumed prior $p(\boldsymbol{z})$
2. $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{z})$: Given a family of deterministic functions $f_{\boldsymbol{\theta}}(\boldsymbol{z}) : \mathbb{R}^d \mapsto \mathbb{R}^D$ parameterized by learnable parameters $\boldsymbol{\theta}$, a deterministic decoder maps the sampled latent variables $\boldsymbol{z}$ to an observable conformation $\boldsymbol{x}$. While the function $f_{\boldsymbol{\theta}}$ is deterministic, $\boldsymbol{x} = f_{\boldsymbol{\theta}}(\boldsymbol{z})$ is now a random variable in the high-dimensional input feature space $\mathbb{R}^D$ due to the randomness of $\boldsymbol{z}$.

[xii] $\sim$ means "is sampled from"

The goal of our model is now to optimize the parameters $\boldsymbol{\theta}$ such that when samples are drawn from the prior $\boldsymbol{z} \sim p(\boldsymbol{z})$ and subsequently decoded, the resulting generated high-dimensional conformations $\boldsymbol{x} = f_{\boldsymbol{\theta}}(\boldsymbol{z})$ closely resemble the conformations in the training data. To formalize this, we seek to maximize the marginal likelihood of each observed conformation under the entire generative process,

$$p(\boldsymbol{x}) = \int \mathrm{d}\boldsymbol{z}\, p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}). \tag{2.23}$$

The marginal $p(\boldsymbol{x})$ (also called evidence) is not only required for the generation of realistic conformations but also ensures that our assumptions about the latent structure are included in the posterior [see Eq. (2.22)]. Unfortunately, it is usually intractable for high-dimensional data such as MD simulations due to the integration over a high-dimensional $\boldsymbol{z}$ and becomes a computational hurdle for computing the posterior. A common strategy in modern Bayesian statistics is thus the use of *varia-*

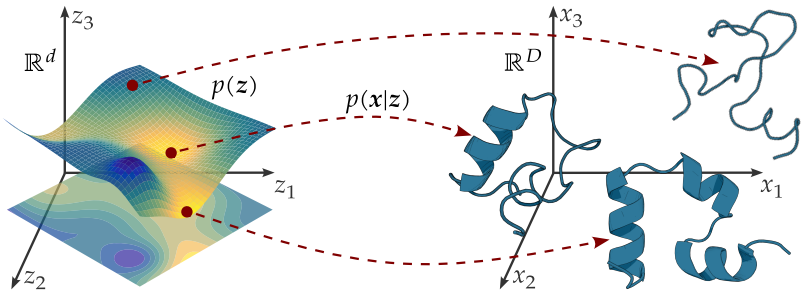


Figure 2.9 | Schematic concept of a latent variable model. On the left, a prior probability distribution $p(\boldsymbol{z})$ is shown. $\boldsymbol{z}$ resides in a latent space with reduced dimensionality $d \ll D$. Each sample $\boldsymbol{z}' \in \mathbb{R}^d$ corresponds to a high-dimensional protein conformation $\boldsymbol{x}' \in \mathbb{R}^D$ via the stochastic decoder $p(\boldsymbol{x}|\boldsymbol{z})$.

tional inference to approximate probability densities in Eq. (2.22) for large datasets.[180,183]

Variational Autoencoders

In order to understand which latent variables $\boldsymbol{z}$ explain the sampled observations $\boldsymbol{x}$, we want to compute the posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$. However, the direct computation via Eq. (2.22) is infeasible due to the intractability of the evidence $p(\boldsymbol{x})$. Variational autoencoders (VAEs) address this issue by the use of variational inference.[112,182,184]

To this end, the idea is to approximate the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ by a family $\mathcal{Q}$ of approximate densities over the latent variables, which are easier to evaluate. While they do not need to be specified yet, we could think of them as e.g. approximating the complex free energy landscape of a protein with a collection of harmonic wells.[115] Mathematically, the goal is to find the function $q(\boldsymbol{z}) \in \mathcal{Q}$ that best approximates the true posterior by minimizing the Kullback-Leibler (KL) divergence[xiii]

$$\tilde{q}(\boldsymbol{z}) = \underset{q(\boldsymbol{z})\in\mathcal{Q}}{\arg\min}\, D_{\mathrm{KL}}\left[q(\boldsymbol{z}) \,\|\, p(\boldsymbol{z}|\boldsymbol{x})\right]. \tag{2.24}$$

[xiii] The Kullback-Leibler divergence measures the proximity of two probability distributions $p(x)$ and $q(x)$ by quantifying how much information is lost when using $q(\boldsymbol{x})$ instead of $p(\boldsymbol{x})$:

$$D_{\mathrm{KL}}[p(x) \parallel q(x)] = \sum_{x\in\mathcal{X}} p(x) \ln \frac{p(x)}{q(x)}.$$

It is non-negative, asymmetric and zero when $p(x) = q(x)$.

The KL divergence is given as

$$D_{\mathrm{KL}}\left[q(\boldsymbol{z}) \,\|\, p(\boldsymbol{z}|\boldsymbol{x})\right] = \left\langle \ln q(\boldsymbol{z}) - \ln p(\boldsymbol{z}|\boldsymbol{x}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z})}, \tag{2.25}$$

where $\langle \dots \rangle_{\boldsymbol{z}\sim q(\boldsymbol{z})}$ denotes the expectation over $\boldsymbol{z}$. We use Bayes' theorem to substitute the intractable posterior and factor out $p(\boldsymbol{x})$ of the expectation since it does not depend on $\boldsymbol{z}$

$$= \left\langle \ln q(\boldsymbol{z}) - \ln p(\boldsymbol{x}|\boldsymbol{z}) - \ln p(\boldsymbol{z}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z})} + \ln p(\boldsymbol{x}). \tag{2.26}$$

Since $q(\boldsymbol{z})$ was not specified and could, in theory, be any distribution, we intentionally construct it such that it depends on $\boldsymbol{x}$ to focus on latent variables $\boldsymbol{z}$ that are likely to have generated $\boldsymbol{x}$, i.e. $q(\boldsymbol{z}|\boldsymbol{x})$. Rearranging Eq. (2.26) yields the *evidence lower bound* (ELBO)

$$\ln p(\boldsymbol{x}) - \underbrace{D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z}|\boldsymbol{x})\right]}_{\text{Approximation error}} = \underbrace{\left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})}}_{\text{Reconstruction}} - \underbrace{D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z})\right]}_{\text{Regularization}} \tag{2.27}$$

This equation nicely summarizes the key concept of a VAE: using a latent variable model, the data [described by the log-evidence $p(\boldsymbol{x})$], is approximated by the ELBO (right-hand side), with the approximation error quantified by $D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z}|\boldsymbol{x})\right]$. Since the KL divergence is strictly non-negative, the ELBO provides a lower bound on $\ln p(\boldsymbol{x})$. Maximizing the ELBO thus implies 1.) the improvement of the reconstruction accuracy and 2.) the regularization of the latent space by aligning $q(\boldsymbol{z}|\boldsymbol{x})$ with the prior.

In practice, VAEs use (deep) neural networks:

- An **encoder network** $q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$, parameterized by learnable parameters $\boldsymbol{\theta}$, maps observations to parameters of distribution over latent variables (typically mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x})$ and variance $\boldsymbol{\Sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x})$)

- A **decoder network** $p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})$, parameterized through learnable parameters $\boldsymbol{\phi}$, reconstructs observations from sampled latent variables

Unlike for deterministic autoencoders which employ a direct mapping $\boldsymbol{x} \to \boldsymbol{z} \to \hat{\boldsymbol{x}}$, standard backpropagation is not possible for VAEs due to the non-differentiable sampling step $\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})$ prior to reconstruction. As a remedy, the *reparameterization trick*[112] can be employed, which transforms the non-differentiable sampling step into a differentiable function with external randomness. Specifically, for the commonly used Gaussian approximate posterior $q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}\left\{\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}), \mathrm{diag}\left[\boldsymbol{\Sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x})\right]\right\}$, this means:

$$\boldsymbol{z} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sqrt{\boldsymbol{\Sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x})} \odot \boldsymbol{\epsilon}, \tag{2.28}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbb{1})$ and $\odot$ indicates element-wise multiplication. Employing a Gaussian distribution for both the approximate posterior $q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ and prior $p(\boldsymbol{z})$ allows us to compute the KL divergence term in Eq. (2.27) in closed form:

$$D_{\mathrm{KL}}\left[q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z})\right] = \frac{1}{2}\Big(\mathrm{tr}\left[\boldsymbol{\Sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x})\right] + \boldsymbol{\mu}^T_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}) - d - \ln\det\left[\boldsymbol{\Sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x})\right]\Big), \tag{2.29}$$

where d is the dimensionality of the latent space. The calculation in closed form significantly improves computation efficiency and training stability.

By connecting latent variable models with deep learning, VAEs offer a powerful probabilistic framework for analyzing MD simulation: rather than providing point estimates for protein conformations like a traditional AE, VAEs naturally capture the uncertainty and thermal fluctuations inherent in sampled trajectories into smooth distributions. VAEs are successfully applied for designing entirely new molecules[185–187] and are extensively used for the identification of CVs for MD simulation data.[115,134,188–193]

2.3.4 Clustering

After having identified suitable CVs to characterize the biomolecular process of interest, analysis and interpretation might profit from partitioning the conformational space into a set $\mathbb{S} = \{S_1, \ldots, S_m\}$ of m distinct *metastable states*—regions or conformations where the protein tends to remain for a while before transitioning into another state. Such a coarse-graining of the dynamics enables the calculation of transition statistics, which allows the estimation of important conformational pathways and/or timescales (see Sec. 2.4).[85]

While a myriad of different clustering algorithms exist,[194,195] their application to MD simulation data demands careful selection and parameterization since there is no problem-agnostic measure that could be assessed for evaluating the suitability of a specific clustering algorithm. Fundamentally, almost all applications of clustering in the field of MD simulations fall into one of the two categories[196]

- **Partitioning schemes** (most notably k-means[117,197] and k-medoids[198]) partition the conformational space through Voronoi-tesselation by minimizing intra-cluster distances (see Fig. 2.10, left). The number of clusters k must be predefined and, therefore, does not reflect an intrinsic characteristic of the system.
  Partitioning schemes not requiring a predefined k are hierarchical/agglomerative methods that iteratively merge data points according to a linkage criterion (such as the smallest inter-cluster distance). Starting with each data point as its own cluster, the merging procedure continues until all data points are contained within one single cluster. This hierarchical sequence of merges can be visualized as a dendrogram, of which the final clustering can be obtained by selecting a cutoff value.[199]
- **Density-based schemes**[118–120,200,201] estimate the probability density of the conformational space. Regions with locally high density are regarded as cluster cores and are expanded to include nearby data points (see Fig. 2.10, right).

Arguably, the most popular clustering algorithm for MD simulation data is k-means, which requires the user to specify the number of states k beforehand. The algorithm then iteratively relocates the position $\{\boldsymbol{\mu}_i\}_{i=1}^{k}$ of the k cluster centers to minimize the within-cluster variance

$$\Sigma^2 = \sum_{i=1}^{k} \sum_{j=1}^{N_i} \left\| \boldsymbol{z}_j - \boldsymbol{\mu}_i \right\|^2, \tag{2.30}$$

where $\boldsymbol{z}_j$[xiv] is the location of the j^{th} data point, and N_i is the number of data points in cluster i.

k-means generally effectively identifies spherical clusters with similar densities through the Voronoi-tesselation approach (see Fig. 2.10), but it struggles when it comes to irregular geometries or distributions with strongly varying densities. For MD simulation data, this poses serious problems: clusters often fail to align with free energy barriers, and metastable states might be split into multiple ones, effectively creating artificial boundaries within a metastable state or, even worse, combining two kinetically distinct states (as e.g. seen for the three-pointed star and its neighbor in Fig. 2.10). Even though some heuristics like the "elbow method"[202] and silhouette score[203] for the optimal choice of k exist, determining an appropriate number of states remains delicate.[xv]

In practice, however, the number of (micro-)states is typically chosen to be very large ($k \gtrapprox 10^3$) in order to avoid combining multiple kinetically

[xiv] While the clustering typically takes place in the latent space with decreased dimensionality $d \ll D$, partitioning approaches like k-means or k-medoids remain applicable in high-dimensional spaces as they do not require density estimates.

[xv] This limitation becomes evident when computing both measures for uniformly distributed data points, which completely lack any inherent structure: both methods propose $k \geq 1$, which would result in a partitioning that introduces artificial boundaries into perfectly homogeneous data.

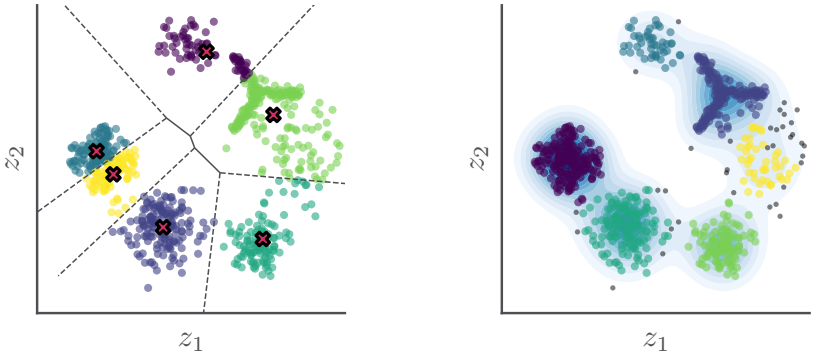


Figure 2.10 | Comparison of two principal clustering approaches for MD data. For both methods, the clusters are indicated by different colors. *Left:* partitioning methods such as k-means or k-medoids divide the conformational space into Voronoi cells, where x markers indicate the cluster centers.
Right: Density-based methods cluster the data points according to their density basin. The blue contour lines represent the density estimate and gray points indicate "noise" not assigned to a cluster.

disconnected states into one. Instead, a dynamical coarse-graining of the (micro-)states is performed afterward, which reduces the number of states down to a few (macro-)states, allowing for a humanly interpretable model.[129] Popular lumping methods for this purpose include the most-probable-path algorithm[121] or Perron-cluster cluster analysis.[86,204,205]

On the other hand, density-based methods were developed and adapted to better align with the physical properties of biomolecular systems. Robust density-based clustering,[150] for example, constructs the free energy landscape $\Delta G(\boldsymbol{z})$ based on a local estimate of the free energy at every data point:

$$\Delta G(\boldsymbol{z}') = -k_{\mathrm{B}} T \ln\left[\frac{p_R(\boldsymbol{z}')}{\max_{\boldsymbol{z}} p_R(\boldsymbol{z})}\right], \tag{2.31}$$

where

$$p_R(\boldsymbol{z}') = \sum_{i=1}^{N} \Theta\left[R - d(\boldsymbol{z}_i, \boldsymbol{z}')\right]. \tag{2.32}$$

Here, $\Theta[\dots]$ is the Heaviside step function and $d(\boldsymbol{z}_i, \boldsymbol{z}')$ is the d-dimensional Euclidean distance defined as $d^2(\boldsymbol{z}, \boldsymbol{z}') = \sum_{j=1}^{d} \left(z_j - z'_j\right)^2$. Starting from the lowest local free energy estimate in Eq. (2.31), the full free energy landscape can be hierarchically constructed, and regions of locally high density serve as seed points for the clusters separated by low-density barriers. This allows to cut metastable states precisely at the free energy barrier, making density-based approaches more adaptable to the irregular geometries common in MD simulation data without requiring a predefined number of states. While these properties make density-based clustering schemes particularly attractive for MD simulation data, they are strongly affected by the curse of dimensionality compared to partitioning schemes: faithful density estimation requires low dimensional spaces of $d \lessapprox 10$ dimensions, which is why a prior feature extraction step is indispensable in this case.

2.4 Markov State Models

Under the core assumption that the system's future evolution depends only on its current state (Markov property), the proteins' dynamics can be approximated in terms of memoryless jumps by a Markov State Model (MSM).[85,89] This assumption heavily depends on the selection of CVs and requires 1.) a clear timescale separation between interstate transitions and intrastate fluctuations and 2.) ergodicity within the state network (no kinetically disconnected subsets in $\mathcal{S}$).

Given a state partitioning $\mathcal{S} = \{S_1, \dots, S_m\}$, the transition statistics can be extracted by simply counting the pairwise transitions after a lag-time τ_{lag} and storing them in the so-called transition count matrix:

$$\mathbf{T}^{\mathrm{c}}(\tau_{\mathrm{lag}}) = \begin{pmatrix} \#(S_1 \to S_1; \tau_{\mathrm{lag}}) & \dots & \#(S_1 \to S_m; \tau_{\mathrm{lag}}) \\ \vdots & \ddots & \vdots \\ \#(S_m \to S_1; \tau_{\mathrm{lag}}) & \dots & \#(S_m \to S_m; \tau_{\mathrm{lag}}) \end{pmatrix} \tag{2.33}$$

The transition count matrix is one of the major strengths of MSMs: rather than relying on one very long trajectory that samples the complete biomolecular process, transition statistics can be aggregated from an ensemble of short trajectories (given that they reach local equilibrium). Transition probabilities of jumping from state i to j within the lag-time τ_{lag} can be obtained by row-normalizing the transition count matrix

$$T_{ij}(\tau_{\text{lag}}) = \frac{T^c_{ij}(\tau_{\text{lag}})}{\sum_{l=1}^{m} T^c_{il}(\tau_{\text{lag}})}. \tag{2.34}$$

The choice of the lag-time τ_{lag} critically impacts the resulting transition matrix $\boldsymbol{T}$ and should be chosen larger than the time required to relax within the metastable states.[126] At the same time, excessive τ_{lag} sacrifices temporal resolution since the MSM cannot resolve dynamics on shorter timescales than τ_{lag}. For equilibrium systems, detailed balance ensures microscopic reversibility:

$$\pi_i T_{ij} = \pi_j T_{ji}, \tag{2.35}$$

where the stationary distribution π_i of state i is defined through the left eigenvalue problem

$$\boldsymbol{\pi T} = \boldsymbol{\pi} \tag{2.36}$$

with dominant eigenvalue $\lambda = 1$. From the remaining eigenvalues $\lambda < 1$, the implied timescales t_i can be derived

$$t_i(\tau_{\text{lag}}) = -\frac{\tau_{\text{lag}}}{\ln \lambda_i(\tau_{\text{lag}})}, \tag{2.37}$$

corresponding to the slowest dynamical processes between the metastable states of the system. Furthermore, they represent interesting kinetic properties that can be directly compared to experiments.[93] This might aid atomistic understanding of biomolecular processes in cases where experimental structural information is unavailable.

The Markov property is validated by the *Chapman-Kolmogorov* equation[86]

$$\boldsymbol{T}(k\tau_{\text{lag}}) = \left[\boldsymbol{T}(\tau_{\text{lag}})\right]^k, \qquad k \in \mathbb{N}^+. \tag{2.38}$$

This equation compares the dynamics in the raw MD data (on the left-hand side) with the model predictions, which were estimated for τ_{lag} and then propagated $k-1$ times. From the Chapman-Kolmogorov equation also follows the eigenvalue relationship $\lambda_i(k\tau_{\text{lag}}) = \left[\lambda_i(\tau_{\text{lag}})\right]^k$, which implies constant implied timescales t_i for Markovian systems[206]

$$t_i = -\frac{\tau_{\text{lag}}}{\ln \lambda_i(\tau_{\text{lag}})} = -\frac{k\tau_{\text{lag}}}{\ln \lambda_i(k\tau_{\text{lag}})}. \tag{2.39}$$

This relation can also be consulted to choose an appropriate lag-time for a Markovian model.

2.5 Protein Systems

Finally, two proteins will be briefly described that appear repeatedly in this thesis, namely, villin headpiece and T4 lysozyme.

2.5.1 Villin Headpiece (HP35)

First is the villin headpiece subdomain HP35, which is a very widely used model system in the realm of MD simulations, primarily due to its rapid folding kinetics (i.e. good statistics) and structural simplicity.[207,208] Composed of 35 amino acids, it forms a compact structure with a hydrophobic core and three α-helices connected by two short loops, as shown in Fig. 2.11a.

While the isolated subdomain HP35 is extensively studied in MD simulations[79,95,124,208–211]—in part due to its free availability through D.E. Shaw Research[212,213]—it is part of the full villin headpiece HP67 in reality, which itself represents an integral component of the globular Villin-1 protein found in the red junglefowl (Gallus gallus).

The ultrafast folding kinetics of HP35 ($\sim 0.7\,\mu$s) were achieved via double mutation of the wild-type HP35,[212] namely Lys24→Nle and Lys29→Nle, which reduce electrostatic repulsion and stabilize the native fold.[214] Compared to the wild-type HP35, these two modifications represent a six-fold acceleration of the wild-type folding time of $\sim 4.3\,\mu$s.

Here, we study a $300\,\mu$s long trajectory by Piana *et al.*,[213] who employed the AMBER ff99SB*-ILDN force field[63,215] and simulated the crystal structure (PDB entry 2f4k) at a temperature $T = 360\,\mathrm{K}$ close to the melting temperature on the special-purpose Anton supercomputer[216] using the TIP3P water model.[217]

2.5.2 T4 Lysozyme

T4 Lysozyme (T4L), a 164-residue enzyme from bacteriophage T4, is responsible for the degradation of Escherichia coli bacteria cell walls by catalyzing the cleavage of glycosidic bonds, ultimately enabling viral replication and host cell lysis.[98,99,218–220] T4L undergoes a hinge-like open↔closed conformational transition, which resembles the motion of a Pac-Man as indicated in Fig. 2.11b.

The conformational change is triggered through a long-range allosteric mechanism originating in the hinge region (H), where the key residue Phe4 (highlighted in red) acts as a locking mechanism. Then, this motion propagates into the mouth region (M), rendering T4L a prime example to study long-range allosteric couplings in proteins.[98] The large-scale motions in the mouth region and the subtle but essential motions in the hydrophobic hinge region pose a challenge to standard dimensionality reduction approaches.

Here we adopt the $50\,\mu$s-long all-atom MD simulation carried out by Ernst et al.,[98] which was simulated using GROMACS 4.6.7[221] at $T = 300\,\mathrm{K}$ in combination with the AMBER ff99SB*-ILDN force field[63,215] and the TIP3P water model.[217]

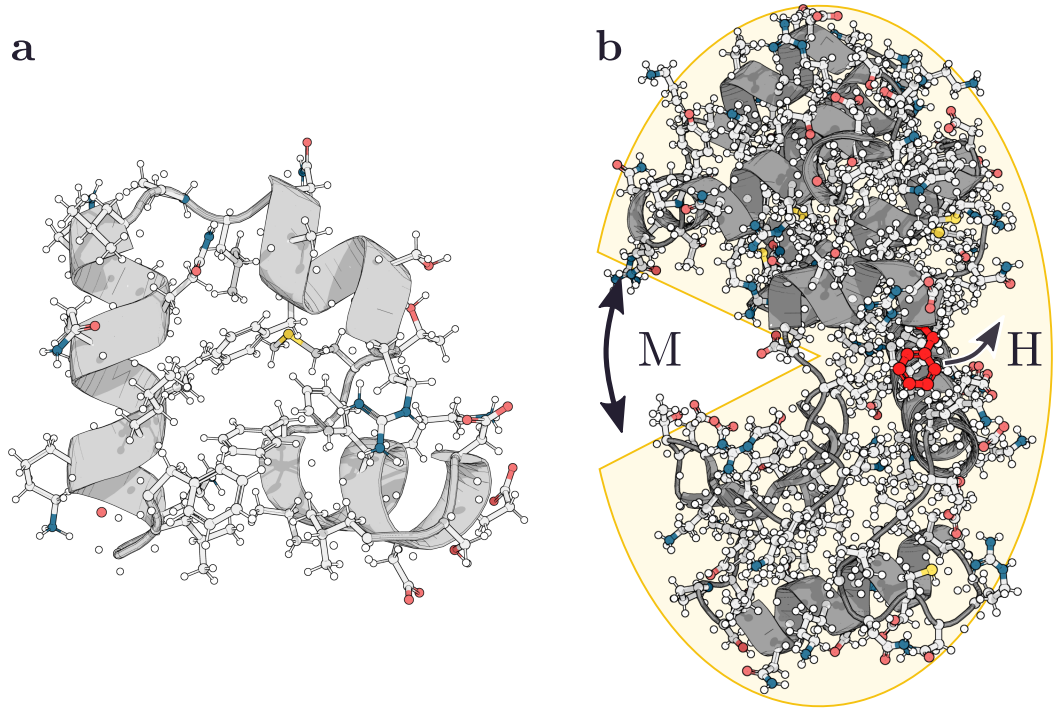


Figure 2.11 | (a) Structure of the villin headpiece subdomain HP35. The cartoon representation of the protein is shown in gray. The carbon atoms are shown in light gray, oxygen in red, nitrogen in blue, sulfur in yellow and hydrogen in white.
(b) Molecular structure of T4L. The open-closed motion of the mouth region (M) is triggered through the key residue Phe4 (marked in red) in the hinge region (H). The atom color is the same as for HP35 apart from the carbon atoms, which are represented by a darker gray for better visualization.

From Noise to Signal: Identification of Functional Dynamics in Proteins

3

Parts of this chapter are based on our publication:

Correlation-Based Feature Selection to Identify Functional Dynamics in Proteins
G. Diez, D. Nagel, and G. Stock, *J. Chem. Theory Comput.* **2022** 18 (8), 5079–5088,
DOI: 10.1021/acs.jctc.2c00337.

As described in Sec. 2.3.1, the Cartesian coordinates of the MD simulation are generally not well suited for dimensionality reduction and model building due to their mixing of biomolecular internal motions with irrelevant global motions of the protein within the simulation box. To address this, internal coordinates such as dihedral angles and interatomic distances are preferred since they are not plagued by the above-mentioned issues. Backbone dihedral angles (ϕ, ψ) are effective in capturing the conformation of secondary structures such as α-helices or β-sheets but only indirectly account for the tertiary structure.[148,149] In contrast, interresidue distances provide a more direct description of the tertiary structure,[95,222,223] which makes them a suitable choice for studying protein dynamics. Therefore, we will concentrate on these in the following.

However, interresidue distances have the drawback of scaling quadratically with the size of the protein and (thus) carry a significant amount of redundant information. This issue can be mitigated by focusing on contact distances,[95,152] which drastically reduces the number of coordinates on the one hand and still allows understanding long-range effects as a consequence of contact patterns on the other hand. Moreover, a major strength of contact distances is that they directly explain conformational changes through physicochemical processes such as contact formation and breaking. Irrespective of how the input coordinates are chosen, we anticipate that only a subset of these coordinates will play a role in the specific biomolecular process of interest.

Therefore, we seek a grouping of all input coordinates—also referred to as "features"—according to their mutual relation in order to identify and divide them into clusters of coordinates describing the same collective motion.[i] These cluster can serve as a direct way of interpreting the biomolecular process in terms of contact distances[158,224] or can be used for further feature extraction and model building.[2,208] Especially when doing the latter, it becomes essential to exclude certain coordinates that can mislead the analysis: specifically, high variance coordinates that lack functional relevance in PCA and coordinates that change slowly over time but are functionally irrelevant in TICA.[ii] Since both methods maximize for variance or timescales, respectively, including such coordinates

[i] An interesting and notable work in this context is the AMINO algorithm developed by Tiwary and coworkers,[167] which uses a mutual information-based distance metric to group related coordinates into order parameters that can subsequently be used for the construction of reaction coordinates.

[ii] Examples include protein termini that show large, but functionally irrelevant fluctuations, and (ϕ, ψ) dihedral angles that capture rare conformational transitions (such as right- to left-handed helix formation in HP35[211,213]) which occur slowly but are biologically insignificant.[156]

would critically impact and bias the resulting model, effectively obstructing the extraction of meaningful information. To summarize, clustering all features according to their mutual relation allows us the:

- Identification of collective motions in the protein via groups of highly correlated features.
- Exclusion of certain coordinates that would make further modeling infeasible due to their bias on PCA/TICA.
- Exclusion of uncorrelated (or independent) features that describe random motion or features that remain roughly constant, such as stable contacts

3.1 Similarity Measures

Before grouping all input features, a similarity measure quantifying their mutual relation must be established. In this chapter, we will focus on the one-dimensional case before extending to the multidimensional case in Chapter 4.

In the following, we will consider two one-dimensional random variables, $X \sim p_X$ and $Y \sim p_Y$, where p_X is the marginal probability distribution over $\mathcal{X} = \{x \mid p_X(x) > 0\}$[iii] (analogous for Y), and their corresponding realizations $\{x_i\}$ and $\{y_i\}$. The joint probability distribution of X and Y is described by $p_{x,y}$.

[iii] In the following we will write p_X instead of $p_X(x)$ for the sake of brevity.

3.1.1 Pearson Correlation Coefficient

The Pearson correlation coefficient ρ measures the *linear correlation* between X and Y[225, iv]

$$\rho(X,Y) = \frac{\langle (X - \langle X \rangle)(Y - \langle Y \rangle) \rangle}{\sigma_X \sigma_Y}, \tag{3.1}$$

where $\langle \dots \rangle$ denotes the expectation and σ the standard deviation. The linear correlation $\rho_{X,Y}$ can take values from -1 to 1, specifically:

- $\rho(X,Y) = 1$ if and only if $Y = aX + b$, where $a > 0$
- $\rho(X,Y) = -1$ if and only if $Y = aX + b$, where $a < 0$

This implies $\rho(X,X) = 1$ by definition. When describing the similarity between two features, distinguishing between correlation and anti-correlation (i.e., whether a is larger or smaller than zero) is not meaningful. To illustrate, consider the motion of a seesaw: as one end rises, the distance from that end to the ground increases, while the distance on the other side simultaneously decreases. Despite their anti-correlation, they describe the same underlying process. Thus, it is the strength of their relationship—rather than its sign—that matters. Therefore, the absolute value $|\rho|$ is used as a similarity measure.

While statistical independence $p_{x,y} = p_x p_y$ implies $\rho(X,Y) = 0$, the converse is not true: zero linear correlation does not generally imply independence. This is because the Pearson correlation coefficient only captures linear relationships through first and second moments, means and

[iv] While Auguste Bravais first derived the mathematical formula for correlation[226] and Francis Galton later conceptualized the application to his heredity studies,[227] Karl Pearson formalized and popularized the idea of correlation based on Galtons ideas. Therefore, the Pearson correlation coefficient serves as an example of *Stigler's Law*, which states that "no scientific discovery is named after its original discoverer."[228] To prove his point, Stigler humorously attributed his Law to sociologist Robert K. Merton, who originally described the phenomenon of cumulative advantage (not only) in academia as *Matthew effect*.

"For to every one who has will more be given, and he will have abundance; but from him who has not, even what he has will be taken away."

-Matthew 25:29, RSV

covariance. Strictly speaking, this is only adequate for Gaussian distributions, while nonlinear dependencies are not necessarily detected (see Fig. 3.1 for some examples). Physically speaking, this limitation is analogous to a quadratic energy landscape with an associated linear force.[53]

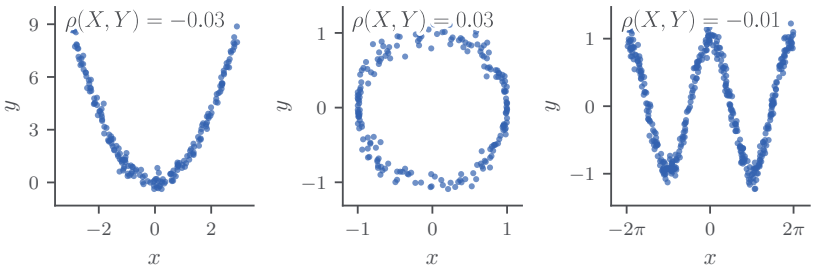


Figure 3.1 | Three sets of (X, Y) data points with added noise and $\rho(X, Y) \approx 0$ despite functional dependencies: *left:* $Y = X^2$, *center:* $X = \cos\theta$ & $Y = \sin\theta$, *right:* $Y = \cos X$.

3.1.2 Mutual Information

Due to the above-mentioned restrictions of linear correlation, we consider a more general measure of dependence between X and Y, namely mutual information (MI).[229] Unlike Pearson correlation, MI avoids any assumptions about the underlying data and directly measures the statistical independence of X and Y by quantifying the dissimilarity of their joint probability distribution $p_{x,y}$ and the product of the marginals $p_x p_y$ through the KL divergence

$$\begin{aligned} I(X,Y) &= D_{\text{KL}}\left[p_{x,y} \,\|\, p_x p_y\right] \\ &= \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{x,y} \ln \frac{p_{x,y}}{p_x p_y}, \end{aligned} \tag{3.2}$$

which vanishes for independent variables $p_{x,y} = p_x p_y$. Introducing the Shannon entropy $H(X)$ and the joint entropy $H(X,Y)$, which are given by

$$\begin{aligned} H(X) &= -\sum_{x\in\mathcal{X}} p_x \ln p_x, \\ H(X,Y) &= -\sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{x,y} \ln p_{x,y}, \end{aligned}$$

the MI can be understood more intuitively:

$$\begin{aligned} I(X,Y) &= H(X) + H(Y) - H(X,Y) \\ &= H(X) - H(X \mid Y). \end{aligned} \tag{3.3}$$

In the last step, the conditional entropy $H(X \mid Y) = H(X,Y) - H(Y)$ was introduced. Interpreting $H(X)$ as the uncertainty about X and $H(X \mid Y)$ as the remaining uncertainty about X after knowing Y, the MI can be understood as the reduction in uncertainty about X due to the knowledge of Y.

From a theoretical point of view, MI is a more versatile measure of mutual relation since it captures both linear and nonlinear relationships while making fewer assumptions about the underlying data. Despite these advantages, there are two practical limitations to MI, which we will discuss in the following.

Normalizing the one-Dimensional Mutual Information

First, the MI is not bound to $[-1, 1]$ but ranges from $[0, \infty)$, which complicates analysis since it is not obvious which range of MI should be considered indicative of high or low similarity. For example, when we observe an MI of $I(X, Y) = 0.3$, it is not immediately evident whether this represents a strong or weak relationship between the two variables X and Y. As described above, the MI measures the decrease in the uncertainty of variable X when taking into account variable Y. Following Eq. (3.3), it is clear, that the answer also depends on the uncertainty of X and Y, that is $H(X)$ and $H(Y)$.[v] Therefore, the normalization of MI by entropy measures is necessary to enable meaningful comparisons of mutual relations across different proteins.[167,230,231] Fortunately, MI satisfies an upper bound that directly incorporates these entropy values:[232]

$$I(X, Y) \leq \sqrt{H(X)H(Y)} \leq H(X, Y), \tag{3.4}$$

where $\sqrt{H(X)H(Y)}$ denotes the geometric mean of the marginal entropies and $H(X, Y)$ the joint entropy. Based on this inequality, we define two normalized MI measures

$$I_{\text{geom.}}(X, Y) = \frac{I(X, Y)}{\sqrt{H(X)H(Y)}}, \tag{3.5}$$

$$I_{\text{joint}}(X, Y) = \frac{I(X, Y)}{H(X, Y)}. \tag{3.6}$$

Alternatively, we can revisit the definition of the MI in Eq. (3.2) and substitute the unbound KL divergence with a bounded divergence measure between $p_{x,y}$ and $p_x p_y$. This leads us to the Jensen-Shannon divergence:

$$\begin{aligned} I_{\text{JS}}(X, Y) &= D_{\text{JS}}\left[p_{x,y} \middle\| p_x p_y\right] \\ &= \frac{1}{2} D_{\text{KL}}\left[p_{x,y} \middle\| M\right] + \frac{1}{2} D_{\text{KL}}\left[p_x p_y \middle\| M\right], \end{aligned}$$

where M is the mixture distribution $M = \frac{1}{2}[p_{x,y} + p_x p_y]$. Finally, we mention the formulation of Gel'fand and Yaglom,[233] who found that ρ captures all relationships in the special case of Gaussian distributions in one dimension or collinear Gaussian distributions of unit variance in three dimensions. In these cases, the linear correlation coefficient completely characterizes the relationship between the variables since Gaussian distributions are fully determined by their first two moments. Since MI measures all statistical dependence and the linear correlation coefficient captures the complete dependencies for Gaussian distributions, we can express the MI as a direct function of ρ in this special case:

$$I(X, Y) = -\frac{d}{2} \log\left[1 - \rho(X, Y)^2\right], \tag{3.7}$$

where d denotes the dimensionality. Based on this equation, we solve for ρ and interpret it as a normalized quantity I_{GY} of MI[234]

$$I_{\text{GY}} = \sqrt{1 - \exp\left(-\frac{2I(X, Y)}{d}\right)}. \tag{3.8}$$

[v] Consider the two scenarios with identical MI, but completely different relationships: In the first scenario, we face a low-entropy variable X with $H(X) = 0.5$ and conditional entropy of $H(X|Y) = 0.2$, which results in a MI of $I(X, Y) = 0.3$. The conditional entropy $H(X|Y)$ is significantly lower than the marginal entropy $H(X)$, which indicates a strong relationship because little uncertainty about X remain after observing Y.
In the second scenario, we are dealing with a high-entropy variable $H(X) = 5$, where observing Y hardly reduces the uncertainty about X, indicated by $H(X|Y) = 4.7$. Despite also yielding a MI of $I(X, Y) = 0.3$, the relationship is weak because most uncertainty still remains.

For completeness, we also mention an alternative normalization scheme that achieves the normalization of the MI through the discretization of the data on a grid.[235] However, this might affect the statistical robustness of the MI and can be significantly slower to compute compared to measures based on local k-nn statistics (see Eq. 3.9 or section 4.3).[236]

Computation of one-Dimensional Mutual Information

The second limitation of MI is that its computation is much more involved compared to linear correlation: rather than simply computing a dot product, MI relies on the estimation of (at least) two-dimensional probability densities. The estimation of higher dimensional probability densities is notoriously difficult—especially in the three-dimensional case, when the joint probability distribution $p_{x,y}$ becomes six-dimensional, which we will address in Chapter 4.

For the one-dimensional case, a simple histogram ansatz is the most straightforward method to estimate the probability densities (see Fig. 3.3a), which, however, converges only slowly with the number of samples and is not very robust since the number of bins heavily affects the resulting estimate. This issue can be partly circumvented by using an optimal bin width d_{bin} as suggested by Freedman and Diaconis[237] with an adjusted prefactor of 2.59

$$d_{\text{bin}} = 2.59 \frac{\text{IQR}(X)}{\sqrt[3]{N}},$$

where N denotes the number of samples and IQR is the interquartile range of the data X. Freedman and Diaconis assumed Gaussian distributed data in their derivation, which is why this rule typically struggles with distributions featuring fat tails. When working with contact distances, we are often faced with distributions featuring one peak for the bound state and a single-sided fat tail for the unbound state (see Fig. 3.2 for an example).

As a simple remedy, the bin width might be rescaled by the following factor

$$d_{\text{bin}} \to d_{\text{bin}} \cdot \frac{100^{\text{th}}\,\text{percentile} - 0^{\text{th}}\,\text{percentile}}{85^{\text{th}}\,\text{percentile} - 15^{\text{th}}\,\text{percentile}},$$

that effectively widens the bins to better capture the tail behavior.

Another option is a kernel density estimation (KDE),[238] which estimates the probability function density function by superposing smooth kernels (see Fig. 3.3b) —typically Gaussian—centered at each data point x_i (with $i = 1, \dots, N$)

$$\hat{p}(x) = \frac{1}{\sqrt{2\pi} N \sigma} \sum_{i=1}^{N} \exp\left(-\frac{1}{2}\left(\frac{x - x_i}{\sigma}\right)^2\right).$$

It converges faster than the histogram ansatz but comes with the price of increased computational cost as well as the need to choose an appropriate bandwidth parameter σ.[239,240] In practice, relying on a single bandwidth parameter σ renders KDE ill-suited for estimating MI since this fixed smoothing parameter often results in over-smoothed probability

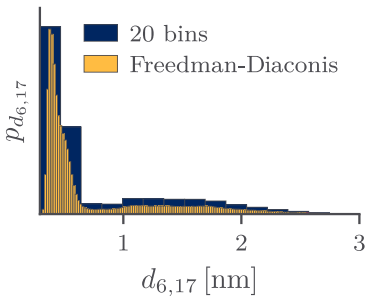


Figure 3.2 | Estimation of the probability distribution of an exemplary distance $d_{6,17}$ of HP35 via the histogram approach. In blue, we choose a naive ansatz with 20 bins, while the red bars indicate the number of bins estimated via the Freedman-Diaconis rule.[237] The height of the bars were equally scaled for better visualization.

density estimates when applied to distributions with strongly varying densities, for example, multimodal distributions.

To overcome this, Kraskov and coworkers proposed an alternative estimator for MI based on local k-nearest neighbor (k-nn) statistics (see Fig. 3.3c):[241, vi]

$$I(X,Y) = \psi(N) + \psi(k) - \frac{1}{N}\sum_{i=1}^{N}\left[\psi(n_{x,i}+1) + \psi(n_{y,i}+1)\right], \qquad (3.9)$$

where ψ is the digamma function satisfying the recursion $\psi(n+1) = \psi(n) + 1/n$, with the Euler-Mascheroni constant as starting value $\psi(1) = -\gamma \approx 0.577$. Here, N denotes the number of data points and $n_{x,i}$ (respectively $n_{y,i}$) is the number of data points j for which $\|x_i - x_j\| < \epsilon_i^k$ (respectively in y). For every data point i, the cutoff value ϵ_i^k is selected as the maximum of the k-nn distances in the x and y directions, i.e. $\epsilon_i^k = \max(\epsilon_{x,i}^k, \epsilon_{y,i}^k)$. This adaptive choice of ϵ_i^k is a major advantage of this estimator since it eliminates the need for a fixed bandwidth—as required in KDE—hence preventing over-smoothing.

[vi] We will revisit this estimator in more detail in Sec. 4.3 since it plays a central role for the multidimensional case.

In order to compare the robustness of the histogram, KDE, and k-nn estimators, we tested these three approaches against two benchmark distributions of which we can analytically calculate the true (non-normalized)[vii] MI value I_{true}. The first involved two uniformly independent random variables in the interval $[0,1)$, featuring no mutual relation, and thus, $I_{\text{true}}(X,Y) = 0$ (see Fig. 3.4). For the second distribution, we considered samples drawn from a bivariate normal distribution with a correlation of $\rho(X,Y) = 0.8$. This special case allows calculating the true MI value as $I_{\text{true}}(X,Y) \approx 0.5$ following Eq. (3.7).

[vii] Here, we consider the non-normalized case, since the normalization of the k-nn estimator is not straightforward and will be addressed in Chapter. 4.

The results clearly indicate that the k-nn and KDE estimates converge much faster toward the true MI value compared to the simple histogram, which uses a bin width determined by the Freedman-Diaconis rule. While relying on larger sample sizes N to achieve comparable accuracy, the histogram approach still provides a reasonable approximation of the MI, especially since we are (only) interested in relative differences among different pairs of features.

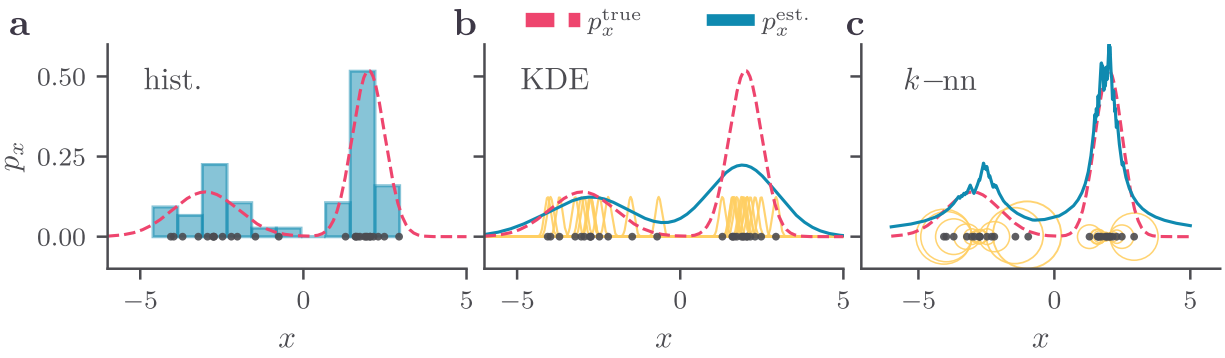


Figure 3.3 | Illustration of the three different probability density function estimators. Data points (gray circles) were drawn according to underlying true probability density function (red dashed line) and the probability density function was estimated based on these points (blue line).

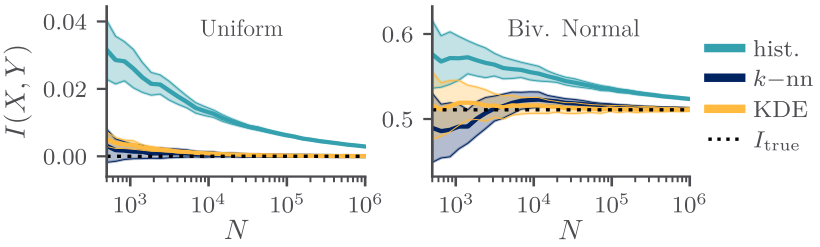


Figure 3.4 | Computing the MI via a simple histogram-based, k-nn and KDE approach for an independent uniform distribution on the left and a bivariate normal distribution on the right. The mean (thick solid line) and standard deviation (shaded area) were estimated from 100 independent runs.

Calculating the full similarity matrix involves quadratic complexity $\mathcal{O}(M^2)$ (or more precisely $\frac{1}{2}M(M-1)$ steps) with respect to the number of features M. Therefore, it is crucial to compare the runtimes for all three methods (histogram, k-nn, and KDE) in order to assess whether they are suitable to compute the MI for a protein characterized by hundreds or even thousands of features. To this end, we computed the runtime of each estimator to compute both MI and $|\rho|$ for two normally distributed variables, as shown in Fig. 3.5 The KDE estimator is by far the most computationally demanding one, requiring approximately a factor of 100 more CPU time than the histogram approach and 10 more times than the k-nn estimator. Generally, calculating the MI is at least two to three orders of magnitude slower than computing the Pearson correlation. Balancing accuracy and computational efficiency, we proceed with the histogram approach for further analysis, as it provides a reasonable approximation for our purpose while maintaining favorable scaling properties. This, however, is only practical in the one-dimensional case, which we will discuss in detail in Chapter 4.

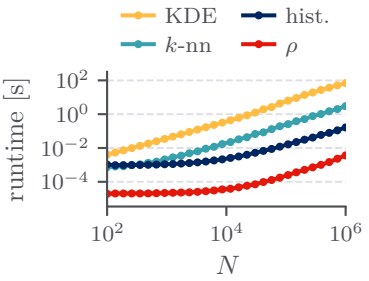


Figure 3.5 | Runtime comparison for the three MI estimators as well as linear correlation for two normally distributed variables. Mean and standard deviation were estimated from 100 independent runs. The computation was performed on a single core of an Intel® Core™ i9-14900K.

3.1.3 Evaluating the Trade-off: MI vs. ρ

Having established the histogram approach for the computation of one-dimensional MI, it remains to evaluate whether the additional computational expenses justify its use over the simple linear correlation ρ. To this end, we calculated the native contacts of T4L and HP35 (see Sec. 2.5) as determined in the references 80, 98, resulting in 402 and 53 contacts, respectively. The results for both systems are quite similar (compare SI, Fig. A.2), which is why we combine these data to facilitate further discussion.

In order to assess whether the MI provides additional insights beyond the absolute Pearson correlation coefficient, we analyze the joint probability between the MI—normalized by its tightest bound $I_{\text{geom.}}$—and the absolute Pearson correlation coefficient $|\rho|$, denoted as $p(I_{\text{geom.}}, |\rho|)$, as shown in Fig. 3.6(a). The joint probability reveals a clear relation between $I_{\text{geom.}}$ and $|\rho|$, and comparable patterns are observed for other normalization schemes of MI, as shown in SI, Fig. A.2. These findings suggest that the nonlinear measures do not contain significant additional information compared to the linear correlation in form of $|\rho|$.

Furthermore, it is noteworthy that the values of $|\rho|$ cover almost the full range from 0 to 1, while values of $I_{\text{geom.}}$ hardly exceed values of 0.2. To further illustrate this effect, Fig. 3.6(b) shows the cumulative probability distribution for all similarity measures, demonstrating that over 90% of the values of $I_{\text{geom.}}$ fall below 0.01. Similar patterns can be observed

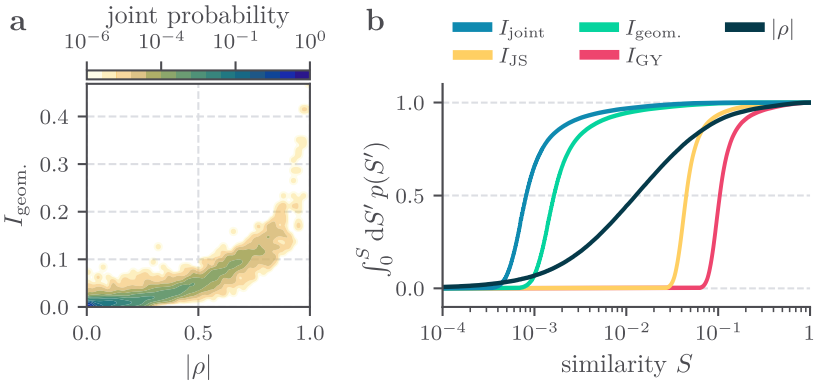


Figure 3.6 | Comparison of linear and nonlinear similarity measures which were computed for the native contacts of T4L and HP35. (a) MI normalized according to Eq. (3.5) compared to the absolute Pearson coefficient $|\rho|$. (b) Cumulative probability distribution of various similarity measures. Adapted with minor changes from Ref. 1. Copyright © 2022 The Authors.

for I_{joint}, while the other nonlinear measures, I_{JS} and I_{GY} predominantly adopt higher values, rarely dropping below $I_{\text{JS}} = 0.03$ or $I_{\text{GY}} = 0.07$. Remarkably, even though all nonlinear measures are derived from the same underlying MI, their normalized distributions show minimal overlap. In contrast, the linear correlation $|\rho|$ uniformly accounts for small and high correlations, spanning the full range. Combined with the fact that the nonlinear measures fail to provide substantial additional insights and their considerably greater computational demands, these results confirm the straightforward and well-established linear Pearson coefficient as an effective choice for measuring the similarity.

