Machine Learning of coarse-grained Molecular Dynamics Force Fields

Jiang Wang,[1,2] Simon Olsson,[3] Christoph Wehmeyer,[3] Adrià Pérez,[4] Nicholas E. Charron,[1,5] Gianni de Fabritiis,[4,6] Frank Noé,[3,1,2,a)] and Cecilia Clementi[1,2,5,3,b)]

[1)] *Rice University, Center for Theoretical Biological Physics, Houston, Texas 77005, United States*

[2)] *Rice University, Department of Chemistry, Houston, Texas 77005, United States*

[3)] *Freie Universität Berlin, Department of Mathematics and Computer Science, Arnimallee 6, 14195 Berlin, Germany*

[4)] *Computational Science Laboratory, Universitat Pompeu Fabra, PRBB, C/ Dr Aiguader 88, 08003, Barcelona, Spain*

[5)] *Rice University, Department of Physics, Houston, Texas 77005, United States*

[6)] *Institucio Catalana de Recerca i Estudis Avanats (ICREA), Passeig Lluis Companys 23, Barcelona 08010, Spain*

SUPPLEMENTARY MATERIAL

A. Decomposition of the force matching error

The decomposition of the force matching error (4) can be achieved by adding and subtracting the mean force (5) and splitting the norm:

$$\begin{aligned}\chi^2\left[U(\mathbf{x})\right] &= \left\langle \left\langle \left\| \xi(\mathbf{F}(\mathbf{r})) - \mathbf{f}(\mathbf{x}) + \mathbf{f}(\mathbf{x}) + \nabla U(\mathbf{x}) \right\|^2 \right\rangle_{\mathbf{r}|\mathbf{x}} \right\rangle_{\mathbf{x}} \\ &= \left\langle \left\langle \left\| \xi(\mathbf{F}(\mathbf{r})) - \mathbf{f}(\mathbf{x}) \right\|^2 \right\rangle_{\mathbf{r}|\mathbf{x}} \right\rangle_{\mathbf{x}} + \left\langle \left\| \mathbf{f}(\mathbf{x}) + \nabla U(\mathbf{x}) \right\|^2 \right\rangle_{\mathbf{x}} \\ &\quad + 2 \left\langle \left\langle (\xi(\mathbf{F}(\mathbf{r})) - \mathbf{f}(\mathbf{x}))^\top (\mathbf{f}(\mathbf{x}) + \nabla U(\mathbf{x})) \right\rangle_{\mathbf{r}|\mathbf{x}} \right\rangle_{\mathbf{x}}.\end{aligned}$$

This expression is equivalent to Eq. (6). as the mixed term is zero:

$$\begin{aligned}&\left\langle \left\langle (\xi(\mathbf{F}(\mathbf{r})) - \mathbf{f}(\mathbf{x}))^\top (\mathbf{f}(\mathbf{x}) + \nabla U(\mathbf{x})) \right\rangle_{\mathbf{r}|\mathbf{x}} \right\rangle_{\mathbf{x}} \\ &= \left\langle \mathbf{f}(\mathbf{x})^\top \mathbf{f}(\mathbf{x}) \right\rangle_{\mathbf{x}} + \left\langle \mathbf{f}(\mathbf{x})^\top \nabla U(\mathbf{x}) \right\rangle_{\mathbf{x}} \\ &\quad - \left\langle \mathbf{f}(\mathbf{x})^\top \mathbf{f}(\mathbf{x}) \right\rangle_{\mathbf{x}} - \left\langle \mathbf{f}(\mathbf{x})^\top \nabla U(\mathbf{x}) \right\rangle_{\mathbf{x}} \\ &= 0\end{aligned}$$

The decomposition of the expected prediction error in the form of Eq. (11) can be achieved by adding and subtracting the mean estimator $\bar{\mathbf{f}}(\mathbf{X}) = \mathbb{E}\left[-\nabla U(\mathbf{X};\boldsymbol{\theta})\right]$:

[a)] Electronic mail: frank.noe@fu-berlin.de

[b)] Electronic mail: cecilia@rice.edu

$$
\begin{aligned}
\mathbb{E}\left[L(\boldsymbol{\theta};\mathbf{R})\right] &= \mathbb{E}_{\mathbf{R}|\mathbf{X}}\left[\left\|\mathbf{f}(\mathbf{X}) + \nabla U(\mathbf{X};\boldsymbol{\theta})\right\|_F^2\right] + \text{Noise} \\
&= \mathbb{E}\left[\left\|\underbrace{\left(\mathbf{f}(\mathbf{X}) - \bar{\mathbf{f}}(\mathbf{X})\right)}_{A} + \underbrace{\left(\bar{\mathbf{f}}(\mathbf{X}) + \nabla U(\mathbf{X};\boldsymbol{\theta})\right)}_{B}\right\|_F^2\right] + \text{Noise} \\
&= \mathbb{E}\left[\|A\|_F^2\right] + \mathbb{E}\left[\|B\|_F^2\right] + 2\mathbb{E}\left[\sum_{i,j}(A*B)_{i,j}\right] + \text{Noise},
\end{aligned}
$$

where $*$ is the element-wise product. We follow standard results for regression. For the mixed term we can use

$$
\mathbb{E}\left[\sum_{i,j}(A*B)_{i,j}\right] = \sum_{i,j}\mathbb{E}\left[(A*B)_{i,j}\right] = \sum_{i,j}\left(\mathbb{E}\left[A*B\right]\right)_{i,j}
$$

