RESEARCH ARTICLE SUMMARY

MACHINE LEARNING

Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning

Frank Noé*†, Simon Olsson*, Jonas Köhler*, Hao Wu

INTRODUCTION: Statistical mechanics aims to compute the average behavior of physical systems on the basis of their microscopic constituents. For example, what is the probability that a protein will be folded at a given temperature? If we could answer such questions efficiently, then we could not only comprehend the workings of molecules and materials, but we could also design drug molecules and materials with new properties in a principled way.

To this end, we need to compute statistics of the equilibrium states of many-body systems. In the protein-folding example, this means to consider each of the astronomically many ways to place all protein atoms in space, to compute the probability of each such "configuration" in the equilibrium ensemble, and then to compare the total probability of unfolded and folded configurations.

As enumeration of all configurations is infeasible, one instead must attempt to sample them from their equilibrium distribution. However, we currently have no way to generate equilibrium samples of many-body systems in "one shot." The main approach is thus to start with one configuration, e.g., the folded protein state, and make tiny changes to it over time, e.g., by using Markov-chain Monte Carlo or molecular dynamics (MD). However, these simulations get trapped in metastable (long-lived) states: For example, sampling a single folding or unfolding event with atomistic MD may take a year on a supercomputer.

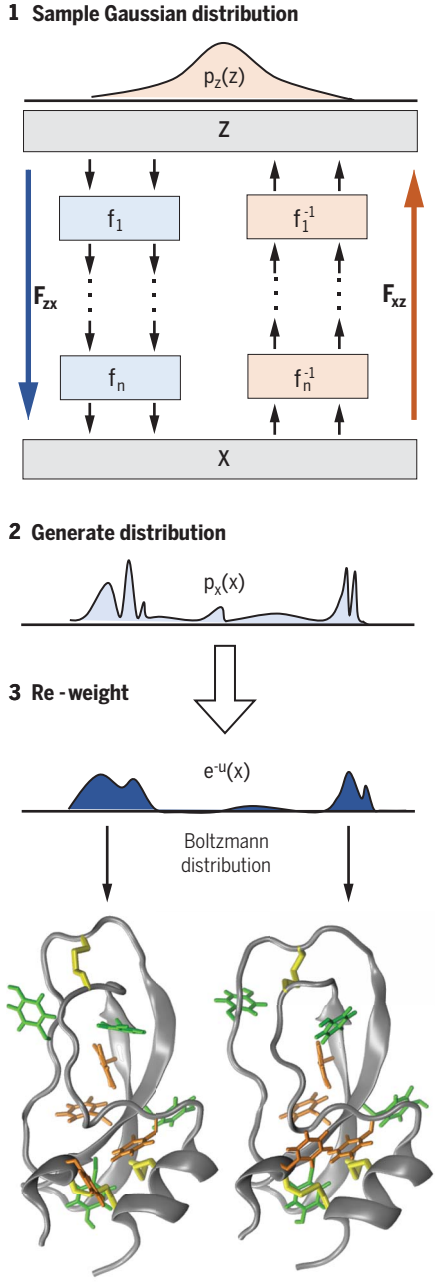
RATIONALE: Here, we combine deep machine learning and statistical mechanics to develop Boltzmann generators. Boltzmann generators are trained on the energy function of a many-body system and learn to provide unbiased, one-shot samples from its equilibrium state. This is achieved by training an invertible neural network to learn a coordinate transformation from a system's configurations to a so-called latent space representation, in which the low-energy configurations of different states are close to each other and can be easily sampled. Because of the invertibility, every latent space sample can be back-transformed to a system configuration with high Boltzmann probability (Fig. 1). We then employ statistical mechanics, which offers a rich set of tools for reweighting the distribution generated by the neural network to the Boltzmann distribution.

ON OUR WEBSITE

Read the full article at http://dx.doi.org/10.1126/science.aaw1147

RESULTS: Boltzmann generators can be trained to directly generate independent samples of low-energy structures of condensed-matter systems and protein molecules. When initialized with a few structures from different metastable states, Boltzmann generators can generate statistically independent samples from these states and efficiently compute the free-energy differences between them. This capability could be used to compute relative stabilities between different experimental structures of protein or other organic molecules, which is currently a very challenging problem. Boltzmann generators can also learn a notion of "reaction coordinates": Simple linear interpolations between points in latent space have a high probability of corresponding to physically realistic, low-energy transition pathways. Finally, by using established sampling methods such as Metropolis Monte Carlo in the latent space variables, Boltzmann generators can discover new states and gradually explore state space.