As a cautionary note, we emphasize that the Pearson correlation coefficient has serious flaws (compare e.g. Fig. 3.1),[242] especially when it comes to high-dimensional data.[3,234] However, for collinear data such as distances and even periodic variables,[95] the Pearson correlation coefficient reliably captures the overall correlation well—given that an appropriate transformation to linear variables is applied.[150,151]

The close agreement observed between $|\rho|$ and MI is particularly remarkable given that $|\rho|$ considers only the first two moments of the underlying distribution. While only Gaussian distributions are fully characterized by these moments, the probability distributions we encountered in MD data are often far from normal, typically exhibiting a pronounced peak at small distances (corresponding to the bound state) and a one-sided fat tail at large distances reflecting unbound conformations.

3.2 Communities of Collective Motion

Having established a robust similarity measure for quantifying the pairwise similarity between our input features, we now seek to reveal groups of coordinates[viii] involved in some specific process of cooperative motion. Representing the features as data points in a similarity space where the features are arranged according to their similarity $|\rho|$ (or distance $1 - |\rho|$), this step translates into a straightforward clustering task and allows the reorganization of the correlation matrix into a block-diagonal structure. This is a critical step, and we demand that our clusters describing collective motion fulfill the following two criteria:[243]

1. **Homogeneity:** Each cluster should exclusively contain features that correspond to a specific collective motion.

[viii] In the following, we will refer to these groups as "clusters".

2. **Completeness:** All coordinates that describe one specific collective motion must be assigned to the same cluster.

In the following, we want to identify an optimal clustering approach that satisfies our criteria. However, clustering a correlation matrix involves two particularities: First, the M features—represented as M-dimensional data points according to their distance $1 - |\rho|$ to all other features— do not naturally lie on a coordinate system, which is why we cannot define a Euclidean distance between any two points in this space. This already excludes k-means as a possible candidate for the clustering scheme since it relies on geometric centroids computed from coordinate averages. Secondly, we are now facing a very sparse data space, where the number of dimensions equals the number of data points, which renders employing density-based clustering methods infeasible (at least without prior feature extraction). In order to avoid introducing unnecessary complexity, we exclude density-based methods as well.

This narrows down the methods mentioned in Sec. 2.3.4 to k-medoids, which still works even in non-metric spaces by designating actual data points as cluster centers and complete linkage clustering, an agglomerative hierarchical clustering method based on maximal pairwise distances. While k-medoids and complete-linkage clustering address the challenges arising from the non-Euclidean and high-dimensional sparse space, they lack guarantees on cluster connectivity—a critical requirement for the completeness of our clustering.[ix]

[ix] Looking at SI, Fig. A.3, we see that neither of these methods assign the coordinates corresponding to the open↔closed motion to one single cluster.

3.2.1 Leiden Community Detection

The Leiden community detection algorithm[244] circumvents the abovementioned problems by operating on a graph rather than in a vector space. We construct this graph by treating every feature as an individual node, and the edges between the nodes reflect on their pairwise similarity (i.e., correlation $|\rho|$). Based on such a graph, the Leiden algorithm then identifies communities of highly correlated features through iterative optimization of an objective function, performing the following three steps until convergence:[x]

1. **Local moving of nodes:** nodes are assigned to the communities that yield the maximal gain of the objective function.
2. **Refinement of the partition:** Communities from step 1 may be split into multiple sub-communities when this improves partition quality. This step also ensures that all communities are internally well-connected.
3. **Aggregation of the network:** The (sub-)communities of step 2 now become super-nodes, creating a coarse-grained network that accelerates subsequent iterations.

[x] This is only a high-level description of the algorithm. For a detailed description, please refer the supporting information in Ref. 244.

The Leiden algorithm is closely related to the very popular Louvain algorithm[245] but addresses its critical flaw of internally disconnected clusters by introducing the refinement phase. Additionally, the Leiden algorithm incorporates some randomness during the refinement phase, which avoids getting stuck in local minima and allows for a broader exploration of the partition space.

Objective Functions

The most widely used objective function Φ is modularity, which quantifies how far the structure of a network (or graph) deviates from a randomly wired graph/network.[246] That is, high modularity values indicate the presence of well-defined communities featuring densely interconnected nodes with sparse inter-community connections. Conversely, low modularity values suggest a lack of pronounced community structure, implying that the network features only little organization or may even be regarded as a randomly wired network. Formally, modularity is defined as

$$\Phi_{\text{mod}} = \frac{1}{2m} \sum_c \left(e_c - \frac{k_c^2}{2m} \right),$$

where the sum is taken over all clusters c, m is the total number of edges in the graph and e_c denotes the sum of edge weights within cluster c. Additionally, k_c denotes the sum of the degrees of all nodes in c. The first term e_c favors the aggregation of nodes into large clusters, while the second term, $k_c^2/2m$, imposes a constraint on the cluster size by penalizing excessively large clusters. The penalty term represents the expected number of edges within cluster c if the entire network were randomly rewired, but each node retained its degree (null model). Therefore, the modularity is only high if an actual cluster is more densely connected than its counterpart in a random network with the same node strengths.

The assumption of randomly rewiring the entire network implies that any node can theoretically be connected to any other node in the network. However, in large networks, this is very unlikely since nodes typically interact only within their local neighborhoods, which—depending on the size of the network—only constitute a vanishingly small fraction of the entire network. Additionally, in large graphs, the expected number of edges between two small distinct clusters is of the order of $\mathcal{O}(M^{-1})$, where M is the number of nodes (or features in our case). Thus, even a single accidental edge between two small and distinct clusters may be interpreted as statistically significant by modularity, and the two clusters would be merged. This issue implies a "resolution limit"[247] for modularity and prevents the identification of small clusters in a graph.

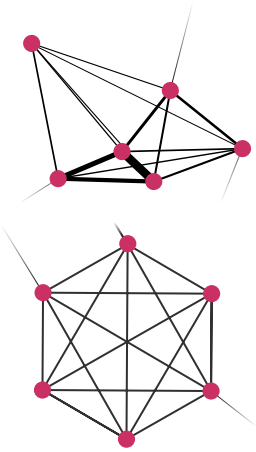


Figure 3.7 | Illustration of the CPM: The total edge weights in an actual cluster (top) are compared to a null model with the same number of nodes in which every edge is γ-correlated.

As a remedy, Traag et al.[248] proposed another objective function, referred to as the "constant Potts model" (CPM). Like modularity, the CPM can be derived from the Potts model,[249] which is itself a generalization of the Ising model.[250] Formally, the CPM objective function is defined as

$$\Phi_{\text{CPM}} = \sum_c \left[e_c - \gamma \binom{n_c}{2} \right], \tag{3.10}$$

where n_c denotes the number of nodes in cluster c and the binomial $\binom{n_c}{2} = (n_c^2 - n_c)/2$ denotes the number of possible edges within c. By weighting the number of possible edges with the resolution parameter γ, the CPM model compares the total observed correlation within each cluster to an expected correlation in a null model cluster characterized by a constant correlation of γ (see Fig. 3.7).

A cluster c that contributes a higher total correlation than expected results in a positive contribution to the objective function, whereas clusters with a smaller total observed correlation would penalize the partition. This way, $\gamma \in [0,1]$ might be interpreted as the minimal average correlation necessary for cluster formation. However, unlike complete linkage clustering, CPM also allows individual correlations to fall below γ, provided that the cluster as a whole still increases Φ_{CPM}. Ultimately, γ determines the clustering resolution: high γ values yield many small, homogeneous clusters, while a lower γ results in fewer but also larger and more heterogeneous clusters.

3.2.2 Optimal Clustering Strategy

Inspired by the patterns in the correlation matrices we observed for HP35 and T4L (for descriptions of the systems, see Sec. 2.5), we developed a benchmark artificial correlation matrix to systematically compare and evaluate the performance of the different clustering methods, namely:

- Leiden clustering using the CPM objective function
- Leiden clustering using the modularity objective function
- complete linkage clustering
- k-medoids

This toy matrix is designed to capture essential characteristics of protein systems, such as three large clusters corresponding to collective motion alongside multiple mini-clusters containing only one feature to represent uncorrelated, noisy coordinates. Furthermore, we included small residual correlations between the clusters to simulate a more realistic scenario. The resulting matrix is shown in Fig. 3.8a and poses a challenging test case for comparing the clustering approaches.

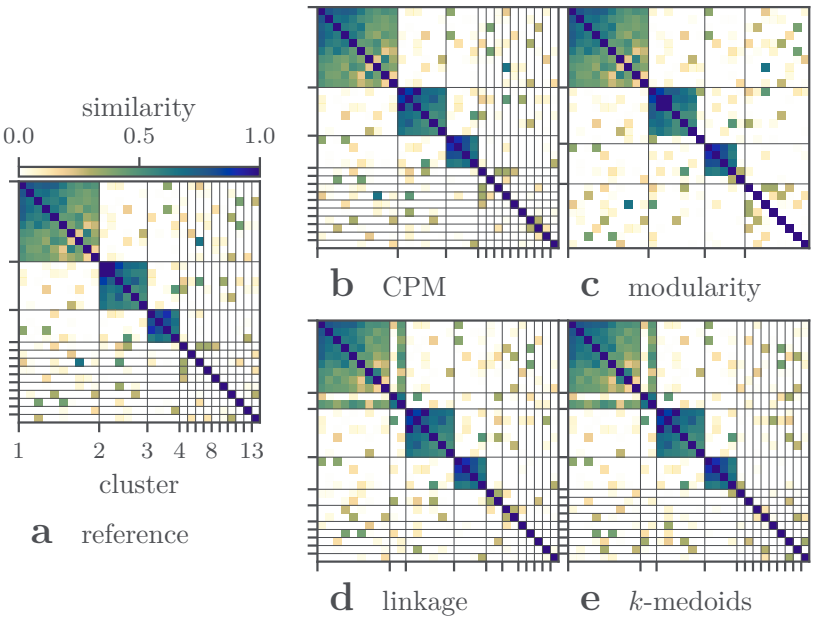


Figure 3.8 | Simple toy model representing a similarity matrix. The matrix consists of three clusters and 10 noise coordinates. Clustering via k-medoids, complete linkage and the Leiden algorithm using modularity and CPM as objective function were performed and compared to the reference. Figure adapted with minor changes from Ref. 1. Copyright © (2022) The Authors.

Since we know the "ground truth" of cluster assignment by construction in this case, we can use it to calculate the V-measure.[243] Comparing the clustering results of each method to the known "true" reference cluster

assignments, the V-measure can be used to estimate the optimal parameters for each method, hence guaranteeing a fair and unbiased comparison. We determined the optimal cluster parameters for each method in SI, Fig. A.1, which were then used to obtain the clustering results in Fig. 3.8b-e.

Generally speaking, all methods broadly capture the main clusters of collective motion and noisy coordinates but differ in detail. In light of the two criteria, *homogeneity*, and *completeness* defined in Sec. 3.2, we notice that the latter is only satisfied by Leiden/CPM. The remaining methods fail to completely resolve cluster 1, which features a strong correlation to a single noise coordinate. Such sporadic correlations with noisy coordinates can lead to spurious formation of new clusters—evident for cluster 2 in complete linkage clustering and k-medoids—which is internally highly correlated but has a disproportionately high cross-correlation to cluster 1. The cause lies in the greedy decision-making of k-medoids and complete linkage clustering, which rely on making locally optimal decisions rather than optimizing a global quantity. Similarly, the Leiden algorithm with modularity as an objective function is plagued by comparable problems since it is based on a k-nearest neighborhood graph, which inevitably limits the decision-making to the local neighborhood. Furthermore, as discussed above, the assumption of randomly rewiring the entire graph introduces a resolution limit for small clusters, which prevents the identification of the noise coordinates in the form of clusters containing only one coordinate.

[xi] hereafter referred to as "Leiden clustering".

Conversely, the Leiden algorithm using the CPM objective function[xi] allows (and enforces) locally suboptimal decisions through its stochastic refinement, which facilitates finding the global maximum of the objective function. This approach ultimately yields the best results, satisfying both completeness and homogeneity (as reflected by achieving a perfect V-measure score; see SI, Fig. A.1). While the differences appear to be subtle for the studied toy model, they become critical in real-world protein systems as e.g. T4L, where other methods systematically fail to meet the completeness criterion, even for the most dominant dynamical processes (see SI, Fig. A.3).

Beyond its robustness, Leiden clustering is convenient to use as it only relies on a single intuitive parameter γ. Besides fine-tuning it by visual inspection of the resulting clustered correlation matrix, γ can be optimized by cross-validation methods such as the Generalized Matrix Rayleigh Quotient approach,[251] which has been successfully applied for constructing MSMs.

3.3 Software

All correlation measures and clustering methods introduced in this chapter have been implemented in the Python package MoSAIC ("Molecular Systems Automated Identification of Cooperativity").[1] The package adapts scikit-learn[252] syntax and is freely available on the Moldyn website[253] or GitHub.[254]

3.4 Applications

In the following, we demonstrate the versatility of MoSAIC through four distinct applications spanning different systems and purposes.

3.4.1 T4 Lysozyme

As a first application, we study the cooperative open↔closed motion of T4L (see Sec. 2.5) to demonstrate the capability of Leiden clustering to distinguish functional coordinates from noise. As previously discussed,[98,99,101] the open↔closed transition of the two domains is allosterically triggered by local motions in its hinge region (compare Fig. 2.11 b), which renders the identification of reaction coordinates underlying this process a challenge to standard dimensionality approaches.

Following Ernst *et al.*,[98] we computed 402 native interresidue contacts. As described in Sec. 2.3.1, a distance forms a contact when the d_{ij} between the closest non-hydrogen atoms falls below 0.45 nm. Native, in this case, means that only two structures—and not the complete trajectory—were used to compute these distances, namely the energy-minimized crystal structure for the open state and the MD structure with the lowest radius of gyration for the closed state.[98]

After computing the associated (time-resolved) contact distances $d_{ij}(t)$, we calculated the linear correlation matrix $|\rho|$ and employed Leiden clustering with a resolution parameter of $\gamma = 0.5$. Clusters that contained $n = 5$ or fewer coordinates were assigned as noise,[xii] which resulted in the block-diagonalized correlation matrix depicted in Fig. 3.9a.

[xii] The choice of n does not impact the clustering, but eases the discussion of the resulting matrix.

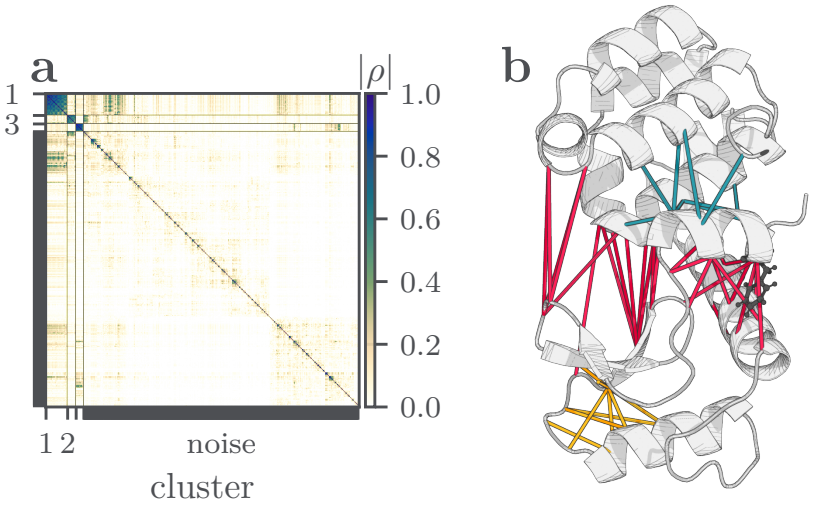


Figure 3.9 | Leiden clustering for T4L using the linear correlation of native contacts and the CPM with $\gamma = 0.5$. (a) shows the block-diagonalized correlation matrix and (b) the structure of T4L in the open state, where the distances of the corresponding clusters are shown: cluster 1 in red, 2 in cyan and 3 in yellow. Phe4 is shown in dark grey. Adapted with minor changes from Ref. 1. Copyright © (2022) The Authors.

Remarkably, only three main clusters represent correlated motion, while the great majority of coordinates (around 90%) are hardly correlated and thus distributed over the remaining ∼ 300 clusters. While these three main clusters feature an average internal correlation of $\langle|\rho|\rangle = 0.69$, the mean residual correlation between them is only about 0.08, and the residual correlation between any two clusters, including noise, is only 0.04 on average. These low values can be explained by the fact, that on the one

hand, most intra-protein contacts in T4L are quite stable and only fluctuate around their mean distance, while on the other hand, contacts on the protein surface frequently form and break and hence fluctuate randomly. Since neither of both are involved in functional dynamics, all these coordinates represent noise that should be excluded from further analysis or model building.

The first three clusters are visualized via their corresponding distances plotted in the T4L structure (Fig. 3.9b). In particular, the 27 highly correlated contact distances contained in cluster 1 describe the open↔closed conformational change of T4L. These distances are shown in red and span the space from the hinge region—located around the key residue Phe4—to the mouth region, therefore reflecting the allosteric coupling between these two distant regions.[2]

Showing very little or no correlation with cluster 1 at all, clusters 2 and 3 represent other correlated motions in T4L. Specifically, Cluster 2 captures a rocking motion driven by the rearrangement of the α_1-helix and the N-terminal domain. Cluster 3 describes a twist-like motion of the two β-sheets and the nearby α_2-helix; this latter motion was previously reported by Hub & de Groot and later by Ernst *et al.*.[98,99]

3.4.2 Villin Headpiece

As an example of folding, where we expect an entirely different picture as for the bistable T4L, we consider HP35. Following Ref. 80, we investigate the 53 native contacts of the crystal structure (PDB 2f4k)[214] and compute their mutual relation via the absolute Pearson correlation coefficient. Following up with Leiden clustering using a resolution parameter of $\gamma = 0.65$, we obtain the block-diagonalized correlation matrix shown in Fig. 3.10a.

The matrix reveals seven highly correlated main clusters and cluster 8, which is almost entirely uncorrelated to the rest of the system. Discarding clusters with $n \leq 2$, 15 coordinates were assigned to noise.[xiii] To facilitate discussion, the individual contact distances associated with the main clusters are displayed in Fig. 3.10b,c. To begin with, cluster 8 reflects on the motions of the N-terminus relative to the α_1-helix, which

[xiii] Other clustering methods produce similar results, although the partition of main and weakly correlated clusters differ, compare SI, Fig. A.4a.

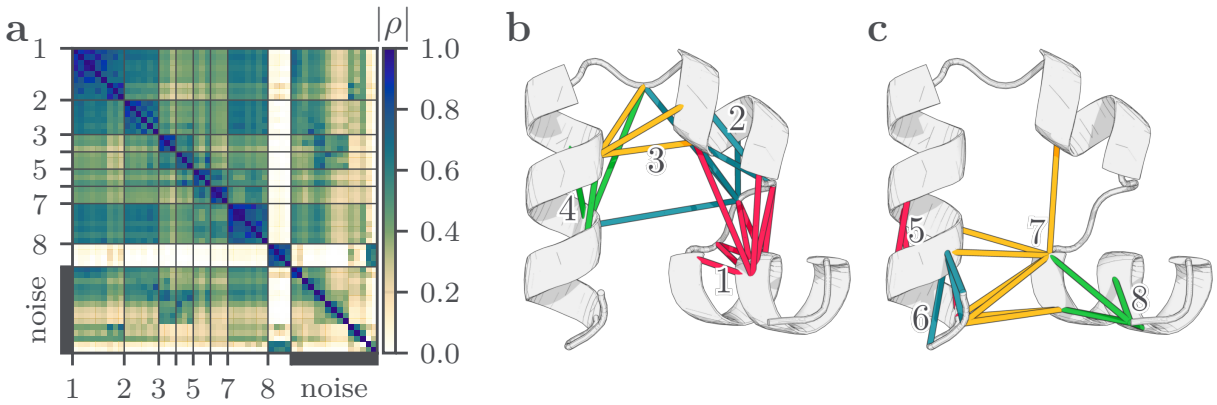


Figure 3.10 | Leiden clustering of HP35 with $\gamma = 0.65$ for 53 native contact distances of the crystal structure. (a) The block-diagonalized matrix reveals seven main clusters, a completely uncorrelated cluster 8 and various noise coordinates. (b and c) Structure of (folded) HP35, where the contact distances included in clusters 1-8 are shown. Adapted from Ref. 1 with minor changes. Copyright © (2022) The Authors.

are almost entirely uncorrelated to the rest of the system. As mentioned earlier, it is crucial to exclude such uncorrelated terminal motion from the analysis since these dangling motions can exhibit large amplitude fluctuations that may dominate the first principal components in a PCA, although not relevant for the folding process. The same applies if the uncorrelated motion displays a two-state behavior, such as transitions between two different orientations of the terminus. If included, these features would consequently double the number of conformational states trivially, making the analysis unnecessarily cumbersome. As with T4L, we, therefore, find that an essential first step in a successful analysis or model building is the identification and rejection of uncorrelated motion or weakly correlated noise coordinates.

Besides serving as a tool for dimensionality reduction, we can also employ Leiden clustering as an aid in interpreting the biomolecular processes. For this purpose, we consider the clusters 1 to 7. By design, they do not only show a high intra-cluster correlation of $\langle|\rho|\rangle = 0.82$ but also exhibit high inter-cluster residual correlations of $\langle|\rho|\rangle = 0.50$ between the main clusters only and 0.4 for all clusters. The high residual correlation of $\langle|\rho|\rangle = 0.50$ between the main seven clusters indicates that all of them describe different aspects of the same process—folding. This presents a different pattern compared to T4L, where the main clusters are largely uncorrelated and describe different processes.

Apart from the small clusters 5 and (in part) 6 that account for motions within the α_3-helix, all remaining main clusters contain tertiary contacts connecting two secondary structures. For example, the α_1- and α_2-helix are connected through cluster 1; cluster 3 reflects on the relative orientation between the α_2 and α_3 helices, while cluster 7 connects the helices α_3 and α_1, thus ultimately reporting on the compactness of HP35 during the folding process.

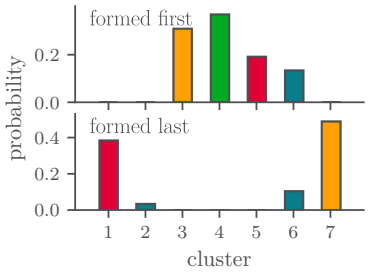


Figure 3.11 | Probability for each cluster to form first or last during folding. Adapted with minor changes from Ref. 1. Copyright © (2022) The Authors.

Because of the strong intra-cluster correlations, these contacts will preferably form and break in a concerted manner, which allows us to assign a state of "1" to a cluster when (most of) its contacts are formed and a "0" otherwise. Representing the complete protein as a product state,[xiv] we can characterize the structures of the folding trajectory using this coarse-grained state description.[152] Unlike a state definition via helicity where, for example, (ffu) indicates that the first two helices are folded and helix 3 is unfolded,[156] the product state description here focuses on tertiary contacts and might, therefore, be better suited to describe cooperative processes and the compactness of the whole protein.

As a straightforward application of this approach, we investigated the temporal order of cluster formation during the 31 successful folding events of HP35, as depicted in Fig. 3.11. Analysis of the MD trajectory reveals that clusters 3 and 4—responsible for stabilizing the connection of helices α_2 and α_3—typically fold first in the folding process. Notably, clusters 1 and 7, which bridge helices α_1 and α_2, and α_2 and α_3, respectively, consistently emerge as the final folding step and thus likely define the transition state of the folding process. This small example illustrates how the organization of contact distances in key clusters can provide valuable first insights into the mechanisms of the considered biomolecular process.

[xiv] For example the notation (1110000) indicates formed contacts in the first three clusters and unformed contacts in cluster 4-7.

HP35 Applications in Independent Work

A notable example of Leiden clustering is the work of D. Nagel *et al.* who used it to analyze their resulting (macro-)state trajectory of an MSM of HP35.[208] They systematically employed Leiden clustering to characterize the structural organization of each state.

In Fig. 3.12, we consider four representative macrostates from the trajectory made available by D. Nagel and coworkers[xv] and follow the visualization approach they introduced in Ref. 208. The macrostates are sorted by decreasing fraction of native contacts, such that macrostate S1 is the most compact (folded) state, S5 and S8 can be considered intermediate states along the folding pathway, while S12 is fully unfolded.

[xv] this reference trajectory is freely accessible at `https://github.com/moldyn/HP35`.

For each macrostate, the distribution of contact distances d_{ij} within the Leiden clusters from Fig. 3.10 provides a concise structural fingerprint. This facilitates a straightforward interpretation of the biomolecular process, for example modeled by an MSM[208] in terms of the selected input coordinates.

In these plots, the interquartile range (iqr)—the range between the 25th and 75th percentiles—captures the central data spread for each contact distance. The whiskers extend from the lower bound $Q_1 - 1.5{\cdot}\text{iqr}$ to the upper bound $Q_3 + 1.5{\cdot}\text{iqr}$,[xvi] indicating the typical range of the contact distances excluding outliers. Furthermore, the median and the corresponding distances d_{ij} of the crystal structure are shown for reference. Returning to the above idea of product space description, we can readily assign (1111111) to S1, (1111000) to S5, (0111100) to S8 and S12 as (0000000), reflecting whether the contact distances in each Leiden cluster are (on average) formed within that specific state.

[xvi] Q_1 is the first quartile, Q_3 the third.

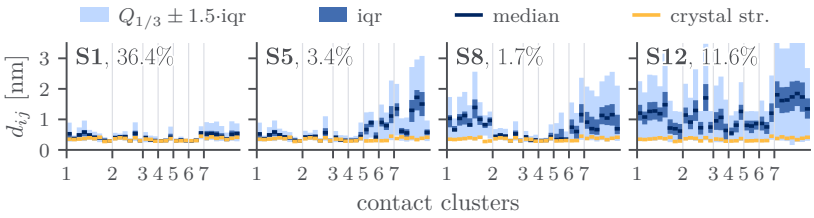


Figure 3.12 | Structural analysis of four representative macrostates (S1, S5, S8 and S12) of the state trajectory provided in Ref. 208. The macrostates are characterized by the distribution of contact distances within the Leiden clusters defined in Fig. 3.10. The percentage for each state indicates its relative population in the trajectory.

3.4.3 C_{10}-Trimer

As a last application example for identifying collective motion, we consider the C_{10}-trimer (1-decene trimer, $C_{30}H_{62}$), which is a synthetic hydrocarbon widely studied for its extreme-condition lubricant properties.[255–257] Formed by linking 1-decene molecules into a branched alkane structure, the C_{10}-trimer is a key component of the PAO4 base oil—a low-viscosity polyalphaolefin widely used in gear, compressor and engine oils, hydraulic fluids, greases, and more[258] due to its performance even under extreme pressures of the order of GPa.[259]

Studying the atomic motion of the C_{10}-trimer may allow the identification of structure-property relationships that link macroscopic quantities, such as viscosity, to atomistic variables. To this end, we used a 1 μs long trajectory simulated by Matthias Post,[260] who simulated a bulk of 260

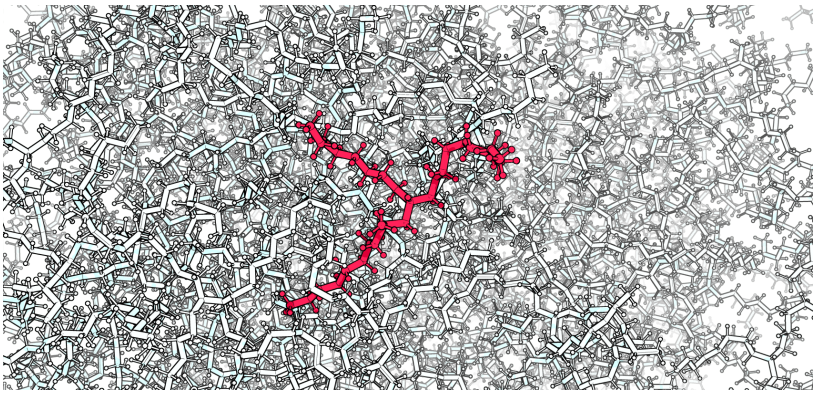


Figure 3.13 | Bulk of 260 C_{10}-trimers. For better visualization, one trimer is marked in red.

trimers within a cuboid of size $5 \times 5.5 \times 8\,\mathrm{nm}^3$ with periodic boundary conditions using GROMACS at a temperature of $T = 500\,\mathrm{K}$.[261, xvii]

As a first step, we calculated all distances between C-atoms within one trimer and performed Leiden clustering with CPM and a resolution parameter of $\gamma = 0.5$. The resulting block-diagonalized correlation matrix is shown in Fig. 3.14 and reveals distinct clusters governing its conformational dynamics (Fig. 3.15). The first three clusters correspond to large-scale relative movements between the three branched chains, while clusters 4-8 describe local intra-chain flexibility such as bending and stretching. The uniformly distributed uncorrelated thermal vibrations (i.e., noise coordinates) along the chains show no spatial coherence. This means that motions localized within single chains (apart from clusters 4-8 bending/stretching, in which the complete chain is included), do not collectively influence the trimer's global dynamics.

In Ref. 255, Falk *et al.* integrated microscopic free-volume dynamics into the Stokes-Einstein framework to predict viscosity under extreme pressure. Their approach describes diffusion motion as a consequence of cage-and-jumps events, where a molecule (described through its center of mass (COM) coordinate) is temporally spatially confined by neighboring molecules and jumps as soon as sufficient volume becomes available in its neighborhood.

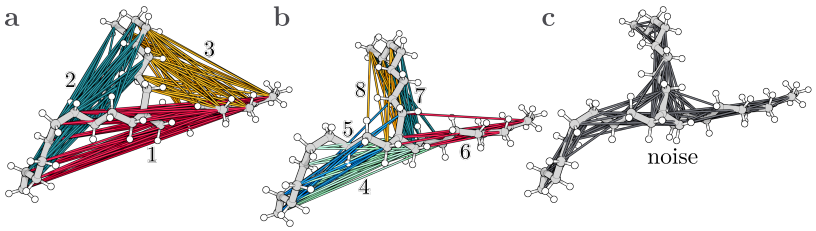


Figure 3.15 | Structural representation of the distances contained in the corresponding Leiden clusters in Fig. 3.14.

To this end, we computed two displacement quantities to see whether internal rearrangements of the trimer are correlated with its jumps in the bulk: the mean squared displacement (MSD) of the COM and the simple Euclidean distance between its initial and current position (see SI, Fig. A.5). The linear correlation between the first principal component of each Leiden cluster[xviii] and these quantities is shown in Tab. 3.1. The throughout low correlation between COM displacement and internal motion suggests that the free-volume availability alone dictates the diffusion. We hypothesize that under high pressure, the trimer cannot undergo coordinated internal rearrangements before translating into a

[xvii] For further details on the simulation consult Ref. 261, Sec. 5.5.

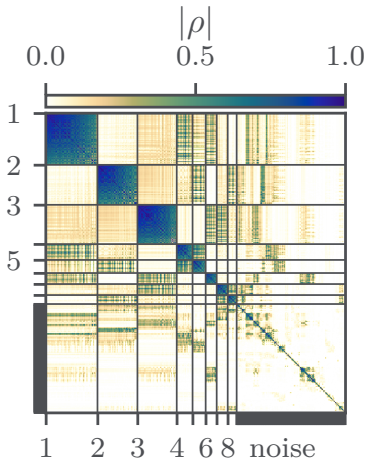


Figure 3.14 | Block-diagonalized correlation matrix of all distances between C-atoms within the C_{10}-trimer. Leiden/CPM clustering with a resolution parameter of $\gamma = 0.5$ was applied.

[xviii] The first principal component for each Leiden cluster has an explained variance ratio roughly around 0.8 thus representing a good one-dimensional approximation.

neighboring free volume. Instead, these jumps are stochastically triggered by external forces resulting from the pressurized environment, which forces the trimer to move into the free volume without prior alignment.

This is in line with the findings of Falk *et al.*,[255] who predict the viscosity solely by relying on COM-derived parameters implicitly assuming that internal dynamics are thermally equilibrated and independent of the diffusion process.

Table 3.1 | Correlation between the first principal component of each Leiden cluster and the mean squared displacement (MSD) and Euclidean distance of the center of mass (COM) of the C_{10}-trimer. The correlation was computed over a 1 μs long trajectory.

cluster	1	2	3	4	5	6	7	8
$\lvert\rho(x_1, \text{MSD})\rvert$	0.06	0.13	0.11	0.05	0.00	0.01	0.04	0.05
$\lvert\rho(x_1, \text{Euc. dist.})\rvert$	0.01	0.03	0.07	0.00	0.00	0.00	0.09	0.06

3.4.4 From Features to Trajectories: Path Separation

In the last sections, we saw that the Leiden clustering greatly facilitates the analysis of biomolecular processes by identifying key collective motions and noise coordinates. However, due to its general framework, MoSAIC can not only be used for feature selection but naturally extends to the identification of different pathways by clustering trajectories.

Targeted MD and the Need for Path Separation

This plays a crucial role in the analysis of biomolecular processes, where a protein-ligand system is studied as, e.g., ligand (un)binding. Here, binding and unbinding events can occur via different pathways in and out of the binding site.[262] In our group, ligand unbinding is typically studied using dissipation-corrected targeted Molecular dynamics,[263] which allows the estimation of free energies and, thus, friction factors Γ. Ligand unbinding is enforced via targeted MD,[65] which applies a constant velocity v constraint along a predefined reaction coordinate x

$$\Phi(t) = x(t) - (x_0 + vt) \stackrel{!}{=} 0.$$

This constraint is realized by the force $f = \lambda \frac{\mathrm{d}\Phi}{\mathrm{d}x}$, where λ represents the Lagrange multiplier. The nonequilibrium work performed during such a pulling process is given by:

$$W(x) = \int_{x_0}^{x} \mathrm{d}x' f(x')$$

Based on this work distribution, the free energy profile $\Delta G(x)$ can be estimated using a cumulant expansion of the Jarzynski equality[264]

$$\Delta G(x) = \langle W(x) \rangle_N - \underbrace{\frac{1}{2k_\mathrm{B}T} \langle \delta W(x)^2 \rangle_N}_{\text{dissipative work}} + \mathcal{O}(\delta W^3). \tag{3.11}$$

Here, $\langle \cdots \rangle_N$ denotes the ensemble average over N independent trajectories initialized from a common equilibrium Boltzmann distribution. The truncation after the second order assumes that the work distribution

is Gaussian—which is only reasonable when trajectories follow similar pathways. Assuming that the approximation holds, the friction can be calculated as

$$\Gamma(x) = \frac{1}{v}\frac{\mathrm{d}}{\mathrm{d}x}W_{diss}(x).$$

Application: A_{2A}

As a single example, we consider ligand unbinding in the A_{2A} adenosine receptor.[265, xix] To capture the dynamics of the unbinding process, contact distances between the ligand and the protein were computed, resulting in 104 input features.
In the next step, we performed a two-step PCA to first establish a common, global coordinate system for all 681 trajectories and then extract each trajectory's own directions of (orthogonal) maximum variance within this global coordinate system. To this end, we combined the mean-free contact distances of all 681 trajectories into a single data matrix $\boldsymbol{X} \in \mathbb{R}^{N\times 104,\,\text{xx}}$. Performing a PCA on this data matrix, we retained the top $d = 3$ eigenvectors $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3\}$ (see Sec. 2.3.3) of the covariance matrix $\boldsymbol{\Sigma} = \frac{1}{N}\boldsymbol{X}^\top\boldsymbol{X}$. In a subsequent step, we extracted the directions of maximum variance for each trajectory $\boldsymbol{X}_i$ individually by first projecting it onto the global subspace

$$\boldsymbol{Z}_i = \boldsymbol{X}_i \cdot \boldsymbol{W}$$

and computed the trajectory-specific covariance matrix in this reduced 3-dimensional space,

$$\boldsymbol{\Sigma_i} = \frac{1}{(N_i - 1)}\boldsymbol{Z}_i^\top \boldsymbol{Z}_i$$

and then finally diagonalizing it

$$\boldsymbol{\Sigma_i} = \boldsymbol{W}_i \cdot \boldsymbol{\Lambda}_i \cdot \boldsymbol{W}_i^\top.$$

The columns of $\boldsymbol{W}_i = \{\boldsymbol{w}_i^{(1)}, \boldsymbol{w}_i^{(2)}, \boldsymbol{w}_i^{(3)}\}$ are three-dimensional vectors containing the orthogonal directions of maximum variance for each trajectory in the same global PCA subspace. This shared reference frame allows us to compute the similarity between two trajectories by computing the overlap of their eigenvectors

$$S_{ij} = \left|\hat{\boldsymbol{w}}_i^{(2)} \cdot \hat{\boldsymbol{w}}_j^{(2)}\right| \cdot \left|\hat{\boldsymbol{w}}_i^{(3)} \cdot \hat{\boldsymbol{w}}_j^{(3)}\right|,$$

where $\hat{w}$ indicates the normalized eigenvectors. In our particular case, we have refrained from including the scalar product between the first eigenvectors since they are predominantly defined by the pulling direction rather than the motion orthogonal to it.

The resulting similarity matrix for A_{2A} is clustered with Leiden/CPM with a resolution parameter of $\gamma = 0.9$ and shown in Fig. 3.16. We focus on clusters 5 and 6 due to their pronounced dissimilarity and show the volume occupied by the ligand across all trajectories within these clusters in Fig. 3.17. The structural representation of the accessed volume shows that the ligand in these clusters exits the binding pocket through

xix For more details on the system and the simulation setup, we refer to Ref. 265.

xx N is the number of all frames in the 681 trajectories.

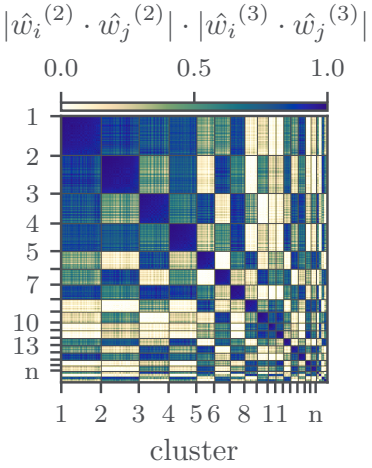


Figure 3.16 | Path separation based on Leiden clustering. First, we computed a PCA-based similarity measure between every pair of trajectories and then applied Leiden clustering with a resolution parameter of $\gamma = 0.9$ to obtain the path separation matrix.

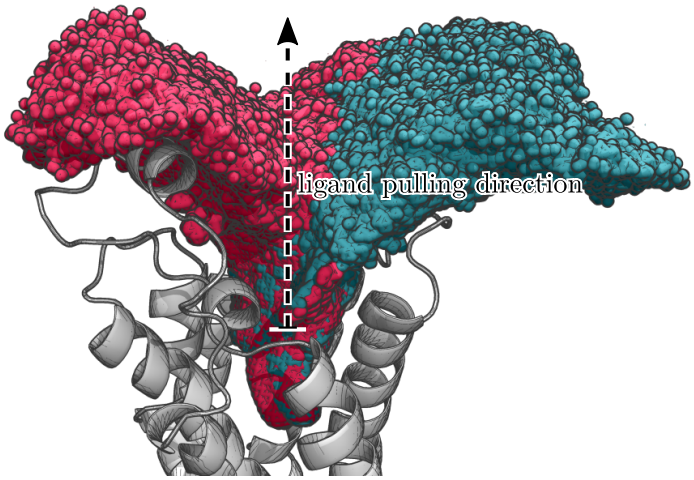


Figure 3.17 | Volume accessed by the ligand during the unbinding process in cluster 5 (red) and 6 (blue) of the path separation matrix shown in Fig. 3.16.

opposite directions, indicating that the supposed approach is suitable to discriminate between different unbinding pathways.

The procedure described above is only meant as proof of concept. While we have arbitrarily limited our analysis to a three-dimensional PCA subspace, we note that the choice could be made more systematically, for example, by selecting enough components to capture a certain amount of cumulative variance (e.g., 90%). Building on this approach, in a subsequent study, Tänzel and coworkers systematically investigated different input coordinates and similarity measures to identify pathways in pulling simulations in Ref. 265.

3.5 Concluding Remarks

We have introduced a correlation-based analysis framework for molecular dynamics simulation data that identifies collective motions underlying functional dynamics while systematically excluding uncorrelated or weakly correlated noise coordinates. Primarily designed for feature selection, this approach is particularly valuable for dimensionality reduction prior to applying feature extraction methods like principal component analysis, time-lagged independent component analysis, or neural-network based autoencoder architectures. High-variance or slow but nonfunctional dynamics can erroneously dominate subsequent analysis and model building and must therefore be discarded before. Our method works completely unsupervised, and, therefore, avoids possible bias due to presumed functional observables,[99,266] conformational states,[101] or variational principles that maximize timescales.[102]

By systematically comparing different similarity measures, we demonstrated that the linear Pearson correlation coefficient provides a robust and computationally efficient measure for collinear biomolecular dynamics data. Even though more sophisticated nonlinear measures such as normalized mutual information are theoretically able to capture more complex relationships, our analysis revealed that they do not offer additional practical insights despite their significantly higher computational cost.

In order to identify collective motions through block-diagonalization of a similarity matrix, we have found that the Leiden algorithm with the constant Potts model objective function consistently outperforms other clustering approaches in both synthetic and real-world examples. Based on the synthetic toy matrix, we demonstrated that the Leiden algorithm was the only method that satisfied both homogeneity and completeness. This is crucial for the faithful assignment of features to their corresponding collective motions, especially in the case of being confronted with challenging residual correlations. This is clearly shown in the SI, Fig. A.3, where all other methods fail to assign all coordinates corresponding to the open↔closed transition of T4L to one single cluster.

We demonstrated the effectiveness and versatility of our approach through diverse applications spanning conformational dynamics of T4L, protein folding in HP35, synthetic polymer dynamics (C_{10}-trimer), and ligand unbinding pathways in A_{2A}. Originally developed for feature selection, the method's ability to be easily extended to trajectory clustering for path separation highlights its adaptability.

All analysis steps are implemented in the open-source Python package MoSAIC, which adapts the Scikit-learn[252] syntax and provides a user-friendly interface for the analysis of functional biomolecular processes, facilitating both mechanistic interpretation (as e.g. in refs. 2, 224, 267) and the construction of accurate low-dimensional dynamical models (for example in refs. 95 and 208).

Normalized Mutual Information

4

Parts of this chapter are based on our publication:

Accurate Estimation of the Normalized Mutual Information of Multidimensional Data
D. Nagel, G. Diez and G. Stock, *J. Chem. Phys.* **2024** 161 (5), 054108, DOI: https://doi.org/10.1063/5.0217960 .

In the preceding chapter, we established the linear Pearson coefficient as a well-suited measure to quantify the interrelation between two one-dimensional stochastic random variables, X and Y. We now explore similarity measures for multidimensional data. This chapter focuses on theoretical aspects of multidimensional similarity measures, whereas their practical application is explored in the second part of chapter 5 through a case study of the protein T4 lysozyme.

In a range of disciplines, the quantification of correlations between different high-dimensional variables is of great interest. Examples include geostatistics, where the interrelationships of geographic variables across different spatial regions are studied.[268–270] In neuroscience, correlations between different brain regions are examined to understand functional pathways underlying cognitive processes.[271] In computer vision, multidimensional similarity measures facilitate image alignment, which is essential for tasks like medical image analysis.[272,273] Similarly, in finance, correlation analysis uncovers connections between different financial assets, market structures, and potential contagion effects during varying market conditions.[274,275]

In the field of chemical and biological physics, correlation measures are employed to investigate dynamic interdependencies among atoms or various parts of a molecular system. For example, correlation analysis serves as the foundation of PCA, which is used to reduce the dimensionality of the system,[105,172,276] while community detection approaches leverage correlations to identify interacting regions within biomolecular systems.[1,167,277] Furthermore, correlation patterns are used for the construction of allosteric networks that aim to model signal transduction pathways in proteins.[278–283]

Traditional linear correlation measures like the Pearson correlation coefficient impose restrictive Gaussian assumptions about the underlying data distribution, limiting their ability to identify relationships beyond linear ones. In this chapter, we therefore explore the limitations of linear correlation measures—that manifest in both one-dimensional and multidimensional spaces. Mutual information (MI) provides a versatile framework to quantify the similarity between variables by measuring statistical dependence without relying on specific distributional assumptions.

A significant drawback of MI, however, lies in the fact that it lacks normalization, making the interpretation of its values difficult and posing problems when comparing different systems. To address this limitation, we propose a novel and scalable normalization scheme for MI that works in any dimension—a challenge substantially more complex than in its one-dimensional counterpart.[3]

4.1 Limits of Multidimensional Linear Correlation

Arguably, the most straightforward measure of the correlation between two Cartesian coordinates is the linear Pearson coefficient extended to the multidimensional case via[234,278,284]

$$\rho(\boldsymbol{X},\boldsymbol{Y}) = \frac{\langle \boldsymbol{X}\cdot\boldsymbol{Y}\rangle}{\sqrt{\langle \boldsymbol{X}^2\rangle\langle \boldsymbol{Y}^2\rangle}}, \tag{4.1}$$

where $\boldsymbol{X} = \boldsymbol{r} - \langle \boldsymbol{r}\rangle$ denotes a displacement vector that measures the deviation of the atom position $\boldsymbol{r}_t = (x_t, y_t, z_t) \in \mathbb{R}^3$ from its time-averaged position.[i] This formulation decomposes into directional components through the dot product:

[i] t denotes the time index of the trajectory. Analogous for $\boldsymbol{Y}$.

$$\langle \boldsymbol{X}_i\cdot\boldsymbol{Y}_j\rangle = \sum_{\alpha=x,y,z}\langle \Delta\alpha_i\Delta\alpha_j\rangle$$

where $\Delta\alpha_i = \alpha_i(t) - \langle\alpha_i\rangle$ represents coordinate fluctuations of atoms i and j (typically C_α-atoms).

For the sake of simplicity, we will slightly abuse notation and express the absolute three-dimensional Pearson correlation coefficient as

$$\left|\rho_{ij}\right| = \frac{1}{3}\left|\sum_{\alpha=x,y,z}\rho_{ij}^{\alpha\alpha}\right|,$$

This means that due to the scalar product in Eq.(4.1), no $\rho_{ij}^{\alpha\beta}$-terms are considered, which is very problematic in the multidimensional case. To illustrate this, we consider a simple example of two particles that are entirely dependent on each other (see Fig. 4.1a), i.e., we can describe the motion of one particle as a function of the other particle:

$$\boldsymbol{r}_t^{(1)} = \begin{pmatrix}\cos(t)\\ \sin(t)\\ \sin(2t)\end{pmatrix} \quad\text{and}\quad \boldsymbol{r}_t^{(2)} = \begin{pmatrix}\cos(t)\\ -\sin(t)\\ \sin(2t)\end{pmatrix} = \begin{pmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 1\end{pmatrix}\boldsymbol{r}_t^{(1)}$$

This is even though a completet dependence between the two particles in all three dimensions exists (see Fig. 4.1b), the absolute Pearson correlation coefficient is only $\left|\rho_{ij}\right| = 1/3\cdot(1-1+1) = 1/3$. This is a severe limitation since the Pearson correlation coefficient fails to capture the perfect functional dependence between both particles due to sign cancellation. We want to emphasize that this is not related to any kind of nonlinearity but rather a consequence of the differentiation between correlation and anti-correlation, which has to be dropped for any meaningful measure of multidimensional correlation.[234] Even though this limitation

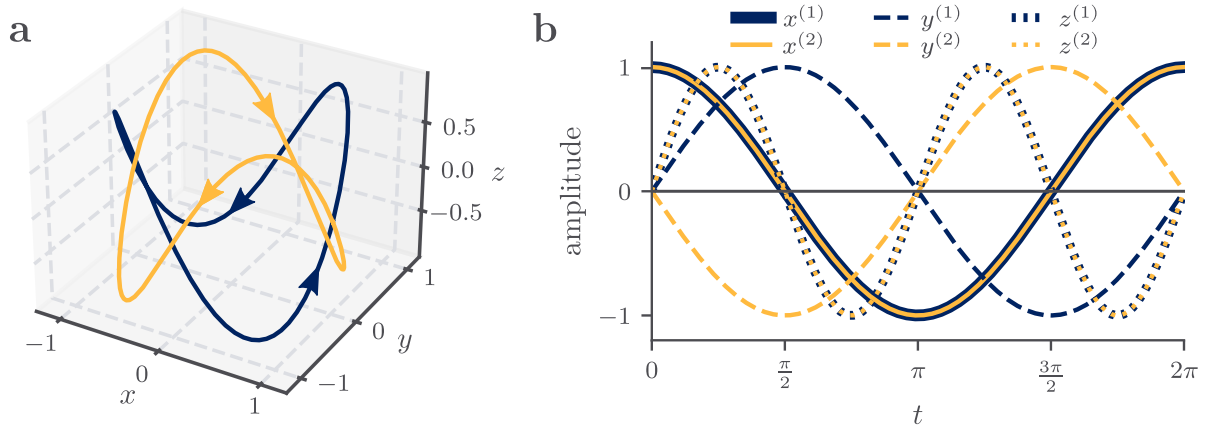


Figure 4.1 | A simple example of a three-dimensional system with two particles which are perfectly linearly (anti-)correlated in each direction, x, y and z. (a) The three-dimensional trajectory of the two particles and (b) the corresponding time series of the three coordinates.

was already pointed out by Lange and Grubmüller,[234] the Pearson correlation coefficient is still widely used for the construction of allosteric networks.[278–281]

As a simple remedy, we may sum the absolute values of the three individual components, that is, $|\rho_{ij}| = \frac{1}{3}\sum_{\alpha=x,y,z}|\rho_{ij}^{\alpha\alpha}|$, which fixes the problems of sign correlation and would indeed lead to a perfect correlation of $|\rho_{ij}| = 1$ for the example above. However, this approach is still limited since it does not take into account cross-correlations between the different directions and still depends on the specific coordinate system chosen.