and this expectation value disappears:

$$
\begin{aligned}
\mathbb{E}\left[A*B\right] &= \mathbb{E}\left[\left(\mathbf{f}(\mathbf{X}) - \bar{\mathbf{f}}(\mathbf{X})\right) * \left(\bar{\mathbf{f}}(\mathbf{X}) + \nabla U(\mathbf{X};\boldsymbol{\theta})\right)\right] \\
&= \mathbb{E}\left[\mathbf{f}(\mathbf{X})\right] * \bar{\mathbf{f}}(\mathbf{X}) + \mathbb{E}\left[\mathbf{f}(\mathbf{X}) * \nabla U(\mathbf{X};\boldsymbol{\theta})\right] - \mathbb{E}\left[\bar{\mathbf{f}}(\mathbf{X}) * \bar{\mathbf{f}}(\mathbf{X})\right] - \bar{\mathbf{f}}(\mathbf{X}) * \mathbb{E}\left[\nabla U(\mathbf{X};\boldsymbol{\theta})\right] \\
&= \mathbf{f}(\mathbf{X}) * \bar{\mathbf{f}}(\mathbf{X}) - \mathbf{f}(\mathbf{X}) * \bar{\mathbf{f}}(\mathbf{X}) - \bar{\mathbf{f}}(\mathbf{X}) * \bar{\mathbf{f}}(\mathbf{X}) + \bar{\mathbf{f}}(\mathbf{X}) * \bar{\mathbf{f}}(\mathbf{X}) \\
&= 0.
\end{aligned}
$$

The remaining terms define bias and variance.

B. Cross-validation for the coarse-graining of the 2d toy model

We report here the results from cross-validation for the choice of hyper-parameters for the coarse-graining of the 2d toy model discussed in the main text.

The feature regression for the coarse-graining of the 2 dimensional toy model is performed with the twenty basis functions listed in Table S1 selected as features. Cross-validation is performed with the Stepwise Sparse Regressor introduced in[1]. The minimum cross-validation error is obtained when the first four functions are used as features, as shown in Fig. S1.

Table S1. Twenty elementary basis functions.

function ID	function, $f(x)$	function ID	function, $f(x)$
1	1	11	x^{10}
2	x	12	$sin(x)$
3	x^2	13	$cos(x)$
4	x^3	14	$sin(6x)$
5	x^4	15	$cos(6x)$
6	x^5	16	$sin(11x)$
7	x^6	17	$cos(11x)$
8	x^7	18	$tanh(10x)$
9	x^8	19	$tanh^2(10x)$
10	x^9	20	e^{-50x^2}

The results from the cross-validation of the CGnet for the toy 2 dimensional system are reported in Tables S2 and Fig. S1.

Table S2. Hyper-parameter optimization for unregularized CGnet of two-dimensional model system. D: network depth, W : network width. The unit of the cross-validation error is $(k_BT)^2$, with the unit of length equal to 1.

D ($W = 20$)	Cross-validation error
1	**0.3785 ± 0.0024**
2	0.5457 ± 0.0973
3	0.7339 ± 0.0298
4	0.5695 ± 0.0172
5	0.8543 ± 0.1227

W ($D = 1$)	Cross-validation error
5	0.5674 ± 0.0044
10	0.8762 ± 0.0048
20	0.3785 ± 0.0024
40	0.3729 ± 0.0017
60	0.3703 ± 0.0013
80	0.3682 ± 0.0013
100	0.3671 ± 0.0013
120	**0.3661 ± 0.0012**
150	0.3661 ± 0.0012

