From Atoms to Dynamics: Learning the Committor Without Collective Variables

Supplementary Information

Contents

1 Theoretical background

1.1 The committor function and the variational principle

The configuration of a molecular system in a specific state is completely determined by Cartesian coordinates $\mathbf{x} \equiv \{\mathbf{x}_i\}_{i=1}^{N} \in \mathbb{R}^{3N}$. The probability density of the system at time t is expressed as $\rho(\mathbf{x}, t)$. The forward propagation step $(\mathbf{x} \to \mathbf{x}')$ for the probability density from the time t to the time $t + \tau$ is

$$\rho\left(\mathbf{x}, t+\tau\right) = \int \mathrm{d}\mathbf{x}\, \mathscr{P}_\tau\left(\mathbf{x} \mid \mathbf{x}'\right) \rho\left(\mathbf{x}', t\right) \tag{S1}$$

where $\mathscr{P}_\tau\left(\mathbf{x} \mid \mathbf{x}'\right)$ is the propagator or transfer operator and represents the transition from $\mathbf{x}$ to $\mathbf{x}'$. The dynamics of the system is assumed to be Markovian with a finite time-lag τ, and $\mathscr{P}_\tau$ obeys the Chapman-Kolmogorov equation $\rho\left(t + n\tau\right) = \mathscr{P}_{n\tau}\rho\left(t\right)$, with $\mathscr{P}_{n\tau} = \left(\mathscr{P}_\tau\right)^n$. It is assumed that the system is in thermodynamic equilibrium and that we have microscopic

detailed balance, $\mathscr{P}_\tau(\mathbf{x} \mid \mathbf{x}')\rho_{\text{eq}}(\mathbf{x}') = \mathscr{P}_\tau(\mathbf{x}' \mid \mathbf{x})\rho_{\text{eq}}(\mathbf{x})$. Let us assume a system with two metastable states A and B, the forward committor, $q(\mathbf{x})$, is the sum of the probability over all paths starting at $\mathbf{x}$ that ultimately reach the states B before ever reach the state A, by definition. The probability of each of these paths is expressed as a product of discrete propagation steps $\mathscr{P}_\tau \cdots \mathscr{P}_{n\tau}$ with time-lag τ, under the restriction that the intermediate states resulting from all of these steps are in $A^c \cap B^c$. Summing over all possible paths, it follows that $q(\mathbf{x})$ can be written as

$$q(\mathbf{x}) = \int \mathrm{d}\mathbf{x}'\, q(\mathbf{x}')\mathscr{P}_\tau(\mathbf{x}'|\mathbf{x}) \tag{S2}$$

with the constraints $q(\mathbf{x}) = 0$ if $\mathbf{x} \in A$, and $q(\mathbf{x}) = 1$ if $\mathbf{x} \in B$, arising from its definition. It should be noticed that the equation for the committor probabilities involving only the elementary propagator $\mathscr{P}_\tau(\mathbf{x}'|\mathbf{x})$ for the time lag, is valid under Markovianity of the dynamics as expressed by the Chapman-Kolmogorov equation. In the context of TPT, one can express the net forward reactive flux from A to B as

$$J_{AB} = \frac{1}{2\tau} \int \mathrm{d}\mathbf{x} \int \mathrm{d}\mathbf{x}' \left(q(\mathbf{x}') - q(\mathbf{x})\right)^2 \mathscr{P}_\tau(\mathbf{x}'|\mathbf{x})\rho_{\text{eq}}(\mathbf{x}), \tag{S3}$$

which, equivalently, can also be expressed in terms of the time-correlation functional[1] of equation (1) in the main text. The expression for J_{AB} can serve as a robust variational principle to optimize a trial committor. Minimizing the quantity J_{AB} in the functional sense with respect to a trial function q, $\delta J_{AB}/\delta q = 0$ recovers equation (S2) that formally defines the committor probability. Thus, defining the committor time-correlation functional $C[q;\tau] = \langle (q(\tau) - q(0))^2 \rangle / 2$, one can use the loss function to refine a trial model function for the committor probability[2,3]. A more detailed discussion about the variational principle can also be found in Ref. 1.