CONCLUSION: Boltzmann generators can overcome rare event–sampling problems in many-body systems by learning to generate unbiased equilibrium samples from different metastable states in one shot. They differ conceptually from established enhanced sampling methods, as no reaction coordinates are needed to drive them between metastable states. However, by applying existing sampling methods in the latent spaces learned by Boltzmann generators, a plethora of new opportunities opens up to design efficient sampling methods for many-body systems. ■



Boltzmann generators overcome sampling problems between long-lived states. The Boltzmann generator works as follows: 1. We sample from a simple (e.g., Gaussian) distribution. 2. An invertible deep neural network is trained to transform this simple distribution to a distribution $p_X(x)$ that is similar to the desired Boltzmann distribution of the system of interest. 3. To compute thermodynamics quantities, the samples are reweighted to the Boltzmann distribution using statistical mechanics methods.

The list of author affiliations is available in the full article online.
*These authors contributed equally to this work.
†Corresponding author. Email: frank.noe@fu-berlin.de
Cite this article as F. Noé *et al.*, *Science* **365**, eaaw1147 (2019). DOI: 10.1126/science.aaw1147

RESEARCH ARTICLE

MACHINE LEARNING

Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning

Frank Noé[1,2,3]*†, Simon Olsson[1]*, Jonas Köhler[1]*, Hao Wu[4,1]

Computing equilibrium states in condensed-matter many-body systems, such as solvated proteins, is a long-standing challenge. Lacking methods for generating statistically independent equilibrium samples in "one shot," vast computational effort is invested for simulating these systems in small steps, e.g., using molecular dynamics. Combining deep learning and statistical mechanics, we developed Boltzmann generators, which are shown to generate unbiased one-shot equilibrium samples of representative condensed-matter systems and proteins. Boltzmann generators use neural networks to learn a coordinate transformation of the complex configurational equilibrium distribution to a distribution that can be easily sampled. Accurate computation of free-energy differences and discovery of new configurations are demonstrated, providing a statistical mechanics tool that can avoid rare events during sampling without prior knowledge of reaction coordinates.

Statistical mechanics is concerned with computing the average behavior of many copies of a physical system based on its microscopic constituents and their interactions. For example, what is the average magnetization in an Ising model of interacting magnetic spins? Or what is the probability of a protein to be folded as a function of the temperature? Under a wide range of conditions, the equilibrium probability of a microscopic configuration $\mathbf{x}$ (setting of all spins, positions of all protein atoms, etc.) is proportional to $e^{-u(\mathbf{x})}$, for example, the well-known Boltzmann distribution. The dimensionless energy $u(\mathbf{x})$ contains the potential energy of the system, the temperature, and, optionally, other thermodynamic quantities.

Except for simple model systems, we presently have no approach to directly draw "one-shot," i.e., statistically independent, samples $\mathbf{x}$ from Boltzmann-type distributions to compute statistics of the system, such as free-energy differences. Therefore, one currently must rely on trajectory methods such as Markov chain Monte Carlo (MCMC) or molecular dynamics (MD) simulations that make tiny changes to $\mathbf{x}$ in each step. These methods sample from the Boltzmann distribution in the long run, but many simulation steps are needed to produce a statistically independent sample. This is because complex systems often have metastable (long-lived) phases or states and the transitions between them are rare events; for example, 10^9 to 10^{15} MD simulation steps are needed to fold or unfold a protein. As a result, MCMC and MD methods are extremely expensive and consume much of the worldwide supercomputing resources.

A common approach to enhance sampling is to speed up rare events by biasing user-defined order parameters, or "reaction coordinates" (RCs), that may be of a mechanical (*1*–*4*), thermodynamic (*5*–*7*), or alchemical nature (*8*, *9*). Applying these techniques to high-dimensional systems with a priori unknown transition mechanisms is challenging, as identifying suitable order parameters and avoiding rare events in other, unbiased directions becomes extremely difficult. For example, the development of enhanced simulation protocols for the binding of small drug molecules to proteins has become a research area in its own right (*10*).

In this study, we set out to develop a "Boltzmann generator" machine that is trained on a given energy function $u(\mathbf{x})$ and then produces unbiased one-shot samples from $e^{-u(\mathbf{x})}$, circumventing the sampling problem without requiring any knowledge of RCs. At first sight, this enterprise seems hopeless for condensed-matter systems and complex polymers. In these systems, strongly repulsive particles are densely packed, such that the number of low-energy configurations is vanishingly small compared with the number of possible ways to place particles.