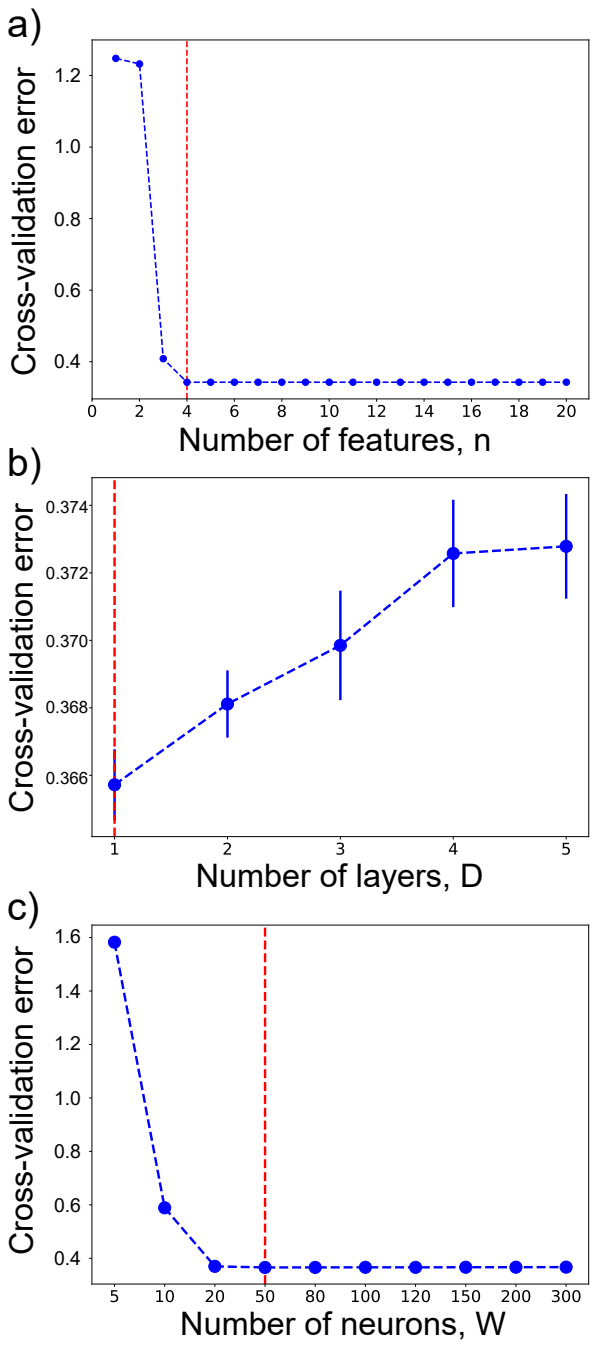

Figure S1. Model selection for CG model of 2D system using cross-validation. a) Choice of the set of feature functions for feature regression. b) First stage of regularized CGnet hyper-parameter selection: the optimal number of layers, D. c). Second stage of regularized CGnet hyperparameter selection: the optimal number of neurons per layer, W. Red dashed lines indicate the minimal cross-validation error. Error bars represent the standard error of the mean cross-validation error over five cross-validation folds, in panels a) and c) the error bars are invisible as they are smaller than the marker. The unit of the cross-validation error is $(k_B T)^2$, with the unit of length equal to 1.

C. Training CG models

Networks were optimized using the Adam adaptive stochastic gradient descent method[2] with default settings using the PyTorch program. The batch-size was 128 for the 2D model and 512 for alanine dipeptide. The convergence of the training error and validation error for the 2d toy model and alanine dipeptide is shown in Fig. S2 below.

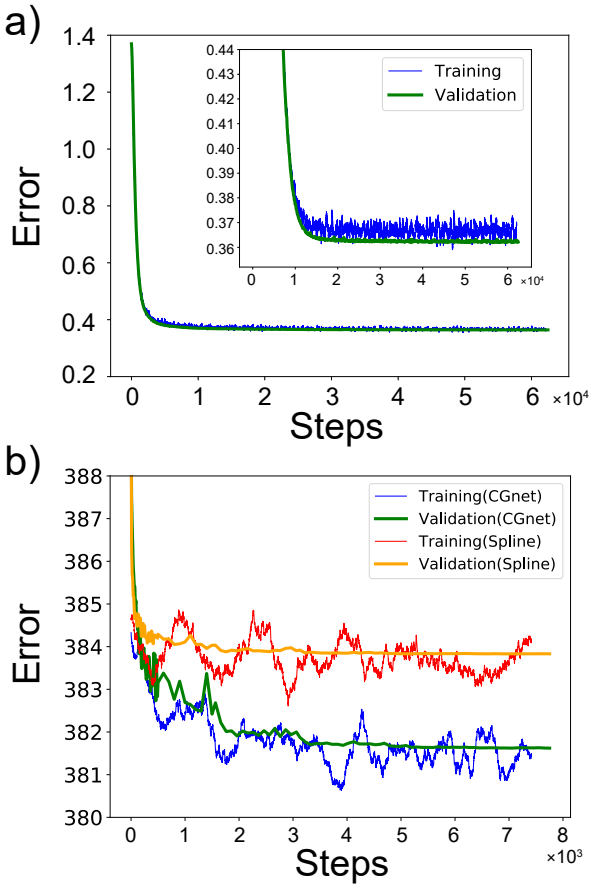

Figure S2. Training error and validation error for (a) the 2D model and (b) alanine dipeptide. In (a), the model is the regularized CGnet, in (b), the model is the regularized CGnet and the spline model, which is also regularized. All errors are averaged over 200000 points – for the training error this is done by averaging over the most recent batches, while the validation error is shown for a fixed validation set. Note that the hyper-parameter choices are made via cross-validation. The unit of the error is $(k_BT)^2$ in (a) and $[kcal/(mol{\cdot}\mathring{A})]^2$ in (b).