1.2 GVP-GNNs architectures

Learning from macromolecular structures in three-dimensional space requires neural architectures that are both rotation-equivariant and expressive in terms of relational reasoning. Traditional GNNs are effective at modeling relational structures, whereas convolutional neural networks handle the geometric information of the systems. Geometric Vector Perceptrons (GVPs) were introduced as a unified method that bridges graph neural networks (GNNs) and convolutional neural networks to simultaneously capture the full geometric and relational information encoded in macromolecular systems, building GVP-GNNs as a new class of GNNs that extend scalar-based message-passing schemes to operate on both scalar and geometric vector features.

The GVP learns functions which output vectors or scalars, using geometric vectors and scalars as inputs as explained in the main text, i.e., given a tuple $(S, \mathbf{V})$ of scalar features $S \equiv \{s_i\}_{i=1}^n \in \mathbb{R}^n$ and vector features $\mathbf{V} \equiv \{\mathbf{v}_i\}_{i=1}^\nu \in \mathbb{R}^{3\times\nu}$—where each $\mathbf{v}_i$ belongs to the three-dimensional space,—we compute new features $(S', \mathbf{V}') \in \mathbb{R}^m \times \mathbb{R}^{3\times\mu}$. In the architecture proposed here, we consider graphs as representations of atomic-level structures in which all node and edge embeddings are tuples $(S, \mathbf{V})$, as detailed in the main text. Information is propagated through the network layers using a message-passing scheme[4–6], where

node features are updated based on information from their neighbors. GVP-GNNs utilize this mechanism by aggregating information from neighboring nodes and edges to iteratively update both scalar and vector node embeddings during each propagation step. We briefly outline the two fundamental components of our model, the GVP layer and the overall GVP-GNN architecture, built by integrating GVPs with message-passing. For full implementation details, we refer the reader to the original works in Ref. 4, 7. Considering that a GVP layer operates on tuples $(S, \mathbf{V})$, this transformation is defined in Algorithm S1, as reported in Ref. 7.

Algorithm S1 Operation of GVP layers[7]

Input: Tuple (scalar and vector features) $(S, \mathbf{V})$.
Output: Tuple (scalar and vector features) $(S', \mathbf{V}')$.
GVP:

$\mathbf{V}_h \leftarrow \mathbf{W}_h \mathbf{V}$
$\mathbf{V}_\mu \leftarrow \mathbf{W}_\mu \mathbf{V}_h$
$S_h \leftarrow \|\mathbf{V}_h\|_2$
$S_{h+n} \leftarrow \text{concat}(S_h, S)$
$S_m \leftarrow \mathbf{W}_m S_{h+n} + \mathbf{b}_m$
$S' \leftarrow \sigma(S_m)$
$\mathbf{V}' \leftarrow \sigma_g(\mathbf{W}_g[\sigma^+(S_m)] + \mathbf{b}) \odot \mathbf{V}_\mu$
return $(S', \mathbf{V}')$

In Algorithm S1, we have introduced σ and σ^+, which are scalar nonlinearities (e.g., ReLU and identity), and σ_g is a gating function (e.g., sigmoid). The operation $\odot$ denotes row-wise multiplication, where each 3D vector in a matrix is scaled by a corresponding scalar, while $\mathbf{W}_h$, $\mathbf{W}_\mu$, $\mathbf{W}_m$ and $\mathbf{W}_g$ are linear learnable transformations, $\mathbf{b}_m$, $\mathbf{b}$ are learnable biases and $\|\cdot\|_2$ is the Euclidean norm. This formulation ensures that the vector and scalar outputs of the GVP are equivariant and invariant, respectively, with respect to an arbitrary composition of rotations and reflections in 3D Euclidean space. Formally, invariance and equivariance can be defined as follows,

Definition 1. *Let G be a group. A map $f : X \to Y$ is called invariant with respect to the G-action $\cdot$ on X and Y, if*

$$f(g \cdot x) = f(x), \quad \forall x \in X, \forall g \in G.$$

Definition 2. *Let G be a group. A map $f : X \to Y$ is called equivariant with respect to the G-action $\cdot$ on X and Y, if*

$$f(g \cdot x) = g \cdot f(x), \quad \forall x \in X, \forall g \in G.$$

As an example, if R is a composition of rotations, and if $\text{GVP}(S, \mathbf{V}) = (S', \mathbf{V}')$, then $\text{GVP}(S, R(\mathbf{V})) = (S', R(\mathbf{V}'))$. This equality stems from the fact that the only permissible operations on vector inputs are scalar scaling, linear combinations, and the Euclidean norm. In addition, the GVP architecture inherits the Universal Approximation Theorem of dense

layers with respect to rotation and reflection equivariant functions as stated by the theorem in Refs. 4, 7, that reads as follows.