Key to the solution is combining the strengths of deep machine learning (*11*) and statistical mechanics (Fig. 1A). We train a deep invertible neural network to learn a coordinate transformation from $\mathbf{x}$ to a so-called "latent" representation $\mathbf{z}$, in which the low-energy configurations of different states are close to each other and can be easily sampled, e.g., using a Gaussian normal distribution. Enhancing MD sampling by user-defined English coordinate transformations has been proposed previously (*12*). The novelty of Boltzmann generators is that this transformation is learned and, owing to the deep transformation network, can be as complicated as needed to represent state changes in the many-body system. As Boltzmann generators are invertible, every sample $\mathbf{z}$ can be back transformed to a configuration $\mathbf{x}$ with high Boltzmann probability. We can improve the ability to find relevant parts of configuration space by "learning from example," where the potential energy $u(\mathbf{x})$ used to train the Boltzmann generator is complemented by relevant samples $\mathbf{x}$, e.g., from the folded or unfolded state of a protein but without knowing the probabilities of these states. Then, we use statistical mechanics, which offers a rich set of tools to generate the target distribution $e^{-u(\mathbf{x})}$ when the proposal distribution is sufficiently similar.

This study demonstrates that Boltzmann generators can be trained to generate low-energy structures of condensed matter systems and protein molecules in one shot, as shown for model systems and a millisecond-timescale conformational change of the bovine pancreatic trypsin inhibitor (BPTI) protein. When the Boltzmann generator is initialized with a few structures from different metastable states, it can generate statistically independent samples from these states and can compute the free-energy profiles of the corresponding transitions without suffering from rare events. Although Boltzmann generators do not require RCs, they can be included in the training to sample continuous free-energy profiles and low-probability states. When trained in this way, Boltzmann generators can also generate physically realistic transition pathways by performing simple linear interpolations in latent space. We also show that multiple independent Boltzmann generators, trained on disconnected MD or MCMC simulations of different states, can be used to compute free-energy differences between these states in a direct and inexpensive way and without requiring any RCs. Finally, we demonstrate that when using established sampling methods such as Metropolis Monte Carlo in the latent space of a Boltzmann generator, efficient methods can be constructed to find new states and gradually explore state space.

Boltzmann generators

Neural networks that can draw statistically independent samples from a desired distribution are called directed generative networks (*13*, *14*). Such networks have been demonstrated to generate photorealistic images (*15*), to produce deceivingly realistic speech audio (*16*), and even to sample formulae of chemical compounds with certain physicochemical properties (*17*). In these domains, the exact target distribution is not known and the network is "trained by example" using large databases of images, audio, or molecules. Here, we are in the inverse situation, as we can compute the Boltzmann weight of each

[1]FU Berlin, Department of Mathematics and Computer Science, Arnimallee 6, 14195 Berlin, Germany. [2]FU Berlin, Department of Physics, Arnimallee 14, 14195 Berlin, Germany. [3]Rice University, Department of Chemistry, Houston, TX 77005, USA. [4]Tongji University, School of Mathematical Sciences, Shanghai, 200092, P.R. China.
*These authors contributed equally to this work.
†Corresponding author. Email: frank.noe@fu-berlin.de