D. Distribution of bond distances and angles for the different models of alanine dipeptide

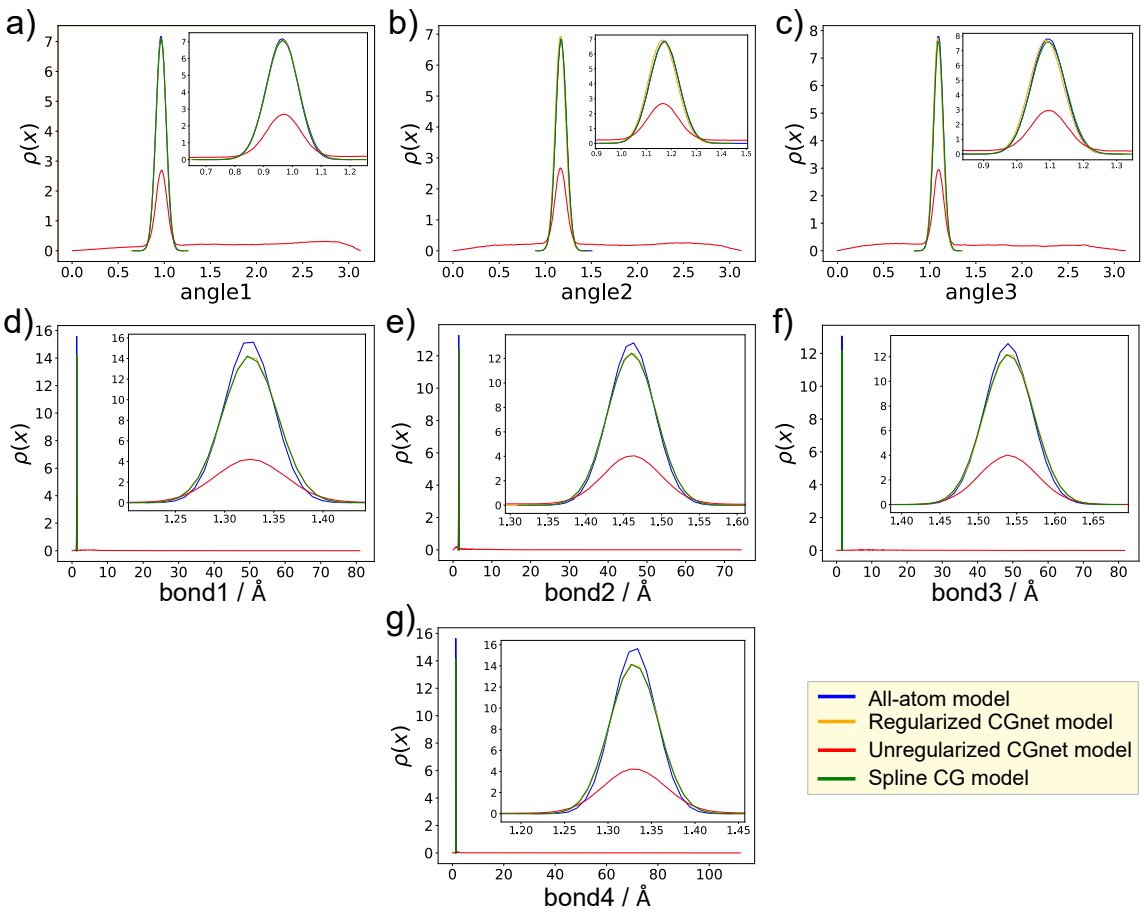

Figure S3. Probability density distribution for three angles a)- c), and four bonds d)-g) for the alanine dipeptide models. Each panel contains the distribution from four models: All-atom model (blue), regularized CGnet model (red), unregularized CGnet model (cyan), spline CG model (green). The distribution for regularized CGnet and spline model (with regularization) agree with the true all-atom one. The distribution for the unregularized CGnet has a wide range, which makes the distributions for the other models appear very narrow in d)-g). The insets in d)-g) present zoomed views of the distributions in the correct range.

E. Changes in the free energy of alanine dipeptide with different hyper-parameters

In order to show how the free energy is approximating the atomistic free energy as the hyper-parameters gradually reach the optimal values, we select five hyper-parameters for CGnet (C1, C2, C3, C4, C5) and four for the Spline model (S1, S2, S3, S4), as indicated in Fig. 5 in the manuscript. For each of these combinations of hyper-parameters, we report the corresponding two dimensional free energy profiles in Fig. S4 and Fig. S5 (in addition to the free energy profile for the global optima reported in Fig. 6). The figures show that as the hyper-parameters get closer to the optimal values the model free energy landscape becomes closer to the atomistic free energy landscape.