Theorem 1. *Let R describe an arbitrary rotation and/or reflection in $\mathbb{R}^3$ and G_s be the vector outputs of a GVP defined with $n = 0$, $\mu = 6$, and sigmoidals σ, σ^+. For $\nu \geq 3$, let $\Omega^\nu \subset \mathbb{R}^{\nu\times 3}$ be the set of all $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_\nu]^\top \in \mathbb{R}^{\nu\times 3}$ such that $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ are linearly independent and $0 \leq \|\mathbf{v}_i\|_2 \leq b$ for all i and for some $0 < b < \infty$. Then, for any continuous and rotation/reflection-equivariant $F : \Omega^\nu \to \mathbb{R}^3$, and for any $\varepsilon > 0$, there exists a form $f(\mathbf{V}) = \mathbf{1}^\top G_s(\mathbf{V})$ such that $|F(\mathbf{V})_i - f(\mathbf{V})_i| < 6bC\varepsilon$, with $0 < C < \infty, \forall i \in \{1, 2, 3\}$ and $\forall \mathbf{V} \in \Omega^\nu$.*

Additionally, this theorem can be generalized to $n \neq 0$, that is, over functions defined on $\mathbb{R}^n \times \mathbb{R}^{\nu\times 3}$.[7]

The full GVP-GNN architecture is built by combining GVPs with message-passing.[4] Each node in the graph carries scalar and vector features. Messages passed from node j to node i are computed via

$$\begin{aligned} \mathbf{h}_m^{(j\to i)} &= g\left(\text{concat}(\mathbf{h}_v^{(j)}, \mathbf{h}_e^{(j\to i)})\right) \\ \mathbf{h}_v^{(i)} &\leftarrow \text{LayerNorm}\left(\mathbf{h}_v^{(i)} + \frac{1}{k'}\sum_{j:e_{j\to i}\in E} \text{Dropout}(\mathbf{h}_m^{(j\to i)})\right) \end{aligned} \tag{S4}$$

where g is a stack of GVPs, k' is the number of neighbors (i.e. incoming messages), $\text{concat}(\cdot,\cdot)$ is the concatenation function, $\text{Dropout}(\cdot)$ is a regularization scheme that helps neural networks to be prevented from overfitting[8] and $\text{LayerNorm}(\cdot)$ is the layer normalization. The $\mathbf{h}_v^{(j)}$ and $\mathbf{h}_e^{(j\to i)}$ stand for nodes and edges embeddings, respectively, while $\mathbf{h}_m^{(j\to i)}$ describes the message from node j to node i. A feedforward step updates node embeddings as,

$$\mathbf{h}_v^{(i)} \leftarrow \text{LayerNorm}\left(\mathbf{h}_v^{(i)} + \text{Dropout}(g(\mathbf{h}_v^{(i)}))\right). \tag{S5}$$

This formulation allows for both scalar and geometric vector features to evolve jointly in an equivariant fashion across graph layers. Finally, since we are primarily interested in learning a global feature, namely, the committor function—that characterizes the state of the entire graph—it is necessary to aggregate information from all nodes and edges into graph-level quantities. This is typically achieved using permutation-invariant pooling functions, such as a simple summation or averaging over the output scalar node features of the network[9].

2 Transition rates

For all the illustrations reported here, we assume a two-state Markov jump between metastable states A and B,

$$A \underset{k_{BA}}{\overset{k_{AB}}{\rightleftharpoons}} B \tag{S6}$$

from which a transition rate can be determined. The equilibrium state probabilities are $p_A = k_{BA}/(k_{AB} + k_{BA})$ and $p_B = k_{AB}/(k_{AB} + k_{BA})$. The unidirectional fluxes are equal at equilibrium, that is, $J_{AB} = p_A k_{AB} = J_{BA} = p_B k_{BA}$. The mean first passage time (MFPT) from A to B can be determined as $1/k_{AB}$.[1,10,11] In our implementation, in the spirit of the VCN,[2] the transition rate can be obtained from A to B from the unidirectional flux $k_{AB} =$

J_{AB}/p_A, where the flux J_{AB} is given by Equation 1 of the main manuscript in terms of the time-correlation function of the committor. Here, we determine the transition rate, k_{AB}, from the slope of the time-correlation function $C_{qq}(\tau)$, as a function of τ and p_A, for the different systems examined.