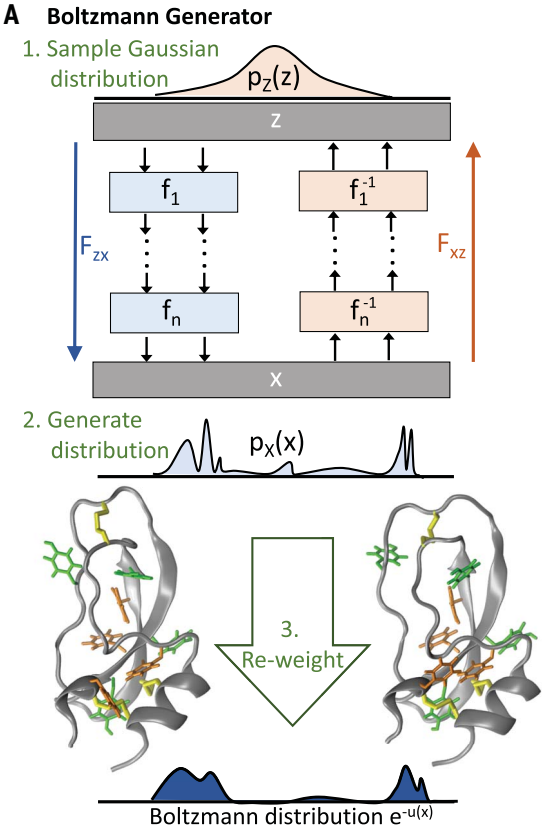


Fig. 1. Boltzmann generators. (**A**) A Boltzmann generator is trained by minimizing the difference between its generated distribution and the desired Boltzmann distribution. Generation proceeds by drawing "latent" space samples **z** from a simple prior distribution (e.g., Gaussian) and transforming them to configurations **x**. The variable transformation is formed by stacking invertible transformations $f_1, ..., f_n$ to a deep neural network F_{zx} and its inverse, F_{xz}. To compute thermodynamics, such as configurational free energies, the samples must be reweighted to the Boltzmann distribution. (**B**) A Boltzmann generator is composed of invertible neural network blocks. Here, a non–volume-preserving transformation block is shown as an example.

generated sample **x**, but we do not have samples from the Boltzmann distribution a priori. The idea of Boltzmann generators is as follows (Fig. 1A):

1) A neural network transformation F_{zx} is learned such that when sampling **z** from a simple prior, e.g., a Gaussian normal distribution, $F_{zx}(\mathbf{z})$ will provide a configuration **x** that has a high Boltzmann weight, i.e., is coming from a distribution $p_X(\mathbf{x})$ that is similar to the target Boltzmann distribution.

2) To obtain an unbiased sample and to compute Boltzmann-weighted averages, the generated distribution $p_X(\mathbf{x})$ is reweighted to the Boltzmann distribution $e^{-u(\mathbf{x})}$. This can be achieved with various algorithms; here, the simplest one is used: assign the statistical weight $w(\mathbf{x}) = e^{-u(\mathbf{x})}/p_X(\mathbf{x})$ to every sample **x** and then compute desired statistics, such as free-energy differences using this weight.

For both training and reweighting, it is important that we can compute the probability $p_X(\mathbf{x})$ of generating a configuration **x**. This can be achieved when F_{zx} is an invertible transformation, which allows us to transform the known prior distribution $p_Z(\mathbf{z})$ to $p_X(\mathbf{x})$ (Fig. 1A, materials and methods) (*18*, *19*). Physically, invertible transformations are analogous to flows of a fluid that transform the probability density from configuration space to latent space, or vice versa. Volume-preserving transformations, comparable to incompressible fluids, were introduced in (*19*). Here, we use the non–volume-preserving transformations introduced in (*20*) (Fig. 1B), as they allow the probability distribution to be scaled differently at different parts of configuration space. Alternatively, Boltzmann generators can be built using more general invertible transformations (*21*–*23*). Invertibility is achieved by adopting special neural network architectures (Fig. 1B; materials and methods). Multiple trainable invertible "blocks" can be stacked, thus encoding complicated variable transformations in the form of a deep invertible neural network (Fig. 1A).

Boltzmann generators are trained by combining two modes: training by energy and training by example. Training by energy is the main principle behind Boltzmann generators, and proceeds as follows: We sample random vectors **z** from a Gaussian prior distribution, and then transform them through the neural network to proposal configurations, $\mathbf{x} = F_{zx}(\mathbf{z})$. In this way, the Boltzmann generator will generate configurations from a proposal distribution $p_X(\mathbf{x})$, which initially will be very different from the Boltzmann distribution, and will include configurations with very high energies. Next, we compute the difference between the generated distribution $p_X(\mathbf{x})$ from the Boltzmann distribution whose statistical weights $e^{-u(\mathbf{x})}$ are known. For Boltzmann generators, a natural measure of this difference is the relative entropy, or Kullback–Leibler (KL) divergence. The KL divergence can be computed as the following expectation value over samples **z** (materials and methods):

$$J_{KL} = \mathbb{E}_{\mathbf{z}}[u(F_{zx}(\mathbf{z})) - \log R_{zx}(\mathbf{z})] \qquad (1)$$

Here, $u(F_{zx}(\mathbf{z}))$ is the energy of the generated configuration. R_{zx} is the determinant of the Boltzmann generator's Jacobian matrix and measures how much the network scales the configuration space volume at **z**. The invertible network layers are designed such that R_{zx} can be easily computed (materials and methods). We treat J_{KL} as a loss function: To train the Boltzmann generator, we approximate J_{KL} using a batch of ~1000 samples, and then change the neural network parameters so as to decrease J_{KL}. A few hundred or thousand such iterations are required to train the Boltzmann generator for the examples in this study. The resulting few million computations of the potential energy in Eq. 1 are the main computational investment and take between 1 min and few hours for the present systems.