4.1.1 Canonical Correlation Analysis

To account for this, canonical correlation analysis (CCA)[285]—a standard tool for the computation of linear correlations between multidimensional data—can be employed. It was already previously successfully applied to compute linear correlation between Cartesian coordinates.[286]

The core idea of CCA is that rather than directly computing correlations between individual directions, the original variables are transformed into new canonical variables that capture the strongest possible linear relationships between the two sets of variables. For Cartesian coordinates $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^3$, CCA finds linear transformations $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{3\times3}$, that project the input coordinates into canonical spaces $\hat{\boldsymbol{X}} = \boldsymbol{A}\boldsymbol{X}$ and $\hat{\boldsymbol{Y}} = \boldsymbol{B}\boldsymbol{Y}$. These transformations are chosen such that they maximize the Pearson correlation coefficient between the two sets of canonical variables

$$\rho_\mathrm{C}(\boldsymbol{X}, \boldsymbol{Y}) \equiv \rho(\hat{\boldsymbol{X}}, \hat{\boldsymbol{Y}}) \stackrel{!}{=} \max, \tag{4.2}$$

and can be obtained by solving a generalized eigenvalue problem derived from the covariance matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$. Eventually, the squared canonical correlation coefficient can be calculated as the average eigenvalue, i.e. $\rho_\mathrm{C}^2 = \frac{1}{3}\sum_i \lambda_i$.[ii] For the example above in Fig. 4.1, we find a perfect correlation of $\rho_\mathrm{C} = 1$ since the canonical variables overlap entirely (see SI, Fig. B.1).

[ii] A recipe how to compute the canonical correlation coefficient is given in the Supporting Information; see SI, chapter B.1.

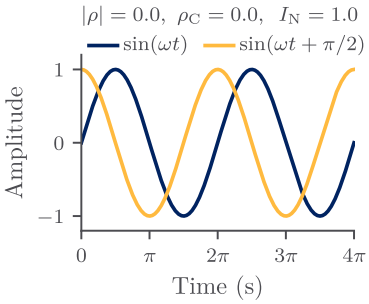


Figure 4.2 | Comparison of ρ, ρ_C and normalized MI I_N for two oscillating particles with a phase shift of $\pi/2$. While both ρ and ρ_C fail to detect any relationship, our estimator I_N—presented in Sec. 4.4—is able to capture the perfect relationship.

4.1.2 General Limitations of Linear Correlation

We have seen that the linear Pearson correlation coefficient is not suitable for capturing the correlation between two multidimensional variables. Nevertheless, it is still widely used for the construction of allosteric networks.[278–281] While the most critical issue—sign cancellation—can easily be fixed by summing absolute values of the directional contributions, this approach still fails to capture correlations between different directions. CCA offers a partial solution by transforming the original variables to canonical coordinates in order to optimize correlation, yet both approaches share a fundamental limitation: they are inherently restricted to linear relationships.

This linear assumption can be problematic since it implies that the underlying data distribution is Gaussian. However, this is rarely the case for most MD data (see Fig. 3.2), which makes the close relationship between linear correlation and MI in Fig. 3.6 even more surprising. From a more theoretical perspective, however, nonlinear relationships in physics are ubiquitous rather than exceptional. Consider the simple case of two particles oscillating with identical frequency in parallel direction. As long as they are in phase, they are perfectly correlated and exhibit a linear correlation of $\rho = 1$, but if we introduce a phase shift of $\pi/2$, the linear correlation vanishes entirely despite the fact that the two particles are still perfectly functionally dependent. The reason for this is that the configurational space is now a circle, which cannot be captured by the Gaussian approximation, regardless of a coordinate transformation.[234]

MI provides a natural solution to these problems as it makes no assumptions about the underlying form of relationships between the variables. As demonstrated in Fig. 4.2, our estimator for the normalized MI I_N—which we will introduce in Sec. 4.4—is able to perfectly capture the relationship between the two oscillating particles despite their phase shift of $\pi/2$; $I_N = 1$. That being said, we want to emphasize that MI does not suffer from directional cancellation, which is why this also works in the multidimensional case. These properties make MI a versatile and powerful measure for dependencies between variables, especially in the context of complex MD dynamics, where the joint probability distribution of any two variables can be highly nonlinear and non-Gaussian. In the following sections, we will present computational and theoretical difficulties that arise when estimating MI in the multidimensional case and finally propose the novel estimator I_N, which was also used in the example above.

4.2 Mutual Information Revisited

As already pointed out in the last chapter, the MI $I(X, Y)$ is a more versatile measure of correlation since it makes no assumption about the underlying distribution of the data. We refer to the section 3.1.2 for details to MI and only recall the definition here:

$$I(X, Y) = H(X) + H(Y) - H(X, Y), \tag{4.3}$$

where we remember that

$$H(X) = -\sum_{x\in\mathcal{X}} p_x \ln p_x \tag{4.4}$$

denotes the marginal entropy and

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{xy} \ln p_{xy} \tag{4.5}$$

represents the joint entropy. As the requirement of normalization was also already discussed in section 3.1.2, we refrain from repeating details and focus solely on problems that arise in the multidimensional case.

4.2.1 Normalizing Multidimensional Mutual Information

To establish an appropriate normalization factor, we recall the normalization factors already discussed for the one-dimensional case in Eq. (3.4):

$$\begin{aligned} I(X,Y) &\leq \min_{Z=X,Y} H(Z) \leq \sqrt{H(X)H(Y)} \\ &\leq \max_{Z=X,Y} H(Z) \leq H(X,Y) \end{aligned} \tag{4.6}$$

We choose the geometric mean $\sqrt{H(X)H(Y)}$ as our normalization due to its analogy to a normalized inner product and, thus, to the Pearson correlation coefficient. This leads to the upper bound of the Normalized Mutual Information (NMI):

$$I_{\mathrm{N}}(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \leq 1. \tag{4.7}$$

So far, nothing new compared to the one-dimensional case, and we could readily apply this normalization scheme to multidimensional MI computed via a histogram approach (compare section 3.1.2). However, in the three-dimensional Cartesian case, the joint probability distribution $P(X,Y)$ is six-dimensional, which renders such a histogram approach prohibitive. Instead, the NMI in the multidimensional case is typically computed using the algorithm developed by Kraskov, Stögbauer, and Grassberger, also known as KSG-estimator,[241] which computes the MI via Eq. (4.3) and estimates the entropies using k-nn statistics.[287–290]

Differential and Relative Entropy

Unfortunately, the above-mentioned inequalities used for the normalization of MI only hold for discrete entropies but do not extend to continuous entropies.[291] This creates a practical challenge since the KSG-estimator relies on continuous k-nn statistics, which might violate the inequalities in Eq. (4.6). Therefore, to properly apply these inequalities, we need to transform the discrete probability distributions (like p_x) into their continuous counterparts [like $P(x)$] when computing the entropies

in Eq. (4.4) and Eq. (4.5). To address this transformation, Shannon[292] suggested using "differential" entropy, defined as

$$H_d(X) = -\int_{\mathcal{X}} \mathrm{d}x\, P(x) \ln P(x), \tag{4.8}$$

for continuous distributions. However, this formulation introduces new complications. From the normalization condition

$$1 = \int_{\mathcal{X}} \mathrm{d}x\, P(x),$$

we note that the probability density $P(x)$ carries the dimension $1/[x]$, which creates issues when computing $\ln P(x)$. Furthermore, unlike discrete entropy, which is always positive, differential entropy can take negative values. This distinction becomes clear when we first consider the discrete case with normalization $1 \stackrel{!}{=} \sum_{x\in\mathcal{X}} p_x$, where every event occurs with a certain probability $p_x \in [0,1]$, which implies that $\ln p_x \leq 0$ and ensures that the entropy is always positive. On the other hand, in the continuous case, the probability density is normalized such that the area under its curve integrates to one, which means that $P(x)$ can take any value in the range $[0,\infty)$. As a consequence, $\ln P(x)$ can take any value in the range $(-\infty,\infty)$, which means that the differential entropy can be negative.

Investigating Shannon's differential entropy, Jaynes identified a fundamental flaw in its definition in Eq. (4.8),[291, iii] namely that it lacks invariance under variable transformations $x \mapsto \tilde{x}$. Jaynes corrected this and derived the correct continuous limit of the Shannon entropy, referred to as "relative" entropy

[iii] Jaynes noted: "Unfortunately, Shannon did not derive this formula, and rather just assumed it was the correct continuous analogue of discrete entropy, but it is not."[293]

$$H_r(X) = -\int_{\mathcal{X}} \mathrm{d}x\, P(x) \ln \frac{P(x)}{m(x)}, \tag{4.9}$$

where the invariant measure $m(x)$ is introduced that transforms identically as the probability density $P(x)$ under the variable transformation:

$$x \mapsto \tilde{x} : \frac{P(x)}{m(x)} = \frac{P(\tilde{x})}{m(\tilde{x})}.$$

This ensures that the relative entropy $H_r(X)$ does not depend on the choice of coordinates or units.

The invariant measure $m(x,y)$ of two coordinates can be chosen to factorize into single-coordinate functions, i.e.

$$m(x,y) = m(x)m(y). \tag{4.10}$$

This factorization guarantees that if either x or y changes, the measure $m(x,y)$ automatically updates to reflect the changes in those coordinates. Consequently, the MI preserves its invariance even after the introduction of the invariant measure

$$\begin{aligned} I_{\mathrm{d}}(X,Y) &= H_{\mathrm{d}}(X) + H_{\mathrm{d}}(Y) - H_{\mathrm{d}}(X,Y) \\ &= H_{\mathrm{r}}(X) + H_{\mathrm{r}}(Y) - H_{\mathrm{r}}(X,Y) \\ &\quad + \int \mathrm{d}(x,y) P(x,y) \underbrace{\left[\ln m(x) + \ln m(y) - \ln m(x)m(y)\right]}_{=0} \\ &= I_{\mathrm{r}}(X,Y). \end{aligned} \tag{4.11}$$

While it does not matter whether we use the differential entropy or the corrected relative entropy for the computation of MI, this does not hold when calculating the NMI because $H_\mathrm{r}(X) \neq H_\mathrm{d}(X)$. Therefore, it is crucial to compute the correct relative entropy for the normalization in Eq. (4.7) Otherwise, the normalization factor would, e.g., depend on whether we measure distances in nanometers or Angström.

4.3 A Nonparametric Estimator for Mutual Information

Attribution:

The estimator presented in the sections 4.3 and 4.4 was originally proposed in the PhD thesis of D. Nagel (Ref. 96).

Due to its scalability, we will use the KSG-estimator to compute the NMI. Since it is already capable of estimating the MI, we need to extend it to compute relative entropies as well.[241] Following Ref. 241, we will briefly revisit the most important steps in their derivation.

We start with a continuous random variable X with values in some metric space; that is, we can define a distance function $d(x, x')$ between any two realizations x and x' of the random variable. The differential entropy of X can be estimated using Eq. (4.8), and we can interpret it (up to a minus sign) as an average of $\ln P(x)$ over all realizations $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$ of the random variable X:

$$\hat{H}_d(X) = -\frac{1}{N}\sum_{i=1}^{N} \ln P(x_i) = -\langle P(x) \rangle$$

Thus, the challenge lies in finding an unbiased estimator for $\ln P(x)$. Instead of directly estimating the probability density $P(x)$, the KSG-estimator uses k-nn distances as a proxy: in a region with high density, the k-nn distances will be small, and in a region with low density, the k-nn distances will be large. To this end, we consider the probability distribution $P_k^i(\epsilon_i)$ for the distance ϵ_i between x_i and its k-th nearest neighbor x_k. This distribution can be characterized by the following conditions occurring simultaneously:

1. $k-1$ data points are located at smaller distances ϵ_i from x_i.
2. $N-k-1$ data points are located at larger distances ϵ_i from x_i.
3. exactly one data point lies within the infinitesimal shell $[\epsilon_i, \epsilon_i + \mathrm{d}\epsilon_i]$.

Using the multinomial theorem, we can assign the data points to one of these three cases:

$$\begin{aligned} P_k^i(\epsilon_i)\,\mathrm{d}\epsilon_i &= \frac{(N-1)!}{1!(k-1)!(N-k-1)!}\frac{\mathrm{d}p_i(\epsilon_i)}{\mathrm{d}\epsilon_i}\,\mathrm{d}\epsilon_i p_i^{k-1}(1-p_i)^{N-k-1} \\ &= k\binom{N-1}{k}\frac{\mathrm{d}p_i(\epsilon_i)}{\mathrm{d}\epsilon_i}\,\mathrm{d}\epsilon_i p_i^{k-1}(1-p_i)^{N-k-1}, \end{aligned} \tag{4.12}$$

where $\binom{n}{k}$ represents the binomial coefficient and p_i denotes the mass of the hypersphere of radius ϵ_i around x_i:

$$p_i(\epsilon_i) = \int_{\|x-x_i\|\leq\epsilon_i} \mathrm{d}x\, P(x)$$

Using Eq. (4.12), we can now compute the expectation value of $\ln p_i(\epsilon_i)$

$$\begin{aligned}\langle \ln p_i(\epsilon_i)\rangle &= \int \mathrm{d}\epsilon_i P_k^i(\epsilon_i) \ln p_i(\epsilon_i) \\ &= k\binom{N-1}{k} \underbrace{\int_0^1 \mathrm{d}p_i p_i^{k-1}(1-p_i)^{N-k-1} \ln p_i}_{=\frac{\Gamma(k)\Gamma(N-k)}{\Gamma(N)}[\psi(k)-\psi(N)]} \\ &= \psi(k) - \psi(N). \end{aligned} \tag{4.13}$$

$\psi(k)$ denotes the digamma function, that satisfies the recursion $\psi(n+1) = \psi(n) + 1/n$, with the Euler-Mascheroni constant defining the starting value $\psi(1) = -\gamma \approx -0.577$.

This is the key result of the KSG-estimator: while direct computation of $P(x)$ is not possible, the expectation value of $\ln p_i(\epsilon)$ can be computed because of the known analytical form of $P_k^i(\epsilon_i)$ in Eq. (4.12). To return to the entropy estimation, we now just need to relate $\langle \ln p_i(\epsilon_i)\rangle$ back to $\ln P(x_i)$. Under the assumption that the probability density is approximately constant within the local neighborhood around x_i, we approximate:

$$p_i(\epsilon_i) \approx c_d (2\epsilon_i)^d P(x_i),$$

where c_d is the volume of a d-dimensional unit ball and $P(x_i)$ the density of the 2ϵ ball around x_i. Taking the logarithm on both sides and using Eq. (4.13), we find

$$\ln P(x_i) \approx \psi(k) - \psi(N) - \ln c_d - d\langle \ln 2\epsilon_i\rangle.$$

This finally leads to the KSG-estimator for the entropies

$$\begin{aligned}\hat{H}_{\mathrm{d}}(X) &= -\psi(k) + \psi(N) + \ln c_d + \frac{d}{N}\sum_{i=1}^{N} \ln 2\epsilon_i. \\ \hat{H}_{\mathrm{d}}(X,Y) &= -\psi(k) + \psi(N) + \ln\left(c_{d_X} c_{d_Y}\right) + \frac{d_X + d_Y}{N}\sum_{i=1}^{N} \ln 2\epsilon_i \\ &= -\psi(k) + \psi(N) + \ln\left(c_{d_X} c_{d_Y}\right) + (d_X + d_Y)\langle \ln 2\epsilon\rangle \end{aligned} \tag{4.14}$$

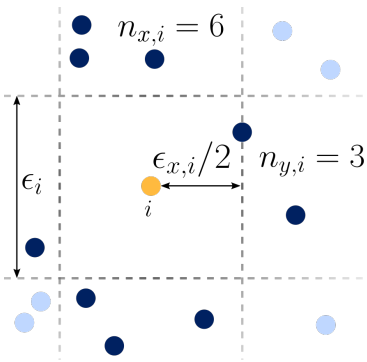


Figure 4.3 | Determination of ϵ_i in the joint space via $\epsilon_i = \max\{\epsilon_{x,i}, \epsilon_{y,i}\}$ (here for a fixed $k = 1$). $n_{x,i}$ and $n_{y,i}$ denote the number of neighbors in the marginal spaces that fall within the distance ϵ_i.

In theory, this would provide us with all the means to compute the MI via Eq. (4.3). However, the k-nn distances in the joint space (X, Y) are systematically larger than in the marginal spaces, which inevitably leads to a bias in the estimation of the MI. To address this, the KSG-estimator fixes the distance scale ϵ_i in the joint space and then counts how many neighbors fall within the same distance in each marginal space n_x and n_y (see Fig. 4.3). This ensures the same distance scales in all subspaces and thus ensures that the finite-sample biases in the entropy estimates fluctuate together and thus largely cancel out in Eq. 4.3. These estimators

lead to the (differential) entropy

$$\begin{aligned}\hat{H}_{\mathrm{d}}(X) &= -\frac{1}{N}\sum_{i=1}^{N}\psi(n_{x,i}+1) + -\psi(N) + \ln c_{d_X} + \frac{d_X}{N}\sum_{i=1}^{N}\ln 2\epsilon_i \\ &= -\langle\psi(n_x+1)\rangle + \psi(N) + \ln c_{d_X} + d_X\langle\ln 2\epsilon\rangle \end{aligned} \tag{4.15}$$

Substituting $H(X)$ and $H(X,Y)$ in Eq. (4.3) with eqs. (4.15) and (4.14), we find the final estimator for the MI

$$I(X,Y) = \psi(N) + \psi(k) - \langle\psi(n_x+1) + \psi(n_y+1)\rangle. \tag{4.16}$$

Importantly, the specific dimensionality d_X and d_Y do not affect the algorithmic structure, as the k-nn framework works in any dimension, allowing the KSG-estimator to be applied to arbitrarily high-dimensional data. While the KSG-estimator in Eq. (4.16) works perfectly fine for the computation of non-normalized MI, it is not suited for the computation of NMI due to its invariance with respect to variable transformations.

4.4 Deriving an Estimator for Normalized Mutual Information

As explained above, we, therefore, need to modify the KSG-estimator such that it computes the relative entropies instead of the differential ones. Employing the relative entropy in Eq. (4.9), this introduces the invariant measures into the marginal entropy estimator

$$\begin{aligned}\hat{H}_{\mathrm{r}}(X) &= -\left\langle\frac{P(X)}{m(X)}\right\rangle = \hat{H}_{\mathrm{d}}(X) + \langle\ln m(X)\rangle \\ &= -\langle\psi(n_x+1)\rangle + \psi(N) + \ln c_{d_X} + d_X\langle\ln 2\epsilon\rangle + \langle\ln m(X)\rangle,\end{aligned} \tag{4.17}$$

and analogously for the joint entropy

$$\begin{aligned}\hat{H}_{\mathrm{r}}(X,Y) &= \hat{H}_{\mathrm{d}}(X,Y) + \langle\ln m(X,Y)\rangle, \\ &= -\psi(k) + \psi(N) + \ln\left(c_{d_X}c_{d_Y}\right) + (d_X+d_Y)\langle\ln 2\epsilon\rangle + \langle\ln m(X,Y)\rangle.\end{aligned} \tag{4.18}$$

4.4.1 Estimation of the Invariant Measure

After establishing the scale and parameterization invariance of the KSG-estimator through invariant measures, we now must determine the specific form of these measures. In order to avoid bias through prior assumptions about the data distribution, Jaynes suggested using an invariant measure $m(X)$ that reflects complete ignorance about the underlying data structure—essentially assuming a uniform probability density.[291] For the one-dimensional case of data $\{x_i\}$ distributed in the interval between a and b, this yields

$$1 = \int_a^b \mathrm{d}x\, m \Rightarrow m = \frac{1}{b-a},$$

meaning that the invariant measure is simply a constant and is defined by the range of the data. While this might work reasonably well in one-dimensions, the assumption of uniform sampling within a $d-$dimensional hypercube becomes more problematic in higher dimensions: as the dimensionality increases, data points become increasingly sparse (hello again, curse of dimensionality!) and tend to cluster near the boundaries of the hypercube and make this assumption increasingly unrealistic. Outliers can dramatically inflate the estimated volume of the hypercube, which leads to a significant underestimation of probability density and, thus, to an overestimation of the entropy. Moreover, the effective dimensionality of MD simulation data is typically far lower than the full dimensionality, as the dynamics are constrained by, e.g., bonds between atoms or hydrophobic interactions, effectively creating highly non-uniform distributions.

To overcome this limitation, we propose an alternative estimator for the invariant measure. Following Jaynes' suggestion of complete ignorance,[291] we define the invariant measure $m(X)$ and $m(X,Y)$ as the inverse of the corresponding volumes enclosed by the data points, i.e.,

$$m(X) = 1/V(X) \quad \text{and} \quad m(X,Y) = 1/V(X,Y).$$

Here, we will only consider quantities that are already computed in the original KSG-estimator, namely the k-nn distances ϵ, the number of data points n whose distance from a considered data point is less than ϵ, and the volume c_d of the $d-$dimensional unit ball. This will maximize computationally efficiency, and it maintains scalability to large data sets.

To get a better estimate of the actual volume spanned by all data points, instead of assuming a $d-$dimensional hypercube, we will approximate the volume as N times the local neighborhood (mean) volume of a single data point $\langle (2\epsilon)^{d_X+d_Y} \rangle$. In order to avoid overcounting, we divide by the number of nearest neighbors k, yielding

$$\hat{V}(X,Y) = \frac{N}{k} c_{d_X} c_{d_Y} \langle (2\epsilon)^{d_X+d_Y} \rangle. \tag{4.19}$$

Again, we demand factorization [see Eq. (4.10)]

$$\hat{V}(X,Y) = \hat{V}(X)\hat{V}(Y),$$

leading to the invariant measures[iv]

$$\ln \hat{m}(X) = -\ln c_{d_X} - \frac{d_X}{d_X+d_Y} \ln \langle (2\epsilon)^{d_X+d_Y} \rangle, \tag{4.20}$$

$$\ln \hat{m}(X,Y) = -\ln\left(c_{d_X} c_{d_Y}\right) - \ln \langle (2\epsilon)^{d_X+d_Y} \rangle. \tag{4.21}$$

Substituting these expressions back into eqs. (4.17) and (4.18), we find the final results for the entropy estimators:

$$\hat{H}_{\mathrm{r}}(X) = -\langle \psi(n_x+1) \rangle + \psi(N) + d_X \langle \ln \tilde{\epsilon} \rangle, \tag{4.22}$$

$$\hat{H}_{\mathrm{r}}(X,Y) = -\psi(k) + \psi(N) + (d_X+d_Y) \langle \ln \tilde{\epsilon} \rangle, \tag{4.23}$$

where $\tilde{\epsilon} = \epsilon / \sqrt[d]{\langle \epsilon^d \rangle}$ represents the scaling invariant k-nn radius and $d = d_X + d_Y$ the dimensionality of the space (X,Y).

[iv] Just like in Jaynes' derivation of the relative entropy,[291] we drop the term $\ln N$. Similarly, we also neglect $\ln k$, since it does not depend on the data and thus does not affect the normalization.

Based on the inequalities given in Eq. (4.6), the MI in Eq. 4.7 can now finally be normalized. Equations (4.20)-(4.23) constitute the main theoretical contributions and address two main challenges in the computation of (normalized) MI: The first challenge concerns the accurate estimation of probability distributions in multidimensional spaces when working with finite datasets, a problem worsened by the curse of dimensionality. Employing k-nearest neighbor statistics, robust local density allows the approximation of probability distributions and their entropy estimates without requiring the explicit construction of probability density functions. The second challenge is then the normalization of the MI, which is necessary for a meaningful interpretation of the MI values and enables comparisons across different datasets. This normalization is achieved through the use of relative entropy, which can conveniently be computed using the same k-nearest neighbor statistics that are already computed as part of the KSG-estimator algorithm. Together, these contributions provide a theoretically robust and computationally efficient framework for the computation of a scale-invariant MI that reliably quantifies the relation between different random variables, even in the multidimensional case.

4.4.2 Validation of the NMI Estimator

It is an obvious test to check whether the volume estimator in Eq. (4.19) reliably approximates the actual volume. To this end, we generate two simple example distributions:

1. a uniform distribution $(x, y) \in [0, 1]^2$ and
2. a "donut" distribution, where we uniformly sample data points in the annulus $0.5 \leq \sqrt{x^2 + y^2} \leq 1$.

For both distributions, we can compute the true volumes analytically so that we can divide the estimated volume $\hat{V}(X, Y)$ by the exact volume $V_{\text{ex}}(X, Y)$ such that a perfect estimate would yield a ratio of $\hat{V}/V_{\text{ex}} = 1$. In Fig. 4.4, we show the results of our volume estimator for various numbers of nearest neighbors k as a function of the sample size N. In order to suppress large fluctuations of $\hat{V}$ for small values of N, we average over 100 independent realizations of both probability distributions.

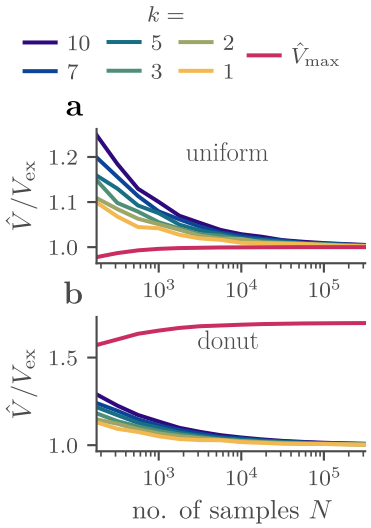


Figure 4.4 | Estimation of the volume of (a) a uniform distribution and (b) a donut-shaped distribution. Adapted with minor changes from Ref. 3. Copyright © (2024) Authors of Ref. 3.

In the case of the uniform distribution, we find that our k-nn volume estimator converges rapidly towards the exact volume and depends only weakly on the number of nearest neighbors k—given that the sample size is sufficiently large (see Fig. 4.4 a). In this specific case, even the simple "naive" volume estimator that relies solely on the data distribution boundaries, $\hat{V}_{\text{max}} = (x_{\text{max}} - x_{\text{min}})(y_{\text{max}} - y_{\text{min}})$, works perfectly well. However, in the second case of the donut distribution (see Fig. 4.4 b), the naive volume estimator obviously fails to capture the actual volume due to the empty space in the center. On the other hand, the k-nn volume estimator reliably approximates the correct volume. These results demonstrate that our k-nn volume estimator provides reliable and robust estimates of the actual volume spanned by the data points. Since we only assume local density homogeneity around each data point, this estimator is particularly well-suited for complex MD data, where the underlying structure is unknown.

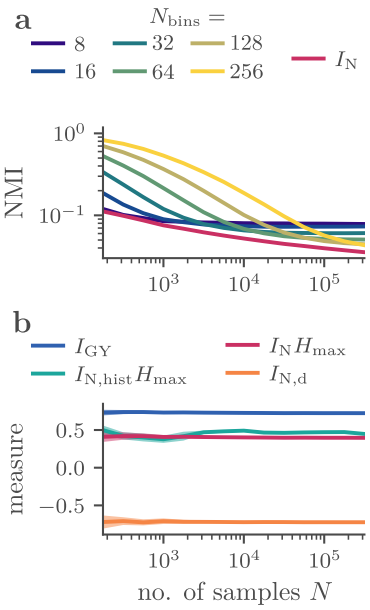


Figure 4.5 | Estimation of the NMI for the 2D toy model defined in Eq. (4.24) for different sample sizes and number of bins per dimension N_{bins}. (a) Comparison between our NMI estimator and the discrete histogram-based approach; both normalized via the geometric mean of the marginal entropies $\sqrt{H(X)H(Y)}$. (b) Comparison of various estimators for NMI, with standard deviations indicated by the shaded areas. The compared estimators include: k-nn estimator $I_{\text{N}} H_{\max}$ ($k = 5$), the 2D histogram-approach $I_{\text{N, hist}} H_{\max}$, the Gel'fand Yaglom NMI I_{G}, and $I_{\text{N, d}}$ using differential entropy. For each value of N, 100 independent realizations were sampled and the NMI computed. Adapted with minor changes from Ref. 3. Copyright ©(2024) Authors of Ref. 3.

To evaluate the accuracy and performance of our NMI estimator defined in Eq. (4.7), we will now employ it to a two-dimensional toy model with the distribution

$$P\left(r = \sqrt{x^2 + y^2}\right) \propto \exp\left[-\frac{(r - r_0)^2}{2\sigma^2}\right], \tag{4.24}$$

where $r_0 = 0.75$ represents the mean and $\sigma = 1/8$ denotes the standard deviation. We subsequently compare the results of our k-nn estimator I_{N} with $k = 5$ against the most established method, that is, the histogram-based approach $I_{\text{N, hist}}$. Since we are considering one-dimensional random variables X and Y, we expect this discrete estimation of the NMI via a binning approach to be accurate for large sample sizes N.[167] Nevertheless, as discussed in Sec. 3.1.2, the precision of probability density estimation, and consequently entropy calculation, critically depends on the choice of bin number per dimension N_{bins}. Comparing the NMI for various choices of N_{bins} and sample sizes N, we confirm that the histogram-based approach indeed heavily depends on both parameters (see Fig. 4.5 a). Generally, this simple binning approach overestimates the NMI, which is why Tiwary and coworkers previously suggested the heuristic of choosing N_{bins} such that the NMI is minimized.[167] Indeed, this heuristic results in the closest approximation of the results from k-nn estimator I_{N}.

The challenge of selecting an appropriate number of bins is the trade-off between avoiding too few bins, which would fail to capture relationships in the data, and too many bins, which drastically overestimates the NMI. The bin-dependency problem becomes most apparent if we consider the (extreme) case where the binning is so fine-grained that every bin in X and Y contains exactly one data point, i.e., $N_{\text{bins}} = N$. In such a scenario, each occupied bin in the joint histogram has equal probability mass of $p_x = 1/N = p_{x,y}$, and every row and column contains exactly one data point (all other bins are empty and thus $p_{x,y} = 0$). The MI calculation then yields:

$$\begin{aligned} I(X,Y) &= \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{x,y} \ln \frac{p_{x,y}}{p_x p_y} = \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} \frac{1}{N} \ln \frac{1/N}{(1/N)(1/N)} \\ &= \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} \frac{1}{N} \ln N = \ln N. \end{aligned}$$

Since $p_x = 1/N$, the marginal entropies become

$$H(Y) = H(X) = -\sum_{x\in\mathcal{X}} p_x \ln p_x = -\sum_{x\in\mathcal{X}} 1/N \ln 1/N = \ln N,$$

resulting in an NMI of

$$I_{\text{N}}(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} = \frac{\ln N}{\sqrt{\ln N \ln N}} = 1.$$

That being said, we are considering one-dimensional random variables X and Y here. In the case of Cartesian coordinates, the joint density $p_{x,y}$ becomes six-dimensional, making the binning approach prohibitive and statistically unreliable due to the low resolution needed to accommodate N_{bins}^6 bins in memory. In contrast, k-nn methods can be applied to arbitrary dimensions without the exponential increase in memory require-

ments, rendering them particularly well-suited for multidimensional MD data.

In order to address the sample size dependency that we can observe for both methods (compare Fig. 4.5 a), we define H_{max} as the highest achievable entropy for a given number of frames N. As analyzed above, this corresponds to $H_{\text{max}} = \ln N$. Both the k-nn estimator I_{N} and the histogram-based estimator $I_{\text{N, hist}}$ exhibit a characteristic $1/H_{\text{max}}$ scaling behavior with N. Therefore, multiplying the results of both estimators with H_{max}, Fig. 4.5 b shows that the resulting NMI measures become largely independent of the sample size N, and the fluctuations of the NMIs (shaded areas) fall below the line width for $N \gtrsim 10^3$, indicating fast statistical convergence.

Finally, we want to highlight the critical importance of introducing an invariant measure when calculating I_{N}. In Fig. 4.5 b, we show the results for $I_{\text{N, d}}$, which are computed using the differential entropy in Eq. (4.8) without appropriate treatment of the scale invariance. This approach yields negative values for the NMI, which is physically meaningless since we require that the NMI is bound between 0 and 1. As a normalized quantity that does not depend on entropies, we recall the Gel'fand-Yaglom NMI[233] defined in Eq. 3.8 as $I_{\text{GY}} = \sqrt{1 - \exp[-2I(X,Y)/(d_X + d_Y)]}$. Analogous to the results in the last chapter, we find that I_{GY} overestimates the NMI values, which are significantly larger than the values obtained from the k-nn estimator and from the binning approach.

4.4.3 Runtime

In order to evaluate the performance and scalability of the NMI estimator, we performed a runtime benchmark using synthetic data generated according to:

$$x_i = \mathcal{N}(\pm 1, 1), \quad \text{with alternating signs for successive samples,}$$
$$y_i = x_i + 0.2\,\mathcal{N}(0, 1),$$

where $\mathcal{N}(\mu, \sigma^2)$ denotes a normal distribution with mean μ and variance σ^2. This creates highly correlated data points, where y is a noisy function of x. In Fig. 4.6 a, we show the runtime of the NMI estimator as a function of the sample size N and the number of parallel jobs. We can observe that the runtime scales approximately in the order of $\mathcal{O}(N \ln N)$, which is expected for the KSG-estimator due to the k-nn search. Parallel jobs can speed up the computation, but the performance improvements are becoming increasingly smaller with the number of parallel jobs.

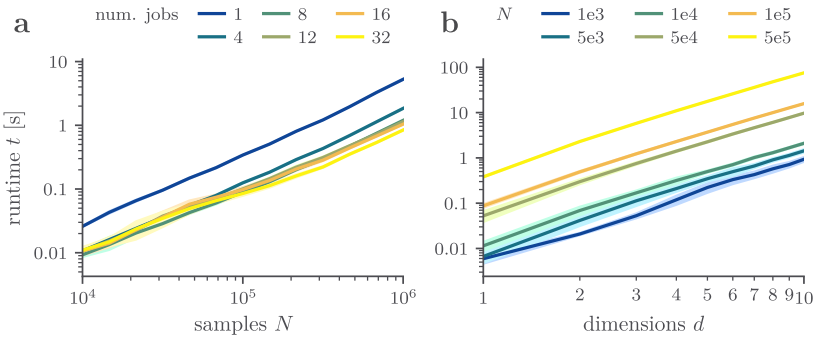


Figure 4.6 | Runtime benchmark of the NMI estimator using synthetic data on an Intel® Core™ i9-14900K CPU. (a) runtime as a function of sample size and number of parallel jobs; (b) runtime as a function of the dimensionality of the data and sample size. For all computations, we used $k = 3$.

Fig. 4.6 b shows the runtime for different sample sizes N as a function of the dimensionality d of the data. Here, we observe a power-law scaling of the runtime with respect to the dimensionality, with theoretical contributions containing Euclidean distance computations [$\mathcal{O}(d)$] and k-nn search operations that degrade with increasing dimensionality [$\mathcal{O}(N \ln N)$]. For MD applications that involve Cartesian coordinates ($d = 3$), this power law scaling remains computationally manageable, even for large sample sizes in the order of $N \approx 10^4 - 10^6$.

Based on these runtime benchmarks, we expect the following runtimes for the computation of the full pairwise NMI matrix for a typical MD trajectory consisting of $N = 10^5$ frames: approximately 24 hours (1 hour) to analyze 1500 (300) internal coordinates such as dihedral angles or inter-residual distances, and approximately 24 hours (1 hour) for a protein consisting of 140 (26) amino acids based on their three-dimensional Cartesian coordinates of e.g. their C_α-atoms.

4.4.4 Software

The k-nn estimator for the NMI has been implemented as an open-source Python package (NorMI) that follows the scikit-learn API conventions.[252] The package can be installed via pip or conda: `pip/conda install normi`, and the source code and documentation can be found here: `https://github.com/moldyn/NorMI`. All analyses in this and the next chapter that involve NMI have been performed using NorMI.

4.5 Concluding Remarks

In this chapter, we have systematically investigated fundamental theoretical and computational challenges involved in quantifying the similarity between multidimensional random variables. We demonstrated that the widely-used multidimensional Pearson correlation suffers from critical flaws, such as the sign cancellation problem, that can obscure perfect functional dependencies. While canonical correlation analysis provides a partial remedy by transforming the data into a new basis, it shares the most fundamental restriction of the Pearson correlation: the assumption that the underlying joint probability distribution of the two random variables is Gaussian distributed.

We showed that mutual information represents a theoretically sound solution by quantifying deviations from statistical independence $P(X, Y) = P(X)P(Y)$ rather than imposing a specific functional form to measure the relationship between X and Y. However, extending mutual information to the multidimensional case leads to new challenges, particularly with respect to proper normalization. Therefore, the central theoretical contribution of this chapter is the development of a scale and coordinate system invariant mutual information estimator that is appropriately normalized. To this end, we extended the Kraskov-Stögbauer-Grassberger to compute relative entropies instead of the scaling-dependent differential entropies while maintaining its efficient k-nearest neighbor framework for estimating mutual information. To do so, we introduced suitable invariant measures based on k-nearest neighbor volume estimation and

showed that the resulting estimator converges reliably to the actual values of known distributions. Furthermore, this framework avoids most issues related to the curse of dimensionality and can be applied to, e.g., Cartesian coordinates in MD simulations without any difficulty.

While this chapter has focused on establishing the theoretical foundations and computational methodology, we will demonstrate the practical impact of these advances in the next chapter, where we thoroughly investigate the protein T4 lysozyme.

A Case Study on T4 Lysozyme

5

Parts of this chapter are based on our publications:

Cooperative Protein Allosteric Transition Mediated by a Fluctuating Transmission Network
M. Post, B. Lickert, G. Diez, S. Wolf, and G. Stock *J. Mol. Biol.* **2022** 434 (17), 167679,
DOI: https://doi.org/10.1016/j.jmb.2022.167679.

Accurate Estimation of the Normalized Mutual Information of Multidimensional Data
D. Nagel, G. Diez and G. Stock, *J. Chem. Phys.* **2024** 161 (5), 054108,
DOI: https://doi.org/10.1063/5.0217960.

In this chapter, we present a comprehensive case study on T4 lysozyme (T4L; see Sec. 2.5), applying the correlation analysis methods developed in the previous chapters to thoroughly investigate the allosteric transition between the open and closed state.

We structure this chapter in four main parts. First, we identify the microscopic mechanism driving T4L's conformational change by constructing a local network of highly correlated inter-residue distances that move in a coordinated manner. This contact-based approach reveals the cooperative nature of the conformational change and allows the identification of essential internal coordinates involved in the long-range allosteric coupling.[2] In the second part, we employ NMI of Cartesian C_α-coordinates to construct a residue interaction network that captures global correlation patterns across the entire protein. Finally, we integrate these two somewhat complementary approaches into a unified picture of the allosteric transition in T4L. We conclude by exploring practical, more technical aspects of multidimensional similarity measures, some of which were already discussed more theoretically in the previous chapter.

Allostery in a Nutshell

Before studying the specific allosteric transition in T4L, we want to briefly introduce the general concept of allostery itself. Allostery represents a form of distant regulation where an effector molecule (such as a ligand) causes a perturbation at one site of the molecule, resulting in a functional change at a remote site through alteration of shape and/or dynamics (see Fig. 5.1).[294] A paradigmatic example is the protein hemoglobin, where oxygen binding at one site enhances oxygen binding affinity at the three remaining sites, ensuring efficient oxygen transport throughout the cardiovascular system.[31,295,296]

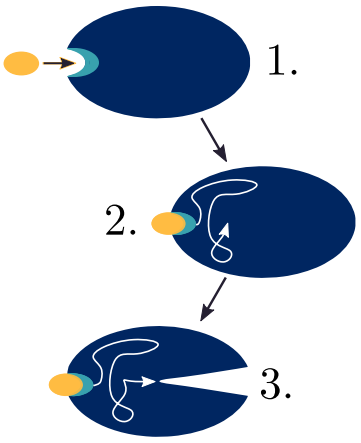


Figure 5.1 | Simple schematic representation of allostery: the binding of a ligand in yellow to the allosteric site of a protein in bright blue (left) induces a conformational change in the protein at a distant site (right).

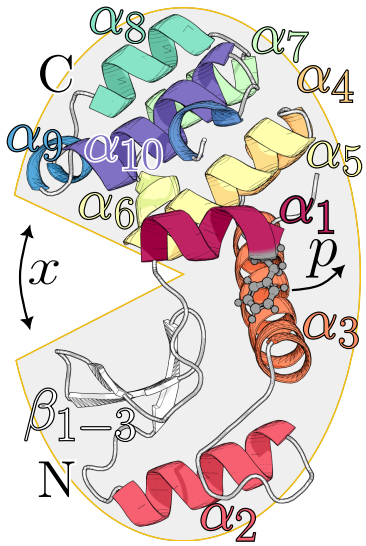


Figure 5.2 | Structure of T4L in the open state, indicating the opening coordinate x and the locking coordinate p.

Despite allosteric couplings appearing to be ubiquitous in protein systems,[297] the underlying dynamical processes—including their microscopic details and temporal evolution during conformational changes—remain poorly understood.[55] This is in stark contrast to the well-characterized field of protein folding, where decades of combined experimental and theoretical research have established a solid understanding of the governing principles, including cooperative two-state and multistate downhill folding,[298] along with dynamical mechanisms such as zipping or diffusion-limited processes.[299] Unlike protein folding, where large-scale conformational changes during folding are relatively straightforward to detect, the subtle local structural changes occurring in many allosteric transitions are much more challenging to observe in experiments and simulations.[93,300]

To this end, we consider T4L as a well-studied example of a bistable two-domain protein.[99,218–220] While it might not be a usual example of an allosteric protein (it lacks a ligand binding site), previous analysis hinted at a hidden locking mechanism that allosterically couples the distant mouth and hinge region of the protein.[98]

The Locking Mechanism in T4L

The open↔closed transition of T4L is typically characterized by the mouth opening width x (see Fig. 5.2), which can, e.g., be quantified by the distance $x \equiv d_{20,145}$ between residues 20 and 145. Analysis of the 50 μs MD trajectory[i] by Ernst *et al.* in Fig. 5.3 b[98] reveals that transitions between open and closed occur on a microsecond timescale, whereas the transition path times are significantly shorter on the order of a few nanoseconds. From this data, we estimate equilibrium populations of approximately 75% for the open state and 25% for the closed state, with mean waiting times of $\tau_{o\to c} \approx 4\,\mu s$ and $\tau_{c\to o} \approx 2\,\mu s$, respectively. These findings are in excellent agreement with recent experimental results.[220] Despite the clear bistable behavior of T4L, with transition times on the microsecond timescale, the free energy profile $\Delta G(x)$ along the mouth opening coordinate exhibits a surprisingly low energy barrier of only $\Delta G^\ddagger \approx 1\,k_B T$. According to transition state theory,[301] the reaction rate is given by $k = \frac{1}{\tau} = k_0 \exp(-\Delta G^\ddagger / k_B T)$, where τ is the mean waiting time between transitions. With such a small barrier of $\Delta G^\ddagger \approx 1\,k_B T$, it is evident that the transition rate is predominantly caused by the prefactor k_0 rather than the thermodynamic barrier height. This indicates that the mouth opening coordinate x solely reflects the consequences of the allosteric transition but not the originating cause itself.

As mentioned earlier, the actual process is mediated by a locking mechanism, which allows the side chain of Phe4 to change from a solvent-exposed to a buried state. This locking mechanism can be characterized by the locking coordinate $p \equiv d_{4,60}$, which is the hydrophobic locking distance between Phe4 and Lys60. Looking at its time trace in Fig. 5.3 b, we confirm that p perfectly discriminates between the open and closed state, with $p \lesssim 0.7$ nm denoting the locked, hydrophobically buried state and $p \gtrsim 0.7$ nm representing the free, solvent-exposed state. The coupling between x and p becomes evident when looking at the free energy landscape $\Delta G(x, p)$ in Fig.5.3 a, which shows four metastable states: predominantly the open state S_1 and closed state S_4 alongside two sparsely

[i] In contrast to Ref. 2, we excluded the last 11 μs of the trajectory because the β-sheets at the lower jaw were temporarily deforming around $t = 51\,\mu s$, potentially introducing non-physical artifacts.[261]

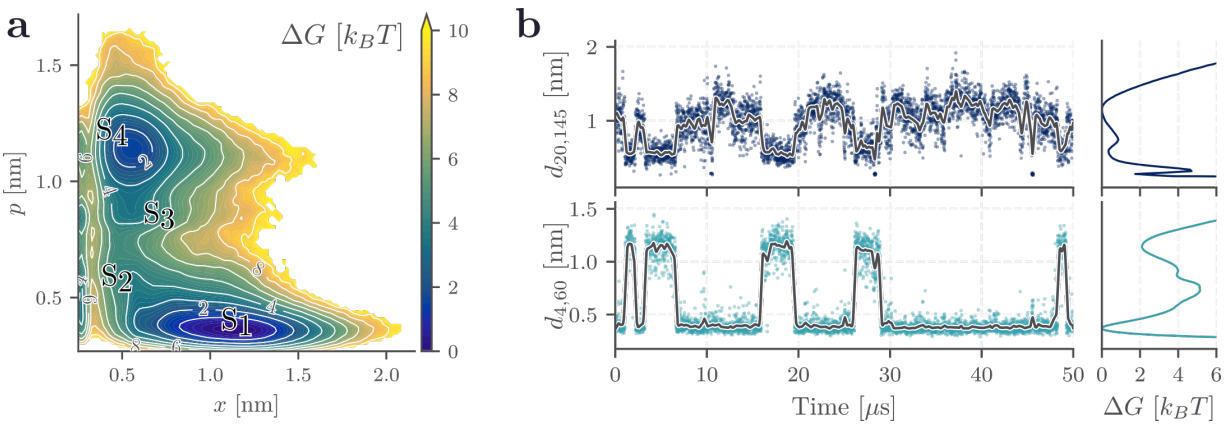


Figure 5.3 | Two-dimensional model of the allosteric transition in T4L using the opening coordinate $x \equiv d_{20,145}$ and the locking coordinate $p \equiv d_{4,60}$. (a) Two-dimensional free energy landscape $\Delta G(x,p)$ suggesting four metastable states S_1-S_4. (b) MD time traces of the opening coordinate x (top) and the locking coordinate p (bottom) and their corresponding free energy landscapes. The gray lines indicate Gaussian filtered time traces $g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$, providing a weighted smoothing for enhanced visualization. Adapted with changes from Ref. 2. Copyright © (2022) The Authors.

populated transition states S_2 and S_3. This already gives some insights into allosteric transitions as the system appears to follow a preferred route $S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4$ when transitioning from the open to the closed state, with the reverse transition following the same route in the opposite direction, $S_4 \rightarrow S_3 \rightarrow S_2 \rightarrow S_1$. Nevertheless, the two-dimensional model still remains incomplete, as the highest free energy barrier around $\Delta G^{\ddagger} \approx 6\,k_{\mathrm{B}}T$ is still too low to account for the observed waiting times in the microsecond range. In the following, we demonstrate that the mechanism underlying the open↔closed transition is a cooperative conformational cascade that propagates from the Phe4 locking site in the hinge region to the distant mouth region.

5.1 Constructing a Contact Network

To investigate the process of allosteric transmission between the mouth and hinge region, we analyze inter-residue contact distances and first side-chain dihedral angles. This choice of coordinates is motivated by the fact that allosteric transitions arise from the propagation of local perturbations, and contact distances capture the strengthening and weakening of bonds between different residues, while side-chain dihedral angles describe local conformational rearrangements of the side chains. Requiring a contact $d_{i,j} \leq 0.45$ nm to be formed for at least 1% of the simulation time, we obtain 556 contact distances, that are shown in the contact map in Fig. 5.4.[ii] As some residues, namely those of the type Ala, Gly, or Pro, do not exhibit or change the side-chain dihedral angle, we ended up with 131 dihedral angles χ_n.

[ii] Here, we consider only heavy atoms, while the Ref. 2 erroneously included hydrogens. We also excluded contacts formed between the i−th and $i+j$-th residues where $j \leq 4$, as they stabilize the α-helical structures and are therefore not of interest.

5.1.1 MoSAIC Analysis

To study their collective behavior, we computed the linear correlation matrix according to Eq. (3.1). We then employed MoSAIC,[1] which uses Leiden clustering[244] with the constant Potts model[248] and a resolution parameter of $\gamma = 0.5$, to rearrange the matrix in an approximately

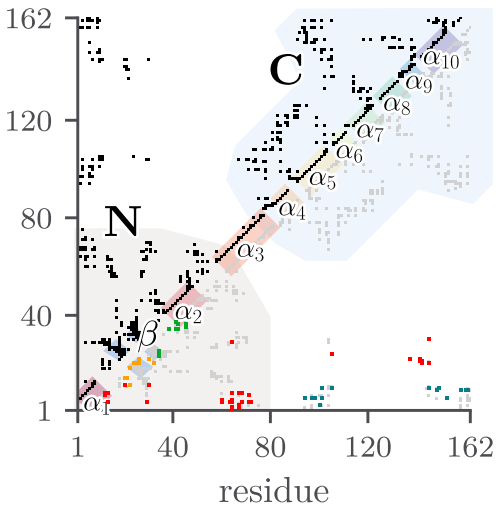


Figure 5.4 | Contact map of T4L, showing all contacts ($d_{i,j} \leq 0.45\,\text{nm}$) that are formed for at least 1% of the simulation time. In the lower diagonal, the contacts are colored according to their MoSAIC cluster membership. N and C denote the N- and C-domain of T4L, respectively.

block-diagonal form. The resulting block-diagonalized correlation matrix (Fig. 5.5 a) shows a clear separation between functionally relevant motions and background fluctuations (or noise). Four main clusters emerge from the MoSAIC analysis, while the majority of contact distances (∼ 85%) exhibit only minor correlations and are therefore assigned to the noise cluster. We attribute this to stable intraprotein contacts in T4L that only fluctuate around their mean distance, while contacts on the protein surface undergo frequent formation and breaking, resulting in random fluctuations.