2.1 NANMA isomerization

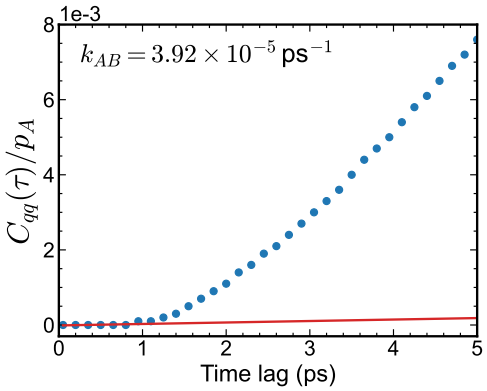

Fig. S1: **Fitting of the rate constant for NANMA.** Rate constant k_{AB} determined from GNN committor using the biased trajectory of NANMA.

2.2 Trialanine conformational equilibrium

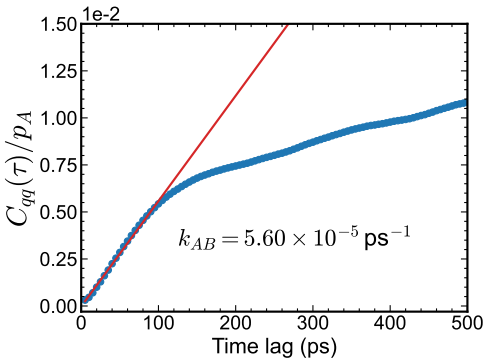

Fig. S2: **Fitting of the rate constant for trialanine.** Rate constant k_{AB} determined from GNN committor using the biased trajectory of trialanine.

2.3 Diels–Alder reaction

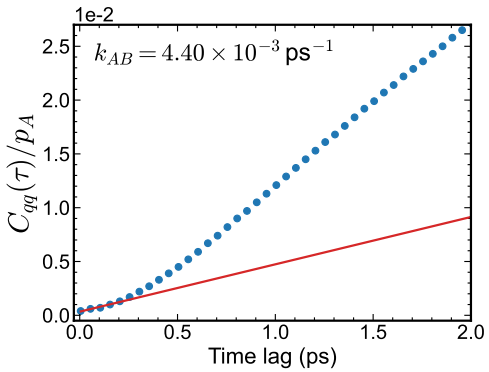

Fig. S3: **Fitting of the rate constant for Diels–Alder reaction.** Rate constant k_{AB} determined from GNN committor using the biased trajectory for the Diels–Alder reaction.

2.4 Trp-cage reversible folding

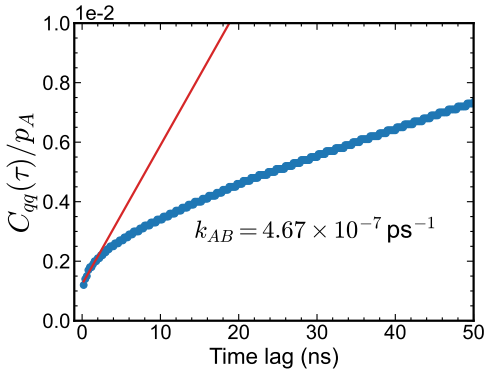

Fig. S4: **Fitting of the rate constant for Trp-cage reversible folding.** Rate constant k_{AB} determined from GNN committor using the biased trajectory for the Trp-cage.

3 Dependence on the restraints applied to the basins (λ)

Here, in Fig. S5–S6, we show how learning differs with different choices of the parameter λ of the training loss function (see equation (2) of the main paper). Only in those cases where λ is exaggerated to values of the order of 10^5, we encounter either inaccurate or incomplete learning. However, through a wide range of values the output is consistent.