The KL divergence (Eq. 1) is equivalent to the free-energy difference of transforming the

Gaussian prior distribution to the generated distribution (materials and methods, supplementary materials): The first term $\mathbb{E}[u(F_{zx}(\mathbf{z}))]$ is the mean potential energy, i.e., the internal energy of the system. The second term $\mathbb{E}[\log R_{zx}(\mathbf{z})]$ is equal to the entropic contribution to the free energy at the chosen temperature plus a constant factor. The terms in Eq. 1 counterplay in an interesting way: the first term tries to minimize the energy and therefore trains the Boltzmann generator to sample low-energy structures. The second term tries to maximize the entropy of the generated distribution and therefore prevents the Boltzmann generator from the so-called mode collapse (*13*), i.e., the repetitive sampling of a single minimum-energy configuration that would minimize the first term.

Despite the entropy term in Eq. 1, training by energy alone is not sufficient, as it tends to focus sampling on the most stable metastable state (fig. S2). We therefore additionally use training by example, which is the standard training method used in other machine learning applications, and is here implemented with the maximum likelihood (ML) principle. We initialize the Boltzmann generator with some "valid" configurations $\mathbf{x}$, e.g., from short initial MD simulations or experimental structures, and transform them to latent space via $\mathbf{z} = F_{xz}(\mathbf{x})$. Maximizing their likelihood in the Gaussian distribution corresponds to minimizing the loss function (*18*, *19*):

$$J_{ML} = \mathbb{E}_{\mathbf{x}}\left[\frac{1}{2}\|F_{xz}(\mathbf{x})\|^2 - \log R_{xz}(\mathbf{x})\right] \quad (2)$$

Here, the first term $\frac{1}{2}\|F_{xz}(\mathbf{x})\|^2$ is the energy of a harmonic oscillator corresponding to the Gaussian prior distribution. Training by example is especially useful in the early stages of training, as it helps the Boltzmann generator to focus on relevant parts of state space.

By combining training by energy and training by example, we can sample configurations that have high probabilities and low free energies. However, sometimes we want to sample states with low equilibrium probabilities, such as transition states along a certain RC whose free-energy profile is of interest. For this purpose, we introduce an RC loss that can optionally be used to enhance the sampling of a Boltzmann generator along a chosen RC (materials and methods).

Results

Illustration on model systems

We first illustrate Boltzmann generators using two-dimensional model potentials that have metastable states separated by high energy barriers: the double well potential and the Mueller potential (Fig. 2, A and G). MD simulations stay in one metastable state for a long time before a rare transition event occurs. Hence, the distributions in configuration space (x_1, x_2) are split into two modes (Fig. 2, A and G; transition state and intermediate state ensembles are shown in yellow for clarity but are not used for training). We are training Boltzmann generators using the two short and disconnected simulations whose samples are shown in Fig. 2, A and G (details in supplementary materials, convergence in fig. S1). Figure 2, B and H, shows the latent spaces learned by the Boltzmann generator; note that their exact appearance varies between different runs due to stochasticity in neural network training. In both latent spaces, the probability densities of the two states and the transition/intermediate states are "repacked" to form a density concentrated around the origin.

We use the Boltzmann generators by sampling from their latent spaces according to Gaussian distributions. After transforming these variables via F_{zx}, this produces uncorrelated and low-energy samples from both stable states without any sampling problem (Fig. 2, C, D, I, and J). A variety of training methods succeed in sampling across the barrier such that the rare event nature of the system is eliminated (fig. S2). Using a Boltzmann generator trained by energy and by example with simple reweighting reproduces the precise free-energy differences of the two metastable states, although no RC is used to indicate the direction of the rare event (Fig. 2, E and K, green). By additionally training with the RC loss to promote sampling along x_1 (double well) or x_2 (Mueller potential), the low-probability transition states are sampled (Fig. 2, D and J, orange) and the full free-energy profile can be reconstructed with high precision (Fig. 2, E and K, orange).