Among the four main clusters, cluster 1 dominates with 34 highly correlated coordinates (32 contact distances and two χ dihedral angles; see SI, Tab. C.1) that mediate the open↔closed transition. The spatial distribution of these contact distances within the molecular structure of T4L (Fig. 5.5 b) already suggests the allosteric communication network: 10 distances monitor the mouth opening/closing, while the remaining 22 are located in the hinge region, with seven of them directly involving Phe4—highlighting its central role in the mechanism. In contrast, the remaining three clusters, 2-4, (SI, Fig. C.1) are significantly smaller and describe local motions that are not related to the global open↔closed transition. The high average correlation of $\langle|\rho|\rangle = 0.7$ within cluster

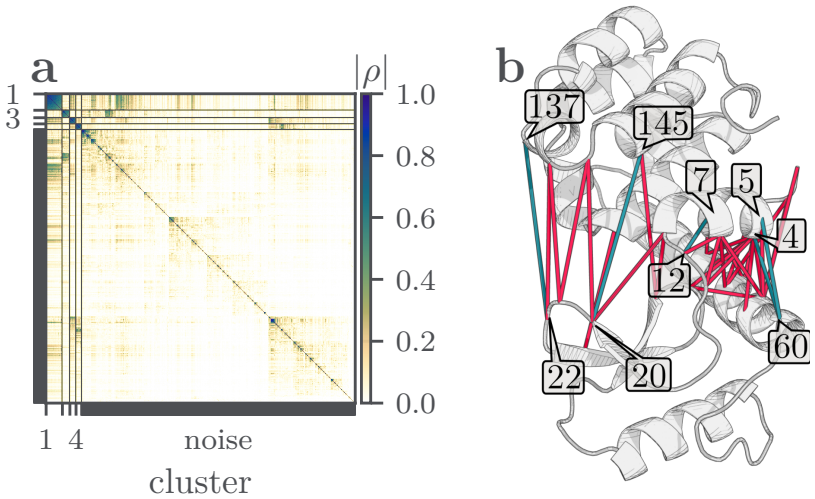


Figure 5.5 | MoSAIC analysis resulting in (a) a block diagonalized correlation matrix containing 556 contact distances and 131 χ dihedral angles. The first block (cluster 1) contains 34 highly correlated coordinates that characterize the open↔closed transition. (b) Illustration of the 32 highly correlated contact distances in cluster 1. Some important distances are highlighted in cyan: $d_{4,60}$, $d_{5,60}$, $d_{22,137}$, $d_{20,145}$ all describing the opening/closing of the mouth and $d_{7,12}$ accounting for the (de)stabilization of the α_1-helix.

1 suggests that the allosteric transition is performed by a cooperative action involving the entire contact network.

This cooperative behavior becomes even more evident when we investigate how the 32 contact distances in cluster 1 behave across both the open and closed conformational state of T4L.[iii] To this end, we analyzed the contact population differences $\Delta p = |p_{\text{open}} - p_{\text{closed}}|$ between the open and the closed state, as presented in SI, Tab. C.2.[302,303] While for 20 contacts, we find $\Delta p \gtrsim 0.7$, which hints to a clear binary switching behavior, 12 contacts exhibit high correlation despite modest population changes of $\Delta p \lesssim 0.2$.[iv] The latter group contains some mechanistically crucial contacts (see next section), including the Glu5-Lys60 salt bridge (directly neighboring the locking coordinate p) and the opening distance $x = d_{20,145}$. This contact population pattern therefore suggests that the allosteric cooperativity is a result of the coordinated network behavior that transcends individual contact properties.

[iii] In order to obtain the open and closed parts of the trajectory, we used the maximum barrier height along the free energy profile of the locking distance p to discriminate between closed ($p > 0.7\,\text{nm}$) and open ($p \leq 0.7\,\text{nm}$) conformations. The detailed procedure is described in the supplementary information (Fig. C.5).

[iv] We note that this grouping depends to some extent on the threshold used to define when a two residues form a contact. Here, we calculated the contact populations using the standard definition of $d \leq 0.45\,\text{nm}$. While widely used, this uniform cutoff may not capture the contact formation perfectly for all distances—the opening distance x, for example, shows its local minimum around $\sim 0.6\,\text{nm}$ (see Fig. 5.3).

Beyond population differences and correlation, we further investigate the temporal evolution of the contact distances in cluster 1. While the coordinates clearly follow a similar temporal behavior (see figs. 5.3, 5.6 or SI, C.2), they exhibit different fluctuations in the open and closed state. For example, the opening coordinate x fluctuates considerably in the open state and only a little in the closed state, while the locking coordinate p shows the opposite behavior. This can be explained by the fact that amino acid side-chains generally fluctuate less when they form a contact with other side-chains, while exhibiting larger fluctuations when not being in contact. The high correlation between all distances in cluster 1 stems from the clear two-state behavior across the microsecond timescale of all coordinates but less so from their rapid fluctuations within each state, which are not necessarily synchronized. As all 34 coordinates must transition from one state into the other in order to achieve a successful open↔closed transition, the process may only take place if the fluctuations of all coordinates cooperatively align by chance. This may explain the rarity of the transition events in the order of microseconds and their relatively short transition path times of a few nanoseconds.

5.1.2 Essential Coordinates and Their Sequential Activation

Although all 34 coordinates are strongly correlated and behave cooperatively, they serve different functional roles depending on their interaction chemistry. To enable a mechanistic understanding of the allosteric transition, we will further reduce the number of coordinates and focus on the contacts that drive the most significant structural rearrangements based on their chemical nature. This selection yields eleven hydrophobic contacts, four salt bridges, and three hydrogen bonds, which are listed in SI, Tab. C.3.

Among these, changes in hydrophobic contacts typically involve complex rearrangements of various atoms that are challenging to interpret structurally. In contrast, salt bridges and hydrogen bonds exhibit a

stronger forming and breaking behavior that can be more easily connected to specific structural changes and makes them particularly valuable for our mechanistic analysis. We identify three salt bridges with a particularly clear role: the first salt bridge (Glu5-Lys60) in the hinge regions complements the above introduced locking coordinate p (Phe4-Lys60), while the second corresponds to the mouth opening width x (Glu20-Lys145) and the third (Glu22-Lys137) is located at the very edge of the mouth (see Fig. 5.5 b).

To understand the temporal relationships among these coordinates, we investigated their time evolution during individual open↔closed transitions (SI, Fig. C.2). Interestingly, this analysis reveals that the fourth salt bridge (Arg8-Glu64), located in the hinge region, seems to compete with the hydrogen bond $d_{2,64}$, which results in a seesaw-like motion of the α_1-helix with respect to the α_3-helix (structural description in SI, Fig. C.3). The distance $d_{7,12}$ shows unique behavior as it reports on the (de)stabilization of helix 1 (more below). Among both side-chain dihedral angles in cluster 1, χ_4 is mechanistically more relevant as it directly describes the re-orientation of the Phe4 side chain. χ_{104} exhibits a similar temporal behavior but shows more fluctuations whose structural implications are investigated below. To sum up, we identified six coordinates (5 distances and 1 dihedral angle) that we can consider essential descriptors of the open↔closed transition: $d_{5,60}$, $d_{4,60}$, χ_4 and $d_{7,12}$ characterize the (un)locking of Phe4 and subsequent α_1-helix rearrangement, while $d_{20,145}$ and $d_{22,137}$ monitor the opening/closing of the mouth. These six coordinates effectively describe the core dynamics and can be thereby considered the most essential internal coordinates.

To demonstrate the synchronized behavior of these six essential coordinates spanning from the hinge to mouth region, we take a close look at a representative open↔closed transition in Fig. 5.6. Setting the transition time to $t = 0$, we see that distance $d_{5,60}$ changes first, followed immediately by the response of distances $d_{4,60}$, $d_{22,137}$ and $d_{20,145}$. After

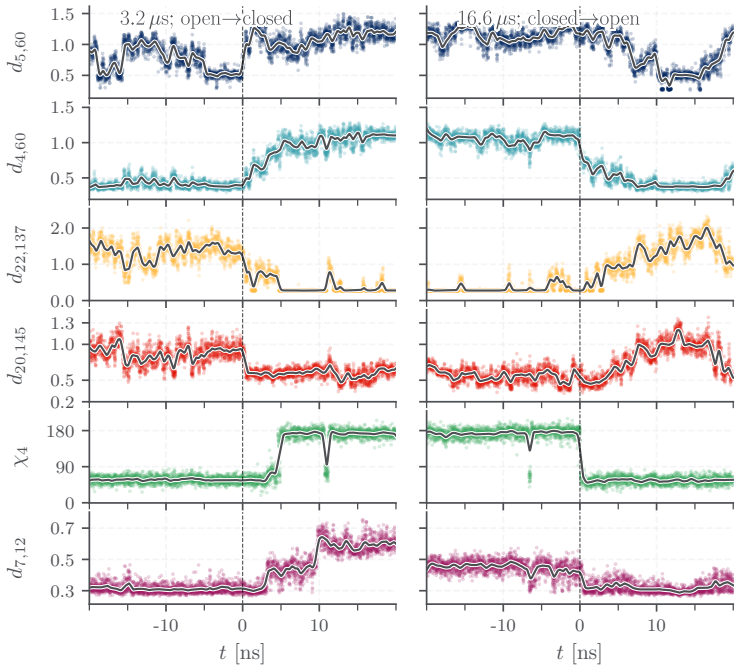


Figure 5.6 | Temporal evolution of six essential coordinates during a representative open→closed transition (left, occurring at 3.2 μs) and closed→open transition (right, occurring at 16.6 μs), respectively. All coordinates are in units of nm, apart from χ_4, which is given in degrees. We used a Gaussian filtering for enhanced visualization. Adapted with minor changes from Ref. 2. Copyright © (2022) The Authors.

a few nanoseconds, the angle χ_4 responds, with $d_{7,12}$ following last, approximately 10 ns later. The simultaneous response of the two distances $d_{4,60}$ and $d_{22,137}$, describing structures that are spatially separated by approximately 2.5 nm, provides evidence for a direct mechanical coupling between the two distant mouth and hinge regions of T4L.

5.1.3 Free Energy Perspective on the Cooperative Transition Mechanisms

Having established the essential six coordinates that describe the allosteric transition, we can now return to our initial question regarding the low free energy barrier of $\Delta G^{\ddagger} \approx 6\, k_{\mathrm{B}}T$ in the simple two-dimensional model $\Delta G(x,p)$. By constructing a free energy landscape using these coordinates, we can see whether this expanded coordinate space now resolves this energetic inconsistency and yields more realistic waiting times. To this end, we computed local free energy estimates for trajectory point $\boldsymbol{X}_t$ employing a local density-based approach. For each time step, we calculated the local free energy as

$$\Delta G(\boldsymbol{X}_t) = -k_{\mathrm{B}}T \ln\left[\frac{P_R(\boldsymbol{X}_t)}{P_R^{\max}}\right],$$

where $P_R(\boldsymbol{X}_t)$ can be estimated using Eq. (2.32). To ensure statistical robustness, particularly in the sparsely populated barrier regions, we dynamically adjusted R to maintain at least 10 neighbors within each hypersphere.

As an example, we revisit the representative open→closed transition at $t = 3.2\,\mu$s, and show the resulting free energy evolution in Fig. 5.7.[v] The analysis includes both the two-dimensional free energy evolution $\Delta G(x,p)$ (upper panel) and the six-dimensional model (lower panel) for comparison. We can observe three distinct regimes in the free energy evolution: before and after the transition, both models show that the system is in the energy minima associated with—respectively—the open and closed conformations, with occasional fluctuations that indicate the

[v] Additional transition examples are provided in the supplementary information (Fig. C.4).

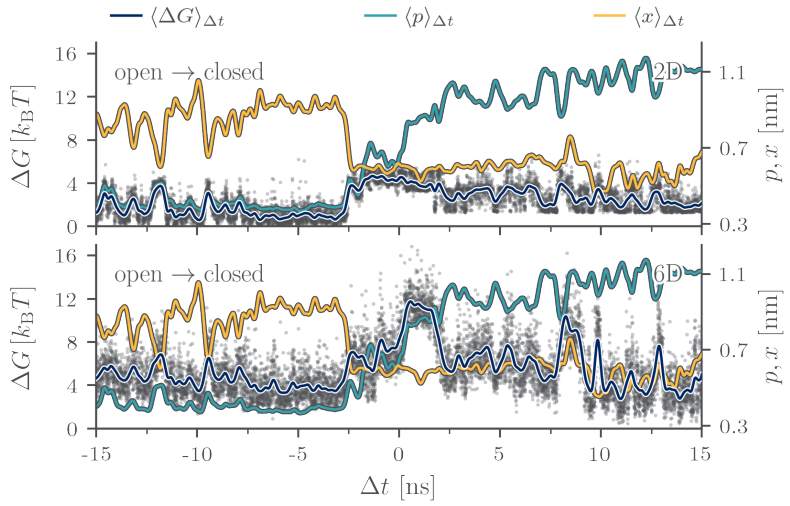


Figure 5.7 | Estimation of the barrier heights of the two-dimensional (upper panel) and six-dimensional (lower panel) free energy landscape of T4L, obtained for the representative open→closed transition at $t = 3.2\,\mu$s. The gray scatter points indicate the free energy values at time t. Adapted with minor changes from Ref. 2. Copyright © (2022) The Authors.

system exploring the boundaries of the energy basins. During the transition, however, the free energy evolution shows significant differences between the two-dimensional and six-dimensional model, as the latter reaches barrier heights as high as $\sim 16\,k_\mathrm{B}T$, whereas the two-dimensional model encounters substantially lower barriers around $\sim 6\,k_\mathrm{B}T$.

This energetic difference becomes more apparent when we analyze the barrier heights across the entire trajectory. To do so, we averaged the free energy evolution over all open→closed and closed→open transitions (see SI, Fig. C.4 for the averaging windows).The results, shown in Fig. 5.8, reveal that the additional four coordinates substantially increase the energy barrier heights. Importantly, this increase occurs symmetrically for both directions of transitions, with the resulting free energy profiles appearing almost identical, suggesting that the forward and backward transitions follow a similar mechanism. For a rough estimate of the barrier heights, we associate the barrier regions with the steep descent in the probability distributions and obtain maximum energies of approximately $(6-8)\,k_\mathrm{B}T$ for the two-dimensional model and $(13-18)\,k_\mathrm{B}T$ for the six essential coordinates. In order to separate the actual energy barriers from thermal fluctuations due to equipartition, we subtract average energies of $1\,k_\mathrm{B}T$ and $3\,k_\mathrm{B}T$, respectively, and obtain estimates for the barrier heights of approximately $6\,k_\mathrm{B}T$ and $12\,k_\mathrm{B}T$ for the two models. These differences arise because the original two-dimensional model captures only the local transition events in the mouth and hinge region, whereas the six essential coordinates model additionally reflects on the long-range transmission of the conformational change.

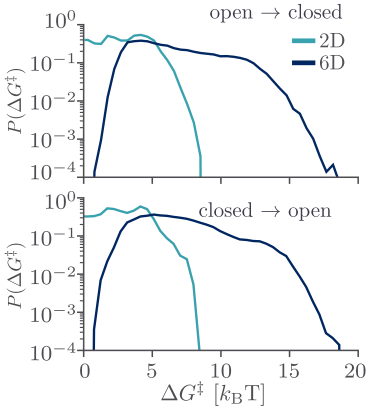


Figure 5.8 | Probability distribution of the free energy values along the transition paths, averaged over all open→closed (top) and closed→open (bottom) transitions. Shown are the probability distributions for the two-dimensional and six-dimensional model. Adapted with minor changes from Ref. 2. Copyright © (2022) The Authors.

The doubling of the barrier heights from $\sim 6\,k_\mathrm{B}T$ to $\sim 12\,k_\mathrm{B}T$ provides quantitative evidence that the allosteric transition in T4L is indeed a cooperative process. This energetic increase reflects the requirement that various coordinates must change simultaneously within the brief transition time of approximately 10 ns, which is roughly one-thousandth of the microseconds long waiting times. Thus, the transition can only occur if the fluctuations of all coordinates align by chance and enable the coincidental switching of all relevant interactions, which represents the hallmark of a cooperative process.

5.2 Constructing a Residue Interaction Network

Complementing this detailed mechanistic understanding of the cooperative allosteric transition, we want to further investigate its global impact on the protein structure of T4L. To this end, we consider the similarity between the Cartesian coordinates of the C_α-atoms.

5.2.1 Cartesian Normalized Mutual Information Reveals Structural Correlation Patterns

Applying our normalized mutual information (NMI) method [eqs. (4.22), (4.23) and (4.16)] to the (globally) aligned MD trajectory yields the NMI

matrix I_N of the Cartesian C_α coordinates in Fig. 5.9 a. The diagonal exhibits the expected large correlation values between neighboring residues, with correlation strengths that reflect the underlying secondary structure. In structured regions, particularly in α-helices, these diagonal correlations extend across 3-4 neighboring residues, creating characteristic broad bands along the diagonal due to stable contacts inside the helices. In contrast, flexible loop regions display much narrower bands since they exhibit less stable contacts and higher structural flexibility.

Moreover, the matrix reveals a clear block structure that reflects the two-domain architecture of T4L. The N-terminal domain ($\alpha_1 - \alpha_3$, $\beta_1 - \beta_3$) and the C-terminal domain ($\alpha_4 - \alpha_9$) [see Fig. 5.2] each form distinct diagonal blocks, indicating strong internal correlations within these relatively rigid subparts of the protein. The presence of off-diagonal blocks hints at further correlated motions between both two domains.

Finally, two secondary structures stand out by exhibiting notably distinct correlation behavior: the α_4-helix, located between the two domains, couples only weakly to the adjacent helices, effectively serving as a flexible linker rather than a rigid connector. This reduced correlation stems from its limited polar interactions with neighboring helices. Similarly, the solvent-exposed α_{10}-helix at the C-terminus correlates weakly with the protein core.

5.2.2 NMI Differences Demonstrate Allosteric Pathways

In addition to a structural characterization of the NMI matrix, we wish to relate it to the process of allosteric transition. To this end, we analyze how correlation patterns change between the two conformational states of T4L. By computing the difference in the NMI between the closed and open state

$$\Delta I_N = I_N^{\text{closed}} - I_N^{\text{open}},$$

we can pinpoint which residues and secondary structures are most affected by the transition, potentially revealing the communication network that mediates the structural change itself. The absolute differences $|\Delta I_N|$ are shown in Fig. 5.9 b and exhibit a general decrease of the NMI in

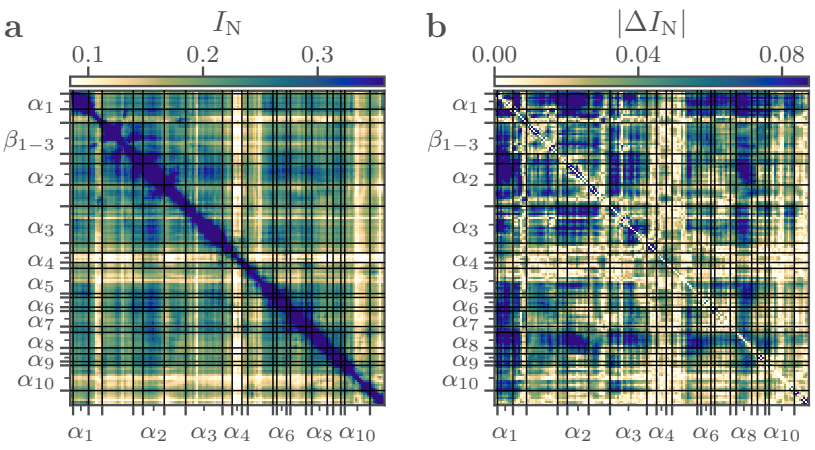


Figure 5.9 | (a) NMI I_N and (b) $|\Delta I_N| = \left|I_N^{\text{closed}} - I_N^{\text{open}}\right|$ computed from Cartesian C_α-atom coordinates of the 50 μs MD trajectory. The colorbar covers values between the 5th and the 95th percentile of the respective matrix.

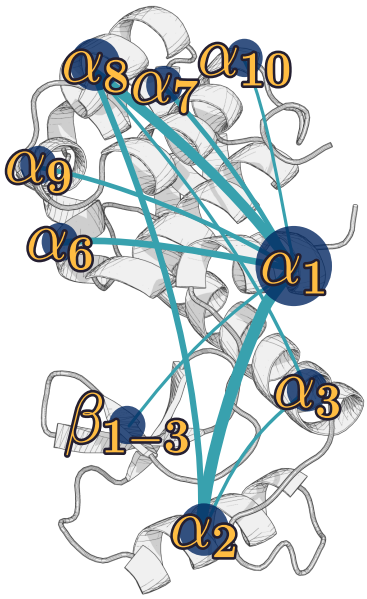


Figure 5.10 | Structural representation of the residue interaction network of T4L, focusing on the most relevant changes in the NMI ($\Delta I_N \geq 0.06$). The edge width indicates the average ΔI_N between two secondary structures. Adapted with minor changes from Ref. 3. Copyright © (2024) Authors.

the closed state (i.e. $\Delta I_N < 0$; not shown) throughout the protein, except the α_4-helix separating the N- and C-domain.

Strong correlations typically arise when two residues rigidly couple and move in a coordinated fashion, while weaker correlations indicate a more flexible connection. The decrease in correlations in the closed state, therefore, indicates that the open state exhibits enhanced coordinated motion between different residues. This interpretation aligns with a root-mean-square-fluctuations analysis (SI, Fig. C.9), which shows increased fluctuations of the residues in the open conformation. Notably, the helices α_4 and α_{10} maintain similar dynamics across both conformations, effectively serving as structural anchors that remain largely unaffected by the allosteric transition.

To facilitate the interpretation of $|\Delta I_N|$, we illustrate the most important interactions (i.e. $|\Delta I_N| \geq 0.06$) in a network representation in Fig. 5.10. This network reveals the α_1-helix as the primary hub for the allosteric transition between open and closed. Nearly all significant correlation differences involve this α-helix, which we already identified as the main mechanistic driver of the allosteric transition relying only on contact distances. Weighted by the average $|\Delta I_N|$ between secondary structures, the edges in the network describe the allosteric communication pathways that enable long-range coupling between the mouth and hinge region.

Centrality-Driven Identification of Allosteric Hubs

As a similar but maybe more systematic approach, we suggest analyzing the importance of individual residues in Cartesian similarity matrices through centrality analysis.[304] The fundamental premise here, consistent with the idea of MoSAIC,[1] is that the importance of a residue is determined by its correlations with other influential residues rather than simply by the number of its connections.

To this end, we transform the $|\Delta I_N|$ matrix into a weighted graph $G = (V, E)$, where the vertices V represent the residues and the edges E correspond to correlation values (or $|\Delta I_N|$ in this case). The eigenvector centrality[304, vi] score ϕ_i for residue i is defined through the eigenvalue problem

$$\boldsymbol{A}\boldsymbol{\phi} = \lambda\boldsymbol{\phi},$$

where $\boldsymbol{A}$ represents the adjacency matrix with elements $a_{i,j} = 1$ if residues i and j are connected, and $a_{i,j} = 0$ otherwise.

At the start, every residue is assumed to be equally important (or central). The centrality score of a residue is subsequently iteratively updated, following

$$\phi_i^{t+1} = \sum_{j \in V} a_{i,j}\phi_j^t.$$

This process propagates centrality across the graph, meaning that residues that are connected to highly central residues will receive higher centralities. According to the Perron-Frobenius theorem, the eigenvector corresponding to the largest eigenvalue λ provides the final centrality ranking.

[vi] Fun fact: Google's famous PageRank algorithm to sort search results is closely related to eigenvector centrality. Just as we identify important residues through their connections to other important residues, PageRank identifies important webpages through links from other important pages.[305]

Fig. 5.11 a shows the 15 most (eigenvector) central residues for the $|\Delta I_N|$ matrix. As expected from previous analysis, the most central residue is Phe4, confirming its crucial role also in the description based on Cartesian C_α coordinates. Apart from Phe4, we also identified several other residues that were found to be essential in the allosteric transition, such as Glu5, Leu7, and Glu11.[2]

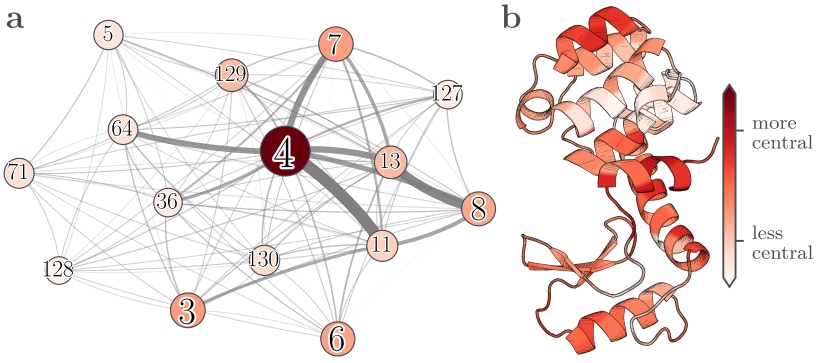


Figure 5.11 | Eigenvector centrality analysis reveals key residue Phe4. (a) Network representation of the 15 most central residues (according to eigenvector centrality based on $|\Delta I_N|$). (b) Three-dimensional structure of T4L, where each residue is colored according to its eigenvector centrality score.

5.3 Some More Technical Remarks on Cartesian Similarity Measures

In this section, we cover some more technical aspects of multidimensional similarity measures that are not directly related to the functional dynamics of T4L.

5.3.1 Translational and Rotational Alignment

In the first step, we must remove the translational and rotational motion from the MD trajectory since this coordinate system is not invariant under rotations and translations. When removing rotational motion from MD trajectories, the alignment procedure relies on the molecule's moment of inertia tensor, which depends on the mass distribution of the protein. For rigid systems, the removal of the roto-translational part of the dynamics is therefore straightforward but less so for flexible systems, such as proteins, as large conformational changes involve a substantial change in the mass distribution. This has the consequence that the reference frame shifts with structural rearrangements, potentially introducing significant artifacts when computing linear correlations. Despite being scale-invariant with respect to linear coordinate transformations,[229] MI also suffers from the above-mentioned problems. To assess the impact of this alignment dependency, we compared two different fitting routines: a global alignment, where both the open and closed parts of the trajectories were aligned to the same reference structure, and a local alignment, where the open and closed reference structures were aligned to their respective mean structures. The analysis of both NMI (Fig. 5.12), as well as linear correlation coefficient (SI, see Fig. C.6), revealed only minor differences, indicating that the standard global rotational fit procedure provides adequate results to describe the functional dynamics of T4L without introducing significant artifacts.

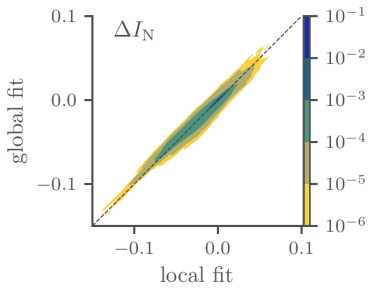


Figure 5.12 | Comparison of local versus global fitting procedures for T4L. Joint probability distribution of the NMI difference $|\Delta I_N| = \left|I_N^{closed} - I_N^{open}\right|$ between the two alignment methods. Local fitting uses RMSD alignment to the minimal average RMSD structure within each conformation (open/closed), while global fitting aligns to the frame with minimal average RMSD across the entire trajectory. Adapted with minor changes from Ref. 3. Copyright © (2024) Authors.

5.3.2 Linear Correlation Cancellation Effect

The linear Pearson correlation coefficient defined in Eq. (4.1) is widely used in the literature[278,279,281–283] despite its known shortcomings discussed in Sec. 4.1 and e.g. Ref. 234. Therefore, comparing the above-found results for NMI with Pearson correlation provides valuable insights into the strengths and weaknesses of both approaches.

The absolute linear correlation matrix, shown in Fig. 5.13 a, differs substantially from I_N in Fig. 5.9 a, exhibiting prominent patterns that are entirely absent in the NMI matrix. Based on a simple two-particle model, we have already illustrated the cancellation effect affecting the multidimensional Pearson correlation in Sec. 4.1, where the directional components ρ^x, ρ^y and ρ^z can have opposing signs that cancel each other out. An analysis of the directional components in SI, Fig. C.7 confirms the presence of this artifact. For example, the helices α_6 and α_8 exhibit large positive correlations in the x-direction and negative correlations in the y- and z-direction, leading to a near-zero net Pearson correlation despite strong coupling between both structures. This behavior explains the spurious patterns in the Pearson correlation matrix.

As a simple remedy to avoid directional cancellation, we may instead sum the moduli of the three components, i.e. $|\rho| = \sum_{\alpha\in\{x,y,z\}} |\rho^\alpha|$. The resulting matrix in Fig. 5.13 b shows much better agreement with the NMI patterns. Similarly, canonical correlation analysis (described in Sec. 4.1.1) yields qualitatively similar results to NMI (SI, Fig. C.8), though with systematically higher values. This demonstrates that the primary difference between NMI and Pearson correlation in the multidimensional case arises from inadequate handling of the relationships between directional components rather than genuine nonlinear correlation effects. That being said, there still remain differences.

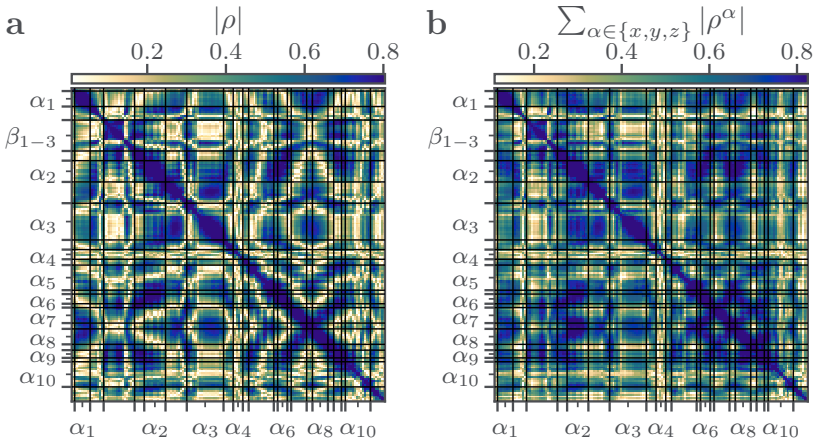


Figure 5.13 | Comparison between linear correlation measures for the Cartesian C_α-atom coordinates of T4L. (a) shows the canonical correlation matrix, while (b) shows the sum of the absolute values of the directional components of the linear correlation.

5.3.3 Comparative Analysis of Similarity Measures for Multidimensional Data

We have already discussed some analogies and differences of various forms of multidimensional similarity measures above. Since correlation matrices often appear visually similar in the standard heatmap representation, we change to an alternative representation to systematically

compare different normalization variants. By sorting the approximately 13000 residue pairs according to increasing values of the unnormalized MI I, we can readily identify relative differences between the different similarity measures. Fig. 5.14 compares the following five distinct normalization schemes:

- $I_{\mathrm{GY}} = \sqrt{1 - \exp(-I(X,Y)/3)}$
- ρ_{C} defined in Eq. (4.2)
- $I_{\mathrm{NM}}^{\mathrm{NaN}}(X,Y) = I(X,Y)/\max I(X,Y)$, where $I_{ij} = \mathrm{NaN}$ if $|i-j| \leq 4$,
- $I_{\mathrm{N}}(X,Y) = I(X,Y)/\sqrt{H(X)H(Y)}$,
- $I_{\mathrm{NM}}(X,Y) = I(X,Y)/\max I(X,Y)$,
- and $I_{\mathrm{NJ}} = I(X,Y)/H(X,Y)$.

I_{N}, I_{NM} and I_{NJ} exhibit remarkably similar behavior, starting at values of approximately 0.06 and showing a gradual, nearly linear increase until around $\sim$ 12000 before rapidly approaching unity. Consequently, approximately 90% of all values fall between 10% and 30% of their maximum value. In contrast, the values in the Gel'fand-Yaglom normalization scheme [described in Eq. (3.8)] are systematically shifted to higher values, such that the majority (say $\sim$ 90%) of the residue pairs show a correlation between 0.5 and 0.8. This systematic inflation of correlation values may lead to misinterpretations when analyzing the correlation's strengths. For example, when we consider residue pairs with a modest value of $I_{\mathrm{N}} \approx 0.2$, the Gel'fand-Yaglom normalization indicates a much higher correlation strength of $I_{\mathrm{GY}} \approx 0.7$. The result from CCA, ρ_{C}, exhibits similar high values but is accompanied by significant vertical spread of the data points. These fluctuations are presumably caused by nonlinear effects that are not described by the linear measure ρ_{C}.

Therefore, we want to focus on I_{N}, I_{NM} and I_{NJ}. Our standard definition I_{N} follows the maximum-normalized I_{NM} very closely and exhibits a rather small average spread of $\sim$ 0.04. I_{NJ} behaves similarly, but is shifted to smaller values, because it is not the tightest bound in Eq. (3.4). The fact that these three measures show such a similar behavior also implies that the respective normalization factors $1/\sqrt{H(X)H(Y)}$ and $1/H(X,Y)$ are largely constant and depend only weakly on the specific residue pair. As we have discussed in the previous chapter [compare eqs. (4.22) and (4.23)], this dependence is introduced only through the scaling invariant k-nn radius $\tilde{\epsilon}$, which apparently is quite similar for most residue pairs in a densely packed protein like T4L.

While the close agreement between I_{N} and I_{NM} initially appears to validate both approaches, we find that the entropy-based normalization I_{N}

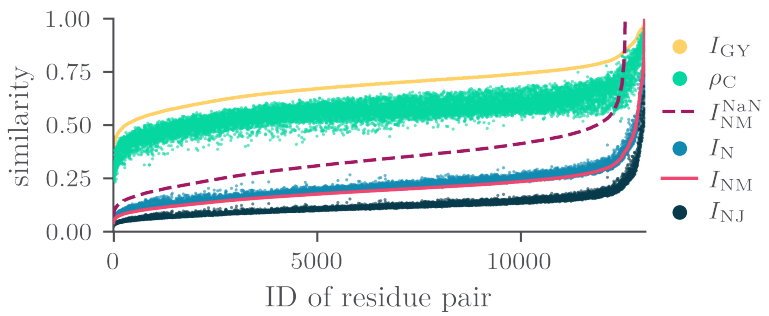


Figure 5.14 | Comparison of various similarity measures, displayed in the order of increasing values of the unnormalized MI $I(X,Y)$ of all residue pairs. Adapted with minor changes from Ref. 3.

remains preferable for the following reason: normalization through the maximum value I_{NM} requires the existence of at least one perfectly correlated data point (i.e. $I_{\mathrm{NM}} \approx 1$) in order to provide a meaningful reference. This is given for proteins, as neighboring residues feature strong spatial correlation due to direct links via the rigid backbone but is generally not guaranteed for other systems. In order to investigate this more closely, we excluded MI values between residue i and its four nearest neighbors, effectively removing the characteristic broad bands discussed earlier. In this case, the maximum value of this modified MI matrix is no longer associated with highly correlated motion, and the close agreement between I_{N} and the newly maximum-normalized MI matrix $I_{\mathrm{NM}}^{\mathrm{NaN}}$ is lost (see Fig. 5.14). This fundamental dependence on the characteristics of the data demonstrates why our entropy-based NMI estimator I_{N} is a more robust and theoretically sound approach. Unlike the maximum normalization, it does not require specific reference correlations and instead provides a mathematically consistent normalization that is applicable across diverse datasets and correlation structures.

5.4 Concluding Remarks

In this chapter, we demonstrated how the combination of contact-based and coordinate-based correlation analysis can provide detailed insights into the complex molecular machinery of proteins. More specifically, by thoroughly investigating the allosteric transition of T4 lysozyme (T4L), we showed how local perturbations in the hinge region propagate to distant sites through cooperative conformational changes.

The initial two-dimensional model relying on the opening coordinate x and the locking coordinate p revealed the two-state behavior and indicated a preferred transition path. Nevertheless, the resulting maximum free energy barrier of approximately $6\,k_{\mathrm{B}}T$ was too low to explain the observed long waiting times in the order of microseconds, suggesting a missing element in the description of the system. Further analysis of all contact distances and all relevant side-chain dihedral angles via MoSAIC revealed that only 34 of the total 687 input coordinates describe the open↔closed transition in a highly correlated fashion. Specifically, the α_1-helix, and even more concretely, its constituent Phe4, were identified as the main hub in this cooperative network of coordinates. Focusing on a subset of coordinates that are particularly suited for a mechanistic analysis, we identified six essential coordinates that are responsible for direct mechanical coupling between the distant hinge and mouth regions, effectively describing a fluctuating transmission network. Employing these coordinates to construct a six-dimensional free energy landscape and following the time evolution of the free energy during different transitions, we found that the energy barriers significantly increased to $\sim (12\text{–}18)\,k_{\mathrm{B}}T$, much higher than the $6\,k_{\mathrm{B}}T$ barrier in the initial model. These substantially increased barriers represent a much more realistic explanation of the observed microsecond waiting times and highlight the cooperative nature of the allosteric transition—where multiple coordinates must align by chance within a narrow temporal window of approximately 10 ns.

Aiming for a complementary global perspective on the allosteric mechanism, we applied the normalized mutual information (NMI) estimator, developed in chapter 4, to the Cartesian C_α-atom positions of T4L. While the MoSAIC analysis revealed the mechanistic details of the transition, the NMI analysis captured global correlated patterns and changes in the rigidity accompanying the transition. The NMI matrix can be interpreted as a residue interaction network, describing how different parts of the protein are affected by the allosteric transition. A detailed analysis of this network by means of eigenvector centrality confirmed the central role of the α_1-helix and particularly of the Phe4 residue.

Beyond the specific case of T4L, we demonstrated how our NMI approach overcomes the limitations of linear correlation measures in multidimensional spaces by providing robust and interpretable results. The suggested entropy-based normalization scheme can be applied independently of the system and of the correlation structure of the data, making this a versatile tool not only for the analysis of MD simulation data. From a computational perspective, the scalability of the NMI estimator [empirically $\mathcal{O}(N \log N)$] allowed for the analysis of the entire 50 μs MD trajectory of T4L without prohibitive computational costs, showing the perspective to analyze longer timescales and larger protein systems in the future.

Physics-Informed Latent Space Models: From Graphs to Gaussian Processes

6

Parts of this chapter are based on our publication:

Recovering Hidden Degrees of Freedom Using Gaussian Processes
G. Diez, N. Dethloff, and G. Stock, *J. Chem. Phys.* **2025** (163) 124105, DOI: https://doi.org/10.1063/5.0282147.

> *Do you want the wrong answer to the right question or the right answer to the wrong question?*
>
> – David Blei

This chapter explores advanced machine learning techniques for the construction of physically more meaningful and temporally coherent free energy landscapes of protein dynamics. Traditional feature extraction methods, such as principal component analysis and classical autoencoders, while powerful, often fail to fully grasp the complex spatio-temporal relationships inherent in molecular dynamics simulations.

We address these limitations in two main subparts of this chapter: First, we explore graph-based representations that naturally capture the protein structure and its spatial molecular relationships. Based on this graph representation, we then propose an autoencoder architecture leveraging graph neural networks. This graph neural network autoencoder effectively captures complex spatial relationships in protein structures through nonlinear local operations between neighboring residues while the graph topology simultaneously regularizes it and prevents overfitting.

Secondly, we will tackle a fundamental limitation of traditional feature extraction methods, that is, the assumption that frames in molecular dynamics simulations are independent and identically distributed, which contradicts the inherent sequential nature of molecular dynamics data. To this end, we will introduce Gaussian processes to incorporate temporal information directly into the latent representation via a variational autoencoder framework. Through time-dependent kernel functions, particularly the Matérn kernel, our proposed model captures temporal correlations between successive frames and is even able to distinguish between dynamically distinct states that appear geometrically identical.

While graph-based approaches explicitly capture the spatial relationships through protein topology, Gaussian processes introduce temporal correlations into the latent space, making both approaches complementary to each other. Combining these two approaches into a unified framework of a Gaussian process variational graph autoencoder in the future can create a powerful tool for the analysis and exploration of

molecular dynamics simulations. This chapter, just like Chapter 4, has a clear focus on methodological development, establishing theoretical and computational foundations for future applications.

6.1 Graph-Based Protein Representations

As described in the Methods part (Sec. 2.3.3), feature extraction methods, such as PCA and AEs, can be used to reduce the dimensionality of the data. While this has proven to be useful in countless applications concerning MD data, these feature extraction techniques do not explicitly capture the spatial relationships between the different features—namely, which contact distances are in physical proximity and, therefore, correlated.[1]

Researchers in computer vision are faced with similar challenges when dealing with images, where pixels exhibit strong spatial correlations and local patterns in the image are crucial for, e.g., object recognition.[306] In images, this problem can be addressed by using, e.g., convolutional neural networks, that apply convolutional filters that are capable of capturing local patterns in the data.[307,308] Such convolutional architectures are applicable due to the regular grid structure of images, where every pixel has a well-defined set of spatial neighboring pixels.[i] While exploiting the spatial structure of images makes these models much more powerful compared to fully connected networks, transferring this idea to proteins is not straightforward. Unlike images, proteins lack a regular structure of features—residues that follow shortly after each other in the sequence may be distant in the three-dimensional structure, while residues far apart in the sequence may be in direct physical contact through the protein fold. Similarly, when using contact distances as features, it remains unclear how to best arrange them in such a way that convolutional filters could be applied. While one could potentially organize residues in such a way that their (contact-) distances are spatially correlated within the data matrix (e.g., by adapting MoSAIC), this strategy would require computing all pairwise distances between these residues, resulting in a quadratic number of features, which is computationally prohibitive for larger proteins. Moreover, even if an optimal spatial arrangement could be found, the relevant spatial relationships in proteins are inherently dynamic, such that the spatial arrangement may change as the protein folds or changes conformations.

[i] Usually, this is referred to as *inductive biases*. In principle, a simple multilayer feedforward network serves as a universal function approximator and can learn local patterns in images.[309] Nevertheless, such a network would be extremely inefficient in this case because it would first have to learn the concept of spatiality that convolutional neural networks already have built-in by design.

6.1.1 From Protein Structures to Graphs

Fortunately, proteins can naturally be represented as graphs, where nodes correspond to individual residues (typically represented by their C_α-atoms), and edges represent the distance between these residues.[ii] Edges can even be temporally removed entirely if the contact is not formed in a given interval.

A major advantage of this representation is of computational nature: unlike data matrices, graphs are not restricted to a regular structure. This allows us to include only the most relevant edges, such as the edges that correspond to contact distances, which avoids the quadratic scaling with

[ii] Typically, the inverse of the distances is used as edge weights so that closer residues have a higher weight.

regard to the number of residues. As illustrated in Fig. 6.1, this graph representation preserves the key topological and geometric properties of the protein structure at a given time step.

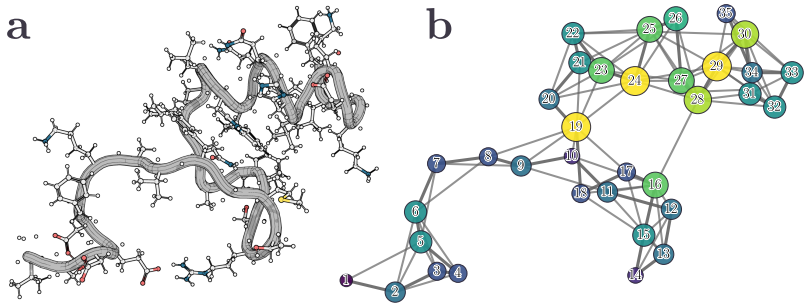


Figure 6.1 | Example of a graph representation of a protein. Due to simplicity, we chose one snapshot of HP35 and showed its structure in (a). Exactly this structure can be represented as a graph in (b), where each node represents one residue and the edges indicate a contact formed between the residues. The nodes are colored and sized according to their degree.

6.1.2 Graph Neural Networks and Autoencoders

Graph neural networks (GNNs) are a special class of neural networks that are specifically designed for graph-structured data.[310,311] In contrast to the traditional fully connected networks, GNNs do not consist of layers that are fully connected to each other but rather operate on the graph structure itself. This way, the model leverages the spatial structure of the data.

Similar to in Sec. 5.2.2, we define a graph as a tuple $G = (V, E)$,[iii] where V is the set of nodes and E is the set of edges. For every time step, for each node $u \in V$, we define[311]

- $\boldsymbol{h}_u^{(k)} \in \mathbb{R}^d$ as the $d-$dimensional node feature vector of node u at layer k,
- $\boldsymbol{e}_{uv} \in \mathbb{R}^p$ as the $p-$dimensional edge feature vector between nodes u and v, and
- $\Gamma(u) = \{v \in V \mid (u, v) \in E\}$ as the set of neighbors of node u.

[iii] In the case of MD trajectories, this graph is time-dependent $G_t = (V, E_t)$, with t being the time step.

In our case, the node feature vector $\boldsymbol{h}_u^{(k)}$ may correspond to physical properties like, e.g., the backbone/side chain dihedral angles of residue u, while $\boldsymbol{e}_{uv}$ corresponds to the (inverse) contact distance between residues u and v—quantities that directly characterize the conformation of the protein.

At the very heart of GNNs is the process of *message passing*,[312] where each node iteratively updates its node feature by exchanging information with its neighbors. This process can be described in three steps:[313]

1. **Message computation:** Each node $v \in \Gamma(u)$ computes a message to node u based on its own feature vector $\psi\left(\boldsymbol{h}_u^{(k)}, \boldsymbol{h}_v^{(k)}, \boldsymbol{e}_{uv}\right)$.
2. **Aggregation:** Node u collects and combines all incoming messages from its neighbors $\Gamma(u)$ using a permutation-invarant aggregation operation $\bigoplus$, which is typically simply a sum or mean operation.
3. **Update:** Node u updates its feature vector using both its current feature vector $\boldsymbol{h}_u^{(k)}$ and the aggregated messages from its neighbors

$$\boldsymbol{h}_u^{(k+1)} = \phi\left[\boldsymbol{h}_u^{(k)}, \bigoplus_{v \in \Gamma(u)} \psi\left(\boldsymbol{h}_u^{(k)}, \boldsymbol{h}_v^{(k)}, \boldsymbol{e}_{uv}\right)\right].$$

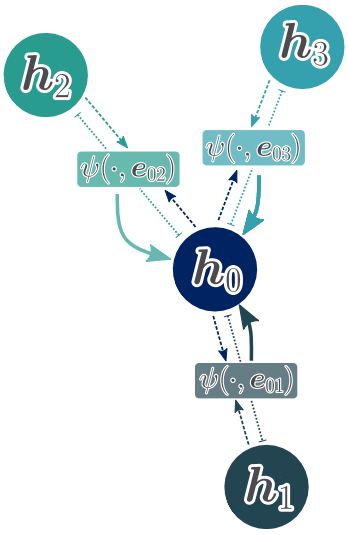


Figure 6.2 | Message passing mechanism showing node 0 receiving messages from its neighborhood $\Gamma(0) = \{1, 2, 3\}$ about their node vector $\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3$ via message the functions $\psi(\boldsymbol{h}_u, \boldsymbol{h}_v, \boldsymbol{e}_{uv})$.

The message function ψ and the update function ϕ are typically implemented as learnable neural networks. By stacking multiple message-passing layers, local information can propagate across spatially distant nodes, creating a physically realistic model of how structural changes are transported through the protein. For example, in the case of T4L, a change in the χ_4 dihedral angle initiates a cooperative conformational cascade that propagates from the hinge region to the mouth region through a series of contact changes—precisely the kind of information transmission that GNNs naturally capture via their message passing framework. While traditional neural networks treat features of proteins as independent elements in high-dimensional vector spaces—making the identification of patterns much more difficult—GNNs are explicitly regularized by the physical structure of the protein, which allows them to learn more efficiently and avoid overfitting.

Similarly to traditional fully connected networks, GNNs can adapt a variety of architectures, including autoencoder architectures (see Sec. 2.3.3) that learn a low-dimensional representation while preserving the essential structural elements and relationships of the graph.[314] In such a graph neural network autoencoder (GNN-AE), the graph has to be coarse-grained into a sequence of progressively coarser graphs and finally into a latent embedding $G^{(0)} \to G^{(1)} \to G^{(2)} \to \ldots \to G^{(L)} \to z$. This coarse-graining is achieved through *pooling* operations. Hierarchical pooling methods, such as Self-Attention Graph Pooling,[315,316] iteratively reduce graph size by computing attention[317] scores for each node and then retaining only the top-k% important nodes in each pooling step.[iv] Finally, the last pooling step transforms the graph into a latent representation $z \in \mathbb{R}^d$.[318]

[iv] Strictly speaking, these pooling operations do not perform coarse-graining, but rather a form of downsampling, where the most important nodes are retained, and the rest are discarded. However, since we combine them here with several layers of message passing, which transport information across the graph, we can interpret the overall process as some kind of coarse-graining. We explain the Self-Attention Graph Pooling in more detail in SI, Sec. D.1

For the reconstruction of the contact distances from the latent representation $z \in \mathbb{R}^d$, we use a fully connected network, conditioned on these distances that are forming a contact in the given time step. A schematic representation of the complete network is shown in Fig. 6.3.