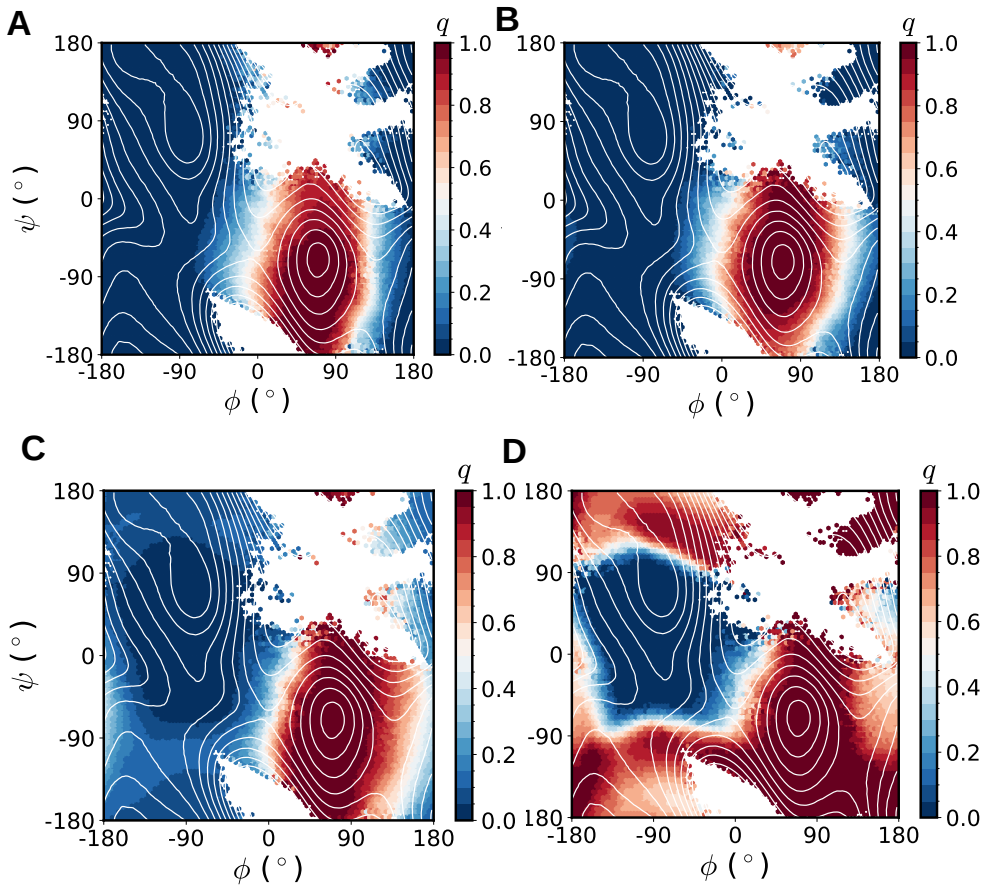

Fig. S5: **Learned committor for NANMA and different values of λ.** Learned committor projected onto (ϕ, ψ) subspace for $\lambda = 0.5$ (A), $\lambda = 10$ (B), $\lambda = 100$ (C) and an extreme value of $\lambda = 10000$ (C).