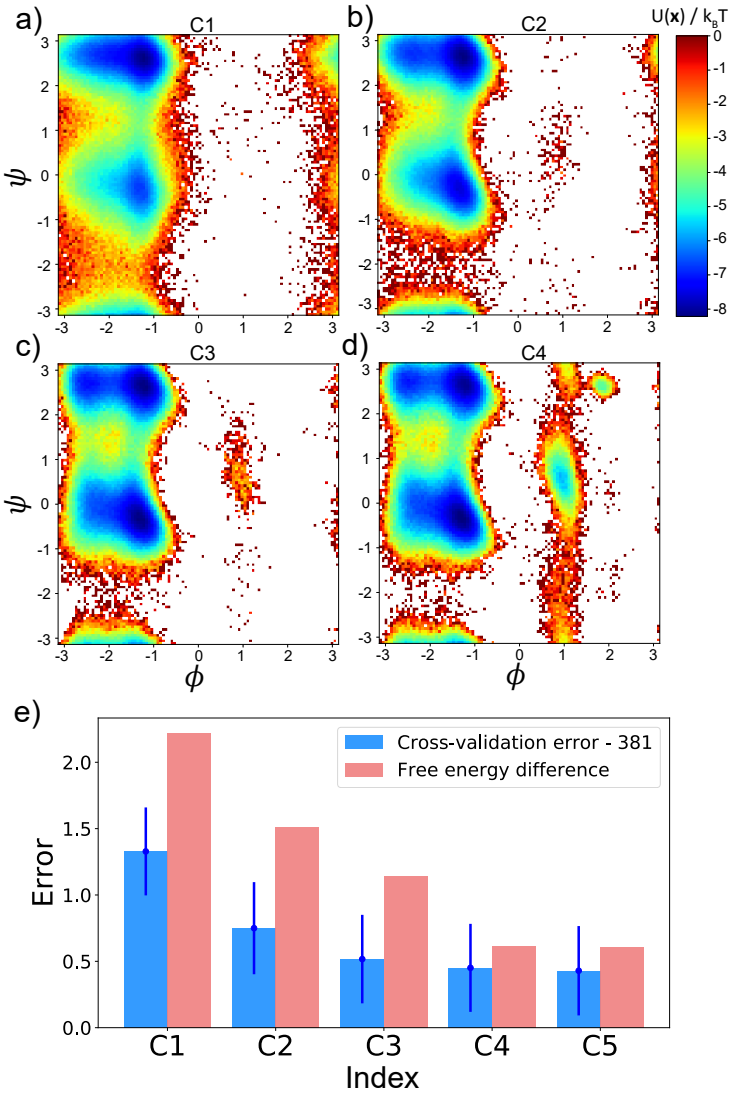

Figure S4. Comparison of the free energy profiles of CGnet models of alanine dipeptide with different choices of hyper-parameters. (a)-(d) Free energy profiles with hyperparameters corresponding to the combination indicated as C1, C2, C3, C4 in Fig. 5. The choice of hyperparameters C5 correspond to the global optimum and is reported in Fig. 6c. (e) Comparison between the cross validation error (in $[kcal/(mol{\cdot}\mathring{A})]^2$) and mean square free energy difference (in $[k_BT]^2$) for the five selected hyperparameters. The value of 381 is subtracted from the cross validation error to obtain values in the similar range as the free energy differences.

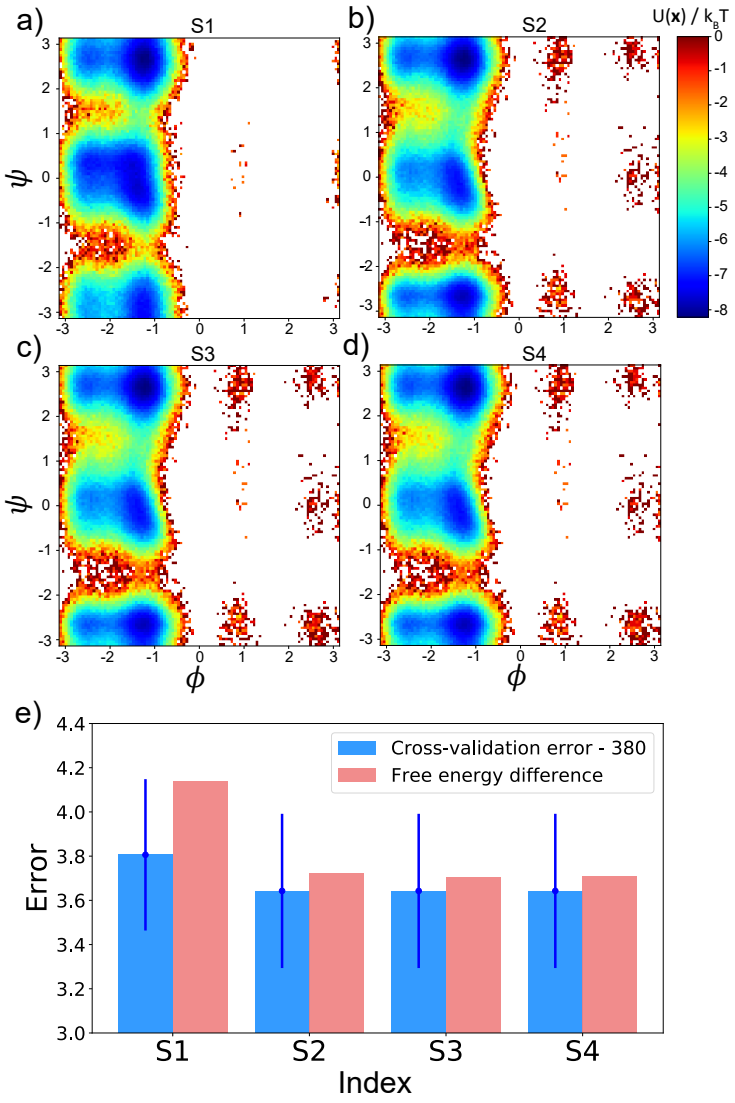

Figure S5. Comparison of the free energy profiles of the spline models of alanine dipeptide with different choices of hyper-parameters. (a)-(d) Free energy profiles with hyperparameters corresponding to the combination indicated as S1, S2, S3, S4 in Fig. 5. The choice of hyperparameters S4 correspond to the global optimum and is also reported in Fig. 6b. (e) Comparison between the cross validation error (in $[kcal/(mol{\cdot}\text{Å})]^2$) and mean square free energy difference (in $[k_BT]^2$) for the five selected hyperparameters. The value of 380 is subtracted from the cross validation error to obtain values in the similar range as the free energy differences.

F. Energy decomposition for the CGnet model of alanine dipeptide.

As discussed in the main text, the use of a baseline energy to enforce physical constraints plays an important role in the CGnet model. Here we report the decomposition of the total CGnet energy into the contribution of the baseline (prior) energy and the energy of the neural network. Figs. S6a-c report the decomposition for each point sampled in the simulations performed with CGnet. Fig. S6d-f report the same quantity averaged over different bins in the space spanned by the dihedral angles. The figures show that the network energy captures the overall features of the free energy landscape for this molecule, while the prior energy seems to play an important role to enforce physical constraints mostly at the edges of the populated regions in the landscape. This is in agreement with the intuition that the prior energy term makes the system avoid high energy regions not visited in the training data.