Graph representations of proteins and the applications of GNNs have proven effective in various applications. For example, Jha *et al.*[319] used GNNs to predict the interactions between different proteins, and Smith *et al.*[320] used GNNs with attention mechanisms to identify druggable binding sites in proteins. Another promising approach is residue interaction networks proposed by Franke *et al.*,[321] which use a graph representation

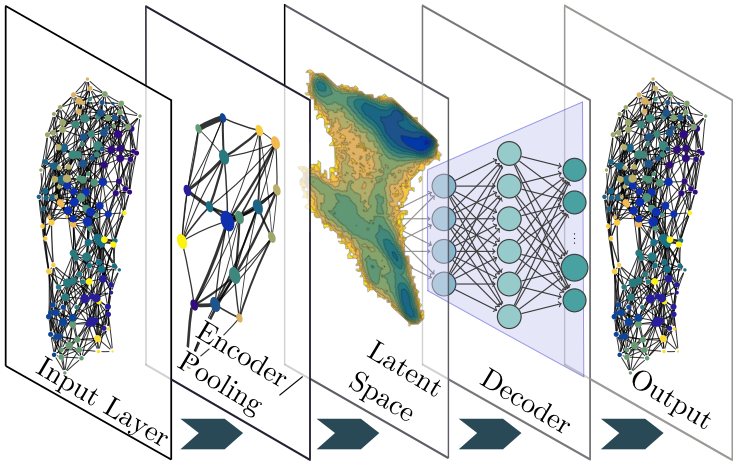


Figure 6.3 | Schematic representation of a GNN autoencoder. The graph neural network operates directly on the graph, which encodes the protein structure (here T4L). For the decoding step, fully connected layers are used.

of each snapshot in an MD trajectory to compute a centrality score (see Fig. 5.11) for each residue and subsequently employ a conventional autoencoder modification called encodermaps[113] to represent the conformation of the protein based on these centrality scores. While this approach used graph representations to encode each time step of the MD trajectory, it ultimately relies on traditional fully connected architecture and does not leverage the regulative power of the GNNs. In contrast to the existing approaches, our GNN-AE framework represents a novel combination of graph neural network autoencoders specifically designed to encode individual time steps of MD trajectories into the latent space.

6.1.3 Application: T4L

In order to demonstrate the power of this architecture, we apply it to the 556 C_α-distances between the residues identified in Fig. 5.4. Using the C_α-distances instead of the contact distances captures the conformation of the backbone in more detail and will allow us to reconstruct the overall protein structure from the latent representation. We note that we are interested in the method development part, which is why we employ only two-dimensional embeddings for the sake of better visualization and comparison. For practical applications, such as, e.g., in Markov state modeling, accurate modeling would most likely require higher-dimensional embeddings with latent dimensions around $d \lessapprox 10$.[78,81]

As a baseline, we compare our GNN-AE model to a traditional PCA. The PCA is applied to the same 556 C_α-distances, and the free energy landscape ΔG from the projection onto the first two principal components are shown in Fig. 6.4 a. The PCA projection reveals two main clusters separated along the first principal component, corresponding to the open ($z_1 \gtrapprox 0$) and closed ($z_1 \lessapprox 0$) conformation (this can, e.g., be verified by coloring the data points by the locking distance, as shown in SI, Fig. D.3). The second principal component, z_2, further separates the open conformation into two subclusters, which can be linked to motion within the MoSAIC cluster 4 (compare Sec. 5.1.1).[322] Following the idea of N. Dethloff in Ref. 322 to visualize transitions occurring within the MoSAIC clusters within the latent embedding, we identified these transitions using the breakpoint analysis that we employed in Ref. 4.[323,324] The detailed methodology and results are provided in SI, Sec. D.2, with results for MoSAIC cluster 4 shown in SI, Fig. D.3. The exact architecture that we used for the GNN-AE model, as well as training parameters, can be found in SI, Sec. D.2.4.

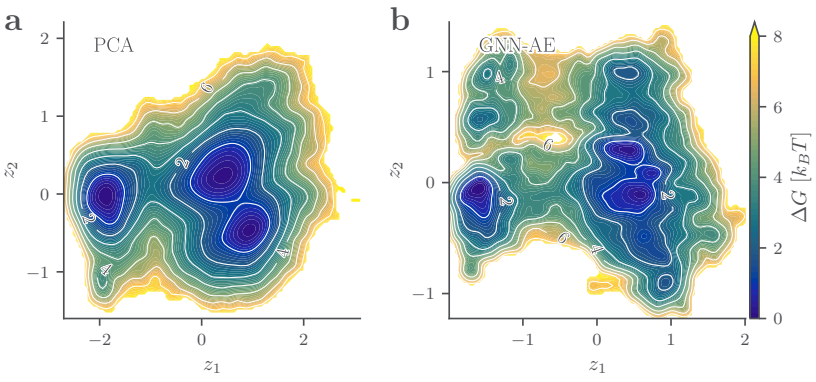


Figure 6.4 | Comparison of the two-dimensional embeddings of the 556 C_α-distances using PCA (a) and GNN-AE (b).

Comparing the PCA results to the GNN-AE model, trained by N. Dethloff,[322] we observe in Fig. 6.4 b that the GNN-AE captures significantly greater structural complexity in its latent representation. Moreover, our analysis (SI, Sec. D.2.3) demonstrates that the additional basins in the GNN-AE embedding are physically meaningful, with the identified free energy basins corresponding to distinct conformation states as validated based on the MoSAIC clustering results.

We believe that this enhanced resolution results from how information propagates through a GNN compared to PCA. In a GNN, the features (in our case, the C_α-distances)[v] are not independent of each other, but changes in one distance propagate locally to neighboring residues and thus influence the distances in their physical vicinity. This makes the GNN-AE model more susceptible to subtle changes in the protein structure, eventually resulting in a more detailed latent representation. That being said, the GNN-AE requires the full representation of a protein structure (or at least a fully connected subpart of it) to effectively leverage the spatial relationships encoded in the graph. In contrast, methods like PCA or AE architectures operating on fully connected layers can work on an arbitrary (sub)set of features, which might be advantageous in cases where an extensive feature selection step was performed beforehand.[4,101–103] Eventually, the nonlinearities in the GNN-AE may also help to "crunch" the information into a more compact representation, but without proper regularization—as imposed by the graph structure—this quickly leads to overfitting and nonphysical representations, as shown for several different architectures in Ref. 322. Furthermore, the regularization through the graph structure of the protein makes the representations learned by the GNN-AE more robust and reproducible, showing only minor variations when trained with different hyperparameters or smaller architectural changes.

[v] The node features, i.e., the backbone dihedral angles, had only a minor influence on the GNN-AE representation.

Navigating the Latent Space

As an additional means to explore the latent space, we can exploit the generative capabilities of the GNN-AE model and generate structures of the proteins' backbone along a path in the latent space. First, we need to compute a path in the latent space that is defined by a starting point $\boldsymbol{z}_{\text{start}}$ and an ending point $\boldsymbol{z}_{\text{end}}$ and respects the underlying free energy landscape. We recognize that this problem can be framed mathematically as an optimal transport problem,[325] which would provide a rigorous framework for finding a minimal-cost mapping between two probability distributions given an underlying topology (e.g., via the Wasserstein distance). However, since we are only interested in a proof of concept, we follow a simpler approach here.

To this end, we first approximate the probability density p_{z_1,z_2} by constructing a two-dimensional histogram over the latent space coordinates. Each histogram bin (n, m) corresponds to a small region in the latent space with average probability density $p_{z_1,z_2}(n, m)$. Next, we construct an undirected graph where each node corresponds to a bin with non-zero probability density $p_{z_1,z_2}(n, m) > 0$. The edges connecting the nodes i and j carry the weight w_{ij}, which is inversely proportional to the sum of

their local densities

$$w_{ij} = \frac{1}{\rho_i + \rho_j},$$

where ρ_i and ρ_j represent the histogram densities at adjacent bins. This weighting scheme effectively converts the free energy landscape into a cost landscape, where high-density (or low free energy) regions are easier to cross, and pathways through low-density regions are penalized. Finally, we can use Dijkstra's algorithm[326] to approximate the shortest path between z_{start} and z_{end}, respecting the underlying free energy.

Building on this path, we can now generate samples along it by, e.g., extracting N equidistant points along the path $\boldsymbol{z}_i$, where $i = 1, \dots, N$. These generated points $\boldsymbol{z}_i$ can then be fed into the decoder $f_{\text{D},\boldsymbol{\phi}}(\boldsymbol{z})$ [compare Eq. (2.19)] of our trained GNN-AE model, yielding 556 reconstructed C_α-distances. These C_α-distances are then used to generate harmonic distance restraints for PyRosetta.[327,328] Using the Rosetta Energy Function 2015,[329] an energy minimization is performed, which balances the distance restraints resulting from the GNN-AE model with physical energy terms including van der Waals interactions, hydrogen bonding, and electrostatics to find physically realistic protein conformations along the transition pathway. We found that this works very well for stable proteins such as T4L, as demonstrated in SI, Fig. D.6, but struggles for folding proteins like HP35, where secondary structures completely fold or unfold. Such transitions that involve significant secondary structure formation or unfolding may require a more sophisticated approach than our current restraint-based approach.

To illustrate this approach, we consider the open→closed allosteric transition of T4L and manually define a starting and ending point in the latent space. The resulting pathway is shown in Fig. 6.5 a, along we extracted three points (represented by the circles) and fed them into the decoder of the GNN-AE model, which generated the restraints for the corresponding C_α-distances. Employing PyRosetta, we obtained the three structures shown in Fig. 6.5 b, yielding a realistic picture of the backbone dynamics during the transition.

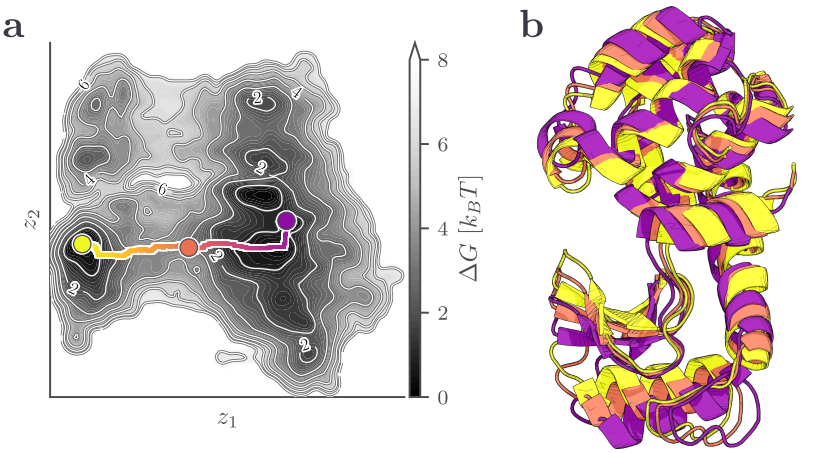


Figure 6.5 | As an example, we consider the open→closed transition of T4L. We manually define a starting and ending point in the latent space (a) and calculate a pathway between these two points. Here, we extracted three points (represented by the circles) and fed them into the decoder of the GNN-AE model, which generated the corresponding C_α-distances. Using PyRosetta,[328] we used these distances as restraints to generate the corresponding protein structures shown in (b).

6.2 Temporal Continuity vs. the i.i.d. Assumption

While the above-suggested approach of using a GNN-AE model substantially increases the resolution of the latent space compared to PCA, it still treats each MD frame as an independent snapshot. The assumption that each frame is independent and identically distributed (i.i.d.) plays a crucial role in machine learning (not only for MD data) since it enables batch-wise and parallelized training of neural networks. Batch-wise training usually implies that the input data is randomly shuffled into smaller batches, typically consisting of several tens to hundreds of frames, which are, depending on the capabilities of the GPU, processed in parallel. Moreover, in MD data, the random shuffling has another advantage: randomly shuffling the frames breaks the temporal correlations between consecutive frames and thus prevents the gradient updates from being dominated by temporally limited trends such as, e.g., metastable state lifetimes. This is because all (randomly shuffled) mini-batches now approximately represent the whole data distribution rather than localized temporal segments, which eventually results in a more stable and efficient training process.

So, while it is mathematically and computationally convenient, the i.i.d. assumption (intentionally) fails to capture the temporal dependencies in MD simulations and, thus, does not account for the time evolution of the system, where each conformation directly depends on previous states. In the rest of the chapter, we will demonstrate that explicitly incorporating these temporal dependencies into the latent space can significantly improve the quality of the latent representation and even allow us to distinguish between dynamically distinct states that appear geometrically identical due to missing degrees of freedom.

6.3 Gaussian Processes

The temporal evolution of the protein in MD simulations is inherently sequential, meaning it is continuous and correlated. Thus, we need a modeling framework that allows us to capture these temporal dependencies explicitly.

Gaussian Processes (GPs) can do exactly this. GPs provide a flexible and probabilistic framework for modeling functions without committing to a fixed functional form.[330] Unlike traditional regression methods, which require the specification of a particular functional form (e.g., linear, polynomial, or exponential[158]), GPs do not assume any specific equation to describe the data. Instead, a GP defines a probability distribution over functions, which are constrained only by the choice of a covariance function (or kernel) that reflects our prior beliefs about the correlation structure in the data (e.g., how strongly data points at different time steps are correlated). Formally, we write[331]

$$f \sim \mathrm{GP}(m, k),$$

which means that the function f is distributed as a GP with mean function m and covariance function (or kernel) k.

Example: Radial Basis Function Kernel

To illustrate how this works in practice, we consider the simple case of the radial basis function (RBF) kernel—or Gaussian kernel—which is defined as

$$k_{\mathrm{RBF}}(x, x'; l) = \exp\left(-\frac{\|x - x'\|^2}{2l^2}\right),$$

where l controls the length scale of the kernel. Intuitively, we can think of this as how fast the correlations between subsequent function values $f(x_i)$ and $f(x_j)$ decay as the distance of their respective input values x_i and x_j increases. To generate sample functions, we first need to select a set of input points, in this case, $x_i \in [0, 10]$ for $i = 1, \ldots, N = 30$ equidistant x-values. For these input values, we then compute the covariance matrix $\boldsymbol{K}_{ij} = k_{\mathrm{RBF}}(x_i, x_j; l)$ for all pairs of x_i, x_j. This results in a 30×30 matrix that encodes the expected similarity between the function values at all pairs of input points. Finally, we can now draw samples from the multivariate Gaussian with zero mean covariance $\boldsymbol{K}$: $f \sim \mathcal{N}(0, \boldsymbol{K})$, where each sample is a 30-dimensional vector representing the function values at the 30 x-positions (see Fig. 6.6). Usually, the mean function $m(x)$ is set to zero for simplicity because the main flexibility and expressiveness of GPs stem from the kernel.[331]

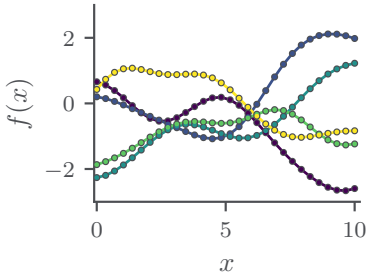


Figure 6.6 | Five functions drawn from a Gaussian Process with $k_{\mathrm{RBF}}(x, x'; l = 2)$. The actual function values are denoted by the circles; solid lines only for the sake of visualization.

From Prior to Posterior: Conditioning on Observations

For more practical applications, we want to condition the GP on observations, such as MD frames, in order to make predictions about function values at unseen input points. In Bayesian terminology, this means that we want to update our initial beliefs (i.e., the prior) in light of the data, resulting in a posterior distribution over functions that both respect the prior assumptions and the observed data. We denote the known function values at observed input points as $\boldsymbol{f}_{\mathrm{o}}$ and $\boldsymbol{f}_{\mathrm{p}}$ are the function values at input points $\boldsymbol{x}_{\mathrm{p}}$ that we want to predict. Since we model $\boldsymbol{f}_{\mathrm{o}}$ and $\boldsymbol{f}_{\mathrm{p}}$ as resulting from the same GP, they are jointly Gaussian distributed

$$\begin{bmatrix} \boldsymbol{f}_{\mathrm{o}} \\ \boldsymbol{f}_{\mathrm{p}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_{\mathrm{o}} \\ \boldsymbol{\mu}_{\mathrm{p}} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{\mathrm{oo}} & \boldsymbol{\Sigma}_{\mathrm{op}} \\ \boldsymbol{\Sigma}_{\mathrm{po}} & \boldsymbol{\Sigma}_{\mathrm{pp}} \end{bmatrix}\right),$$

where we use the shorthand notation $\boldsymbol{\mu}_k = m(\boldsymbol{x}_k)$ and $\boldsymbol{\Sigma}_{kl} = k(\boldsymbol{x}_k, \boldsymbol{x}_l)$ for convenience ($k, l \in \{\mathrm{o}, \mathrm{p}\}$). Assuming that we want to predict the values $\boldsymbol{f}_{\mathrm{p}}$ at the points $\boldsymbol{x}_{\mathrm{p}}$, we can obtain the posterior distribution through conditioning on the observations

$$\boldsymbol{f}_{\mathrm{p}} | \boldsymbol{f}_{\mathrm{o}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathrm{p|o}}, \boldsymbol{\Sigma}_{\mathrm{p|o}}),$$

where

$$\begin{aligned}\boldsymbol{\mu}_{\mathrm{p|o}} &= \boldsymbol{\mu}_{\mathrm{p}} + \boldsymbol{\Sigma}_{\mathrm{po}}\boldsymbol{\Sigma}_{\mathrm{oo}}^{-1}(\boldsymbol{f}_{\mathrm{o}} - \boldsymbol{\mu}_{\mathrm{o}}),\\ &= \boldsymbol{\Sigma}_{\mathrm{po}}\boldsymbol{\Sigma}_{\mathrm{oo}}^{-1}\boldsymbol{f}_{\mathrm{o}}\\ \boldsymbol{\Sigma}_{\mathrm{p|o}} &= \boldsymbol{\Sigma}_{\mathrm{pp}} - \boldsymbol{\Sigma}_{\mathrm{po}}\boldsymbol{\Sigma}_{\mathrm{oo}}^{-1}\boldsymbol{\Sigma}_{\mathrm{op}},\end{aligned}$$

where we assumed $\boldsymbol{\mu}_{\mathrm{k}} = 0$ and used standard properties of conditioning in multivariate Gaussian distributions.[131] The posterior mean $\boldsymbol{\mu}_{\mathrm{p|o}}$ provides optimal predictions for the function values at the prediction points $\boldsymbol{x}_{\mathrm{p}}$, while the diagonal of the posterior covariance $\boldsymbol{\Sigma}_{\mathrm{p|o}}$ quantifies prediction uncertainty.

[vi] we assume additive i.i.d. Gaussian noise.

In case the observations $\boldsymbol{f}_{\mathrm{o}}$ are noisy,[vi] GPs can readily account for this by modifying the covariance structure: since the noise is assumed to be independent, each observation $f(x)$ has an additional covariance with itself only $\boldsymbol{\Sigma}_{\mathrm{oo}}^{\mathrm{noise}} = \boldsymbol{\Sigma}_{\mathrm{oo}} + \sigma^2\boldsymbol{I}$, where σ^2 is the noise variance, and $\boldsymbol{I}$ is the identity matrix. The remaining equations remain structurally unchanged, with only this modified covariance matrix.

To demonstrate the effectiveness of GP regression, we drew values of the function $f(x) = \sin(x) + \frac{x}{2}$ for five randomly sampled x-values in the interval $x \in [-\frac{\pi}{2}, \frac{5\pi}{2}]$ and added Gaussian noise with $\sigma^2 = 0.2$. Again, we assumed the same covariance structure of $k_{\mathrm{RBF}}(x, x'; l = 2)$ as above and conditioned the GP on these five noisy observations $\boldsymbol{f}_{\mathrm{o}}$. The resulting posterior mean $\boldsymbol{\mu}_{\mathrm{p|o}}$ and the 68% and 95% confidence intervals are shown in Fig. 6.7. We see that the GP successfully captures the underlying function despite the sparse and noisy observations—especially in the core region around these observations. Moreover, the confidence intervals become wider in regions where the GP regression can not rely on observations, such as, e.g., for values $x < \pi/2$.

GPs are a flexible tool for modeling sequential data, such as, e.g., MD simulations, but we note that GPs can be applied to any kind of data, which allows the definition of a kernel that captures the underlying correlation structure. Examples include computer vision,[332] genomics and molecular discovery,[333,334] robotics,[335] climate modeling,[336] and finance.[337]

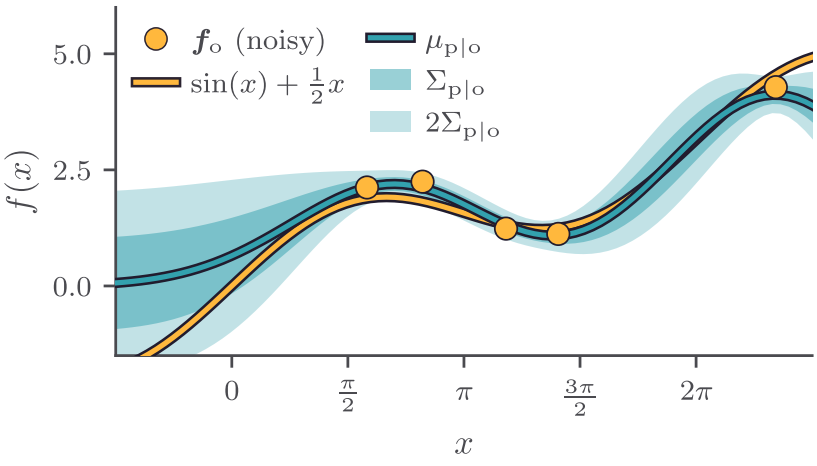


Figure 6.7 | Gaussian process regression on five noisy realizations of $f(x) = \sin(x) + \frac{1}{2}x$. The confidence intervals at a given x'-value indicate that there is a 68% and 95% probability that the function value $f(x')$ of any sampled function from the posterior distribution lies within the corresponding interval.

6.4 VAEs with GP Priors

While GPs are very effective for modeling temporal correlations directly in the observed space, we want to use them to encode temporal correlations directly into the latent space. This naturally leads us to the framework of variational autoencoders (VAEs, see Sec. 2.3.3),[112] which, just like GPs, operate within a Bayesian probabilistic framework. While GPs provide the means to model dynamics via prior beliefs about temporal correlations, VAEs use variational inference to approximate the posterior distribution over latent variables. As they both share the same probabilistic foundation, we can seamlessly incorporate the temporal structure of our MD data by using GPs as priors in the latent space of a VAE.

In standard VAEs, the latent variables $\boldsymbol{z}$ are regularized through a simple factorized (i.i.d.) Gaussian prior, that assumes independence across both latent dimensions and time steps:

$$p(\boldsymbol{z}) = \prod_{i}^{N} \mathcal{N}(\boldsymbol{z}_i|0, \boldsymbol{I})$$

As described in Sec. 6.2, this i.i.d. assumption contradicts the sequential nature of MD data. The rationale for replacing this factorized prior with a GP is straightforward: introducing a GP instead of a simple i.i.d. Gaussian prior enables us to embed temporal correlations directly into the latent space through time-dependent kernels $k(t, t')$. When working with temporal kernels, the correlation decay, specified through the kernel's length scale l, directly translates into memory decay. Consequently, this means that temporal correlations in the MD trajectory are expressed as spatial proximity in the latent space: frames close in time are considered similar—and their latent representations $\boldsymbol{z}$ are therefore located close to each other—while temporally distant ones exhibit weaker correlations.

6.4.1 Kernel Choice: The Matérn Kernel

The choice of a suitable kernel is crucial for embedding physically meaningful constraints into our Gaussian Process Variational Autoencoder (GP-VAE) framework, as the kernel's characteristics and timescales directly affect the position of a data point in the latent space. Since the dynamics of our system are Markovian in full space, we seek to preserve a simpler Markov-like structure also within our compressed latent representation. To this end, the Matérn kernel emerges as particularly well-suited for this application due to its close connection to the Ornstein-Uhlenbeck process, which was introduced as a mathematical model of the velocity of a particle undergoing Brownian motion. The choice of the Matérn kernel is physically motivated: it inherently captures the continuous, yet stochastic nature of the conformational dynamics of proteins. Simultaneously, it preserves the desired Markovian structure. The Matérn kernel is given as

$$k_\nu(t, t'; l) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{|t - t'|}{l} \right)^{\nu} K_\nu \left(\sqrt{2\nu} \frac{|t - t'|}{l} \right), \tag{6.1}$$

where ν specifies the smoothness of the kernel function, l is the characteristic length scale (i.e. memory timescale), and K_ν is a modified Bessel

function and $\Gamma(\nu)$ is the Gamma function.[330] Crucially, the Matérn kernel allows to independently control two different aspects of memory through the parameters ν and l. Through the smoothness parameter ν, we can directly control the structure of memory, that is, the order of the underlying Markov process, while l controls the rate of memory decay.

Here, we are considering three cases for $\nu = 1/2, 3/2, \infty$:

$$k_{\nu=1/2}(t,t';l) = \exp\left(-\frac{|t-t'|}{l}\right),$$
$$k_{\nu=3/2}(t,t';l) = \left(1+\frac{\sqrt{3}|t-t'|}{l}\right)\exp\left(-\frac{\sqrt{3}|t-t'|}{l}\right),$$
$$\lim_{\nu\to\infty} k_\nu(t,t';l) = \exp\left(-\frac{|t-t'|^2}{2l^2}\right).$$

In the first case, where $\nu = 1/2$, the covariance function simplifies to an exponentially decaying memory, which we find for first-order Markov processes, where the future state only depends on the current state. This is also the covariance function of the Ornstein-Uhlenbeck process. For higher half-integer values $\nu = 3/2, 5/2, \ldots$, the Matérn kernel corresponds to higher-order Markov processes, e.g., for $\nu = 3/2$, we obtain a second-order Markov process, for $\nu = 5/2$ a third-order and so forth. As we employ higher ν-values, the realizations become smoother, indicating that more information about the past is retained in the memory of the stochastic process. For $\nu \to \infty$, the Matérn kernel converges to the RBF (Gaussian) kernel—that we discussed earlier—which corresponds to an infinitely smooth process with no Markovianity at all.

On the other hand, large values of l correspond to a slow decay of memory, meaning that the current state is highly predictive of future states. Small values of l correspond to a fast decay of memory, meaning weakly predictive power of future states given the current state. This makes the length scale l a crucial parameter for capturing the relevant timescales of protein dynamics, where different dynamics occur across vastly different timescales. Fast local fluctuations in the picosecond regime to slow allosteric transitions in the microseconds to milliseconds regime: the length scale l must be appropriately chosen to match the dynamical processes of interest

Fig. 6.8 illustrates the effects of both parameters. The left panels show how different values of $l \in \{1, 3, 10\}$ control the width of the memory decay, while different values of $\nu \in \{1/2, 3/2, \infty\}$ affect the shape of memory decay. The corresponding GP realizations on the right show how these two independent effects combine: for any fixed length scale l, a higher ν-value results in smoother trajectories, while for any given ν-value, a larger l leads to slower variations and thus increased predictive power of the current state.

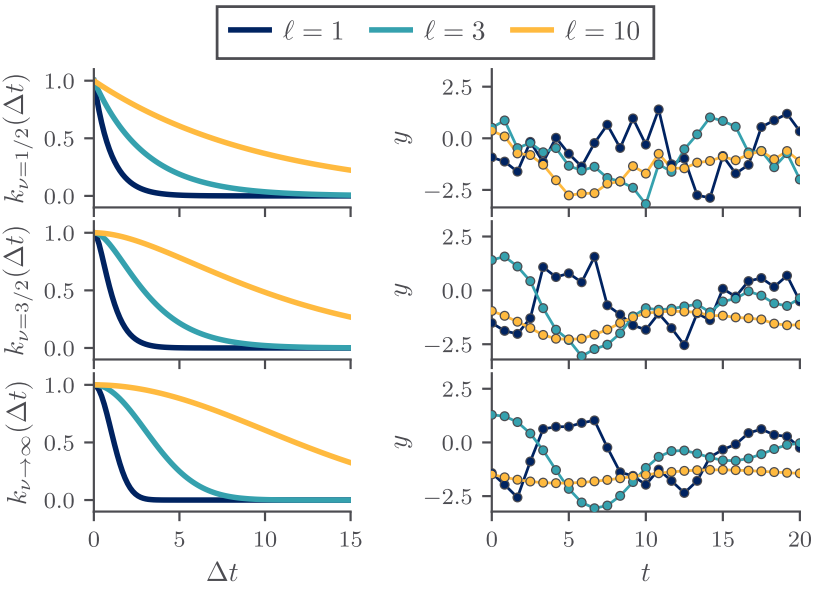


Figure 6.8 | Matérn kernel properties and Gaussian process realizations. Left: Kernel correlation functions $k_\nu(t, t_0; l)$ for different length scales $\in \{1, 3, 10\}$ and smoothness parameters $\nu \in \{1/2, 3/2, \infty\}$. Right: Sample realizations from GP priors using the corresponding Matérn kernels.

6.4.2 Changing the Objective

Let us briefly recap the core idea of a classical latent variable model. The generative model is given as

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}), \tag{6.2}$$

where the likelihood $p(\boldsymbol{x}|\boldsymbol{z})$ encodes how a specific latent variable $\boldsymbol{z}$ leads to an observation $\boldsymbol{x}$. Hence, it acts as a "recipe" for reconstructing the protein's full conformation $\boldsymbol{x}$ given the latent representation (or collective variable) $\boldsymbol{z}$. As described in Sec. 2.3.3, this leads to the ELBO objective function of a VAE, which is given as

$$\ln p(\boldsymbol{x}) \geq \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}} = \underbrace{\left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})}}_{\text{Reconstruction}} - \underbrace{D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p(\boldsymbol{z})\right]}_{\text{Regularization}}.$$

However, now we want to replace the factorized prior $p(\boldsymbol{z}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{z}_i|\boldsymbol{0}, \boldsymbol{I})$ with a GP that explicitly captures the temporal correlations through the time-dependent kernel $k(t, t')$. This modification transforms the generative model in Eq. (6.2) to

$$p(\boldsymbol{x}, \boldsymbol{z}|t) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}|t). \tag{6.3}$$

The crucial difference lies in the temporal conditional of the prior $p(\boldsymbol{z}|t)$, where the latent variables now follow a Gaussian Process: $\boldsymbol{z} \sim \mathrm{GP}[0, k_\nu(t, t')]$. This modifies the standard VAE ELBO objective function to

$$\mathcal{L} = \left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})} - \beta D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x}, t) \,\|\, p(\boldsymbol{z}|t)\right], \tag{6.4}$$

where we compare the temporally-aware posterior distribution $q(\boldsymbol{z}|\boldsymbol{x}, t)$ against the GP prior $p(\boldsymbol{z}|t)$. Similarly to the β-VAE,[338] we also introduce a weighting factor β to control the impact of the regularization term on the overall objective.

However, this apparent straightforward conceptual modification introduces significant computational challenges that distinguish GP-VAEs

from standard VAEs. For large data sets, such as MD simulations, where we are often faced with $10^5 - 10^6$ frames, the $\mathcal{O}(N^3)$ computational complexity of the GP prior becomes computationally prohibitive.[vii] Moreover, unlike standard VAEs that can be trained in batches, GP-VAEs require the full trajectory at once during training because all frames are temporally connected through the kernel matrix.

[vii] Computing the full kernel matrix $\boldsymbol{K}_{NN} \in \mathbb{R}^{N\times N}$, for all N data points requires $\mathcal{O}(N^2)$ operations, and its inversion leads to the cubic complexity $\mathcal{O}(N^3)$.

In order to overcome these computational bottlenecks, several key contributions of different authors have led to the development of sparse GP methods that 1.) scale to very large datasets[339,340] and 2.) allow the integration of these sparse GP techniques into the VAE framework.[133,332,341] These methods use inducing points to approximate the full GP prior and variational inference techniques to reduce the computational complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(BM^2 + M^3)$, where M is the number of inducing points and B denotes the batch size.[133] The resulting sparse GP-VAE objective function is given as

$$\mathcal{L}_{\text{GP-VAE}} = \underbrace{\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \rangle_{z\sim q(\boldsymbol{z}|\boldsymbol{x})}}_{\text{Reconstruction}} - \beta \underbrace{\left[\text{CE}\left[\mathcal{N}(\boldsymbol{m},\boldsymbol{B}) \| \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\sigma}}^2)\right] - \frac{\overbrace{b}^{\text{sparse}}}{N}\mathcal{L}_{\text{H}} \right]}_{\text{GP regularization}}. \tag{6.5}$$

The GP posterior $q(\boldsymbol{z}|\boldsymbol{x}, t) = \mathcal{N}(\boldsymbol{m}, \boldsymbol{B})$ captures temporal structures through its mean $\boldsymbol{m}$ and posterior covariance matrix $\boldsymbol{B}$. This is in contrast with the standard factorized VAE posterior $q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\sigma}}^2)$, which treats each time step independently through its diagonal covariance structure. Both posteriors are brought together and combined through the cross-entropy (CE), which regularizes the encoder to maintain consistency with the temporal correlations induced by the Matérn kernel. Additionally, the sparse approximation term $b/N\mathcal{L}_{\text{H}}$ covers terms resulting from the introduction of inducing points, which are crucial to enable the scaling of this framework to large datasets.[340] The complete mathematical derivation of Eq. (6.5), including the recipes to compute all required quantities, can be found in SI, Sec. D.3.

6.4.3 Proof of Concept: Toy Model

Having established the theoretical foundation of GPs and their suitability for modeling temporal correlations in the latent space—especially using the Matérn kernel—we now demonstrate the practical application of our GP-VAE framework. We aim to isolate and study the effects of non-Markovian behavior that arise when important degrees of freedom are missing in the input data. Such scenarios commonly occur when analyzing MD simulations by means of dimensionality reduction, either through suboptimal feature selection or insufficient latent representations, which can introduce memory effects that violate the Markov assumption. By constructing an analytical toy potential, where we can systematically introduce non-Markovian effects through projection artifacts, we can rigorously study the ability of our framework to recover these hidden degrees of freedom.

To this end, we simulate a three-dimensional trajectory $\boldsymbol{x}_t$ using the overdamped Langevin equation

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{\Delta t}{\gamma}\nabla\Phi(x,y,z) + \sqrt{\frac{2k_\mathrm{B}T\Delta t}{\gamma}}\boldsymbol{\xi}_t,$$

where γ denotes the friction constant and $\boldsymbol{\xi}_t$ represents Gaussian white noise drawn from a normal distribution with zero mean and unit variance. The potential $\Phi(x,y,z)$ that we used in our simulations is given by

$$\begin{aligned}\Phi = &-11.5\left[\mathrm{e}^{-x^2-(y+1.2)^2-(z+1.2)^2} + \mathrm{e}^{-x^2-(y-1.2)^2-(z-1.2)^2}\right] \\ &- 17\left[\mathrm{e}^{-(x+1.8)^2-(y+0.12)^2-(z+2.5)^2} + \mathrm{e}^{-(x+1.8)^2-(y-0.12)^2-(z-2.5)^2}\right] \\ &+ x^2 + y^2 + z^2. \end{aligned} \tag{6.6}$$

It is designed to feature well-separated basins, which we denote as states 1-4. Simulating a trajectory for 10^6 time steps, we obtain the three-dimensional trajectory, whose time trace is shown in Fig. 6.9. Looking at the three-dimensional trajectory representation in Fig. 6.11 a, we can see that states 2 and 3 are dynamically separated in all three dimensions, while states 3 and 4 are only distinguishable along the z-axis, with almost identical x- and y-coordinates. This can readily be verified in the two-dimensional projection onto the xy-plane in Fig. 6.11 b. However, dynamically, states 3 and 4 are well separated since a direct transition from state 4 to 3 is not possible and must pass through states 2 and 1, i.e., $4 \to 2 \to 1 \to 3$.

The original three-dimensional system exhibits Markovian dynamics because we used a Markovian Langevin equation to simulate the trajectory. Upon dimensionality reduction to the xy-plane, however, the dynamically distinct states 3 and 4 become spatially superimposed (indicated by the new combined state 3+4), which breaks the Markovian property in this lower dimensional representation. The loss of Markovianity arises because the transition probabilities become path-dependent in the two-dimensional projection. Specifically, the conditional transition probability $p(i|j = 3{+}4)$ is no longer solely determined by the current state 3+4 but depends on whether state 3+4 was entered via state 1 or state 2—violating the Markov property, which requires that the future state of

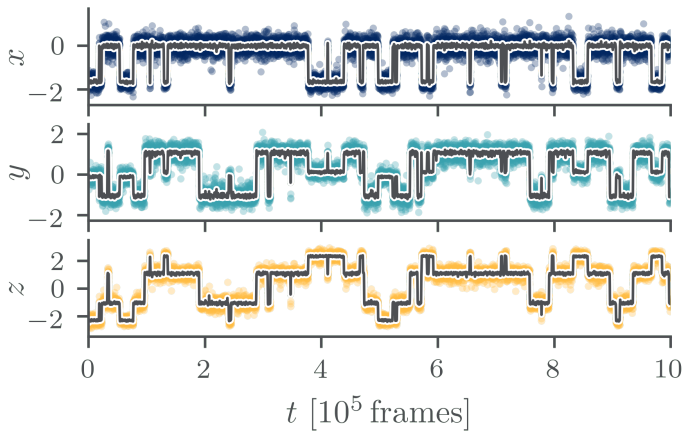


Figure 6.9 | Time trace obtained from a Langevin simulation of the potential $\Phi(x,y,z)$ defined in Eq. (6.6). We simulated 10^6 time steps with a time step size of $\Delta = 5 \cdot 10^{-3}$, a friction coefficient of $\gamma = 1$ and a temperature of $T = 1$ (in dimensions-less units). The solid line represents the Gaussian filtered time trace. Adapted with minor changes from Ref. 4.

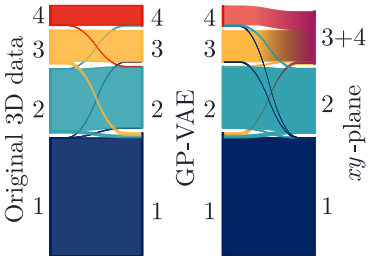


Figure 6.10 | Sankey diagram showing state correspondence across the original 3D data (left), GP-VAE latent embedding (center), and xy-plane projection (right). Band widths indicate the fraction of frames where corresponding states temporally coincide. Adapted with minor changes from Ref. 4.

the system only depends on its current state. Consequently, this path dependence renders traditional Markov State Models (MSMs) unsuitable for modeling the dynamics.

By design, this two-dimensional xy-projection of the three-dimensional dynamics serves as the perfect test case for our GP-VAE framework. In the first step, we used a change point detection algorithm to extract $m = 89$ representative frames from the trajectory using solely information from the xy-coordinates. (for details, see SI, Sec. D.2 and SI, Fig. D.2). Leveraging both spatial data exclusively from the xy-plane and temporal information, we trained our GP-VAE model using the parameters specified in SI, Sec. D.4. Fig. 6.11 c shows the resulting two-dimensional latent representation demonstrating that the GP-VAE is indeed capable of separating the overlapping state 3+4 into its two dynamically distinct substates.

To facilitate better comparison, Fig. 6.10 displays a Sankey diagram that illustrates state correspondence between the states in the full, three-dimensional space and in the reduced two-dimensional xy-projection with the GP-VAE latent representation in the middle. Only minor deviations are notable, mostly for state 1, which are likely to result from dynamical coring[124] that we performed prior to the construction of the MSM for further analysis.

To this end, we employed k-means clustering with $k = 1000$ to achieve a fine state partition in all three representations. To coarse-grain this partition, we used MPP lumping[121] and (iterative) dynamical coring with a lag and coring time of $\tau_{\text{lag}} = \tau_{\text{coring}} = 10$ frames. We then computed transition matrices $\boldsymbol{T}(\tau)$ for each of the three MSMs and calculated the corresponding implied timescales using $t_i(\tau) = -\tau / \ln(\lambda_i)$, where λ_i represents the eigenvalues associated with eigenvector ν_i of $\boldsymbol{T}(\tau)$. The implied timescales are shown in Fig. 6.11 d-f, and the corresponding eigenvector contributions in g-i. Notably, both the full three-dimensional MSM and the GP-VAE embedding MSM converge to identical implied timescale values at large τ_{lag}-balues. However, the GP-VAE embedding demonstrates significantly faster convergence, which we attribute to the inherent Markovian structure enforced by the Matérn kernel used in our GP-VAE framework. As mentioned earlier, temporal correlations in the trajectory are encoded as spatial proximity in the latent space, which is why the GP regression in the latent space effectively acts as a time-aware filtering that filters out non-Markovian noise components. Consequently, the microstates exhibit significantly increased metastability because the rapid, noise-driven fluctuations are suppressed. This increased metastability accelerates the memory decay in the system and therefore facilitates the construction of MSMs with shorter lag times, yielding better resolved models. For our toy model specifically, this improvement is clearly demonstrated in the dendrograms illustrating the metastability of the microstates during hierarchical lumping towards macrostates; see SI, Fig. D.7.

We further evaluate the quality of the latent embedding of our model by comparing the eigenvectors of the MSMs constructed in three dimensions and in the two-dimensional space from the GP-VAE. Despite relying solely on limited information about the dynamics of the system through the xy-plane projection, the GP-VAE accurately recovers the

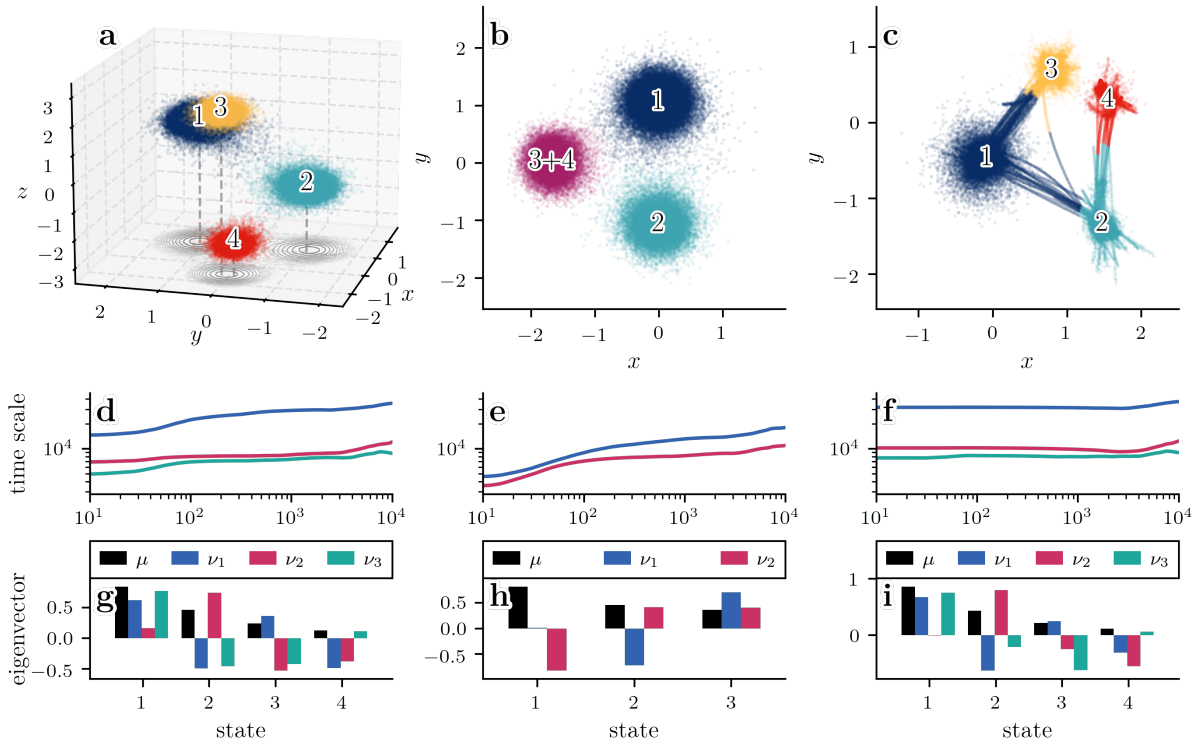


Figure 6.11 | (a) The three-dimensional visualization of the simulated trajectory displays four distinct metastable basins, with states labeled 1 (blue), 2 (cyan), 3 (yellow), and 4 (red), where contour lines projected onto the xy-plane illustrates the relative potential depth. When projected onto the two-dimensional xy-plane (b), states 3 and 4 overlap spatially, rendering them indistinguishable using positional coordinates alone. (c) The GP-VAE successfully separates the originally overlapping states 3 and 4 by incorporating temporal information alongside the xy-plane coordinates, thereby reconstructing the underlying Markovian dynamics. The bottom panels (d-i) present the results from MSM analysis for each of the three scenarios (column-wise) depicted above, showing the implied timescales (d-f) and corresponding eigenvector contributions (g-i) for each case. Reprinted from Ref. 4.

system's dynamic structure. So, for instance, the equilibrium state populations, shown as μ, precisely match those of the three-dimensional MSM, suggesting that the GP prior preserves the underlying thermodynamic properties. Moreover, the first eigenvector ν_1, which characterizes the slowest dynamical process in the system, shows close agreement between both representations. This demonstrates the GP-VAE's capability to identify dominant transition pathways even when the corresponding states are geometrically indistinguishable. The higher-order eigenvectors ν_2 and ν_3 maintain reasonable qualitative agreement, with expected deviations given the missing information in the z-dimension.

To summarize, our GP-VAE framework demonstrates the remarkable ability to extract meaningful collective variables even when input data lack essential geometric information due to flawed dimensionality reduction steps. Beyond accurately reconstructing the correct state assignment from incomplete spatial information, our approach also accelerates the convergence of implied timescales in MSM analysis. This improved performance is due to two key intertwined factors: the mathematical framework of Markovian kernels—such as the employed Matérn kernel—naturally enforces Markovian properties in the latent representation. Secondly, the GP regression operates as a temporal filtering, systematically removing non-Markovian noise components. The synergy between these two aspects enhances MSM construction and can help to

disentangle distinct dynamical processes, even when they are geometrically indistinguishable.

6.4.4 Software

We implemented the GP-VAE framework using PyTorch,[10] and it is freely available for public use at `https://github.com/moldyn/GP-TEMPEST`

6.5 Concluding Remarks

We have introduced two advanced physics-informed machine learning frameworks created to obtain more interpretable and dynamically coherent latent representations from molecular dynamics (MD) simulations. Both methods are complementary and address two different drawbacks of common feature extraction methods used in MD simulations:

1. While nonlinear feature extraction techniques are more powerful than linear methods at capturing complex relationships in the data,[113–115,342] they typically lack explicit physical regularization through their architectural design. This absence of physical constraints in these neural networks often results in latent representations that are not meaningful from a physical perspective, e.g. because the latent representation exhibits completely disconnected regions.[322] This may not come as a surprise, since these methods operate in vastly high-dimensional feature spaces that offer excessive flexibility, effectively allowing the models to fit arbitrary patterns potentially preventing them from learning physically compact and meaningful representations. Furthermore, this unrestricted flexibility makes these models also notoriously difficult to train, and small changes in hyperparameters might lead to drastically different results,[322] meaning that results are often not robust and reproducible.
2. The great majority of all feature extraction methods, with a few notable exceptions,[107,134] rely on the assumption that the data is independent and identically (i.i.d.) distributed. This assumption fundamentally contradicts the sequential nature of MD simulations, where each frame directly depends on the previous one through the Newtonian equations of motion.

Graph Neural Network Autoencoder

In the first part of this chapter, we tackled the first issue by introducing graph-based representations that explicitly capture the underlying physicochemical structure of the protein conformation at each time step in the trajectory. These graph representations naturally encode the spatial relationships between the different residues in the protein.

Based on this foundation, we developed a graph neural network autoencoder that operates directly on these graph representations and learns

a low-dimensional latent representation. Regularized by the underlying graph structure of the protein, this model leverages powerful local and nonlinear operations—including message passing and attention mechanisms—while avoiding the overfitting issues that plague common nonlinear feature extraction methods. The local propagation of information in a graph neural network forces the model to obey to physical principles like the local propagation of conformational changes.

Our application to T4 lysozyme proves the effectiveness of this approach: the resulting two-dimensional free energy landscape reveals significantly enhanced structural complexity compared to the principal component analysis. Crucially, validation of the additional energy basins revealed that these are indeed physically meaningful and correspond to genuine conformational states rather than being spurious artifacts resulting from nonlinearity. Furthermore, we demonstrated the generative capabilities of this model by suggesting a routine to generate realistic protein backbone structures along pathways in the latent space. Navigating the free energy landscape by using the generative capabilities of the graph neural network autoencoder can help to systematically explore dominant conformational transition pathways and investigate how the protein's structure changes with respect to different regions in the free energy landscape. However, as the side chain dynamics are not included in the model, a detailed analysis of the underlying mechanisms in terms of contact formation or breaking is still indispensable for a detailed physical interpretation of a conformational change.

Gaussian Process Variational Autoencoder

In the second part of this chapter, we addressed the fundamental limitation posed by the i.i.d. assumption in common feature extraction methods for molecular dynamics simulations. To this end, we explored the use of Gaussian processes to explicitly model temporal dependencies in molecular dynamics simulations. Their kernel-based approach provides direct control over temporal correlation structures, and their probabilistic foundation enables seamless integration with variational autoencoders.

Central to our approach is the use of Markovian kernel functions, particularly the Matérn kernel, which imposes the preservation of Markovian properties during dimensionality reduction. This renders the resulting embedding a perfect starting point for further dynamical analysis, such as Markov state models. We demonstrate the effectiveness of our framework with a carefully designed analytical toy model that systematically introduces non-Markovian behavior through deliberate projection artifacts. In this controlled test case, we challenged the Gaussian process variational autoencoder by relying solely on incomplete spatial information from a two-dimensional projection in which two dynamically distinct states appear spatially indistinguishable. Despite the essential degree of freedom being hidden, the Gaussian process variational autoencoder successfully recovered the correct state assignments by leveraging temporal information from the trajectory.