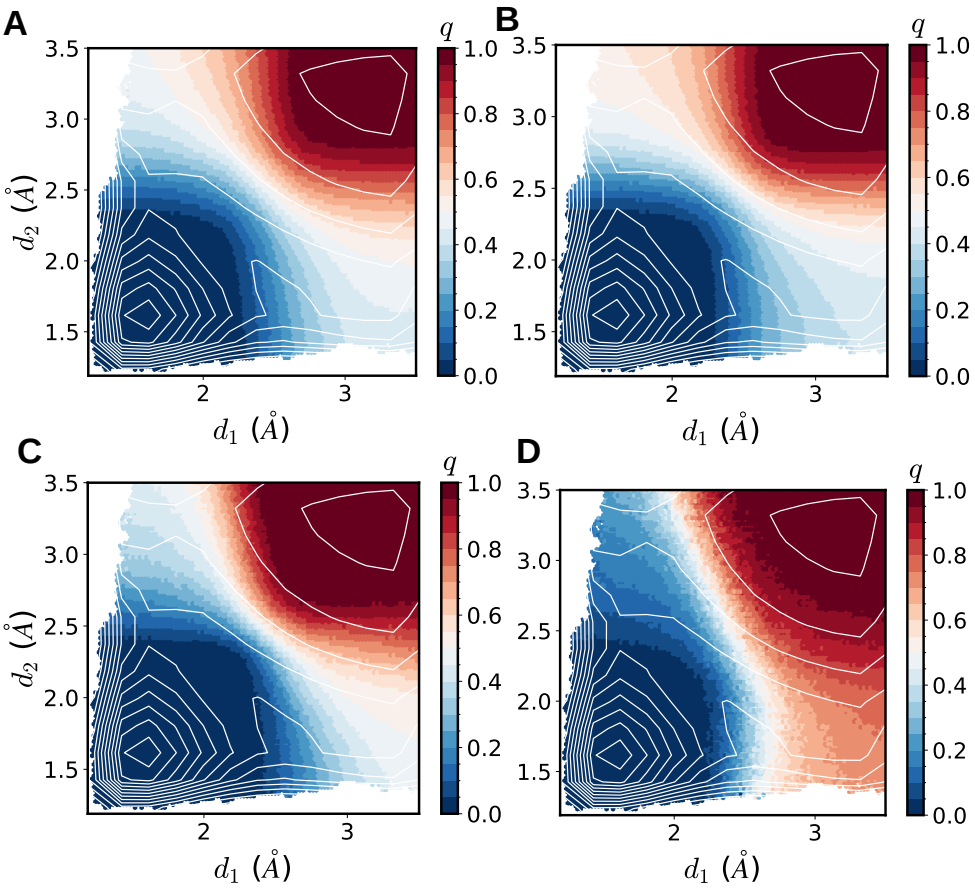

Fig. S6: **Learned committor for Diels–Alder reaction and different values of λ.** Learned committor projected onto (d_1, d_2) subspace for $\lambda = 0.5$ (A), $\lambda = 10$ (B), $\lambda = 100$ (C) and extreme values of $\lambda = 10000$ (C).

4 Trp-cage system

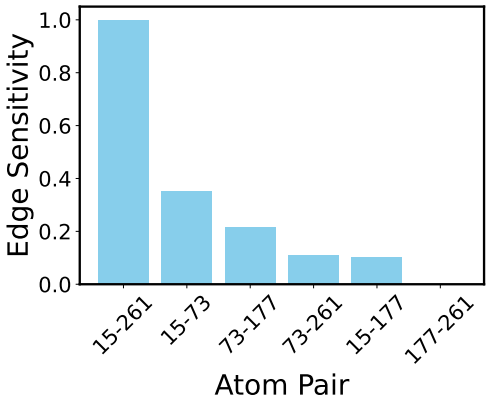

Fig. S7: **Edge-sensitivity of distances in Trp-cage from the unbiased simulation.** Edge sensitivity computed from a short biased simulation of 10 ns.

In addition to the unbiased simulation taken from the previous study on the fast-folding protein Trp-cage [12], we also tested the performance of our model on a WTM-eABF enhanced sampling of the folding dynamics of the Trp cage protein using the CHARMM36m force field. [13] To do so, we performed a 10-ns biased simulation along the RMSD with respect to the folded structure limited to the C_α atoms. (PDB: 2JOF) [14] The range of the CV extended from 0.0 Å to 10.0 Å with a bin width of 0.2 Å, and a harmonic wall was set at 10.5 Å with a force constant of 100 kcal/mol Å^2. The simulation was maintained using a Langevin thermostat with a damping coefficient of 10 ps^{-1} at a constant temperature of 300 K. The biasing force was applied once 10,000 samples were collected in each bin. The parameters of the extended Langevin dynamics were a damping coefficient of 1 ps^{-1}, an extended fluctuation of 0.1 Å and an extended time constant of 100 fs. For the well-tempered metadynamics part of the algorithm, the bias temperature was set to 1000 K, with a hill frequency of 1,000, a hill width of [0.2 Å, 0.2 Å], and a height of 0.1 kcal/mol.

Our results for the short-biased simulation and long-unbiased simulations are generally similar. However, the differences in node sensitivity may indicate that adding CV-dependent biases during the simulations can influence the model's prediction of the primary degrees of freedom. The reader is, therefore, advised to proceed with care when running biased simulations.