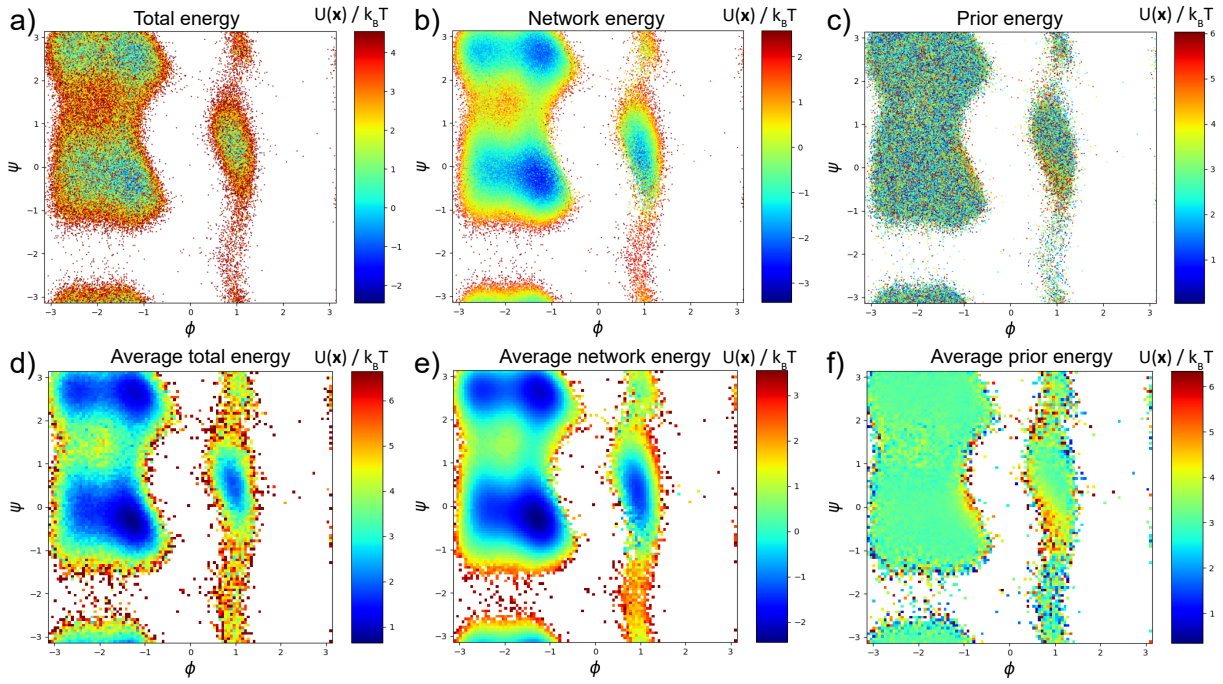

Figure S6. CGnet energy decomposition for the alanine dipeptide. In each simulated point, the total CGnet energy (a) is decomposed in the energy contribution from the dense net (b), and the baseline (or prior) energy (c). In each bin in the dihedral angles space, the average total energy (d) is decomposed into the average dense net energy (e), and average prior energy (f).

G. Chignolin setup and simulation

The initial structure of Chignolin was generated starting from the cln025 peptide[3], with sequence TYR-TYR-ASP-PRO-GLU-THR-GLY-THR-TRP-TYR. The structure was solvated in a cubic box of 40 Å, containing 1881 water molecules and two Na^+ ions to neutralize the peptide's negative charge, as described in[4]. MD simulations were performed with ACEMD[5], using CHARMM22*[6] force field and TIP3P[7] water model at 350K temperature. A Langevin integrator was used with a damping constant of 0.1 ps^{-1}. Integration time step was set to 4 fs, with heavy hydrogen atoms (scaled up to four times the hydrogen mass) and holonomic constrains on all hydrogen-heavy atom bond terms[8]. Electrostatics were computed using Particle Mesh Ewald with a cutoff distance of 9 Å and grid spacing of 1 Å. Ten NVT simulations of 1 ns length were carried out, with a dielectric constant of 80 and temperature of 350K to generate ten different starting conformations for the production runs. Production simulations consisted of 3744 independent simulations of 50 ns, for a total aggregate time of 187.2 μs. All the simulations were run using the GPUGRID[9] distributed computing

platform. The first 1000 simulations were spawned from the 10 conformations obtained previously. The remaining 2744 simulations were spawned using the adaptive sampling[10] protocol implemented in HTMD[11]. In adaptive sampling, multiple rounds of simulations are performed, and each round the available trajectories are analyzed to select the initial coordinates for the next round of simulations. Each round was done every 10 to 20 simulations, respawning an equivalent amount of new simulations. Initial coordinates for the respawned simulations were selected proportionally to the inverse of the number of frames per macrostate as explained in[11]. The Markov State Model[12–16] constructed during the analysis was done using atom distances as projected metric, TICA[17,18] for dimensionality reduction method and k-Centers for clustering. Force data used for training CGnet was obtained from the MD simulation trajectories. ACEMD was used to read the Chignolin trajectories and compute forces for all atoms for each simulation frame, using the same parameters used for the MD simulations.