Crucially, our approach provides significant advantages for the subsequent dynamical analysis by enabling Markov state model construction

with substantially faster convergence of implied timescales, and consequently, shorter lag times. This improvement is due to the inherent Markovian structure imposed by the Matérn kernel Gaussian process prior, acting as a time-aware filter that systematically removes non-Markovian noise components while preserving the essential dynamical structure. The resulting microstates in the latent representation exhibit increased metastability, accelerating memory decay and thus facilitating more efficient construction of Markov state models.

While this toy model represents an idealized scenario, it demonstrates the great potential of incorporating temporal information into dimensionality reduction frameworks for sequential data, such as coming from molecular dynamics simulations. This time-aware perspective extends purely geometric approaches and offers new insights into the dynamics of complex biomolecular systems where conventional collective variables may miss the complete picture of the dynamics. The frameworks' capability of recovering hidden degrees of freedom through temporal correlations opens new avenues for modeling complex biomolecular systems where essential degrees of freedom are hidden from projection artifacts or incomplete feature selection.

Moving Forward

The two approaches presented in this chapter—graph neural network for physically meaningful spatial regularization and Gaussian processes for temporal coherence—are complementary and can be straightforwardly combined into a powerful, unified framework. Such a Gaussian process variational graph autoencoder would simultaneously leverage protein topology and temporal dependencies to create robust and physically informed latent representations.

In a larger context, most of the classical feature extraction methods, including principal component analysis, time-lagged independent component analysis as well as various autoencoder architectures, implicitly operate within a covariance or correlation framework. Whether considering instantaneous covariance matrices in principal component analysis or time-lagged covariance matrices in time-lagged independent component analysis, these methods are fundamentally unable to capture asymmetric temporal relationships or directional dependencies that are required to characterize causal mechanisms. The Gaussian process framework suggested here, while demonstrated using only stationary kernels, can be extended to non-stationary kernels that break the time-reversal symmetry. Kernels of the form[viii]

$$k(t,t') = k_{\text{stationary}}\left(|t-t'|\right) \cdot k_{\text{causal}}\left(t,t'\right),$$

where $k_{\text{causal}}(t,t')$ explicitly depends on the absolute time positions rather than their difference, introduce time asymmetry which might be useful to capture temporal directional dependencies. Employing such non-stationary kernels in the Gaussian process variational autoencoder or in the unified framework could potentially further strengthen the model's ability to find low-dimensional representations of molecular dynamics simulations by capturing not only temporal correlations but also explicitly causal relationships. This is particularly relevant in the context of

[viii] Similarly, the kernel hyperparameters can be made time-dependent, i.e. $k(t,t';l) = k\left[t,t';l(t,t')\right]$.

allosteric transitions, where the directionality of changes is crucial for understanding the underlying molecular machinery.

7 Conclusion and Outlook

Der Mensch ist zu einer beschränkten Lage geboren. Einfache, nahe, bestimmte Zwecke vermag er einzusehen, und er gewöhnt sich, die Mittel zu benutzen, die ihm gleich zur Hand sind. Sobald er aber ins Weite kommt, weiß er weder, was er will, noch was er soll.

– Johann Wolfgang von Goethe (Wilhelm Meisters Lehrjahre)

As integral components of every living organism, proteins orchestrate the fundamental processes that sustain life through their remarkable ability to dynamically transition between various conformational states. Understanding these molecular machines through their dynamical behavior—that spans timescales from picoseconds to seconds—represents one of the grand challenges of theoretical biophysics.

Through remarkable progress in computational power and algorithmic developments—in both simulation and analysis—over recent decades, molecular dynamics simulations have emerged as a transformative tool to study the molecular machinery underlying protein function. By providing spatio-temporal trajectories of protein folding, conformational changes, and interactions at an atomistic scale, molecular dynamics simulations shed light on molecular mechanisms that remain challenging to observe experimentally. Modern molecular dynamics simulations routinely generate terabytes of trajectory data spanning the motion of millions of atomic coordinates across microseconds to milliseconds timescales, and yet their effective dynamics are often described by only a small fraction of these degrees of freedom. This dimensionality reduction challenge motivates the central question of this thesis: *How can we extract meaningful information from this overwhelming wealth of high-dimensional data?*

Contributions

Roughly, the contributions of this thesis can be categorized into two parts of methodological advances, both of which share the common ground of similarity: First, we developed correlation-based approaches (MoSAIC and a nonparametric normalized mutual information estimator) that identify functional relationships between dynamical observables. Second, we introduced physics-informed feature extraction methods (graph neural network autoencoders and Gaussian processes variational autoencoders) that result in robust and interpretable low-dimensional representations. While the proposed graph neural networks architecture operates directly on the graph structure of the protein and thereby regularizes the latent representation in a physically meaningful way, Gaussian processes variational autoencoders preserve the Markovianity of the input data in the low-dimensional representation. Here, the concept of similarity is

used in a different but complementary way: instead of measuring structural similarity among dynamical observables, the covariance kernel in Gaussian processes defines a temporal similarity among conformations of the protein.

Correlation-Based Feature Selection

Conceptually, MoSAIC—introduced in chapter 3—aims to systematically identify functional relationships by distinguishing collective motions from noisy coordinates. Building on the premise that biologically meaningful dynamics organize into groups of coordinates that follow coordinated motion, this community-based perspective represents a conceptual advance in how we understand protein function: Rather than investigating isolated essential coordinates, we recognize that functional motion in proteins emerges from organized networks of correlated interactions. By analyzing the linear correlation structure among internal coordinates, such as contact distances and dihedral angles, MoSAIC systematically identifies groups of coordinates that describe collective motions—a critical step that facilitates both modeling and interpretation in subsequent steps.

This approach proved particularly effective for allosteric systems, where MoSAIC revealed that the vast majority of coordinates (often around 80% – 90%) do not contribute to the functional allosteric transition, and only a small fraction of coordinates is relevant to describe these processes.[i] Systematically removing these redundant coordinates yields substantial improvements in signal-to-noise ratios and hence improves subsequent modeling. For folding proteins, strong inter-cluster correlations indicate that the clusters describe different aspects of structural organization within the folding process. This is in contrast to the low inter-cluster correlation observed for allosteric systems.

[i] This is not only observed for T4 lysozyme, but also for various PDZ domains in independent works.[158,224,343,344]

As the field of biomolecular dynamics increasingly tackles larger and more complex systems, a principled approach to feature selection becomes essential for extracting biologically relevant information. Rather than blindly following variance-based or timescale-based selection criteria that optimize statistically prominent but functionally irrelevant motions, MoSAIC provides a rigorous framework for identifying that motion that truly matters.

Beyond Linear Correlations

While the linear correlation coefficient has proven to be highly effective in identifying functional relationships in one-dimensional internal coordinates, it assumes that relationships manifest as linear changes between co-linear variables. When dealing with higher-dimensional coordinates, such as Cartesian C_α coordinates, this assumption breaks down. Specifically, cancellation effects between different directional components lead to a systematic failure of Pearson correlation: despite clear dependencies in all spatial dimensions, the relationship is not captured.

Without assuming any specific functional form of the relationship between two variables, mutual information provides all the means to overcome these limitations, as it captures all statistical dependencies—whether

they arise from nonlinear relationships or involve complex directional interactions. To address the major drawback of mutual information, namely that it is not normalized and, therefore, difficult to interpret, we introduced a nonparametric estimator of normalized mutual information in chapter 4 that is scalable to large protein systems with extensive simulation data.

Combined Methodological Framework

Having established these two methodological frameworks for correlation analysis, we put these complementary approaches to test whether we can truly gain valuable insights into the complex biological mechanisms that underlie protein function. To this end, we thoroughly investigated the open↔closed allosteric transition in T4 lysozyme in chapter 5 and established a complete picture that bridges the mechanistic details provided through the MoSAIC analysis with global relationships captured by normalized mutual information: MoSAIC's correlation-based feature selection employed on contact distances and side-chain dihedral angles revealed a fluctuation transition network that explains how local perturbations propagate through the protein via cooperative conformational changes. Complementing this mechanistic view, normalized mutual information captured the global relationships and revealed how the entire protein responds to the allosteric transition, providing insights into changes in rigidity and flexibility that extend beyond the scope of MoSAIC's local focus. Together, both approaches result in a comprehensive understanding of the allosteric transition.

Physics-Informed Feature Extraction

While the correlation-based approaches were able to provide very valuable insights into the functional relationships between dynamical observables, they ultimately operate as feature selection methods and identify which coordinates—or parts—of the protein exhibit strong statistical dependencies at any given moment in time. From a modeling perspective, these instantaneous similarity measures provide the essential foundation for the understanding protein function by identifying the relevant degrees of freedom, but the dynamics itself—the very essence of protein function—requires additional approaches that capture how these degrees of freedom evolve over time. Rather than following the dynamics along all selected coordinates individually, we typically seek to construct dynamical models such as Markov state models, which provide a coarse-grained description of the temporal evolution of the system. Unfortunately, the *curse of dimensionality* prevents the direct construction of these models in the high-dimensional feature space but necessitates reducing the number of dimensions to a level where reliable density estimation becomes feasible, and conformational substates can be defined in a meaningful way.

To preserve the essential physics of the system in the low-dimensional representation, we, therefore, developed two novel physics-informed feature extraction methods in chapter 6 that combine the representational power of deep learning with physical constraints. By designing a graph

neural network autoencoder that directly operates on the graph structure of the protein, we obtain robust latent representations that are able to capture far more details in the complex free energy landscapes than traditional methods. Notably, a major advantage of the graph neural network autoencoder is consistent embeddings across different hyperparameters, which is typically not the case for many deep learning approaches. In an additional step, we addressed a common fundamental limitation of most feature extraction methods—including this graph neural network autoencoder and principal component analysis—that they treat each time step independently, violating the inherent sequential nature of molecular dynamics. To this end, we introduced a Gaussian process variational autoencoder that models temporal dependencies in the data via Gaussian processes. This approach not only produces latent representations that exhibit increased Markovianity and allow for more efficient Markov state models but also demonstrates the remarkable ability to recover important degrees of freedom that may have been lost due to erroneous feature selection.

Outlook

Most of the research conducted during this thesis can somehow, more and less directly, be linked to the concept similarity. While this perspective has proven to be quite fruitful in answering some questions, it has also raised new questions that remain open for future research.

Unified Framework for Feature Extraction

The most immediate step would certainly be combining the regularization power of the graph neural network with the Gaussian process variational autoencoder into a unified framework for physics-informed feature extraction of protein dynamics. Such a hybrid architecture would incorporate local information transfer through the graph structure—mimicking the physical propagation of perturbations through the protein—while simultaneously respecting the sequential nature of molecular dynamics simulations. Both complementary strengths identified in this thesis would synergize: the hyperparameter robustness of the graph neural network and the ability of the Gaussian process variational autoencoder to recover lost degrees of freedom.

Causality

Another promising direction is to extend the Gaussian process to non-time-reversible kernels. A kernel that depends not only on the time difference between two conformations, but also on their absolute time, would make the kernel matrices non-symmetric and break the time symmetry of the Gaussian process. This would potentially introduce directionality into the latent representation and might open up new avenues for causal inference directly in the latent space, potentially scaling towards large systems. This would seamlessly connect to the non-equilibrium nature of functional protein dynamics. While equilibrium

simulations obey detailed balance and exhibit time-reversibility, most biologically relevant processes—including allosteric communication—are inherently non-equilibrium processes that feature directional conformational flow.

Similarly, also the MoSAIC analysis would benefit from introducing causal directionality. While MoSAIC identifies clusters of highly correlated motions, it remains an open question how these clusters communicate with each other. A systematic understanding of inter-cluster communication would be particularly relevant to the understanding of allosteric pathways. This would require moving from purely instantaneous similarity measures to time-lagged asymmetric measures, such as, e.g., transfer entropy.[345]

Appendix

Supporting Information for Chapter 3

A

A.1 Optimization of Clustering Parameters

In the case of the simple model of a correlation matrix (Sec. 3.2.2), we by design know the ground truth of the clusters. The matrix is available at our homepage.[i] Hence, we can calculate the V-measure according to[243]

[i] See https://www.moldyn.uni-freiburg.de/software.html.

$$V = \frac{2hc}{h+c}, \tag{A.1}$$

where c is completeness and h homogeneity, defined as

$$c = \begin{cases} 1 & \text{if } H(K,C) = 0 \\ 1 - \frac{H(C|K)}{H(K)} & \text{else} \end{cases} \tag{A.2}$$

and

$$h = \begin{cases} 1 & \text{if } H(C,K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{else .} \end{cases} \tag{A.3}$$

Here C is the set of ground truth (compare Fig. 3.8 in the main paper), and K is the set of the resulting clusters. $H(K,C)$ is the joint entropy and $H(K|C)$ the conditional entropy of the resulting cluster partition given the ground truth.[243] The results for all possible clustering parameters can be found in Fig. A.1.

Usually however, we do not know the ground truth. In this case, we can use the so-called silhouette method as a heuristic.[203] For each feature $i \in C_I$ belonging to cluster C_I we can define the mean distance between i and all other features belonging to the same cluster by

$$a_i = \frac{1}{|C_I| - 1} \sum_{\substack{j \in C_I \\ j \neq i}} d_{ij} \, , \tag{A.4}$$

and its average distance to its nearest neighbor cluster

$$b_i = \min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d_{ij} \, . \tag{A.5}$$

Therewith, the silhouette coefficient can be defined for all M features by

$$SC = \langle s_i \rangle_{i \in M} \, , \tag{A.6}$$

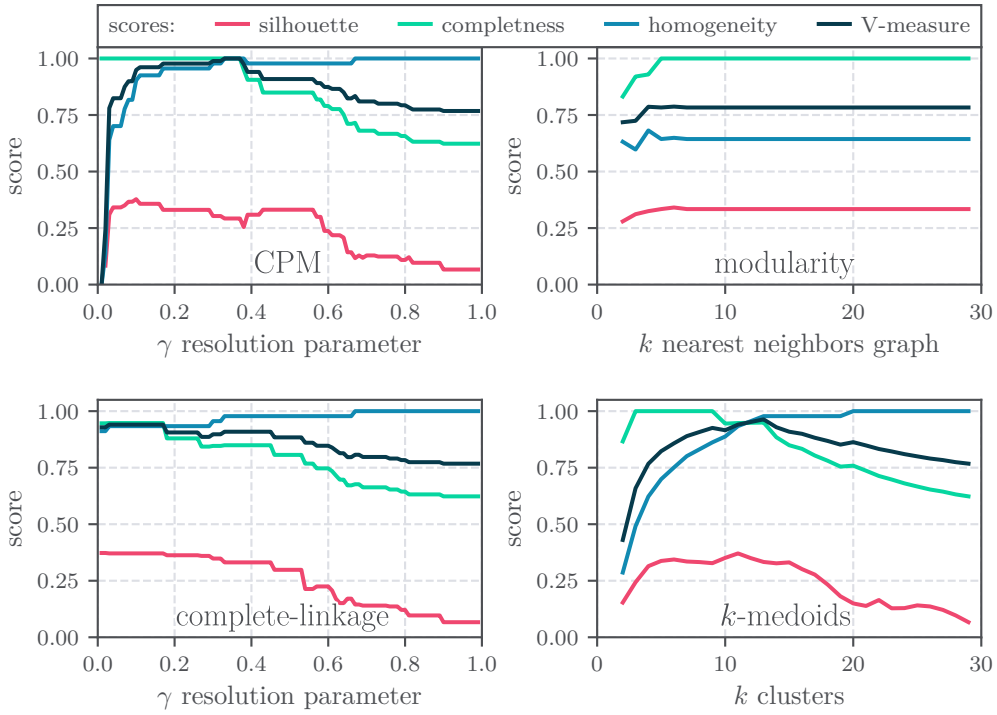


Figure A.1 | Comparison of the performance of Leiden/CPM, Leiden/modularity, complete-linkage and k-medoids clustering, using the silhouette method [Eq. (A.6)] and the V-measure [Eq. (A.1)]. The resulting parameters are $\gamma = 0.35$ for Leiden/CPM, $k = 4$ for Leiden/modularity, $\gamma = 0.15$ for the complete linkage clustering and $k = 13$ for $k-$medoids. Figure reprinted from Ref. 1. Copyright © (2022) The Authors.

where the contribution of each feature i is defined by

$$s_i = \begin{cases} 1 - \frac{a_i}{b_i} & \text{if } a_i < b_i \\ 0 & \text{if } a_i = b_i \\ \frac{b_i}{a_i} - 1 & \text{if } a_i > b_i \,. \end{cases} \tag{A.7}$$

In Fig. A.1 we study the effect of changing the clustering parameters on the silhouette coefficient. Comparing it to the previous results of V-measure we find similar results. *E.g.*, when the latter is minimal also the silhouette coefficient is minimal. Nevertheless, we find a slight shift of the maxima. Hence, we advise to use the silhouette method only as a guide to find a first estimate on the clustering parameter, but not necessarily as a way to determine the final clustering parameters.

A.2 Results

A.2.1 Linear vs. Nonlinear Correlation Measures

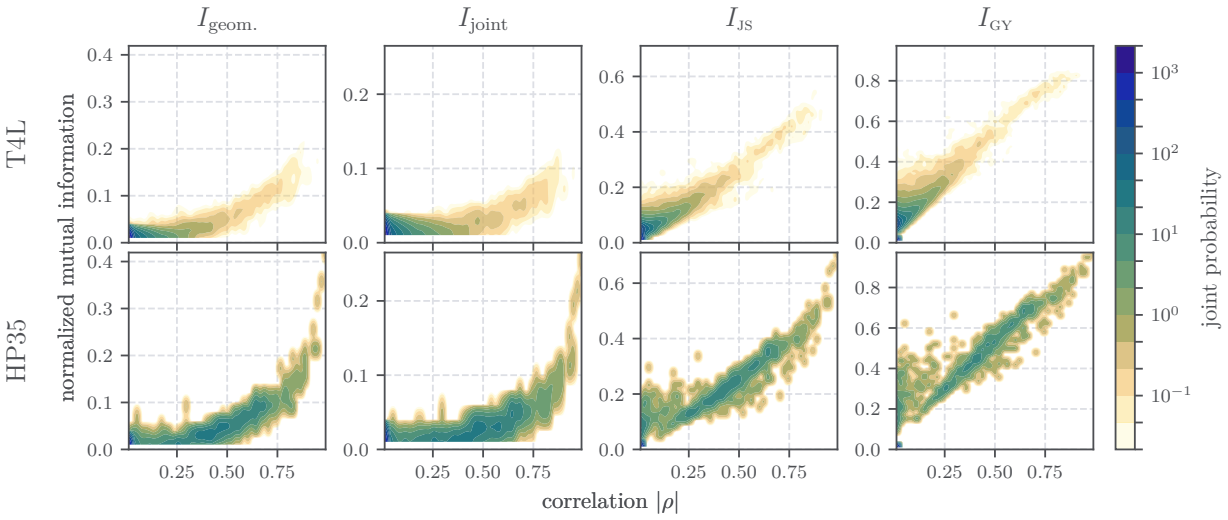
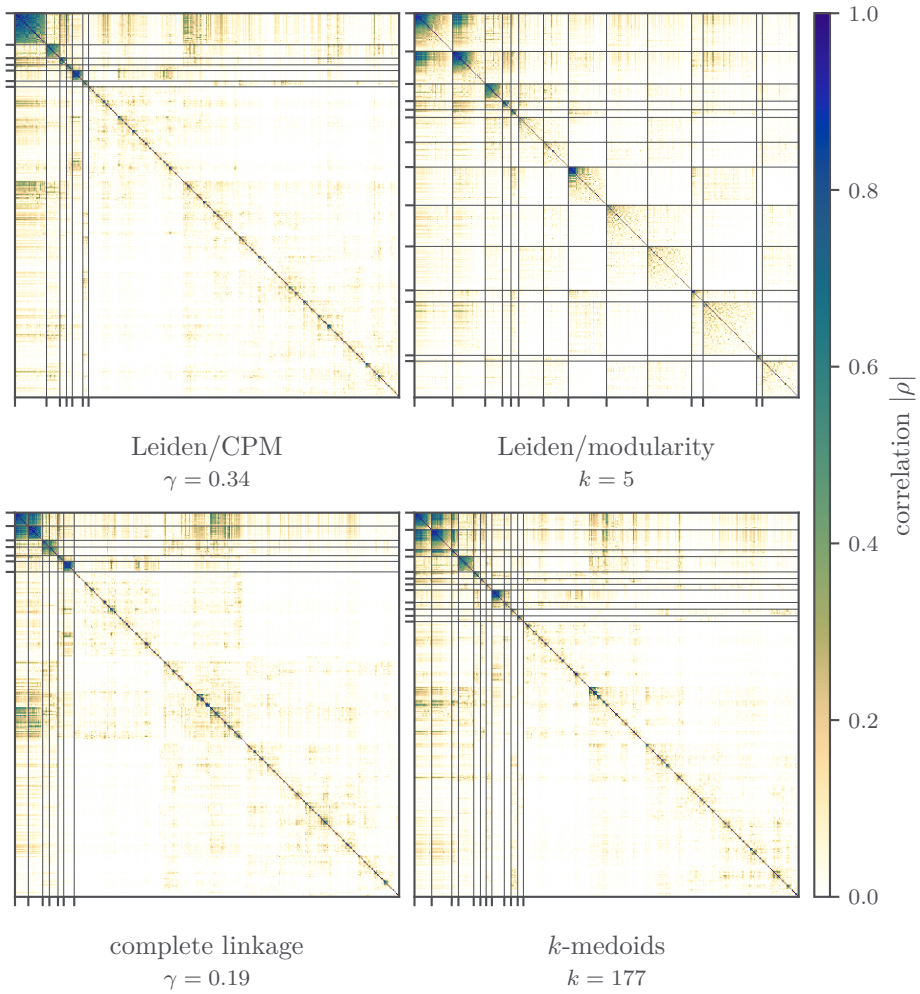


Figure A.2 | Comparison of the nonlinear correlations $I_{\text{geom.}}$, I_{joint}, I_{JS}, and I_{GY} to the absolute Pearson coefficient $|\rho|$ for HP35 and,T4L. Figure reprinted from Ref. 1. Copyright © (2022) The Authors.

A.2.2 Clustering of T4L

a



b

Contact distances $r_{i,j}$ of the clusters using Leiden/CPM with $\gamma = 0.50$:

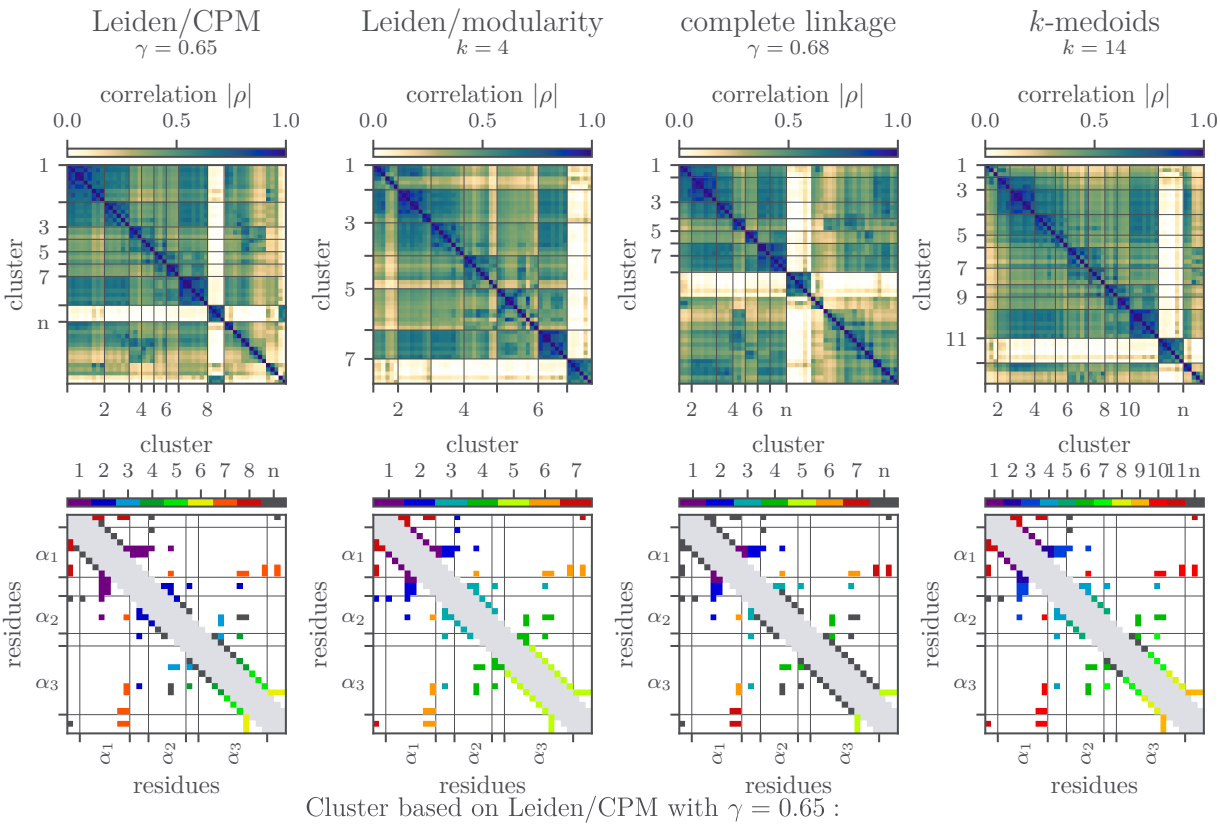
cluster 1 : $r_{4,60}, r_{4,63}, r_{32,106}, r_{21,141}, r_{22,141}, r_{32,105}, r_{4,29}, r_{32,107}, r_{21,142}, r_{4,64}$
$r_{8,67}, r_{7,71}, r_{22,142}, r_{35,106}, r_{32,103}, r_{7,12}, r_{4,71}, r_{8,64}, r_{32,104}, r_{4,68}$
$r_{8,13}, r_{3,67}, r_{8,12}, r_{11,30}, r_{30,104}, r_{22,105}, r_{66,70}$

cluster 2 : $r_{10,101}, r_{9,161}, r_{10,105}, r_{10,145}, r_{6,97}, r_{10,149}, r_{3,100}, r_{6,101}, r_{9,148}, r_{10,104}$

cluster 3 : $r_{34,38}, r_{36,42}, r_{34,42}, r_{36,45}, r_{34,41}, r_{25,34}, r_{24,34}, r_{37,41}, r_{34,45}$

Figure A.3 | (a) Clustering the 402 contact distances of T4L using Leiden/CPM, Leiden/modularity, complete-linkage and k-medoids with optimized parameters according to silhouette score. (b) Description of all clusters in Fig. 3.9 with more than 5 coordinates. Figure reprinted from Ref. 1.

A.2.3 Clustering of HP35



Cluster based on Leiden/CPM with $\gamma = 0.65$:

cluster 1 : $r_{6,12}, r_{7,12}, r_{6,14}, r_{6,13}, r_{7,13}, r_{6,17}, r_{6,11}, r_{7,11}, r_{6,10}$

cluster 2 : $r_{12,17}, r_{12,16}, r_{12,20}, r_{13,17}, r_{12,28}, r_{14,18}$

cluster 3 : $r_{18,25}, r_{17,25}, r_{20,25}$

cluster 4 : $r_{24,28}, r_{25,29}, r_{20,28}$

cluster 5 : $r_{27,31}, r_{28,32}, r_{26,30}$

cluster 6 : $r_{29,34}, r_{29,33}, r_{29,35}$

cluster 7 : $r_{10,32}, r_{10,29}, r_{10,28}, r_{10,34}, r_{9,32}, r_{9,34}, r_{10,17}$

cluster 8 : $r_{1,6}, r_{1,9}, r_{1,10}, r_{1,5}$

Figure A.4 | Clustering the 53 contact distances of HP35 using Leiden/CPM, Leiden/modularity, complete-linkage and k-medoids with optimized parameters according to silhouette score. Where the resulting clusters are visualized by their corresponding (top) correlation matrix and (bottom) contact map. Bottom: Description of all clusters in Fig. 3.10 with more than two coordinates. Figure reprinted from Ref. 1. Copyright © (2022) The Authors.

A.2.4 Displacement of a C_{10}-Trimer

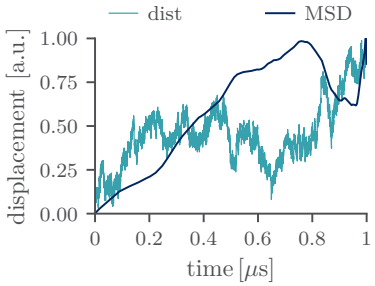


Figure A.5 | MSD and the Euclidean distance between initial and current position of one of the C_{10}-trimer (both in terms of the COM coordinate).

Supporting Information for Chapter 4 B

B.1 Computation of the Canonical Correlation Coefficient

Numerically, it is more efficient to compute the canonical correlation using covariance matrices rather than solving for the projection operators $\boldsymbol{A}$ and $\boldsymbol{B}$ and their eigenvalues.[286] Relying on only the covariance matrices, the approach described here avoids the computationally intensive eigendecomposition step.[346]

For standardized coordinates $\delta\alpha_i = \frac{(\alpha_i - \langle\alpha_i\rangle)}{\sqrt{\langle(\alpha_i - \langle\alpha_i\rangle)^2\rangle}}, \alpha = x, y, z$, ρ_C can be computed as

$$\rho_C = \sqrt{\frac{1}{3}\mathrm{tr}(R)}, \tag{B.1}$$

where R is constructed from the correlation matrices

$$R = R_{11}^{-1} R_{12} R_{22}^{-1} R_{21}, \tag{B.2}$$

and R_{ij} can be calculated as

$$R_{nm} = \begin{pmatrix} \langle\delta x_n \delta x_m\rangle & \langle\delta x_n \delta y_m\rangle & \langle\delta x_n \delta z_m\rangle \\ \langle\delta y_n \delta x_m\rangle & \langle\delta y_n \delta y_m\rangle & \langle\delta y_n \delta z_m\rangle \\ \langle\delta z_n \delta x_m\rangle & \langle\delta z_n \delta y_m\rangle & \langle\delta z_n \delta z_m\rangle \end{pmatrix}. \tag{B.3}$$

Here, $n, m \in \{1, 2\}$ denote the sets of variables 1 and 2, that is, the Cartesian coordinates of two C_α-atoms.

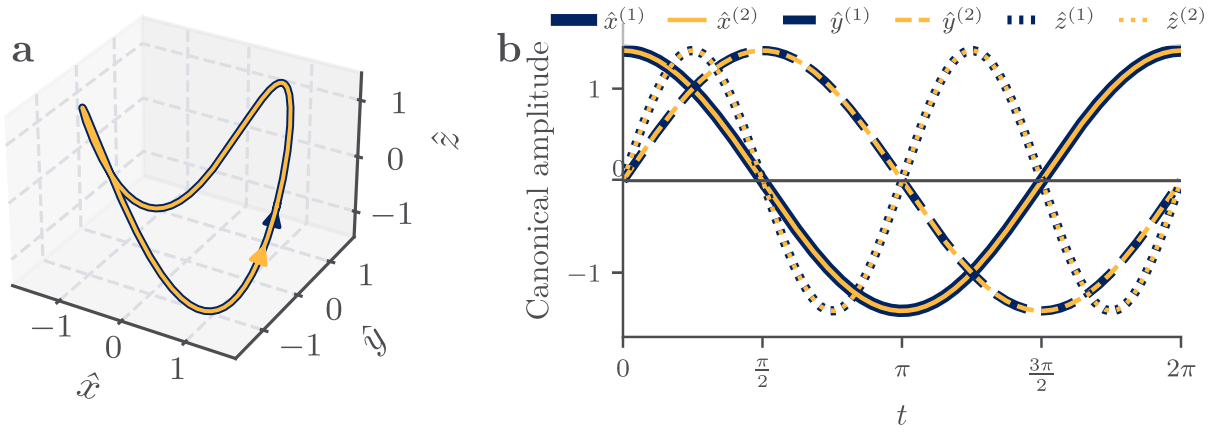


Figure B.1 | The same example system as shown in Fig. 4.1, but with canonical coordinates. CCA now rightfully captures the correlation between the two particles, which is 1 in all three dimensions.

Supporting Information for Chapter 5

C

C.1 Contact Based Analysis

C.1.1 MoSAIC Clustering Results

Tab. C.1 lists the coordinates contained in the clusters resulting from the MoSAIC analysis of T4L in Sec. 5.1.1. The constant Potts model with a resultion parameter of $\gamma = 0.5$ was used and clusters containing less than 10 coordinates attributed to the noise cluster.

Cluster	Coordinates
1	$d_{4,60}$, $d_{4,63}$, $d_{4,13}$, $d_{4,29}$, $d_{4,72}$, $d_{22,137}$, $d_{4,64}$, $d_{20,142}$, $d_{8,67}$, $d_{8,68}$ $d_{22,141}$, $d_{21,141}$, $d_{2,64}$, $d_{7,71}$, $d_{1,64}$, $d_{4,71}$, $d_{30,145}$, $d_{21,142}$, $d_{5,60}$ $d_{7,12}$, $d_{8,64}$, $d_{20,145}$, $d_{4,68}$, $d_{8,13}$, $d_{3,67}$, χ_4, $d_{5,64}$, $d_{24,105}$, $d_{8,12}$, $d_{29,64}$, χ_{104}, $d_{11,20}$, $d_{2,67}$, $d_{11,30}$
2	$d_{10,101}$, $d_{6,98}$, $d_{6,97}$, $d_{9,161}$, $d_{6,94}$, $d_{9,160}$, $d_{10,149}$, $d_{10,105}$, $d_{9,158}$, $d_{10,145}$, $d_{6,152}$, $d_{9,148}$, $d_{6,101}$, $d_{3,100}$
3	$d_{20,24}$, $d_{20,25}$, $d_{18,22}$, $d_{22,26}$, $d_{14,20}$, $d_{14,21}$, $d_{22,30}$, $d_{20,32}$, $d_{20,26}$ χ_{20}, $d_{11,22}$
4	$d_{36,42}$, $d_{25,34}$, $d_{36,45}$, $d_{24,34}$, $d_{34,38}$, $d_{34,41}$, $d_{34,42}$, $d_{37,41}$, $d_{23,34}$ $d_{35,45}$

Table C.1 | Inter-residue distances and first side-chain dihedral angles within clusters found by MoSAIC, sorted by their average correlation within the cluster.

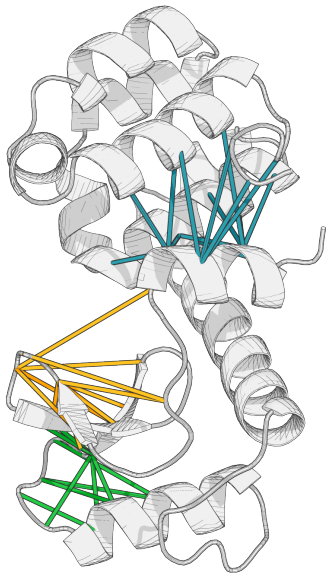


Figure C.1 | The clusters 2-4 of the MoSAIC analysis of T4L with a resolution parameter of $\gamma = 0.5$: cluster 2 in cyan, cluster 3 in yellow and cluster 4 in green. The corresponding coordinates in the clusters are listed in Tab. C.1.

Table C.2 | List of 43 inter-residue contacts that mediate the open↔closed transition of T4L: (Top:) 20 highly correlated contacts that are most important for the open-closed transition as they are highly correlated $\langle|\rho|\rangle_{C_1}$ in the MoSAIC[1] analysis of cluster 1 and feature a high change in contact probability $\Delta p = |p_{\text{open}} - p_{\text{closed}}|$. (Middle): 12 contacts, that are also highly correlated, but exhibit contact probability changes $\Delta p_C \leq 0.3$. (Bottom): 11 contacts that feature a high contact probability change $\Delta p_C \geq 0.3$, but are not significantly correlated to the cordinates describing the open-closed transition $\langle|\rho|\rangle_{C_1} \leq 0.5$. Table reprinted from Ref. 3. Copyright © (2024) Authors.

Contacts	p_{open}	p_{closed}	Δp	$\langle\|\rho\|\rangle_{C_1}$
$d_{4,60}$	0.94	0.00	0.94	0.82
$d_{4,63}$	0.92	0.00	0.92	0.81
$d_{4,13}$	0.91	0.00	0.91	0.81
$d_{4,29}$	0.96	0.00	0.96	0.78
$d_{22,137}$	0.00	0.89	0.88	0.77
$d_{4,64}$	1.00	0.06	0.93	0.76
$d_{8,67}$	0.00	0.96	0.96	0.76
$d_{22,141}$	0.00	0.83	0.83	0.76
$d_{21,141}$	0.00	0.80	0.80	0.76
$d_{2,64}$	0.52	0.00	0.52	0.76
$d_{7,71}$	0.01	0.97	0.96	0.74
$d_{4,71}$	0.00	0.92	0.92	0.72
$d_{21,142}$	0.00	0.59	0.59	0.71
$d_{7,12}$	1.00	0.17	0.82	0.70
$d_{8,64}$	0.00	0.84	0.84	0.69
$d_{4,68}$	0.00	0.84	0.84	0.66
$d_{8,13}$	0.97	0.21	0.76	0.65
$d_{3,67}$	0.98	0.13	0.85	0.64
$d_{8,12}$	1.00	0.26	0.73	0.62
$d_{11,30}$	0.26	0.92	0.66	0.57
$d_{4,72}$	0.00	0.12	0.12	0.77
$d_{20,142}$	0.00	0.08	0.08	0.77
$d_{8,68}$	0.00	0.05	0.05	0.76
$d_{1,64}$	0.05	0.00	0.05	0.72
$d_{30,145}$	0.00	0.05	0.05	0.72
$d_{5,60}$	0.15	0.00	0.15	0.71
$d_{20,145}$	0.00	0.09	0.09	0.70
$d_{5,64}$	0.04	0.00	0.04	0.64
$d_{24,105}$	0.00	0.08	0.08	0.63
$d_{29,64}$	0.00	0.12	0.12	0.60
$d_{11,20}$	0.03	0.26	0.24	0.58
$d_{2,67}$	0.03	0.00	0.03	0.58
$d_{75,88}$	0.85	0.31	0.54	0.48
$d_{11,18}$	0.17	0.85	0.68	0.46
$d_{3,75}$	0.00	0.32	0.32	0.45
$d_{10,104}$	0.00	0.35	0.35	0.42
$d_{7,100}$	0.12	0.60	0.47	0.38
$d_{29,104}$	0.58	0.98	0.40	0.38
$d_{84,103}$	0.26	0.81	0.56	0.37
$d_{31,69}$	0.44	0.07	0.37	0.34
$d_{104,145}$	0.18	0.52	0.34	0.29
$d_{14,20}$	0.36	0.02	0.34	0.21
$d_{81,108}$	0.53	0.84	0.30	0.13

C.1.2 Functional Coordinates in Cluster 1

Table C.3 | Verified functional contacts in cluster 1 arranged starting from the open end of the mouth region towards the hinge region. sc denotes contacts in the side-chain while bb stands for contacts in the backbone.

Distance	Type
$d_{20,145}$	salt bridge
$d_{22,141}$	h-bond sc
$d_{22,137}$	salt bridge
$d_{21,142}$	hydrophobic
$d_{11,30}$	hydrophobic
$d_{20,26}$	hydrophobic
$d_{30,145}$	hydrophobic
$d_{5,60}$	salt bridge
$d_{4,64}$	hydrophobic
$d_{4,71}$	hydrophobic
$d_{4,60}$	hydrophobic
$d_{2,64}$	h-bond sc
$d_{7,71}$	hydrophobic
$d_{7,12}$	h-bond bb
$d_{8,13}$	hydrophobic
$d_{8,64}$	salt bridge
$d_{4,13}$	hydrophobic
$d_{4,29}$	hydrophobic

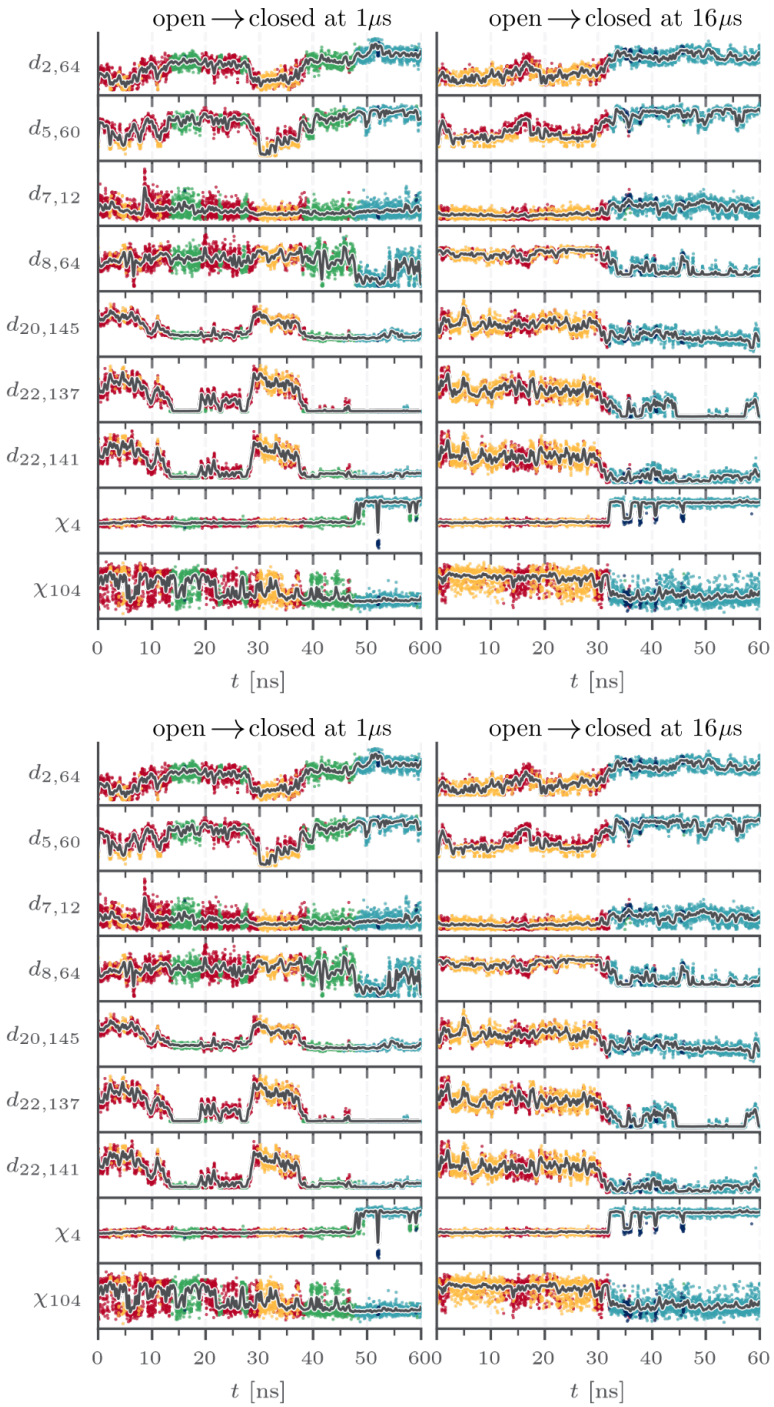


Figure C.2 | Time evolution of the nine selected coordinates of cluster 1 of the MoSAIC analysis. Shown are two transitions from open→closed and two from closed→open.

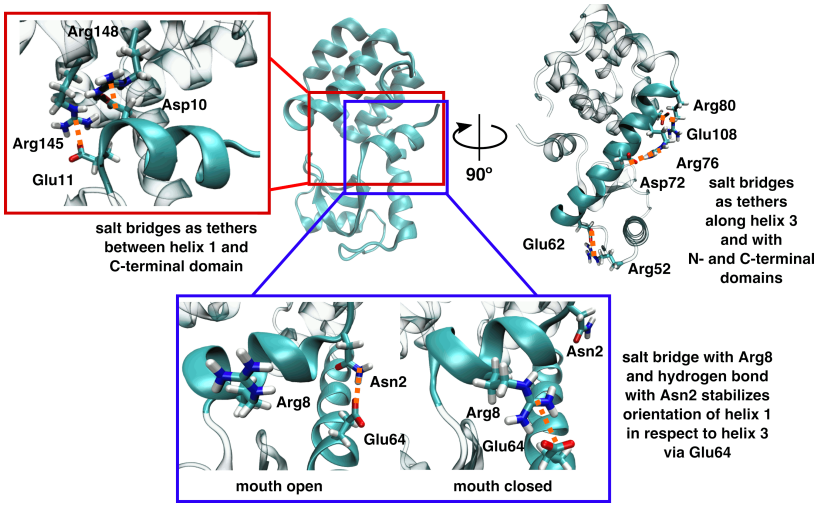


Figure C.3 | T4L structural tethers based on salt bridges, showing helix 1/C-terminal domain connection in red inset. A salt bridge complex between Asp10, Glu11, and Arginines 145 and 148 tethers helix 1 to the C domain. While the salt bridge of Arg52 and Glu62 strengthens the connection between helix 3 and the N domain, Arg80 and Glu108 do so between helix 3 and the C domain. The salt bridge between Asp72 and Arg76 strengthens helix 3 above the kink induced by Phe67. Glu64 switches between a hydrogen bond with Asn2 in the mouth open state and a salt bridge with Arg8 in the mouth closed state to stabilize the respective orientations of helix 1 in respect to helix 3.
Reprinted with kind permission from Ref. 2. Copyright © (2022) The Authors.

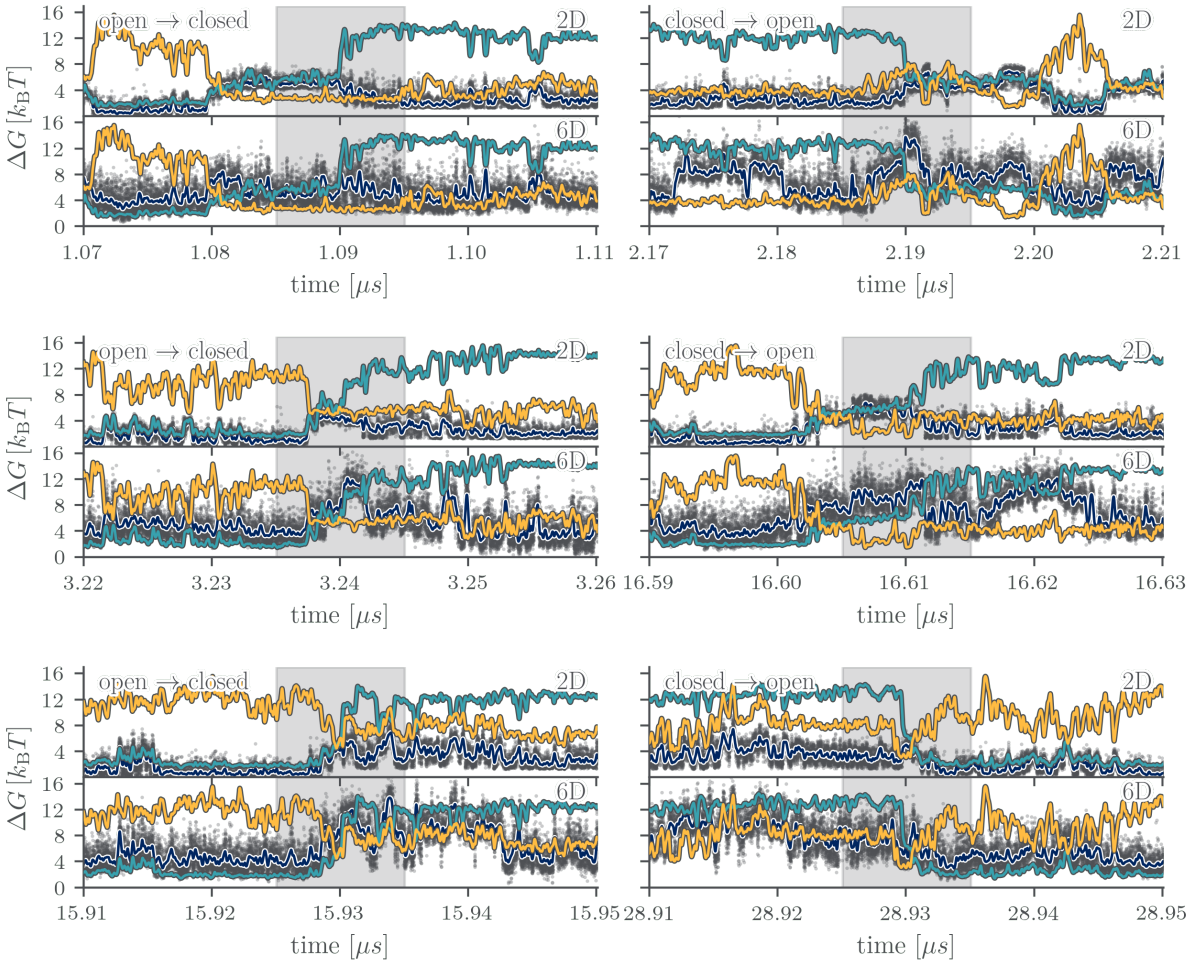


Figure C.4 | Estimation of the barrier heights of the two-dimensional (upper panel) and six-dimensional (lower panel) free energy landscape of T4L, obtained for some representative open→closed (on the left) and closed→open (on the right) transitions. Shown are the local free energy estimates for each time step as gray dots, their time average as blue line (Gaussian filter applied), the opening coordinate x as yellow line (without units) and the locking coordinate p as cyan line (without units as well). The gray windows indicate the time intervals used for averaging over all transitions in order to estimate the barrier heights. Adapted with minor changes from Ref. 2. Copyright © (2022) The Authors.