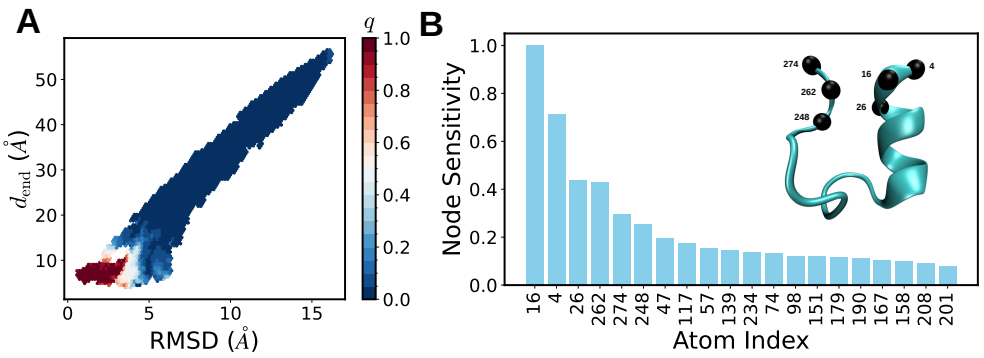

Fig. S8: **Trp-cage committor from a biased trajectory.** Learned committor from the biased simulation projected onto RMSD and end-to-end distance, from atomics coordinates of a biased trajectory of Trp-cage system (A). Node sensitivity computed from a short biased simulation fo 10 ns (B).

Committor projections onto the (d_1, d_3)- and (d_2, d_3)-subspaces. In Fig. S9, one can observe the committor projections onto the (d_1, d_3)- and (d_2, d_3)-subspaces to complete the scheme provided by the first three main distances in the edge-sensitivity analysis.

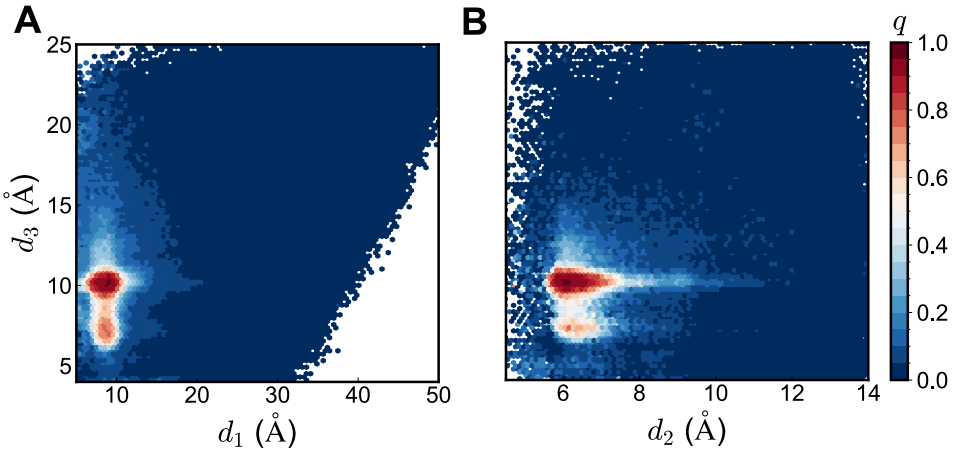

Fig. S9: **Role of d_3 in the committor of Trp-cage reversible folding from the unbiased simulation.** Projection of the learned committor using unbiased trajectory [12] onto the (d_1, d_3) (A) and (d_2, d_3)-subspaces (B).

Clustering at the separatrix. A clustering of the simulation frames in the vicinity of the separatrix, i.e. with a committor value between 0.45 and 0.55, was performed, using the K-means algorithm, as implemented in the sklearn python package [15]. By analyzing the clustering quality via the silhouette, [16] Calinski-Harabazs, [17] and Davies-Bouldin scores, [18] we

observed, as shown in Fig. S10, an elbow in the values as the cluster number increases. Thus, we determined that 4 represents a suitable number of clusters.

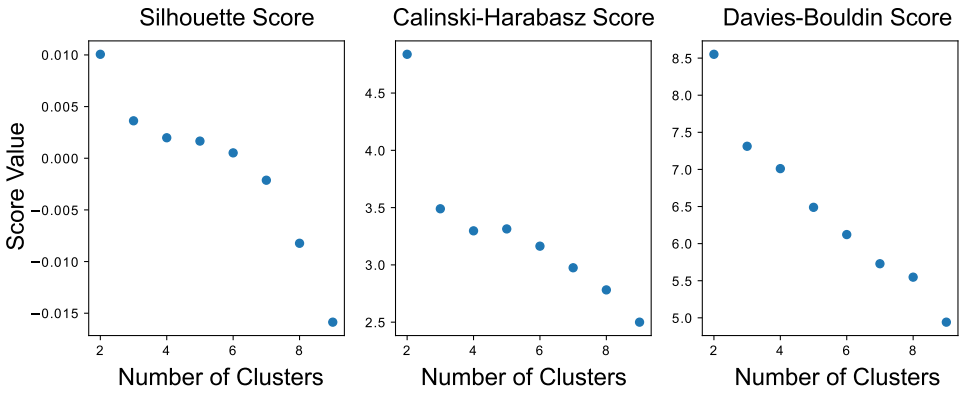

Fig. S10: Trp-cage cluster analysis. Cluster quality metrics for the Kmeans clustering along the separatrix.