H. Markov State Model analysis of Chignolin all-atom simulations

MD simulation data of Chignolin from GPUGrid was featurized into all pairwise C_α distances excluding pairs of nearest neighbors residues (a total of 45 distances). Time-lagged independent component analysis (TICA)[17,18] was carried out with a lag $\tau = 25\,\text{ns}$. By using kinetic-map[19,20] and a kinetic variance cutoff of 95%, 4 TICs were retained for further analysis. The 4 TICs were clustered into 350 discrete states using the k-means algorithm. All MD data was mapped onto their discrete states and used for Markov state model (MSM) estimation. The implied-timescales, $t_i = -\frac{\tau}{\log|\lambda_i|}$, become constant as a function of lag-time (τ) within statistical uncertainty for lag-times above approximately 20 ns. Spectral analysis of a Markov state model estimated at a lagtime $\tau = 37.5\,\text{ns}$ reveal a spectral gap after the third implied-timescale suggesting 4 meta-stable states (Fig. S7). Plotting the populations of the meta-stable states as function of lag-time show that these are stable for $\tau > 10\,\text{ns}$, and that three of the four meta-stable states have significant probability mass $> 1\%$. These three most stable meta-stable states were used as reference states a, b and c, ordered alphabetically from most to least populated (shown in Fig. 7a). To account for the non-equilibrium nature of the multiple short molecular dynamics trajectories, we used the estimated MSM ($\tau = 37.5\,\text{ns}$) to reweighed data prior to calculating the reference free energy profiles. These analyses were carried out using the PyEMMA[21] and MDTraj[22] software packages.

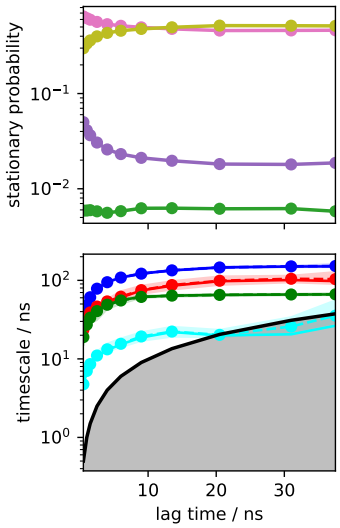

Figure S7. Validation of a convergence of the Chignolin all-atom Markov model, which is estimated at $\tau = 37.5\,\text{ns}$. Top: Stationary probabilities of metastable states. Bottom: MSM implied time scales.