C.2 Cartesian Coordinates Analysis

C.2.1 Definition of Open and Closed Conformations

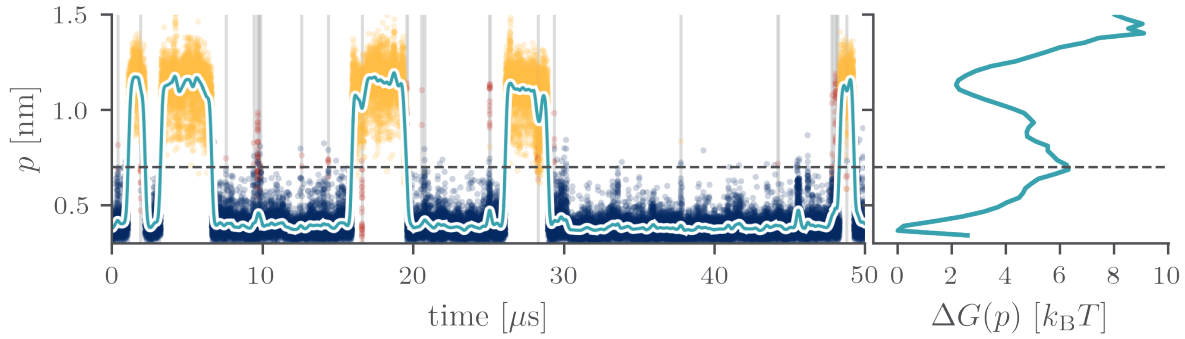


Figure C.5 | Classification of the 50 μs T4L trajectory into open (orange) and closed (blue) conformational states. (Left) Time series of the locking distance $p = d_{4,60}$ between residues 4 and 60. The cyan line shows the running average obtained through Gaussian filtering. Based on the maximum barrier height in the free energy profile at $p = 0.7$ nm (dashed gray line), we used this value as a threshold to discriminate between open ($p > 0.7$ nm) and closed ($p \leq 0.7$ nm) conformations. Short sequences that remained in one state for less than 5 ns were discarded to avoid misclassifications due to noise (gray areas). (Right) Free energy profile $\Delta G(p)$ derived from the locking distance distribution, showing the barrier separating the two conformational states.

C.2.2 Local vs. Global Correlation Fitting

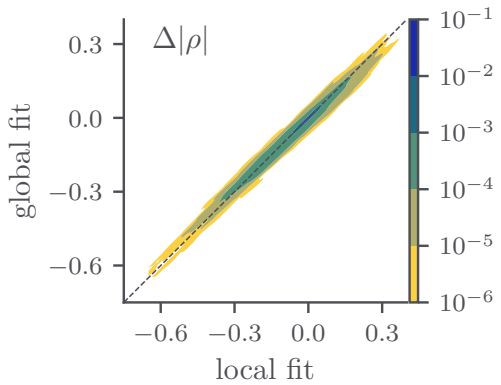


Figure C.6 | Comparison of the relative differences between the local and the global fitting procedure for T4L. Shown is the joint probability distribution of the linear correlation coefficient difference $\Delta|\rho| = |\rho^{\mathrm{closed}}|-|\rho^{\mathrm{open}}|$ computed using a local and global alignment procedure. In the local fitting procedure, the collective rotation and translation effects were eliminated by executing a Root Mean Square Deviation (RMSD) fit to the structure featuring the minimal average RMSD concerning all other frames within each distinct conformation (open and closed). Conversely, in the global fitting approach, the frame characterized by the minimum average RMSD relative to the complete trajectory was employed.

C.2.3 Cartesian Similarity Matrices of T4L

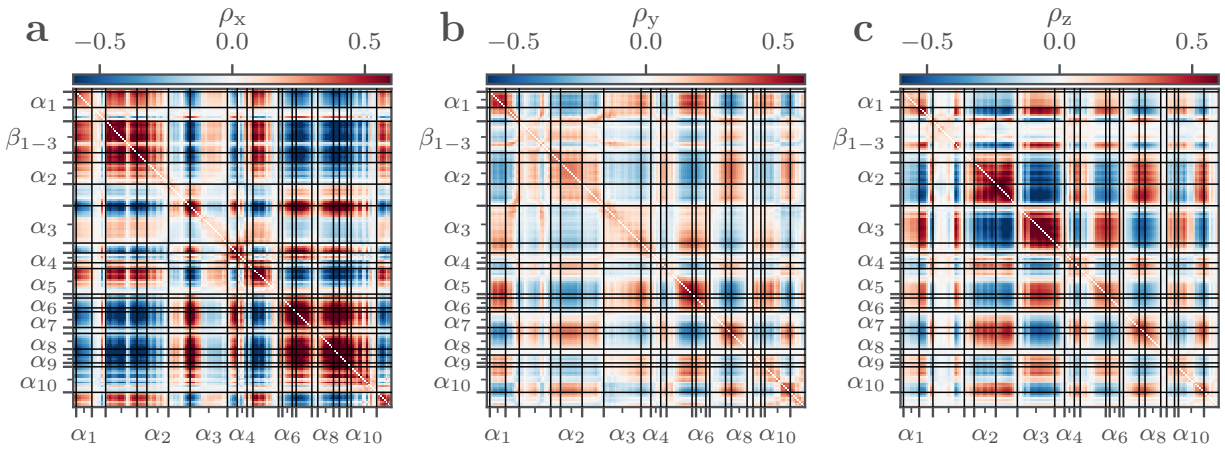


Figure C.7 | Analysis of directional components (x, y, z) of linear correlation for T4 lysozyme C_α-atoms. Some discrepancy between the linear Pearson correlation and NMI can be explained by cancellation effects between the directional components ρ_x (a), ρ_y (b), and ρ_z (c). Focusing e.g. on the correlation between α_6 and α_8 helices, we note a large different between $|\rho|$ and I_N in the figs. 5.9a and 5.13a in the main text. While these regions show positive correlation in the x-direction and negative correlations in the y- and z-directions, the opposing signs lead to cancellation in the scalar Pearson coefficient, effectively masking the true correlation that is properly captured by NMI. Adapted with minor changes from Ref. 3. Copyright © (2024) Authors.

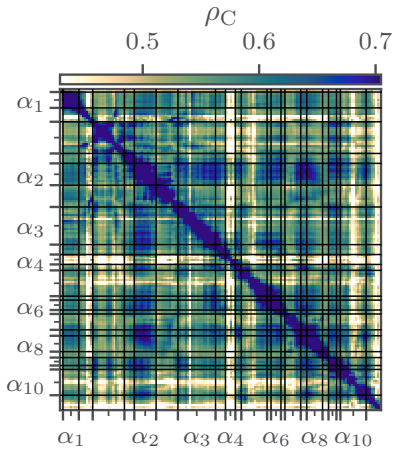


Figure C.8 | Canonical correlation matrix capturing the linear correlation between the canonical coordinates computed from Cartesian C_α-atoms of the 50 μs trajectory of T4L.[98]

C.2.4 Rigidity Analysis

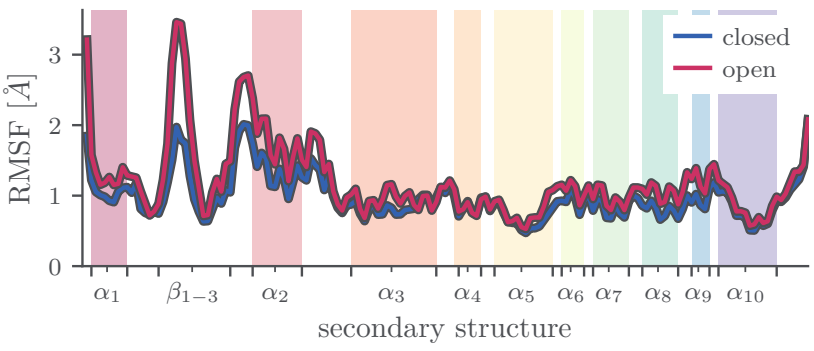


Figure C.9 | The root-mean-square-fluctuation (RMSF) $\sqrt{\langle(\boldsymbol{x}_i - \langle \boldsymbol{x}_i \rangle)^2\rangle}$ as a function for every C_α-atom in T4L. The RMSF was calculated for both the closed and open conformation. While the lower part of T4L ($\alpha_1 - \alpha_2$) shows a larger difference in RMSF in both conformations, the secondary structures within the upper part seem to be more stable. This is especially the case for the α_4 and $\alpha_1 0$ helices. Colors of the bars indicating the α-helices according to the structure shown in Fig. 5.2. Adapted with minor changes from Ref. 3. Copyright © (2024) Authors.

Supporting Information for Chapter 6

D

D.1 Self-Attention Graph Pooling

Self-attention graph pooling[315,316] (SAGPool) is a method for pooling and downsizing graphs in a way that graph topology is preferably preserved. SAGPool computes attention scores[317] for each node in the graph based on its node features and the graph topology. The attention scores $\boldsymbol{Z} \in \mathbb{R}^{N_{\text{res}}}$, where N_{res} is the number of residues in the protein/graph, are computed as[315]

$$\boldsymbol{Z} = \sigma\left(\tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}\boldsymbol{X}\Theta_{\text{att}}\right). \tag{D.1}$$

$\tilde{\boldsymbol{A}} \in \mathbb{R}^{N_{\text{res}}\times N_{\text{res}}}$ denotes the adjacency matrix with self-connections, i.e. $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_{N_{\text{res}}}$. $\tilde{\boldsymbol{D}} \in \mathbb{R}^{N_{\text{res}}\times N_{\text{res}}}$ is the degree matrix of $\tilde{\boldsymbol{A}}$, and $\boldsymbol{X} \in \mathbb{R}^{N_{\text{res}}\times F}$ is the node feature matrix with F-dimensional features (which, in our case, are the ϕ, ψ-dihedral angles). $\Theta_{\text{att}} \in \mathbb{R}^{F\times 1}$ is the only learnable parameter in the SAGPool layer. $\sigma(\cdot)$ is a nonlinear activation function, typically the tanh function.

$\boldsymbol{X}\Theta_{\text{att}}$ computes every residue's F-dimensional feature vector into a scalar value, which indicates the unweighted importance score. The learnable parameter Θ_{att} can be thought of as a trainable filter that selects the most relevant combinations of the F-dimensional features. In our case, Θ_{att} might learn that residues with a specific (ϕ, ψ)-angle conformation deserve greater "attention" than others.

In the next step, the topology of the graph is taken into account via the adjacency matrix $\tilde{\boldsymbol{A}}$. To avoid the dominance of a few highly connected nodes, the degree matrix $\tilde{\boldsymbol{D}}$ is used to normalize the adjacency matrix $\tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}$. This ensures that information from low-degree residues gets amplified ($\hat{=}$ this information carries more weight) while information from high-degree residues is dampened. Effectively, each edge weight between nodes u and v in the adjacency matrix becomes $1/\sqrt{\text{degree}_u \times \text{degree}_v}$, guaranteeing a balanced information exchange where neither highly connected nor low-degree residues dominate the attention scoring. Multiplying this normalized adjacency matrix with the projected features $\tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}\boldsymbol{X}\Theta_{\text{att}}$, we make sure that each residue's importance score is influenced by its neighborhood, meaning that a residue with a modest score, which is connected to several highly important neighbors, will receive a boosted score. In contrast, a residue with a high importance score, which is rather isolated, might get a lower score. Finally, the sigmoid activation function $\sigma(\cdot)$ introduces nonlinearity and ensures that the final (weighted) attention scores are bounded between 0 and 1.

Once we have computed attention scores for each residue, only the top k% residues with the highest scores are retained, where k is a hyperparameter. Downsizing the graph using attention scores makes sure that

the most structurally important elements and their relationships in the graph are retained.

D.2 Estimation of the Inducing Points

Identifying the time points in a time trace where the system undergoes significant changes, such as, e.g., conformational transitions in MD data, is crucial for analyzing its dynamics. In order to reliably estimate these time points in an automated fashion, we employ a *change point detection* algorithm.[324] The Pruned Exact Linear Time[323] (PELT) algorithm effectively detects significant changes in the time traces of the system by minimizing the following cost function using dynamic programming:

$$F(t) = \min_{\tau<t} \left[F(\tau) + C(\mathbf{x}_{\tau+1:t}) + \beta\right] \tag{D.2}$$

Here, $F(t)$ denotes the optimal partitioning up to time t, $\mathbf{x}_{1:T} = (\mathbf{x}_1, \dots, \mathbf{x}_\mathbf{T})$ is the (multi-dimensional and ordered) trajectory, $C(\mathbf{x}_{\tau+1:t})$ a cost function measuring the homogeneity of $\mathbf{x}$ within the segment $\tau + 1$ till t, and β is a penalty term preventing over-segmentation. This approach automatically identifies time points when significant changes (i.e., conformational changes) occur.

D.2.1 T4L MoSAIC Cluster 4

In order to identify the collective switching events in the coordinates contained in MoSAIC cluster 4 (compare Sec. 5.1.1), we apply the PELT algorithm to the first PCA projection z_1 of the corresponding coordinates. As shown in Fig. D.1, the PELT algorithm identified $m = 9$ transitions in z_1, which are shown as gray vertical lines.

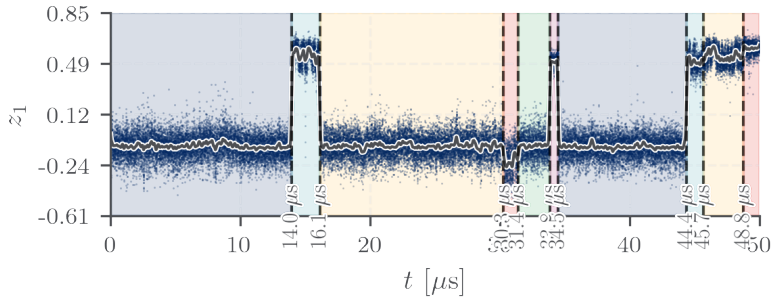


Figure D.1 | Inducing Points identified by the PELT algorithm (gray vertical lines), capturing major switching events in MoSAIC cluster 4.

D.2.2 Toy Model

As we rely on sparse approximations for the latent GP regression, we need to estimate m inducing points. In order to obtain representative time points for the dynamics of the toy model, we again apply the PELT algorithm. By relying only on data from the $xy-$plane, where states 3 and 4 overlap, we mimic a scenario in real MD simulations analysis where important degrees of freedom may be hidden or overlooked in post-simulation analysis. The PELT algorithm identified $m = 89$ inducing points, shown as gray vertical lines in the time traces in Fig. D.2.

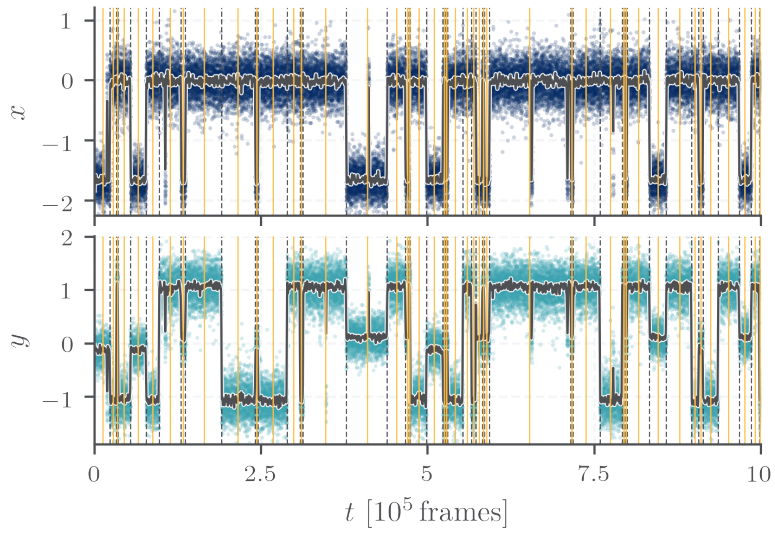


Figure D.2 | Inducing Points identified by the PELT algorithm (gray dashed vertical lines), capturing major transitions between the metastable states. Additional time points were added at the midpoint of each segment in order to reflect the metastable conformation as well (yellow vertical lines). Adapted with minor changes from Ref. 4.

D.2.3 T4L Embeddings by PCA and GNN-AE

PCA

Fig. D.3 shows the two-dimensional PCA projection of the 556 C_α distances of T4L, colored by the locking distance $d_{4,60}$. Furthermore, we show one representative transition of MoSAIC cluster 4 (at 14 μs, see Fig. D.1), where the protein switches between the two metastable open sub-states. The node's positions are averaged over 0.5 ns, and their color indicates the evolving time around the transition.

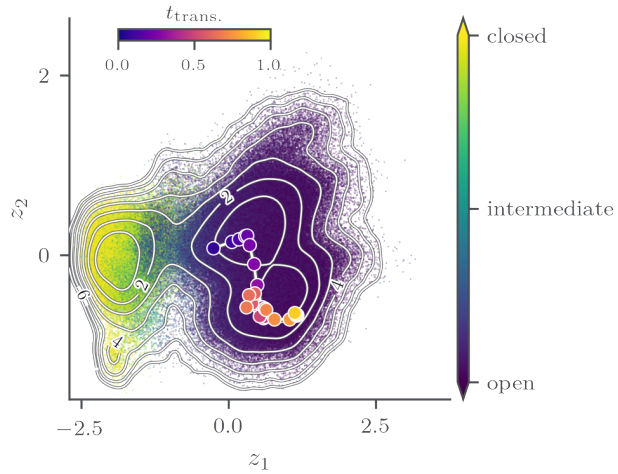


Figure D.3 | Two-dimensional PCA projection of the 556 C_α distances of T4L; the data points are colored by the locking distance $d_{4,60}$. Using the identified switching events of MoSAIC cluster 4, we visualize one representative transition, where the nodes position are a temporal averaged over 0.5 ns and their color represent the evolving time.

GNN-AE

To assess whether the differences between PCA and GNN-AE are physically meaningful, we employed HDBSCAN[201] to identify the frames belonging to some distinct free energy basins. We manually selected a subset of them and colored the corresponding areas in Fig. D.4 accordingly.

To evaluate how well these clusters resolve the conformational differences of T4L, we leveraged the four MoSAIC clusters previously identified in Sec. 5.1.1. For each MoSAIC cluster, we extracted its first principal component (where i represents the i-th MoSAIC cluster) to represent a representative coordinate for the dominant dynamics within that cluster. Even though the first PCs of each cluster caption only a part of the dynamics contained within each MoSAIC cluster—accounting for 72.7% (cluster 1), 46.4% (cluster 2), 59.8% (cluster 3), and 64.2% (cluster 4) of the total variance—they should still provide a good approximation of the respective dynamics.

In Fig. D.5, we show the free energy along these PCs in the lower diagonal as well as how the HDBSCAN clusters are distributed along these PCs in the upper diagonal. The conformational analysis reveals a distinct separation between the identified basins. Considering the open basin of T4L with clusters C1-C3, and C6, it is evident that C6 and C3 are clearly distinguishable from each other and from C1 and C2. While C1 and C2 show substantial overlap in most of the projections, they indicate to capture slightly different regions in the $PC_1^{(2)}$-$PC_1^{(3)}$ projection. For the closed basin of T4L consisting of clusters C4, C5, and C7, C4 is clearly distinct from C5 and C7 along $PC_1^{(4)}$, where C5 and C7 are partly overlapping.

Although this analysis examines only the first principal components of each MoSAIC cluster, the observed separation of the HDBSCAN clusters provides strong evidence for the physical meaningfulness of the GNN-AE representation. Some clusters may exhibit more distinct separation in higher-order PCs or in different parts of the protein not captured by the four MoSAIC clusters analyzed here.

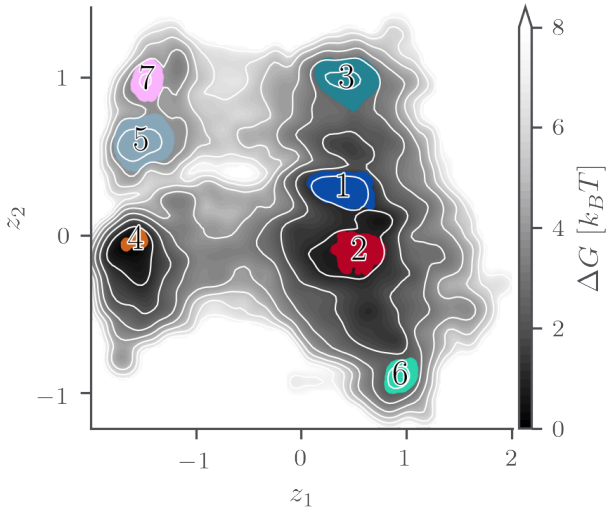


Figure D.4 | Evaluation of the two-dimensional latent space of the GNN-AE, trained on the 556 C_α distances of T4L. HDBSCAN[201] was used to identify seven distinct free energy basins.

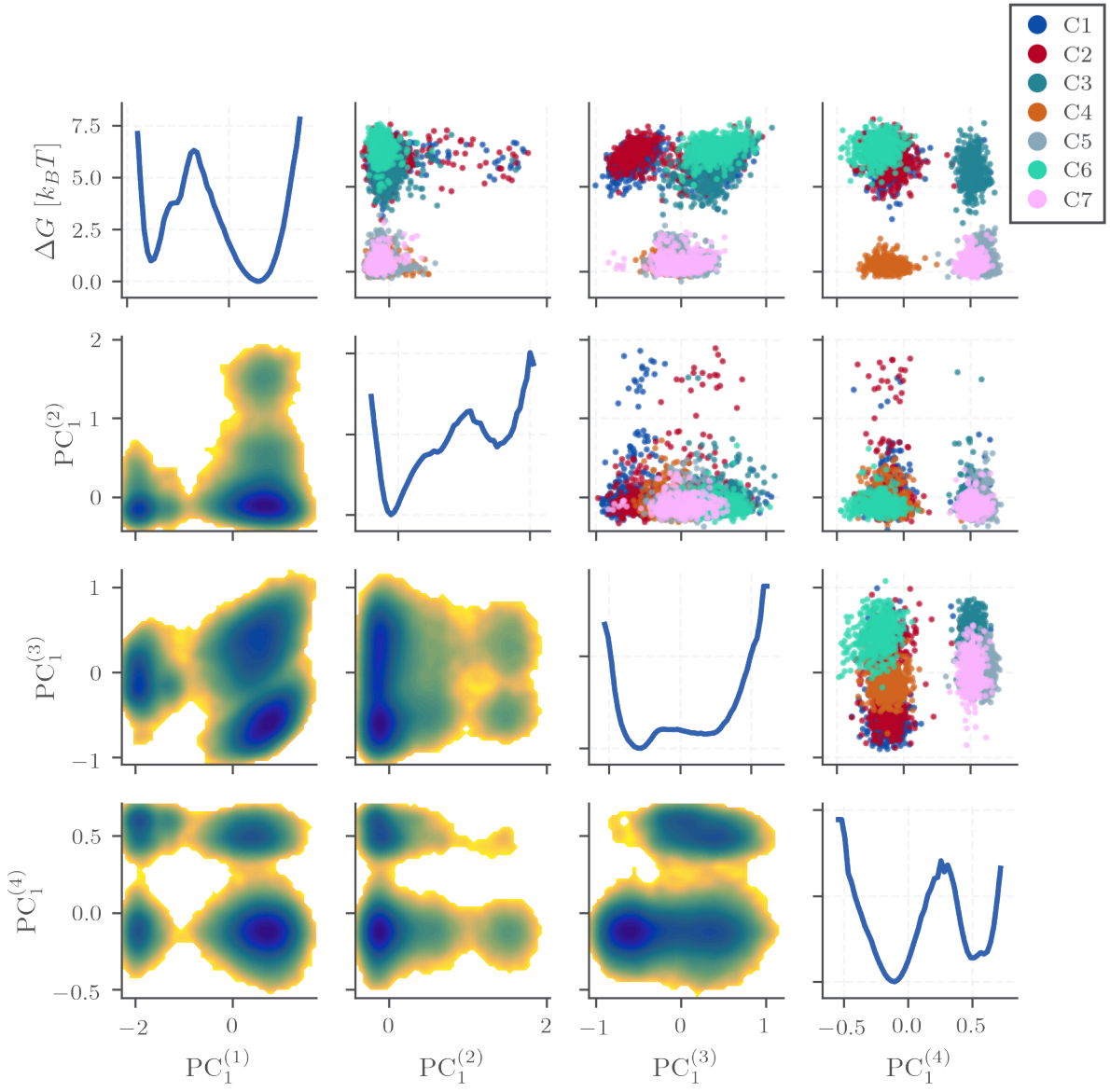


Figure D.5 | Evaluation of GNN-AE basins separation along principal components derived from our MoSAIC analysis. Lower diagonal shows free energy landscapes projected onto pairs of principal components $\mathrm{PC}_1^{(i)}$ from MoSAIC cluster i. Upper diagonal displays the distribution of the seven HDBSCAN clusters (C1-C7) of Fig. D.4 in the same PC spaces, with diagonal elements showing one-dimensional free energy profiles.

D.2.4 GNN-AE Architecture Specifications

For the GNN-AE architecture we used for T4L (Sec. 6.1.3), we used the following parameters:

- **Input Features:**
  - Node features: 4D (backbone dihedral angles ϕ, ψ), represented in the 2D vector space $\mathbb{R}^2$ as, e.g., $\phi \mapsto [\sin(\phi), \cos(\phi)]$. This avoids discontinuities at the boundaries 360° and 0°.[150]
  - Edge features: 1D (inverse C_α distances)
  - Graph size: 164 nodes, 556 edges
- **Encoder Layers:**
  - EdgeAttention module: 1D → 80D → 1D weighted edges
  - GATConv: 4D → 80D with multi-head attention[347]
  - SAGPooling: Graph coarsening with attention-based node selection[315]
  - LeakyReLU activation + BatchNorm1D(80D)
  - Set2Set aggregation: 80D → 160D with 5 processing steps[318]
  - Linear projection: 160D → 2D latent space
- **Decoder Layers:**
  - Linear: 2D → 50D
  - Linear: 50D → 556D (reconstructed C_α distances)
  - Dropout: $p = 0.5$ for regularization
- **Training:**
  - Mean Squared Error (MSE) loss for reconstruction
  - Learning rate: 1×10^{-4} using Adam optimizer[175]
  - trained for 100 epochs (converged)

Batch normalization (BatchNorm1D) is used to standardize the input between the different layers by re-centering them around zero and rescaling the data to unit variance. This allows faster training while increasing stability as it reduces the risk of vanishing or exploding gradients.[348] Dropout sets a fraction of the nodes to zero during training to zero with probability p, which helps to avoid overfitting as it forces the model to learn patterns across multiple nodes and edges rather than relying on specific single ones.[349]

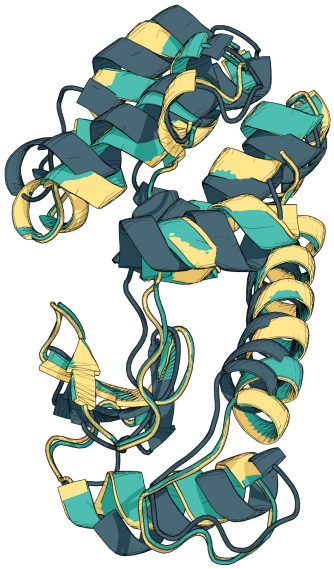


Figure D.6 | PyRosetta structure generation of T4L from GNN-AE C_α-distances restraints.

D.2.5 PyRosetta Structure Generation Validation

PyRosetta[328] was used to compute the three-dimensional structure of proteins from harmonic constraints resulting from the GNN-AE reconstruction. As a benchmark, we computed the 556 C_α-distances from a closed reference structure shown in bright green and used the open conformation (dark green) as a starting point for PyRosetta. The resulting structure generated by PyRosetta applying the harmonic constraints is shown in yellow and closely matches the closed reference structure in green.

D.3 Derivation of the Sparse GP-VAE Loss Function

Following refs. 133, 332, 333, 339, 340, 341, we summarize the derivation of the GP-VAE loss function in Eq. (6.5). For the sake of brevity, we will use the notation $\langle\cdot\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)}$ instead of $\langle\cdot\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x},t)}$ when referring to expectations over distributions to improve readability.

Our starting point is Eq. (6.4):

$$\mathcal{L} = \left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})} - \beta D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x},t)\,\|\,p(\boldsymbol{z}|t)\right],$$

of which we will first focus on the Kullback-Leibler (KL) divergence term

$$D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x},t)\,\|\,p(\boldsymbol{z}|t)\right] = \left\langle \ln \frac{q(\boldsymbol{z}|\boldsymbol{x},t)}{p(\boldsymbol{z}|t)} \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)} = -\left\langle \ln \frac{p(\boldsymbol{z}|t)}{q(\boldsymbol{z}|\boldsymbol{x},t)} \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)},$$

where both the approximate posterior $q(\boldsymbol{z}|\boldsymbol{x},t)$ and the prior $p(\boldsymbol{z}|t) = \mathrm{GP}\left[0, k_\nu(t,t')\right]$ are time-dependent in contrast to the classical VAE. To make the inference tractable, we follow the variational approximation introduced by Pearce,[341] which factorizes the posterior into

$$q(\boldsymbol{z}|\boldsymbol{x},t) = \frac{p(\boldsymbol{z}|t)\tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x})}{Z(\boldsymbol{x},t)}.$$

This factorization separates the temporal GP prior $p(\boldsymbol{z}|t)$ from the data-driven part $\tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\sigma}}^2)$, which remains identical to the classical VAE encoder, where $\tilde{\boldsymbol{\mu}} = f_\mu[f_{\boldsymbol{\theta}}(\boldsymbol{x})]$ and $\tilde{\boldsymbol{\sigma}}^2 = f_\sigma[f_{\boldsymbol{\theta}}(\boldsymbol{x})]$ are learned by neural networks. The normalization constant $Z(\boldsymbol{x},t) = \int \mathrm{d}\tilde{\boldsymbol{z}}\, p(\boldsymbol{z}|t)\tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x})$[i] ensures that the posterior distribution $q(\boldsymbol{z}|\boldsymbol{x},t)$ is a valid probability distribution. Substituting this factorization into the KL term, we obtain

$$\begin{aligned} D_{\mathrm{KL}}\left[q(\boldsymbol{z}|\boldsymbol{x},t)\,\|\,p(\boldsymbol{z}|t)\right] &= -\left\langle \ln \frac{p(\boldsymbol{z}|t)Z(\boldsymbol{x},t)}{p(\boldsymbol{z}|t)\tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x})} \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)} \\ &= \left\langle \ln \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)} - \ln Z(\boldsymbol{x},t), \end{aligned}$$

which we can use to rewrite our starting point equation as

$$\mathcal{L} = \underbrace{\left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x})}}_{\text{reconstruction}} - \beta\Big[\underbrace{\left\langle \ln \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)}}_{\text{GP regularization}} - \underbrace{\ln Z(\boldsymbol{x},t)}_{\text{normalization}}\Big]. \tag{D.3}$$

The direct computation of Eq. (D.3) is still computationally prohibitive, since both the expectation $\left\langle \ln \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)}$ and the normalization constant $Z(\boldsymbol{x},t)$ involve operations on the full kernel matrix that scale as $\mathcal{O}(N^3)$. Therefore, it is necessary to use sparse approximations for the GP,[339,340] which employ a reduced set of $m \ll N$ inducing points with vectors $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m] \in \mathbb{R}^m$ that are representative of the data.[ii] The premise here is that a GP regression based on these inducing points $\boldsymbol{f}_m \equiv f(\boldsymbol{U}) \sim \mathcal{N}(f(\boldsymbol{U})|\boldsymbol{\mu}, \boldsymbol{A})$ faithfully approximates the full GP regression $\boldsymbol{f}_N$. Here, $p(\boldsymbol{f}_m) = \mathcal{N}(\boldsymbol{f}_m|0, \boldsymbol{K}_{mm})$ denotes the GP prior over the inducing points, while $q(\boldsymbol{f}_m|\boldsymbol{x},t) = \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{\mu}, \boldsymbol{A})$ denotes the variational posterior distribution over these same inducing points, with $\boldsymbol{\mu}$ and $\boldsymbol{A}$ being learnable variational parameters. Based on Titsias work,[339] Jazbec *et al.* suggested to compute these (intermediate) variational quantities at

[i] both $\boldsymbol{z}$ and $\tilde{\boldsymbol{z}}$ denote the same latent embedding, however we notationally distinguish between the purely data-driven $\tilde{\boldsymbol{z}}$ as in classical VAEs and the time-dependent posterior $\boldsymbol{z}$ resulting from the GP regularization.

[ii] To estimate appropriate inducing points, we used change point detection to identify the time points in which the system undergoes significant conformational changes (see Sec. D.2). In order to make them more representative, we also added midpoints to reflect the metastable states.

the m inducing points as stochastic estimates for each latent dimension $l \in \{1, \dots, L\}$ and for each batch b:[133]

$$\boldsymbol{\mu}_b^l = \frac{N}{b} \boldsymbol{K}_{mm} \left(\boldsymbol{\Sigma}_b^l\right)^{-1} \boldsymbol{K}_{mb} \mathrm{diag}\left(\tilde{\sigma}_{b,l}^{-2}\right) \tilde{\boldsymbol{\mu}}_b^l,$$
$$\boldsymbol{A}_b^l = \boldsymbol{K}_{mm} \left(\boldsymbol{\Sigma}_b^l\right)^{-1} \boldsymbol{K}_{mm},$$

where $\boldsymbol{\Sigma}_l^b$ is given as

$$\boldsymbol{\Sigma}_b^l = \boldsymbol{K}_{mm} + \frac{N}{b} \boldsymbol{K}_{mb} \mathrm{diag}\left(\tilde{\sigma}_{b,l}^{-2}\right) \boldsymbol{K}_{bm}.$$

In this expression, $\boldsymbol{K}_{mb} = k_{\nu,l}(\boldsymbol{U}, \boldsymbol{t}_b) \in \mathbb{R}^{m \times b}$ represents the kernel matrix computed between the m inducing points $\boldsymbol{U}$ and the b data points $\boldsymbol{t}_b$ within current the batch. Importantly, these estimators converge to the true values for very large batch sizes $b \to N$. Following Titsias and Hensman *et al.*, we obtain the final posterior distribution parameters in form of the posterior mean $\boldsymbol{m}$ and covariance $\boldsymbol{B}$ of the GP latent embedding at all data points in the batch b

$$\boldsymbol{m}_b^l = \frac{N}{b} \boldsymbol{K}_{bm} \left(\boldsymbol{\Sigma}_b^l\right)^{-1} \boldsymbol{K}_{mb} \mathrm{diag}\left(\tilde{\sigma}_{b,l}^{-2}\right) \tilde{\boldsymbol{\mu}}_b^l,$$
$$\boldsymbol{B}_b^l = \mathrm{diag}\left(\boldsymbol{K}_{bb} - \boldsymbol{K}_{bm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mb} + \boldsymbol{K}_{bm} \left(\boldsymbol{\Sigma}_b^l\right)^{-1} \boldsymbol{K}_{mb}\right).$$

Since we assumed independence across latent dimensions, we can characterize the posterior distribution of the latent embedding

$$q(\boldsymbol{z}|\boldsymbol{x}, t) = \prod_{l=1}^{L} q(\boldsymbol{z}^l|\boldsymbol{x}, t) = \mathcal{N}(\boldsymbol{m}, \boldsymbol{B}). \tag{D.4}$$

This allows us to calculate the term $\left\langle \ln \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)}$ in Eq. (D.3) as

$$\begin{aligned} \left\langle \ln \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x},t)} &= \int \mathrm{d}\boldsymbol{z}\, \mathcal{N}(\boldsymbol{m}, \boldsymbol{B}) \ln \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\sigma}^2) \\ &= \mathrm{CE}\left[\mathcal{N}(\boldsymbol{m}, \boldsymbol{B}) \| \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\sigma}^2)\right] \end{aligned}$$

The cross-entropy (CE) between two Gaussian distributions can be analytically computed as[333]

$$\begin{aligned} \mathrm{CE}\left[\mathcal{N}(\boldsymbol{m}, \boldsymbol{B}) \| \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\sigma}^2)\right] = \frac{1}{2} \{ & L \ln 2\pi + \ln \det \mathrm{diag}(\tilde{\sigma}^2) \\ & + (\boldsymbol{m} - \tilde{\boldsymbol{\mu}})^\top \mathrm{diag}(\tilde{\sigma}^{-2}) (\boldsymbol{m} - \tilde{\boldsymbol{\mu}}) \\ & + \mathrm{tr}\left[\mathrm{diag}(\boldsymbol{B}) \mathrm{diag}(\tilde{\sigma}^{-2})\right] \}, \end{aligned} \tag{D.5}$$

of which we can compute all quantities: $\tilde{\boldsymbol{\mu}}$ and $\tilde{\sigma}^2$ are the variational parameters learned by the classical VAE encoder and $\boldsymbol{m}$ and $\boldsymbol{B}$ are the posterior mean and covariance of the GP latent embedding at all data points.

Lastly, the normalization term $Z(\boldsymbol{x}, t)$ remains to be calculated. Recall from the factorization, that the direct calculation of $Z(\boldsymbol{x}, t) = \int \mathrm{d}\tilde{\boldsymbol{z}}\, p(\boldsymbol{z}|t) \tilde{q}(\tilde{\boldsymbol{z}}|\boldsymbol{x})$ is intractable, since it involves the full kernel matrix in $p(\boldsymbol{z}|t)$. Following Hensman *et al.*,[340] this intractability can be circumvented by constructing an Evidence Lower BOund (ELBO) that 1.) serves as a tractable lower

bound to $\ln Z(\boldsymbol{x}, t)$ and 2.) can be computed using mini-batches

$$\ln Z(\boldsymbol{x}, t) \geq \mathcal{L}_{\mathrm{H}} = \sum_{i=1}^{N} \left\{ \ln \mathcal{N}\left(\tilde{\mu}_i \mid \boldsymbol{k}_i \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu}, \tilde{\sigma}_i^2\right) - \frac{1}{2\tilde{\sigma}_i^2}\left[\tilde{k}_{ii} + \operatorname{Tr}(\boldsymbol{A}\boldsymbol{\Lambda}_i)\right] \right\} - D_{\mathrm{KL}}\left[q(\boldsymbol{f}_m|\boldsymbol{x}, t) \,\|\, p(\boldsymbol{f}_m)\right].$$

Here, $\boldsymbol{k}_i$ represents the i-th row of $\boldsymbol{K}_{Nm}$, $\tilde{k}_{ii}$ denotes the i-th diagonal element of $\boldsymbol{K}_{NN} - \boldsymbol{K}_{Nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mN}$, and $\boldsymbol{\Lambda}_i = \boldsymbol{K}_{mm}^{-1}\boldsymbol{k}_i\boldsymbol{k}_i^\top\boldsymbol{K}_{mm}^{-1}$. $p(\boldsymbol{f}_m)$ and $q(\boldsymbol{f}_m|\boldsymbol{x}, t)$ are the sparse GP prior and posterior over the inducing points and were defined above. This ELBO contains a KL divergence term between the variational posterior and GP prior over the inducing variables. To compute this tractably, we evaluate:

$$\begin{aligned} D_{\mathrm{KL}}\left[q(\boldsymbol{f}_m|\boldsymbol{x}, t) \,\|\, p(\boldsymbol{f}_m)\right] &= \left\langle \ln \frac{q(\boldsymbol{f}_m|\boldsymbol{x}, t)}{p(\boldsymbol{f}_m)} \right\rangle_{q(\boldsymbol{f}_m|\boldsymbol{x}, t)} \\ &= \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{\mu}, \boldsymbol{A}) \right\rangle_{q(\cdot)} - \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|0, \boldsymbol{K}_{mm}) \right\rangle_{q(\cdot)}. \end{aligned} \tag{D.6}$$

Using the standard expression for the log-probability of a Gaussian distribution, we can compute the first term as

$$\begin{aligned} \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{\mu}, \boldsymbol{A}) \right\rangle_{q(\cdot)} &= \left\langle -\frac{1}{2}\left[(\boldsymbol{f}_m - \boldsymbol{\mu})^\top \boldsymbol{A}^{-1}(\boldsymbol{f}_m - \boldsymbol{\mu}) + \ln\det \boldsymbol{A} + m \ln 2\pi\right] \right\rangle_{q(\cdot)} \\ &= -\frac{1}{2}\left[m + \ln\det \boldsymbol{A} + m \ln 2\pi\right]. \end{aligned} \tag{D.7}$$

Similarly, we can simplify the second term:

$$\begin{aligned} \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|0, \boldsymbol{K}_{mm}) \right\rangle_{q(\cdot)} &= \left\langle -\frac{1}{2}\left(\boldsymbol{f}_m^\top \boldsymbol{K}_{mm}^{-1}\boldsymbol{f}_m + \ln\det \boldsymbol{K}_{mm} + m \ln 2\pi\right) \right\rangle_{q(\cdot)}, \\ &= -\frac{1}{2}\left[\operatorname{tr}(\boldsymbol{K}_{mm}^{-1}\boldsymbol{A}) + \boldsymbol{\mu}^\top \boldsymbol{K}_{mm}^{-1}\boldsymbol{\mu} + \ln\det \boldsymbol{K}_{mm} + m \ln 2\pi\right], \end{aligned} \tag{D.8}$$

where we used the expectation of the quadratic form of a Gaussian. Substituting Eq. (D.7) and Eq. (D.8) back into Eq. (D.6), we obtain

$$\begin{aligned} D_{\mathrm{KL}}\left[q(\boldsymbol{f}_m|\boldsymbol{x}, t) \,\|\, p(\boldsymbol{f}_m)\right] &= \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|\boldsymbol{\mu}, \boldsymbol{A}) \right\rangle_{q(\boldsymbol{f}_m)} - \left\langle \ln \mathcal{N}(\boldsymbol{f}_m|0, \boldsymbol{K}_{mm}) \right\rangle_{q(\boldsymbol{f}_m)} \\ &= \frac{1}{2}\left[-m + \operatorname{tr}(\boldsymbol{K}_{mm}^{-1}\boldsymbol{A}) + \boldsymbol{\mu}^\top \boldsymbol{K}_{mm}^{-1}\boldsymbol{\mu} + \ln \frac{\det \boldsymbol{K}_{mm}}{\det \boldsymbol{A}}\right]. \end{aligned} \tag{D.9}$$

Having derived the KL divergence term analytically, we can now return to the Hensman ELBO and reformulate it relying only on computable quantities:

$$\begin{aligned} \mathcal{L}_{\mathrm{H}} = \sum_{i=1}^{N} \left\{ \ln \mathcal{N}\left(\tilde{\mu}_i \mid \boldsymbol{k}_i \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu}, \tilde{\sigma}_i^2\right) - \frac{1}{2\tilde{\sigma}_i^2}\left[\tilde{k}_{ii} + \operatorname{Tr}(\boldsymbol{A}\boldsymbol{\Lambda}_i)\right] \right\} \\ - \frac{1}{2}\left[-m + \operatorname{tr}(\boldsymbol{K}_{mm}^{-1}\boldsymbol{A}) + \boldsymbol{\mu}^\top \boldsymbol{K}_{mm}^{-1}\boldsymbol{\mu} + \ln \frac{\det \boldsymbol{K}_{mm}}{\det \boldsymbol{A}}\right], \end{aligned} \tag{D.10}$$

where we have substituted our derived expression for the KL divergence from Eq. (D.9).

This completes the derivation of all necessary components. We can now return to Eq. (D.3) and combine the three tractable terms we derived

- reconstruction term (unchanged from standard VAE)
- GP regularization: Cross-entropy term between sparse GP posterior and VAE encoder [Eq. (D.5)]
- normalization: Hensman ELBO $\mathcal{L}_{\mathrm{H}}$ [Eq. (D.10)]

into the final loss function, serving as a lower bound to the log evidence of the sparse GP-VAE model

$$\Rightarrow \mathcal{L}_{\text{GP-VAE}} \equiv \left\langle \ln p(\boldsymbol{x}|\boldsymbol{z}) \right\rangle_{q(\boldsymbol{z}|\boldsymbol{x})} - \beta \left[\mathrm{CE}\left[\mathcal{N}(\boldsymbol{m}, \boldsymbol{B}) \| \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\sigma}}^2) \right] - \mathcal{L}_{\mathrm{H}} \right].$$

This is the result presented in Eq. (6.5) in the main text.

D.4 Parameters for the GP-VAE

For the GP-VAE toy-model experiments, we implemented the model using PyTorch[10] with the following architecture and hyperparameters:

- Neural Network Architecture:
  - network with hidden dimensions 10-32-64-32-10
- Training configuration:
  - Optimizer: AdamW algorithm[176]
  - Learning rate: 10^{-5}
  - Weight decay: 10^{-2}
  - Batch size: 5000
  - Training epochs: 100
- Model Hyperparameters:
  - KL div. weight β: 20
  - Matérn kernel smoothness parameter ν: 3/2
  - Matérn kernel length scale l: $7.5 \cdot 10^4$
  - $m = 89$ inducing points (see Sec. D.2)

D.5 Most Probable Path Coarse Graining

To systematically coarse grain the microstates into a few macrostates, we employed a hierarchical lumping procedure based on metastability and transition probabilities, namely the most probable path (MPP) algorithm.[121] MPP progressively hierarchically combines microstates into larger macrostates by increasing a minimum metastability threshold $Q_{\min}$ from 0 to 1. Once a microstate i has a metastability $T_{ii} < Q_{\min}$, this microstate is merged with another (branch of) microstate(s) j that has the highest transition probability $\max_j T_{i\to j}(\tau)$.

This lumping procedure can be visualized as a dendrogram, which shows at which metastability two microstates are merged. In Fig. D.7 a, we show the dendrogram for the lumping of the microstates of our analytical toy model (see Sec. 6.4.3) in the full three-dimensional space and in b for the GP-VAE embedding space.

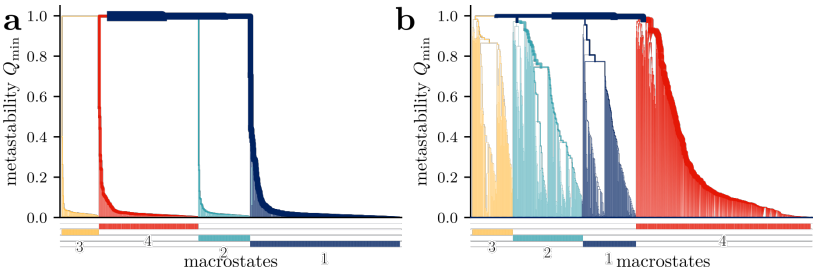


Figure D.7 | (a) Hierarchical clustering of microstates in the full three-dimensional space and (b) corresponding lumping in the GP-VAE embedding space.

Bibliography

List of all references in citation order.

[1] G. Diez, D. Nagel, and G. Stock, "Correlation-based feature selection to identify functional dynamics in proteins," J. Chem. Theory Comput. **18**, 5079–5088 (2022).

[2] M. Post, B. Lickert, G. Diez, S. Wolf, and G. Stock, "Cooperative protein allosteric transition mediated by a fluctuating transmission network," J. Mol. Bio. **434**, 167679 (2022).

[3] D. Nagel, G. Diez, and G. Stock, "Accurate estimation of the normalized mutual information of multidimensional data," J. Chem. Phys. **161**, 054108 (2024).

[4] G. Diez, N. Dethloff, and G. Stock, "Recovering hidden degrees of freedom using Gaussian processes," J. Chem. Phys. **163**, 124105 (2025).

[5] D. Nagel, *Prettypyplot: publication ready matplotlib figures made simple*, version v0.10.0, 2023.

[6] J. D. Hunter, "Matplotlib: A 2D graphics environment," Comput. Sci. Eng. **9**, 90–95 (2007).

[7] C. R. Harris *et al.* "Array programming with NumPy," Nature **585**, 357–362 (2020).

[8] P. Virtanen *et al.* "SciPy 1.0: Fundamental algorithms for scientific computing in python," Nat. Methods **17**, 261–272 (2020).

[9] F. Pedregosa *et al.* "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res. **12**, 2825–2830 (2011).

[10] A. Paszke, "Pytorch: An imperative style, high-performance deep learning library," arXiv:1912.01703 (2019).

[11] D. Nagel and G. Stock, "Msmhelper: A Python package for Markov state modeling of protein dynamics," J. Open Source Soft. **8**, 5339 (2023).

[12] Schrödinger, LLC, "The PyMOL molecular graphics system, version 2.4," 2020.

[13] K. Dill, R. L. Jernigan, and I. Bahar, *Protein actions: principles and modeling* (Garland Science, 2017).

[14] B. Alberts *et al. Essential cell biology* (Garland Science, 2015).

[15] A. V. Finkelstein and O. Ptitsyn, *Protein physics: A course of lectures* (Elsevier, 2016).

[16] E. Brini, C. Simmerling, and K. Dill, "Protein storytelling through physics," Science **370**, eaaz3041 (2020).

[17] R. J. O'brien and P. C. Wong, "Amyloid precursor protein processing and Alzheimer's disease," Annu. Rev. Neurosci. **34**, 185–204 (2011).

[18] M. Gómez-Benito *et al.* "Modeling Parkinson's disease with the alpha-synuclein protein," Front. pharmacol. **11**, 356 (2020).

[19] R. J. Harding and Y.-f. Tong, "Proteostasis in Huntington's disease: Disease mechanisms and therapeutic opportunities," Acta Pharmacol. Sin. **39**, 754–769 (2018).

[20] E. Giusto, T. Yacoubian, E. Greggio, and L. Civiero, "Pathways to Parkinson's disease: A spotlight on 14-3-3 proteins," NPJ Parkinson's dis. **7**, 85 (2021).

[21] K. Manolakou *et al.* "Genetic and environmental factors modify bovine spongiform encephalopathy incubation period in mice," Proc. Natl. Acad. Sci. USA **98**, 7402–7407 (2001).

[22] E. Karran and B. De Strooper, "The amyloid hypothesis in Alzheimer disease: New insights from new therapeutics," Nat. Rev. Drug Discov. **21**, 306–318 (2022).

[23] Y. Zhang *et al.* "Amyloid β-based therapy for Alzheimer's disease: Challenges, successes and future," Signal Transduct. Target. Ther. **8**, 248 (2023).