References

[1] Roux, B.: Transition rate theory, spectral analysis, and reactive paths. J. Chem. Phys. **156**(13), 134111 (2022) https://doi.org/10.1063/5.0084209

[2] Chen, H., Roux, B., Chipot, C.: Discovering reaction pathways, slow variables, and committor probabilities with machine learning. J. Chem. Theory Comput. **19**(14), 4414–4426 (2023) https://doi.org/10.1021/acs.jctc.3c00028

[3] Megías, A., Contreras Arredondo, S., Chen, C.G., Tang, C., Roux, B., Chipot, C.: Iterative variational learning of committor-consistent transition pathways using artificial neural networks. Nat. Comput. Sci. (2025) https://doi.org/10.1038/s43588-025-00828-3

[4] Jing, B., Eismann, S., Suriana, P., Townshend, R.J.L., Dror, R.: Learning from protein structure with geometric vector perceptrons. In: International Conference on Learning Representations (2021). https://doi.org/10.48550/arXiv.2009.01411

[5] Satorras, V.G., Hoogeboom, E., Welling, M.: E(n) equivariant graph neural networks. In: Proceedings of the 38th International Conference on Machine Learning (PMLR) (2021). https://doi.org/10.48550/arXiv.2102.09844

[6] Schütt, K.T., Unke, O.T., Gastegger, M.: Equivariant message passing for the prediction of tensorial properties and molecular spectra. In: Proceedings of the 38 Th International Conference on Machine Learning (PMLR) (2021). https://doi.org/10.48550/arXiv.2102.03150

[7] Jing, B., Eismann, S., Soni, P.N., Dror, R.O.: Equivariant graph neural networks for 3d macromolecular structure. In: Proceedings of the 38 Th International Conference on Machine Learning (PMLR) (2021). https://doi.org/10.48550/arXiv.2106.03843

[8] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(56), 1929–1958 (2014) https://doi.org/10.5555/2627435.2670313

[9] Grattarola, D., Zambon, D., Bianchi, F.M., Alippi, C.: Understanding pooling in graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **35**(2), 2708–2718 (2024) https://doi.org/10.1109/TNNLS.2022.3190922

[10] Weiss, G.H.: First passage times for correlated random walks and some generalizations. J. Stat. Phys. **37**, 325–330 (1984) https://doi.org/10.1007/BF01011837

[11] Hänggi, P., Talkner, P., Borkovec, M.: Reaction-rate theory: fifty years after Kramers. Rev. Mod. Phys. **62**(2), 251 (1990) https://doi.org/10.1103/RevModPhys.62.251

[12] Lindorff-Larsen, K., Piana, S., Dror, R.O., Shaw, D.E.: How Fast-Folding Proteins Fold. Science **334**(6055), 517–520 (2011) https://doi.org/10.1126/science.1208351

[13] Huang, J., Rauscher, S., Nawrocki, G., Ran, T., Feig, M., De Groot, B.L., Grubmüller,

H., MacKerell Jr, A.D.: Charmm36m: an improved force field for folded and intrinsically disordered proteins. Nat. Methods **14**(1), 71–73 (2017)

[14] Barua, B., Lin, J.C., Williams, V.D., Kummler, P., Neidigh, J.W., Andersen, N.H.: The trp-cage: optimizing the stability of a globular miniprotein. Protein Eng. Des. Sel. **21**(3), 171–185 (2008) https://doi.org/10.1093/protein/gzm082

[15] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.*: Scikit-learn: Machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)

[16] Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**, 53–65 (1987) https://doi.org/10.1016/0377-0427(87)90125-7

[17] Caliński, T., Harabasz, J.: A dendrite method for cluster analysis. Commun. Stat. **3**(1), 1–27 (1974) https://doi.org/10.1080/03610927408827101

[18] Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-1**(2), 224–227 (1979) https://doi.org/10.1109/TPAMI.1979.4766909