I. Hyper-parameter optimization for Chignolin CG models

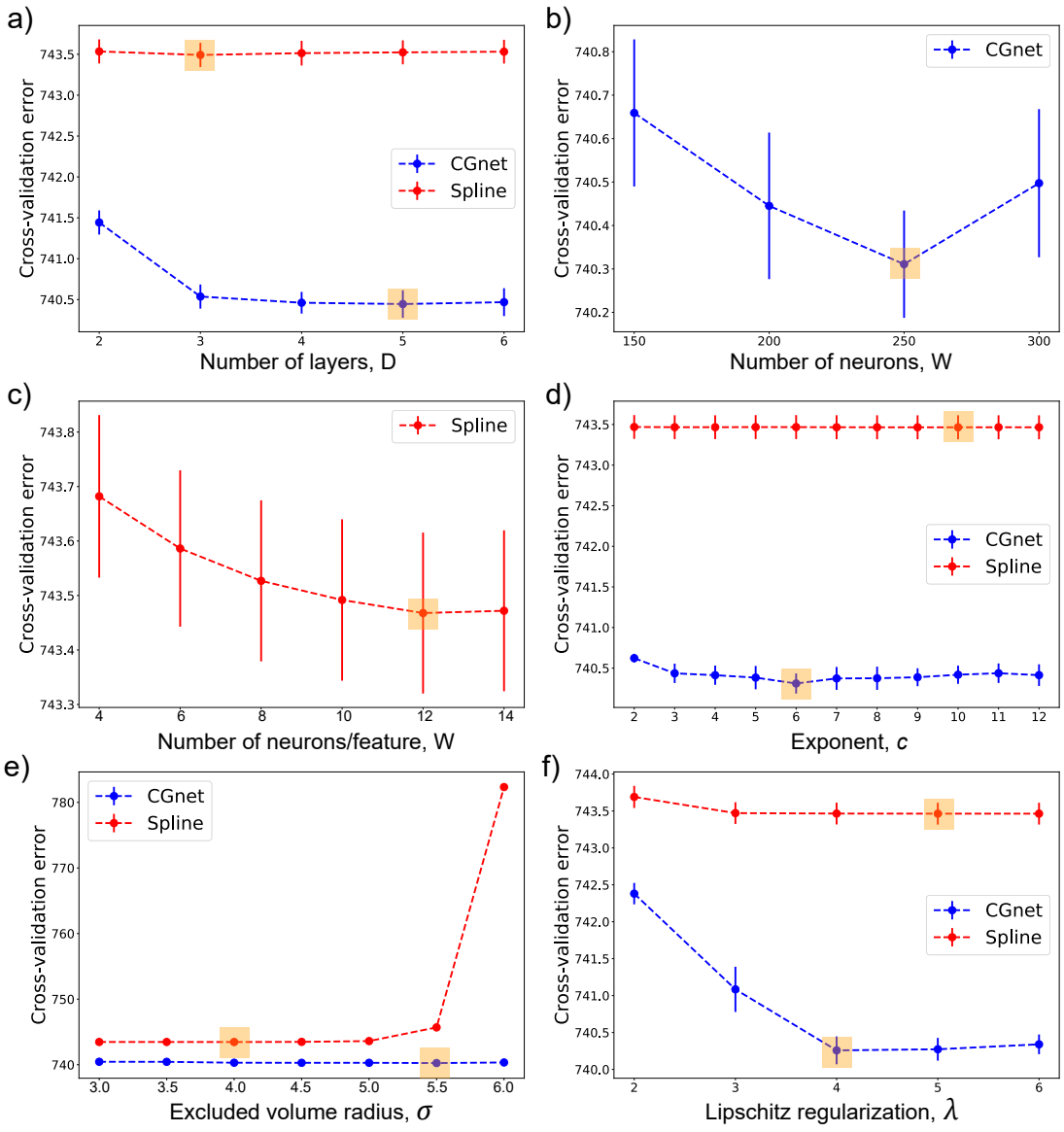

Figure S8. Five-stage cross-validation of the hyper-parameters for the CG models of Chignolin. (a) Selection of the number of layers, D. (b) and (c) Selection of the number of neurons per layer, W. (d) Selection of the exponent of the excluded volume term, c. (d) Selection of the effective excluded volume radius, σ. (f) Selection of of the Lipschitz regularization strength, λ. The optimal values are indicated by orange squares and are used to generate the results reported in Fig. 7.

REFERENCES

[1] L. Boninsegna, F. Nüske, and C. Clementi. Sparse learning of stochastic dynamical equations. *J. Chem. Phys.*, 148(24):241723, 2018.

[2] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[3]S. Honda, T. Akiba, Y. S. Kato, Y. Sawada, M. Sekijima, M. Ishimura, A. Ooishi, H. Watanabe, T. Odahara, and K. Harata. Crystal structure of a ten-amino acid protein. *J. Am. Chem. Soc.*, 130(46):15327–15331, 2008.
[4]K. Lindorff-Larsen, S. Piana, R. O. Dror, and D. E. Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–20, 2011.
[5]M. J. Harvey, G. Giupponi, and G. De Fabritiis. ACEMD: Accelerating biomolecular dynamics in the microsecond time scale. *J. Chem. Theory Comput.*, 5(6):1632–1639, 2009.
[6]S. Piana, K. Lindorff-Larsen, and D. E. Shaw. How robust are protein folding simulations with respect to force field parameterization? *Biophys. J.*, 100(9):L47–L49, 2011.
[7]W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.*, 79(2):926–935, 1983.
[8]K. A. Feenstra, B. Hess, and H. J. Berendsen. Improving efficiency of large time-scale molecular dynamics simulations of hydrogen-rich systems. *J. Comput. Chem.*, 20(8):786–798, 1999.
[9]I. Buch, M. J. Harvey, T. Giorgino, D. P. Anderson, and G. De Fabritiis. High-throughput all-atom molecular dynamics simulations using distributed computing. *J. Chem. Inf. Model.*, 50(3):397–403, 2010.
[10]S. Doerr and G. De Fabritiis. On-the-Fly Learning and Sampling of Ligand Binding by High- Throughput Molecular Simulations. *J. Chem. Theory Comput.*, 10(5):2064–2069, 2014.
[11]S. Doerr, M. J. Harvey, F. Noé, and G. De Fabritiis. HTMD: High-Throughput Molecular Dynamics for Molecular Discovery. *J. Chem. Theory Comput.*, 12(4):1845–1852, 2016.
[12]J.-H. Prinz, H. Wu, M. Sarich, B. G. Keller, M. Senne, M. Held, J. D. Chodera, C. Schütte, and F. Noé. Markov models of molecular kinetics: Generation and validation. *J. Chem. Phys.*, 134:174105, 2011.
[13]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.
[14]J. D. Chodera, K. A. Dill, N. Singhal, V. S. Pande, W. C. Swope, and J. W. Pitera. Automatic discovery of metastable states for the construction of Markov models of macromolecular conformational dynamics. *J. Chem. Phys.*, 126:155101, 2007.
[15]F. Noé, I. Horenko, C. Schütte, and J. C. Smith. Hierarchical Analysis of Conformational Dynamics in Biomolecules: Transition Networks of Metastable States. *J. Chem. Phys.*, 126:155102, 2007.
[16]N. V. Buchete and G. Hummer. Coarse Master Equations for Peptide Folding Dynamics. *J. Phys. Chem. B*, 112:6057–6069, 2008.
[17]G. Perez-Hernandez, F. Paul, T. Giorgino, G. D Fabritiis, and F. Noé. Identification of slow molecular order parameters for markov model construction. *J. Chem. Phys.*, 139:015102, 2013.
[18]C. R. Schwantes and V. S. Pande. Improvements in markov state model construction reveal many non-native interactions in the folding of ntl9. *J. Chem. Theory Comput.*, 9:2000–2009, 2013.
[19]F. Noé and C. Clementi. Kinetic distance and kinetic maps from molecular dynamics simulation. *J. Chem. Theory Comput.*, 11:5002–5011, 2015.
[20]F. Noé, R. Banisch, and C. Clementi. Commute maps: separating slowly-mixing molecular configurations for kinetic modeling. *J. Chem. Theory Comput.*, 12:5620–5630, 2016.
[21]M. K. Scherer, B. Trendelkamp-Schroer, F. Paul, G. Perez-Hernandez, M. Hoffmann, N. Plattner, C. Wehmeyer, J.-H. Prinz, and F. Noé. PyEMMA 2: A software package for estimation, validation and analysis of Markov models. *J. Chem. Theory Comput.*, 11:5525–5542, 2015.
[22]R. T. McGibbon, K. A. Beauchamp, M. P. Harrigan, C. Klein, J. M. Swails, C. X. Hernández, C. R. Schwantes, L. P. Wang, T. J. Lane, and V. S. Pande. Mdtraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophys J.*, 109:1528–1532, 2015.