[24] G. J. Mulder, "Sur la composition de quelques substances animales," Bull. Sci. phys. nat. neerl. **104**, 9 (1838).

[25] J. B. Sumner, "Note. The recrystallization of urease," J. Biol. Chem. **70**, 97–98 (1926).

[26] F. Sanger, E. Thompson, and R. Kitai, "The amide groups of insulin," Biochem. J. **59**, 509 (1955).

[27] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker, "Protein structure prediction using rosetta," in *Methods in enzymology*, Vol. 383 (Elsevier, 2004), pp. 66–93.

[28] J. Jumper *et al.* "Highly accurate protein structure prediction with alphafold," Nature **596**, 583–589 (2021).

[29] J. Abramson *et al.* "Accurate structure prediction of biomolecular interactions with alphafold 3," Nature, 1–3 (2024).

[30] J. C. Kendrew *et al.* "A three-dimensional model of the myoglobin molecule obtained by X-ray analysis," Nature **181**, 662–666 (1958).

[31] M. F. Perutz *et al.* "Structure of hæmoglobin: a three-dimensional Fourier synthesis at 5.5-Å. resolution, obtained by X-ray analysis," Nature **185**, 416–422 (1960).

[32] K. A. Dill and J. L. MacCallum, "The protein-folding problem, 50 years on," Science **338**, 1042–1046 (2012).

[33] C. B. Anfinsen, "Principles that govern the folding of protein chains," Science **181**, 223–230 (1973).

[34] L. Rebecchi *et al.* "Resistance of the anhydrobiotic eutardigrade Paramacrobiotus richtersi to space flight (LIFE–TARSE mission on FOTON-M3)," J. Zool. Syst. Evol. Res. **49**, 98–103 (2011).

[35] T. C. Boothby *et al.* "Tardigrades use intrinsically disordered proteins to survive desiccation," Mol. Cell **65**, 975–984 (2017).

[36] V. N. Uversky and A. K. Dunker, "Understanding protein non-folding," Biochim. Biophys. Acta - Proteins Proteom. **1804**, 1231–1264 (2010).

[37] R. Van Der Lee *et al.* "Classification of intrinsically disordered regions and proteins," Chem. Rev. **114**, 6589–6631 (2014).

[38] P. E. Wright and H. J. Dyson, "Intrinsically disordered proteins in cellular signalling and regulation," Nat. Rev. Mol. Cell Biol. **16**, 18–29 (2015).

[39] K. Dill and S. Bromberg, *Molecular driving forces: statistical thermodynamics in biology, chemistry, physics, and nanoscience* (Garland Science, 2010).

[40] M. Saunders, A. Wishnia, and J. G. Kirkwood, "The nuclear magnetic resonance spectrum of ribonuclease1," J. Am. Chem. Soc. **79**, 3289–3290 (1957).

[41] W. P. Aue, E. Bartholdi, and R. R. Ernst, "Two-dimensional spectroscopy. Application to nuclear magnetic resonance," J. Chem. Phys. **64**, 2229–2246 (1976).

[42] K. Wüthrich, "Protein structure determination in solution by NMR spectroscopy.," J. Biol. Chem. **265**, 22059–22062 (1990).

[43] D. Marion, "An introduction to biological NMR spectroscopy," Mol. Cell. Proteom. **12**, 3006–3025 (2013).

[44] K. A. Taylor and R. M. Glaeser, "Electron diffraction of frozen, hydrated protein crystals," Science **186**, 1036–1037 (1974).

[45] M. Van Heel and J. Frank, "Use of multivariates statistics in analysing the images of biological macromolecules," Ultramicroscopy **6**, 187–194 (1981).

[46] E. Nogales, "The development of cryo-EM into a mainstream structural biology technique," Nat. Methods **13**, 24–27 (2016).

[47] R. Berera, R. van Grondelle, and J. T. Kennis, "Ultrafast transient absorption spectroscopy: Principles and application to photosynthetic systems," Photosynth. Res. **101**, 105–118 (2009).

[48] H. N. Chapman *et al.* "Femtosecond X-ray protein nanocrystallography," Nature **470**, 73–77 (2011).

[49] P. M. Kraus *et al.* "The ultrafast X-ray spectroscopic revolution in chemical dynamics," Nat. Rev. Chem. **2**, 82–94 (2018).

[50] B. Buchli *et al.* "Kinetic response of a photoperturbed allosteric protein," Proc. Natl. Acad. Sci. U.S.A. **110**, 11725–11730 (2013).

[51] J. A. McCammon, B. R. Gelin, and M. Karplus, "Dynamics of folded proteins," Nature **267**, 585–590 (1977).

[52] M. Karplus and J. A. McCammon, "Molecular dynamics simulations of biomolecules," Nat. Struct. Biol. **9**, 646–652 (2002).

[53] H. J. Berendsen, "Simulating the physical world," Simulating the Physical World, 540 (2004).

[54] M. Karplus and J. Kuriyan, "Molecular dynamics and protein function," Proc. Natl. Acad. Sci. U.S.A. **102**, 6679–6685 (2005).

[55] G. Stock and P. Hamm, "A non-equilibrium approach to allosteric communication," Philos. Trans. R. Soc. B: Biol. Sci. **373**, 20170187 (2018).

[56] S. A. Hollingsworth and R. O. Dror, "Molecular dynamics simulation for all," Neuron **99**, 1129–1143 (2018).

[57] D. E. Shaw *et al.* "Anton 3: Twenty microseconds of molecular dynamics simulation before lunch," in Proc. int. conf. high perform. comput. netw. storage anal. (2021), pp. 1–11.

[58] L. Casalino *et al.* "Beyond shielding: The roles of glycans in the SARS-CoV-2 spike protein," ACS Cent. Sci. **6**, 1722–1734 (2020).

[59] L. Casalino *et al.* "AI-driven multiscale simulations illuminate mechanisms of SARS-CoV-2 spike dynamics," Int. J. High Perform. Comput. Appl. **35**, 432–451 (2021).

[60] D. Perez *et al.* "Breaking the mold: Overcoming the time constraints of molecular dynamics on general-purpose hardware," arXiv:2411.10532 (2024).

[61] R. B. Best, N.-V. Buchete, and G. Hummer, "Are current molecular dynamics force fields too helical?" Biophys. J. **95**, L07–L09 (2008).

[62] B. R. Brooks *et al.* "CHARMM: the biomolecular simulation program," J. Comput. Chem. **30**, 1545–1614 (2009).

[63] K. Lindorff-Larsen *et al.* "Improved side-chain torsion potentials for the Amber ff99SB protein force field," Proteins: Struct., Funct., Bioinf. **78**, 1950–1958 (2010).

[64] G. M. Torrie and J. P. Valleau, "Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling," J. Comput. Phys. **23**, 187–199 (1977).

[65] J. Schlitter, M. Engels, and P. Krüger, "Targeted molecular dynamics: A new approach for searching pathways of conformational transitions," J. Mol. Graph. **12**, 84–89 (1994).

[66] Y. Sugita and Y. Okamoto, "Replica-exchange molecular dynamics method for protein folding," Chem. Phys. Lett. **314**, 141–151 (1999).

[67] B. Isralewitz, M. Gao, and K. Schulten, "Steered molecular dynamics and mechanical functions of proteins," Curr. Opin. Struct. Biol. **11**, 224–230 (2001).

[68] S. Wolf, B. Lickert, S. Bray, and G. Stock, "Multisecond ligand dissociation dynamics from atomistic simulations," Nat. Commun. **11**, 2918 (2020).

[69] F. Noé, A. Tkatchenko, K.-R. Müller, and C. Clementi, "Machine learning for molecular simulation," Annu. Rev. Phys. Chem. **71**, 361–390 (2020).

[70] S. Mehdi *et al.* "Enhanced sampling with machine learning," Annu. Rev. Phys. Chem. **75** (2024).

[71] H. Sidky, W. Chen, and A. L. Ferguson, "Machine learning for collective variable discovery and enhanced sampling in biomolecular simulation," Mol. Phys. **118**, e1737742 (2020).

[72] O. T. Unke *et al.* "Machine learning force fields," Chem. Rev. **121**, 10142–10186 (2021).

[73] A. Perez, J. A. Morrone, C. Simmerling, and K. A. Dill, "Advances in free-energy-based simulations of protein folding and ligand binding," Curr. Opin. Struct. Biol. **36**, 25–31 (2016).

[74] M. M. Waldrop, "The chips are down for Moore's law," Nature **530**, 144 (2016).

[75] P. W. Anderson, "More is different: Broken symmetry and the nature of the hierarchical structure of science.," Science **177**, 393–396 (1972).

[76] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," Science **290**, 2323–2326 (2000).

[77] F. Sittel and G. Stock, "Perspective: Identification of collective variables and metastable states of protein dynamics," J. Chem. Phys. **149** (2018).

[78] R. Hegger, A. Altis, P. H. Nguyen, and G. Stock, "How complex is the dynamics of peptide folding?" Phys. Rev. Lett. **98**, 028102 (2007).

[79] S. Piana and A. Laio, "Advillin folding takes place on a hypersurface of small dimensionality," Phys. Rev. Lett. **101**, 208101 (2008).

[80] M. Ernst, F. Sittel, and G. Stock, "Contact- and distance-based principal component analysis of protein dynamics," J. Chem. Phys. **143** (2015).

[81] E. Facco, M. d'Errico, A. Rodriguez, and A. Laio, "Estimating the intrinsic dimension of datasets by a minimal neighborhood information," Sci. Rep. **7**, 12140 (2017).

[82] J. D. Chodera *et al.* "Automatic discovery of metastable states for the construction of Markov models of macromolecular conformational dynamics," J. Chem. Phys. **126** (2007).

[83] N.-V. Buchete and G. Hummer, "Coarse master equations for peptide folding dynamics," J. Phys. Chem. B **112**, 6057–6069 (2008).

[84] V. S. Pande, K. Beauchamp, and G. R. Bowman, "Everything you wanted to know about Markov state models but were afraid to ask," Methods **52**, 99–105 (2010).

[85] G. R. Bowman, V. S. Pande, and F. Noé, *An introduction to Markov state models and their application to long timescale molecular simulation*, Vol. 797 (Springer Science & Business Media, 2013).

[86] J.-H. Prinz *et al.* "Markov models of molecular kinetics: Generation and validation," J. Chem. Phys. **134** (2011).

[87] B. E. Husic and V. S. Pande, "Markov state models: From an art to a science," J. Am. Chem. Soc. **140**, 2386–2396 (2018).

[88] F. Noé and E. Rosta, "Markov models of molecular kinetics," J. Chem. Phys. **151** (2019).

[89] T. Schilling, "Coarse-grained modelling out of equilibrium," Phys. Rep. **972**, 1–45 (2022).

[90] F. Noé *et al.* "Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations," Proc. Natl. Acad. Sci. U.S.A. **106**, 19011–19016 (2009).

[91] G. R. Bowman and P. L. Geissler, "Equilibrium fluctuations of a single folded protein reveal a multitude of potential cryptic allosteric sites," Proc. Natl. Acad. Sci. U.S.A. **109**, 11681–11686 (2012).

[92] U. Sengupta and B. Strodel, "Markov models for the elucidation of allosteric regulation," Philos. Trans. R. Soc. B **373**, 20170178 (2018).

[93] O. Bozovic *et al.* "Real-time observation of ligand-induced allosteric transitions in a PDZ domain," Proc. Natl. Acad. Sci. U.S.A., 26031–26039 (2020).

[94] E. Suárez *et al.* "What Markov state models can and cannot do: Correlation versus path-based observables in protein-folding models," J. Chem. Theory Comput. **17**, 3119–3133 (2021).

[95] D. Nagel, S. Sartore, and G. Stock, "Selecting features for Markov modeling: a case study on HP35," J. Chem. Theory Comput. **19**, 3391–3405 (2023).

[96] D. Nagel, "Markov state modeling of biophysical mechanisms: Markovianity meets spatiotemporal resolution," PhD thesis (Albert-Ludwigs-Universität Freiburg im Breisgau, 2023).

[97] F. Sittel, A. Jain, and G. Stock, "Principal component analysis of molecular dynamics: On the use of Cartesian vs. internal coordinates," J. Chem. Phys. **141** (2014).

[98] M. Ernst, S. Wolf, and G. Stock, "Identification and validation of reaction coordinates describing protein functional motion: Hierarchical dynamics of T4 lysozyme," J. Chem. Theory Comput. **13**, 5076–5088 (2017).

[99] J. S. Hub and B. L. de Groot, "Detection of functional modes in protein dynamics," PLoS Comput. Biol. **5**, 1–13 (2009).

[100] R. T. McGibbon, B. E. Husic, and V. S. Pande, "Identification of simple reaction coordinates from complex dynamics," J. Chem. Phys. **146** (2017).

[101] S. Brandt, F. Sittel, M. Ernst, and G. Stock, "Machine learning of biomolecular reaction coordinates," J. Phys. Chem. Lett. **9**, 2144–2150 (2018).

[102] M. K. Scherer *et al.* "Variational selection of features for molecular kinetics," J. Chem. Phys. **150** (2019).

[103] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," Neurocomputing **300**, 70–79 (2018).

[104] O. Fleetwood, M. A. Kasimova, A. M. Westerlund, and L. Delemotte, "Molecular insights from conformational ensembles via machine learning," Biophys. J. **118**, 765–780 (2020).

[105] A. Amadei, A. B. Linssen, and H. J. Berendsen, "Essential dynamics of proteins," Proteins: Struct., Funct., Bioinf. **17**, 412–425 (1993).

[106] Y. Mu, P. H. Nguyen, and G. Stock, "Energy landscape of a small peptide revealed by dihedral angle principal component analysis," Proteins: Struct., Funct., Bioinf. **58**, 45–52 (2005).

[107] G. Pérez-Hernández *et al.* "Identification of slow molecular order parameters for Markov model construction," J. Chem. Phys. **139** (2013).

[108] W. S. Torgerson, "Multidimensional scaling: I. Theory and method," Psychometrika **17**, 401–419 (1952).

[109] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE.," J. Mach. Learn. Res. **9** (2008).

[110] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," arXiv:1802.03426 (2018).

[111] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science **313**, 504–507 (2006).

[112] D. P. Kingma, "Auto-encoding variational Bayes," arXiv:1312.6114 (2013).

[113] T. Lemke and C. Peter, "Encodermap: Dimensionality reduction and generation of molecule conformations," J. Chem. Theory Comput. **15**, 1209–1215 (2019).

[114] Y. Wang, J. M. L. Ribeiro, and P. Tiwary, "Past–future information bottleneck for sampling molecular reaction coordinate simultaneously with thermodynamics and kinetics," Nat. Commun. **10**, 3573 (2019).

[115] Y. B. Varolgüneş, T. Bereau, and J. F. Rudzinski, "Interpretable embeddings from molecular simulations using Gaussian mixture variational autoencoders," Mach. Learn.: Sci. Technol. **1**, 015012 (2020).

[116] F. Noé, S. Olsson, J. Köhler, and H. Wu, "Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning," Science **365**, eaaw1147 (2019).

[117] J. A. Hartigan, M. A. Wong, *et al.* "A k-means clustering algorithm," Appl. Stat. **28**, 100–108 (1979).

[118] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," Science **344**, 1492–1496 (2014).

[119] F. Sittel and G. Stock, "Robust density-based clustering to identify metastable conformational states of proteins," J. Chem. Theory Comput. **12**, 2426–2435 (2016).

[120] A. M. Westerlund and L. Delemotte, "InfleCS: clustering free energy landscapes with Gaussian mixtures," J. Chem. Theory Comput. **15**, 6752–6759 (2019).

[121] A. Jain and G. Stock, "Identifying metastable states of folding proteins," J. Chem. Theory Comput. **8**, 3810–3819 (2012).

[122] G. R. Bowman, "Improved coarse-graining of Markov state models via explicit consideration of statistical uncertainty," J. Chem. Phys. **137** (2012).

[123] S. Röblitz and M. Weber, "Fuzzy spectral clustering by PCCA+: application to Markov state models and data classification," Adv. Data Anal. Classif. **7**, 147–179 (2013).

[124] D. Nagel, A. Weber, B. Lickert, and G. Stock, "Dynamical coring of Markov state models," J. Chem. Phys. **150** (2019).

[125] F. Noé, H. Wu, J.-H. Prinz, and N. Plattner, "Projected and hidden Markov models for calculating kinetics and metastable states of complex molecules," J. Chem. Phys. **139** (2013).

[126] R. Zwanzig, "From classical dynamics to continuous time random walks," J. Stat. Phys. **30**, 255–262 (1983).

[127] S. Cao *et al.* "On the advantages of exploiting memory in Markov state models for biomolecular dynamics," J. Chem. Phys. **153** (2020).

[128] S. Sartore, F. Teichmann, and G. Stock, "Markov-type state models to describe non-Markovian dynamics," J. Chem. Theory Comput. **21**, 2757–2765 (2025).

[129] G. Hummer and A. Szabo, "Optimal dimensionality reduction of multistate kinetic and Markov-state models," J. Phys. Chem. B. **119**, 9029–9037 (2015).

[130] C. R. Schwantes and V. S. Pande, "Modeling molecular kinetics with tICA and the kernel trick," J. Chem. Theory Comput. **11**, 600–608 (2015).

[131] K. P. Murphy, *Probabilistic machine learning: an introduction* (MIT press, 2022).

[132] J. Tomczak and M. Welling, "VAE with a VampPrior," in Int. conf. artif. intell. stat. (PMLR, 2018), pp. 1214–1223.

[133] M. Jazbec *et al.* "Scalable Gaussian process variational autoencoders," in Int. conf. artif. intell. stat. (PMLR, 2021), pp. 3511–3519.

[134] D. Wang, Y. Wang, L. Evans, and P. Tiwary, "From latent dynamics to meaningful representations," J. Chem. Theory Comput. **20**, 3503–3513 (2024).

[135] L. Pauling and R. B. Corey, "Configurations of polypeptide chains with favored orientations around single bonds: Two new pleated sheets," Proc. Natl. Acad. Sci. U.S.A. **37**, 729–740 (1951).

[136] W. D. Cornell *et al.* "A second generation force field for the simulation of proteins, nucleic acids, and organic molecules," J. Am. Chem. Soc. **117**, 5179–5197 (1995).

[137] D. Van Der Spoel *et al.* "GROMACS: fast, flexible, and free," J. Comput. Chem. **26**, 1701–1718 (2005).

[138] S. Pronk *et al.* "GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit," Bioinformatics **29**, 845–854 (2013).

[139] B. J. Alder and T. Wainwright, "Decay of the velocity autocorrelation function," Phys. Rev. A **1**, 18 (1970).

[140] K. Moritsugu and J. C. Smith, "Temperature-dependent protein dynamics: a simulation-based probabilistic diffusion-vibration Langevin description," J. Phys. Chem. B **110**, 5807–5816 (2006).

[141] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Vol. 2 (Springer, 2009).

[142] E. Parzen, "On estimation of a probability density function and mode," Ann. Math. Stat. **33**, 1065–1076 (1962).

[143] B. W. Silverman, *Density estimation for statistics and data analysis* (Routledge, 2018).

[144] C. Fefferman, S. Mitter, and H. Narayanan, "Testing the manifold hypothesis," J. Am. Math. Soc. **29**, 983–1049 (2016).

[145] A. N. Gorban and I. Y. Tyukin, "Blessing of dimensionality: mathematical foundations of the statistical physics of data," Philos. trans., Math. phys. eng. sci. **376**, 20170237 (2018).

[146] A. D. McLachlan, "A mathematical procedure for superimposing atomic coordinates of proteins," Acta Crystallogr., Sect. A: Found. Crystallogr. **28**, 656–657 (1972).

[147] Y. Zhou, M. Cook, and M. Karplus, "Protein motions at zero-total angular momentum: the importance of long-range correlations," Biophys. J. **79**, 2902–2908 (2000).

[148] A. Altis *et al.* "Construction of the free energy landscape of biomolecules via dihedral angle principal component analysis," J. Chem. Phys. **128** (2008).

[149] R. B. Fenwick *et al.* "Correlated motions are a fundamental property of β-sheets," Nat. Commun. **5**, 4070 (2014).

[150] F. Sittel, T. Filk, and G. Stock, "Principal component analysis on a torus: Theory and application to protein dynamics," J. Chem. Phys. **147** (2017).

[151] A. Altis, P. H. Nguyen, R. Hegger, and G. Stock, "Dihedral angle principal component analysis of molecular dynamics simulations," J. Chem. Phys. **126** (2007).

[152] R. B. Best, G. Hummer, and W. A. Eaton, "Native contacts determine protein folding mechanisms in atomistic simulations," Proc. Natl. Acad. Sci. U.S.A. **110**, 17874–17879 (2013).

[153] P. G. Wolynes, "Recent successes of the energy landscape theory of protein folding and function," Q. Rev. Biophys. **38**, 405–410 (2005).

[154] J. Li *et al.* "Feature selection: A data perspective," ACM Comput. Surv. **50**, 1–45 (2017).

[155] D. Theng and K. K. Bhoyar, "Feature selection techniques for machine learning: A survey of more than two decades of research," Knowl. Inf. Syst. **66**, 1575–1637 (2024).

[156] D. Nagel, A. Weber, and G. Stock, "MSM-Pathfinder: Identification of pathways in Markov state models," J. Chem. Theory Comput. **16**, 7874–7882 (2020).

[157] S. Omori, S. Fuchigami, M. Ikeguchi, and A. Kidera, "Latent dynamics of a protein molecule observed in dihedral angle space," J. Chem. Phys. **132** (2010).

[158] A. A. Ali, E. Dorbath, and G. Stock, "Allosteric communication mediated by protein contact clusters: A dynamical model," J. Chem. Theory Comput. **20**, 10731–10739 (2024).

[159] W. Stacklies, C. Seifert, and F. Graeter, "Implementation of force distribution analysis for molecular dynamics simulations," BMC Bioinform. **12**, 1–5 (2011).

[160] P. G. Bolhuis, D. Chandler, C. Dellago, and P. L. Geissler, "Transition path sampling: Throwing ropes over rough mountain passes, in the dark," Annu. Rev. Phys. Chem. **53**, 291–318 (2002).

[161] A. Ma and A. R. Dinner, "Automatic method for identifying reaction coordinates in complex systems," J. Phys. Chem. B. **109**, 6769–6779 (2005).

[162] T. Chen *et al.* "Xgboost: extreme gradient boosting," R package version 0.4-2 **1**, 1–4 (2015).

[163] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," Ann. Stat. **28**, 337–407 (2000).

[164] R. Patel *et al.* "Oasis: adaptive column sampling for kernel matrix approximation," arXiv:1505.05208 (2015).

[165] F. Litzinger *et al.* "Rapid calculation of molecular kinetics using compressed sensing," J. Chem. Theory Comput. **14**, 2771–2783 (2018).

[166] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," Adv. Neural Inf. Process. Syst. **13** (2000).

[167] P. Ravindra, Z. Smith, and P. Tiwary, "Automatic mutual information noise omission (AMINO): Generating order parameters for molecular systems," Mol. Syst. Des. Eng. **5**, 339–348 (2020).

[168] H. Frauenfelder, S. G. Sligar, and P. G. Wolynes, "The energy landscapes and motions of proteins," Science **254**, 1598–1603 (1991).

[169] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" In Proc. 7. int. conf. database theory (Springer, 1999), pp. 217–235.

[170] A. Zimek, E. Schubert, and H.-P. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data," Stat. Anal. Data Min. **5**, 363–387 (2012).

[171] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemom. Intell. Lab. Syst. **2**, 37–52 (1987).

[172] I. T. Jolliffe, *Principal component analysis for special types of data* (Springer, 2002).

[173] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," AICHE J. **37**, 233–243 (1991).

[174] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436–444 (2015).

[175] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," arXiv:1412.6980 (2014).

[176] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," arXiv:1711.05101 (2017).

[177] P. Baldi and K. Hornik, "Neural networks and principal component analysis: learning from examples without local minima," Neural Netw. **2**, 53–58 (1989).

[178] J. Karhunen and J. Joutsensalo, "Generalizations of principal component analysis, optimization problems, and neural networks," Neural Netw. **8**, 549–562 (1995).

[179] M. Ceriotti, G. A. Tribello, and M. Parrinello, "Simplifying the representation of complex free-energy landscapes using sketch-map," Proc. Natl. Acad. Sci. USA **108**, 13023–13028 (2011).

[180] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," J. Am. Stat. Assoc. **112**, 859–877 (2017).

[181] K. P. Murphy, *Probabilistic machine learning: advanced topics* (MIT press, 2023).

[182] C. Doersch, "Tutorial on variational autoencoders," arXiv:1606.05908 (2016).

[183] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," Mach. Learn. **37**, 183–233 (1999).

[184] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in Int. conf. mach. learn. (PMLR, 2014), pp. 1278–1286.

[185] R. Gómez-Bombarelli *et al.* "Automatic chemical design using a data-driven continuous representation of molecules," ACS Cent. Sci. **4**, 268–276 (2018).

[186] R. R. Eguchi, C. A. Choe, and P.-S. Huang, "Ig-VAE: Generative modeling of protein structure by direct 3D coordinate generation," PLoS Comput. Biol. **18**, e1010271 (2022).

[187] A. Hawkins-Hooker *et al.* "Generating functional protein variants with variational autoencoders," PLoS Comput. Biol. **17**, e1008736 (2021).

[188] J. M. L. Ribeiro, P. Bravo, Y. Wang, and P. Tiwary, "Reweighted autoencoded variational Bayes for enhanced sampling (RAVE)," J. Chem. Phys. **149** (2018).

[189] C. X. Hernández *et al.* "Variational encoding of complex dynamics," Phys. Rev. E **97**, 062412 (2018).

[190] D. Wang and P. Tiwary, "State predictive information bottleneck," J. Chem. Phys. **154** (2021).

[191] S. Adhikari and J. Mondal, "Elucidating protein dynamics through the optimal annealing of variational autoencoders," bioRxiv, 2025–01 (2025).

[192] S. Xiao, Z. Song, H. Tian, and P. Tao, "Assessments of variational autoencoder in protein conformation exploration," J. Comput. Biophys. Chem. **22**, 489–501 (2023).

[193] M. Ghorbani, S. Prasad, J. B. Klauda, and B. R. Brooks, "Variational embedding of protein folding simulations using Gaussian mixture variational autoencoders," J. Chem. Phys. **155** (2021).

[194] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," ACM Comput. Surv. **31**, 264–323 (1999).

[195] A. Saxena *et al.* "A review of clustering techniques and developments," Neurocomputing **267**, 664–681 (2017).

[196] A. Glielmo *et al.* "Unsupervised learning methods for molecular simulation data," Chem. Rev. **121**, 9722–9758 (2021).

[197] S. Lloyd, "Least squares quantization in PCM," IEEE Trans. Inf. Theory **28**, 129–137 (1982).

[198] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis* (John Wiley & Sons, 2009).

[199] E. M. Voorhees, "Implementing agglomerative hierarchic clustering algorithms for use in document retrieval," Inf. Process. Manag. **22**, 465–476 (1986).

[200] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.* "A density-based algorithm for discovering clusters in large spatial databases with noise," in Kdd, Vol. 96, 34 (1996), pp. 226–231.

[201] L. McInnes, J. Healy, S. Astels, *et al.* "HDBSCAN: hierarchical density based clustering.," J. Open Source Softw. **2**, 205 (2017).

[202] R. L. Thorndike, "Who belongs in the family?" Psychometrika **18**, 267–276 (1953).

[203] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," J. Comput. Appl. Math. **20**, 53–65 (1987).

[204] P. Deuflhard and M. Weber, "Robust Perron cluster analysis in conformation dynamics," Linear Algebra Its Appl. **398**, 161–184 (2005).

[205] M. Lange *et al.* "CellRank for directed single-cell fate mapping," Nat. Methods **19**, 159–170 (2022).

[206] W. C. Swope, J. W. Pitera, and F. Suits, "Describing protein folding kinetics by molecular dynamics simulations. 1. Theory," J. Phys. Chem. B **108**, 6571–6581 (2004).

[207] W. A. Eaton, "Modern kinetics and mechanism of protein folding: A retrospective," **125**, 3452–3467 (2021).

[208] D. Nagel, S. Sartore, and G. Stock, "Toward a benchmark for Markov state models: The folding of HP35," J. Phys. Chem. Lett. **14**, 6956–6967 (2023).

[209] S. Piana *et al.* "Predicting the effect of a point mutation on a protein fold: The villin and advillin headpieces and their Pro62Ala mutants," J. Mol. Biol. **375**, 460–470 (2008).

[210] J. Kubelka, J. Hofrichter, and W. A. Eaton, "The protein folding 'speed limit'," Curr. Opin. Struct. Biol. **14**, 76–88 (2004).

[211] J. Kubelka *et al.* "Chemical, physical, and theoretical kinetics of an ultrafast folding protein," Proc. Natl. Acad. Sci. USA **105**, 18655–18662 (2008).

[212] S. Piana, K. Lindorff-Larsen, and D. E. Shaw, "How robust are protein folding simulations with respect to force field parameterization?" Biophys. J. **100**, L47–L49 (2011).

[213] S. Piana, K. Lindorff-Larsen, and D. E. Shaw, "Protein folding kinetics and thermodynamics from atomistic simulation," Proc. Natl. Acad. Sci. USA **109**, 17845–17850 (2012).

[214] J. Kubelka *et al.* "Sub-microsecond protein folding," J. Mol. Biol. **359**, 546–553 (2006).

[215] V. Hornak *et al.* "Comparison of multiple amber force fields and development of improved protein backbone parameters," Proteins: Struct., Funct., Bioinf. **65**, 712–725 (2006).

[216] D. E. Shaw *et al.* "Anton, a special-purpose machine for molecular dynamics simulation," Comput. Archit. News **35**, 1–12 (2007).

[217] W. L. Jorgensen *et al.* "Comparison of simple potential functions for simulating liquid water," J. Chem. Phys.

[218] M. Dixon *et al.* "Structure of a hinge-bending bacteriophage T4 lysozyme mutant, Ile3→ Pro," J. Mol. Biol. **227**, 917–933 (1992).

[219] E. A. Shank *et al.* "The folding cooperativity of a protein is controlled by its chain topology," Nature **465**, 637–640 (2010).

[220] H. Sanabria *et al.* "Resolving dynamics and function of transient states in single enzyme molecules," Nat. Commun. **11**, 1231 (2020).

[221] B. Hess, C. Kutzner, D. Van Der Spoel, and E. Lindahl, "GROMACS 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation," J. Chem. Theory Comput. **4**, 435–447 (2008).

[222] J. Lätzer, T. Shen, and P. G. Wolynes, "Conformational switching upon phosphorylation: a predictive framework based on energy landscape principles," Biochemistry **47**, 2110–2122 (2008).

[223] N. Hori, G. Chikenji, R. S. Berry, and S. Takada, "Folding energy landscape and network dynamics of small globular proteins," Proc. Natl. Acad. Sci. USA **106**, 73–78 (2009).

[224] A. A. Ali, A. Gulzar, S. Wolf, and G. Stock, "Nonequilibrium modeling of the elementary step in PDZ3 allosteric communication," J. Phys. Chem. Lett. **13**, 9862–9868 (2022).

[225] K. Pearson, "Note on regression and inheritance in the case of two parents," Proc. R. Soc. Lond. **58**, 240–242 (1895).

[226] A. Bravais, *Analyse mathématique sur les probabilités des erreurs de situation d'un point* (Impr. Royale, 1844).

[227] F. Galton, "Co-relations and their measurement, chiefly from anthropometric data," Proc. R. Soc. Lond. **45**, 135–145 (1889).

[228] S. M. Stigler, "Stigler's law of eponymy," Trans N Y Acad Sci. **39**, 147–157 (1980).

[229] T. M. Cover, *Elements of information theory* (John Wiley & Sons, 1999).

[230] C. Studholme, D. Hill, and D. Hawkes, "An overlap invariant entropy measure of 3D medical image alignment," Pattern Recognit. **32**, 71–86 (1999).

[231] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," J. Mach. Learn. Res. **3**, 583–617 (2002).

[232] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," J. Mach. Learn. Res. **11**, 2837–2854 (2010).

[233] I. M. Gel'fand and A. M. Yaglom, "Computation of the amount of information about a stochastic function contained in another such function," Transl. Am. Math. Soc. **12** (1959).

[234] O. F. Lange and H. Grubmüller, "Generalized correlation for biomolecular dynamics," Proteins: Struct., Funct., Bioinf. **62**, 1053–1061 (2006).

[235] D. N. Reshef *et al.* "Detecting novel associations in large data sets," Science **334**, 1518–1524 (2011).

[236] J. B. Kinney and G. S. Atwal, "Equitability, mutual information, and the maximal information coefficient," Proc. Natl. Acad. Sci. U.S.A. **111**, 3354–3359 (2014).

[237] D. Freedman and P. Diaconis, "On the histogram as a density estimator:L2 theory," Probab. Theory Relat. Fields **57**, 453–476 (1981).

[238] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," Ann. Math. Statist. **27**, 832–837 (1956).

[239] N.-B. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth selection for kernel density estimation: A review of fully automatic selectors," Adv. Stat. Anal. **97**, 403–433 (2013).

[240] D. M. Bashtannyk and R. J. Hyndman, "Bandwidth selection for kernel conditional density estimation," Comput. Stat. Data Anal. **36**, 279–298 (2001).

[241] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," Phys. Rev. E **69**, 066138 (2004).

[242] J. Matejka and G. Fitzmaurice, "Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing," in Proc. chi conf. hum. factors comput. syst. (2017), pp. 1290–1294.

[243] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in Proc. conf. empir. methods nat. lang. process. (2007), pp. 410–420.

[244] V. A. Traag, L. Waltman, and N. J. Van Eck, "From Louvain to Leiden: Guaranteeing well-connected communities," Sci. Rep. **9**, 1–12 (2019).

[245] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," J. Stat. Mech.: Theory Exp. **2008**, P10008 (2008).

[246] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," Phys. Rev. E **69**, 026113 (2004).

[247] J. M. Kumpula, J. Saramäki, K. Kaski, and J. Kertész, "Limited resolution in complex network community detection with Potts model approach," Eur. Phys. J. B **56**, 41–45 (2007).

[248] V. A. Traag, P. Van Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," Phys. Rev. E **84**, 016114 (2011).

[249] R. B. Potts, "Some generalized order-disorder transformations," Math. Proc. Camb. Philos. Soc. **48**, 106–109 (1952).

[250] E. Ising, "Beitrag zur Theorie des Ferromagnetismus," Z. Phys. **31**, 253–258 (1925).

[251] R. T. McGibbon and V. S. Pande, "Variational cross-validation of slow dynamical modes in molecular kinetics," J. Chem. Phys. **142** (2015).

[252] F. Pedregosa *et al.* "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res. **12**, 2825–2830 (2011).

[253] See `https://www.moldyn.uni-freiburg.de/software.html`.

[254] See `https://github.com/moldyn/MoSAIC`.

[255] K. Falk, D. Savio, and M. Moseler, "Nonempirical free volume viscosity model for alkane lubricants under severe pressures," Phys. Rev. Lett. **124**, 105501 (2020).

[256] A. Codrignani *et al.* "Toward a continuum description of lubrication in highly pressurized nanometer-wide constrictions: The importance of accurate slip laws," Sci. Adv. **9** (2023).

[257] L. B. Kruse, K. Falk, and M. Moseler, "Calculating high-pressure PAO4 viscosity with equilibrium molecular dynamics simulations," Tribol. Lett. **72**, 40 (2024).

[258] *Synfluid® pao 4 technical data sheet*, Accessed: 2025-04-25, Chevron Phillips Chemical Company LP (2020).

[259] S. Bair and S. Flores-Torres, "The viscosity of polyalphaolefins mixtures at high pressure and stress," J. Tribol. **141**, 021802 (2019).

[260] M. Post, S. Wolf, and G. Stock, "Molecular origin of driving-dependent friction in fluids," J. Chem. Theory Comput. **18**, 2816–2825 (2022).

[261] M. Post, "Dynamical models of bio-molecular systems from constrained molecular dynamics simulations," PhD thesis (Albert-Ludwigs-Universität Freiburg im Breisgau, 2022).

[262] S. Wolf, M. Post, and G. Stock, "Path separation of dissipation-corrected targeted molecular dynamics simulations of protein–ligand unbinding," J. Chem. Phys. **158** (2023).

[263] S. Wolf and G. Stock, "Targeted molecular dynamics calculations of free energy profiles using a nonequilibrium friction correction," J. Chem. Theory Comput. **14**, 6175–6182 (2018).

[264] C. Jarzynski, "Nonequilibrium work theorem for a system strongly coupled to a thermal environment," J. Stat. Mech.: Theory Exp. **2004**, P09005 (2004).

[265] V. Tänzel, M. Jäger, and S. Wolf, "Learning protein–ligand unbinding pathways via single-parameter community detection," J. Chem. Theory Comput. **20**, 5058–5067 (2024).

[266] I. C. Unarta *et al.* "Role of bacterial RNA polymerase gate opening dynamics in DNA loading and antibiotics inhibition elucidated by quasi-Markov state model," Proc. Natl. Acad. Sci. USA **118**, e2024324118 (2021).

[267] A. Das, K. Sinha, and S. Chakrabarty, "Elucidating the molecular mechanism of noncompetitive inhibition of acetylcholinesterase by an antidiabetic drug chlorpropamide: Identification of new allosteric sites," Phys. Chem. Chem. Phys. **26**, 28894–28903 (2024).

[268] A. D. Cliff and J. K. Ord, *Spatial processes: models & applications* (Pion, 1981).

[269] D. Wartenberg, "Multivariate spatial correlation: a method for exploratory geographical analysis," Geogr. Anal. **17**, 263–283 (1985).

[270] P. Legendre, "Spatial autocorrelation: trouble or new paradigm?" Ecology **74**, 1659–1673 (1993).

[271] B. B. Averbeck, P. E. Latham, and A. Pouget, "Neural correlations, population coding and computation," Nat. Rev. Neurosci. **7**, 358–366 (2006).

[272] P. Viola and W. M. Wells III, "Alignment by maximization of mutual information," Int. J. Comput. Vis. **24**, 137–154 (1997).

[273] J. P. Pluim, J. A. Maintz, and M. A. Viergever, "Mutual-information-based registration of medical images: a survey," IEEE Trans. Med. Imaging **22**, 986–1004 (2003).

[274] O. E. Barndorff-Nielsen and N. Shephard, "Econometric analysis of realized covariation: high frequency based covariance, regression, and correlation in financial economics," Econometrica **72**, 885–925 (2004).

[275] X. Guo, H. Zhang, and T. Tian, "Development of stock correlation networks using mutual information and financial big data," PloS one **13**, e0195941 (2018).

[276] M. Post, S. Wolf, and G. Stock, "Principal component analysis of nonequilibrium molecular dynamics simulations," J. Chem. Phys. **150** (2019).

[277] C. L. McClendon, A. P. Kornev, M. K. Gilson, and S. S. Taylor, "Dynamic architecture of a protein kinase," Proc. Natl. Acad. Sci. USA **111**, E4623–E4631 (2014).

[278] A. Sethi, J. Eargle, A. A. Black, and Z. Luthey-Schulten, "Dynamical networks in tRNA: Protein complexes," Proc. Natl. Acad. Sci. USA.

[279] C. L. McClendon *et al.* "Quantifying correlations between allosteric sites in thermodynamic ensembles," J. Chem. Theory Comput. **5**, 2486–2502 (2009).

[280] M. Bhattacharyya, S. Ghosh, and S. Vishveshwara, "Protein structure and function: looking through the network of side-chain interactions," Curr. Prot. Pept. Sci. **17**, 4–25 (2016).

[281] S. J. Wodak *et al.* "Allostery in its many disguises: From theory to applications," Structure **27**, 566–578 (2019).

[282] K. Kasahara, I. Fukuda, and H. Nakamura, "A novel approach of dynamic cross correlation analysis on molecular dynamics simulations and its application to Ets1 dimer–DNA complex," PloS one **9**, e112419 (2014).

[283] C. Okamoto and K. Ando, "Molecular dynamics simulation analysis of structural dynamic cross correlation induced by odorant hydrogen-bonding in mouse eugenol olfactory receptor," Biophys Physicobiol. **21**, e210007 (2024).

[284] T. Ichiye and M. Karplus, "Collective motions in proteins: a covariance analysis of atomic fluctuations in molecular dynamics and normal mode simulations," Proteins **11**, 205–217 (1991).

[285] D. Hardoon, S. Szedmak, and Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," Neural Comput. **16**, 2639–2664 (2004).

[286] F. Briki and D. Genest, "Canonical analysis of correlated atomic motions in DNA from molecular dynamics simulation," Biophys. Chem. **52**, 35–43 (1994).

[287] L. F. Kozachenko and N. N. Leonenko, "Sample estimate of the entropy of a random vector," Probl. Peredachi. Inf. **23**, 9–16 (1987).

[288] A. B. Tsybakov and E. C. van der Meulen, "Root-n consistent estimators of entropy for densities with unbounded support," Scand. J. Stat. **23**, 75–83 (1996).

[289] H. Singh *et al.* "Nearest neighbor estimates of entropy," Am. J. Math. **23**, 301–321 (2003).

[290] D. Lombardi and S. Pant, "Nonparametric k-nearest-neighbor entropy estimator," Phys. Rev. E **93**, 013310 (2016).

[291] E. T. Jaynes, "Prior probabilities," IEEE Trans. Syst. Sci. Cybern. **4**, 227–241 (1968).

[292] C. E. Shannon, "A mathematical theory of communication," Bell Syst. Tech. J. **27**, 379–423 (1948).

[293] E. T. Jaynes, "Information theory and statistical mechanics," in *Statistical physics*, Brandeis University Summer Institute Lectures in Theoretical Physics, Vol. 3, Sec. 4b (W. A. Benjamin, Inc., New York, NY, 1963), pp. 181–218.

[294] R. Nussinov and C.-J. Tsai, "Allostery in disease and in drug discovery," Cell **153**, 293–305 (2013).

[295] C. Bohr, K. Hasselbalch, and A. Krogh, "Ueber einen in biologischer Beziehung wichtigen Einfluss, den die Kohlensäurespannung des Blutes auf dessen Sauerstoffbindung übt," Skandinavisches Archiv Für Physiologie **16**, 402–412 (1904).

[296] J. Monod, J. Wyman, and J.-P. Changeux, "On the nature of allosteric transitions: A plausible model," J. Mol. Biol.

[297] K. Gunasekaran, B. Ma, and R. Nussinov, "Is allostery an intrinsic property of all dynamic proteins?" Proteins: Struct., Funct., Bioinf. **57**, 433–443 (2004).

[298] M. Gruebele, "Fast protein folding," in *Protein folding, misfolding and aggregation: classical themes and novel approaches*, edited by V. Muñoz (The Royal Society of Chemistry, Cambridge, 2008), pp. 106–138.

[299] K. A. Dill, S. B. Ozkan, M. S. Shell, and T. R. Weikl, "The protein folding problem," Annu. Rev. Biophys. **37**, 289–316 (2008).

[300] S. Brüschweiler *et al.* "Direct observation of the dynamic process underlying allosteric signal transmission," J. Am. Chem. Soc. **131**, 3063–3068 (2009).

[301] R. B. Best and G. Hummer, "Coordinate-dependent diffusion in protein folding," Proc. Natl. Acad. Sci. USA **107**, 1088–1093 (2010).

[302] X.-Q. Yao and D. Hamelberg, "Detecting functional dynamics in proteins with comparative perturbed-ensembles analysis," Acc. Chem. Res. **52**, 3455–3464 (2019).

[303] X.-Q. Yao and D. Hamelberg, "Residue–residue contact changes during functional processes define allosteric communication pathways," J. Chem. Theory Comput. **18**, 1173–1187 (2022).

[304] P. Bonacich, "Power and centrality: A family of measures," Am. J. Sociol. **92**, 1170–1182 (1987).

[305] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," Comput. Netw. ISDN Syst. **30**, 107–117 (1998).

[306] Y. LeCun *et al.* "Backpropagation applied to handwritten zip code recognition," Neural Comput. **1**, 541–551 (1989).

[307] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Adv. neural inf. process. syst. (2012).

[308] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in Med. image comput. comput. assist. interv. (2015), pp. 234–241.

[309] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Netw. **2**, 359–366 (1989).

[310] M. M. Bronstein *et al.* "Geometric deep learning: Going beyond Euclidean data," IEEE Signal Process. Mag. **34**, 18–42 (2017).

[311] Z. Wu *et al.* "A comprehensive survey on graph neural networks," IEEE Trans. Neural Netw. Learn. Syst. **32**, 4–24 (2020).

[312] J. Gilmer *et al.* "Neural message passing for quantum chemistry," in Int. conf. mach. learn. (PMLR, 2017), pp. 1263–1272.

[313] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, "Geometric deep learning: grids, groups, graphs, geodesics, and gauges," arXiv:2104.13478 (2021).

[314] T. N. Kipf and M. Welling, "Variational graph auto-encoders," arXiv:1611.07308 (2016).

[315] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in Int. conf. mach. learn. Vol. 97 (2019), pp. 3734–3743.

[316] B. Knyazev, G. W. Taylor, and M. R. Amer, "Understanding attention and generalization in graph neural networks," in Adv. neural inf. process. syst. (2019).

[317] A. Vaswani *et al.* "Attention is all you need," in Adv. neural inf. process. syst. Vol. 30 (2017), pp. 6000–6010.

[318] O. Vinyals, S. Bengio, and M. Kudlur, "Order matters: sequence to sequence for sets," in Int. conf. learn. represent. (2016).

[319] K. Jha, S. Saha, and H. Singh, "Prediction of protein–protein interaction using graph neural networks," Sci. Rep. **12**, 8360 (2022).

[320] Z. Smith, M. Strobel, B. P. Vani, and P. Tiwary, "Graph attention site prediction (GrASP): Identifying druggable binding sites using graph neural networks with attention," J. Chem. Inf. Model. **64**, 2637–2644 (2024).

[321] L. Franke and C. Peter, "Visualizing the residue interaction landscape of proteins by temporal network embedding," J. Chem. Theory Comput. **19**, 2985–2995 (2023).

[322] N. Dethloff, "Decoding protein dynamics: dimensionality reduction and generative modeling via autoencoders," Faculty of Mathematics and Physics, Institute of Physics. Supervisor: Prof. Dr. Gerhard Stock, Master's thesis (Albert-Ludwigs-Universität Freiburg, Freiburg, Germany, 2024).

[323] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," J. Am. Stat. Assoc. **107**, 1590–1598 (2012).

[324] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," Signal Process. **167**, 107299 (2020).

[325] G. Peyré, M. Cuturi, *et al.* "Computational optimal transport: With applications to data science," Found. Trends Mach. Learn. **11**, 355–607 (2019).

[326] E. W. Dijkstra, "A note on two problems in connexion with graphs," Numer. Math. **1**, 269–271 (1959).

[327] R. Das and D. Baker, "Macromolecular modeling with Rosetta," Annu. Rev. Biochem. **77**, 363–382 (2008).

[328] S. Chaudhury, S. Lyskov, and J. J. Gray, "PyRosetta: A script-based interface for implementing molecular modeling algorithms using Rosetta," Bioinformatics **26**, 689–691 (2010).

[329] R. F. Alford *et al.* "The rosetta all-atom energy function for macromolecular modeling and design," J. Chem. Theory Comput. **13**, 3031–3048 (2017).

[330] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, Vol. 2, 3 (MIT press Cambridge, MA, 2006).

[331] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning* (Springer, 2003), pp. 63–71.

[332] F. P. Casale *et al.* "Gaussian process prior variational autoencoders," Adv. Neural Inf. Process. Syst. **31** (2018).

[333] T. Tian *et al.* "Dependency-aware deep generative models for multitasking analysis of spatial omics data," Nat. Methods **21**, 1501–1513 (2024).

[334] B. Mohr *et al.* "Data-driven discovery of cardiolipin-selective small molecules by computational active learning," Chem. Sci. **13**, 4498–4511 (2022).

[335] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," IEEE Trans. Pattern Anal. Mach. Intell. **37**, 408–423 (2013).

[336] N. Cressie and C. K. Wikle, *Statistics for spatio-temporal data* (John Wiley & Sons, 2011).

[337] J. Han, X.-P. Zhang, and F. Wang, "Gaussian process regression stochastic volatility model for financial time series," IEEE J. Sel. Top. Signal Process. **10**, 1015–1028 (2016).

[338] I. Higgins *et al.* "β-VAE: Learning basic visual concepts with a constrained variational framework," in 5th int. conf. learn. represent. (iclr) (2017).

[339] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in Int. conf. artif. intell. stat. (2009), pp. 567–574.

[340] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," arXiv:1309.6835 (2013).

[341] M. Pearce, "The Gaussian process prior VAE for interpretable latent dynamics from pixels," in Symposium on advances in approximate Bayesian inference (PMLR, 2020), pp. 1–12.

[342] A. Mardt, L. Pasquali, H. Wu, and F. Noé, "VAMPnets for deep learning of molecular kinetics," Nat. Commun **9**, 5 (2018).

[343] E. Dorbath, "A hierarchical dynamical model of protein allosteric communication," PhD thesis (Albert-Ludwigs-Universität Freiburg im Breisgau, 2025).

[344] F. Rudolf, "Modeling the non-equilibrium allosteric response of protein domains," Master's thesis (Albert-Ludwigs-Universität Freiburg, 2025).

[345] T. Schreiber, "Measuring information transfer," Phys. Rev. Lett. **85**, 461 (2000).

[346] G. Saporta, *Probabilités, analyse des données et statistique* (Editions technip, 2006).

[347] P. Veličković *et al.* "Graph attention networks," in Int. conf. learn. represent. (2018).

[348] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Int. conf. mach. learn. (PMRL, 2015), pp. 448–456.

[349] N. Srivastava *et al.* "Dropout: A simple way to prevent neural networks from overfitting," J. Mach. Learn. Res. **15**, 1929–1958 (2014).