The Boltzmann generator repacks the high-probability regions of configuration space into a concentrated latent space density. We therefore wondered about the physical interpretation of direct paths in latent space. Specifically, we interpolate linearly between the latent space representations of samples from different energy

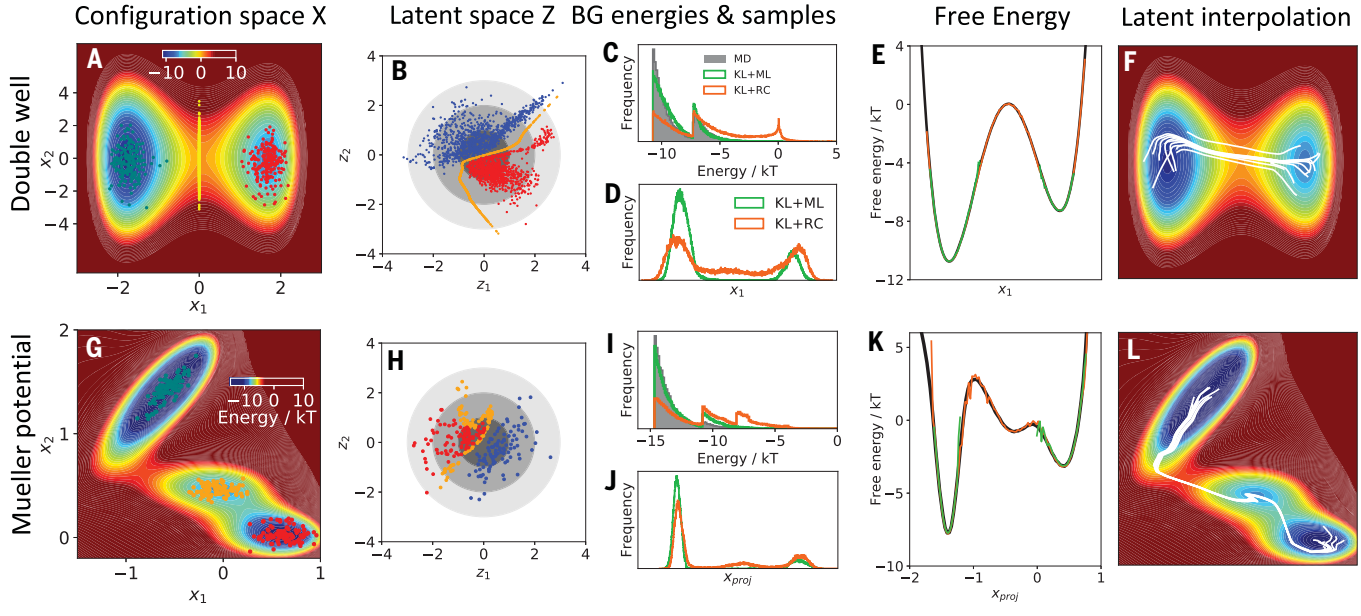


Fig. 2. Application of Boltzmann generators on two-dimensional bistable systems. (**A** and **G**) Two-dimensional potentials: double well (x_1 is the slow coordinate) and Mueller potential. Two short MD simulation trajectories (blue, red) stay in their metastable states without crossing. Transition state and intermediate state ensembles are shown (orange) but were not used for training. (**B** and **H**) Latent-space distribution of trajectories shown in (A) and (G) when mapped through trained F_{xz}. (**C** and **I**) Potential energy distribution sampled by MD simulation (gray) and by Boltzmann generators trained by energy and by example (KL+ML, green) and using RC training (KL+RC, orange). (**D** and **J**) Boltzmann generator sample distribution along the slow coordinates. For the Mueller potential, x_{proj} is defined as projection along the vector (1,–1). (**E** and **K**) Free-energy estimates obtained from Boltzmann generator samples after reweighting. (**F** and **L**) Paths generated by linear interpolation in Boltzmann generator latent space (B and H) between random pairs of "blue" and "red" MD samples.

minima, similar to what is done with generative networks in other disciplines (*22*, *17*). When mapping these linear interpolations back to configuration space, they result in nonlinear pathways that have low energies and high probabilities (Fig. 2, F and L). Although there is no general guarantee that linear paths in latent space will result in low energies, this result indicates that the latent spaces learned by Boltzmann generators can be used to provide candidates of order parameters for bias-enhanced or path-based sampling methods (*1*, *3*, *24*).

For the double-well system, the unbiased MD simulation needs, on average, 4×10^6 MD steps for a single return trip between the two states (supplementary materials), and ~100 such crossings are required to compute the free-energy difference with the same precision as the Boltzmann generator results shown in Fig. 2E. The total effort of training the Boltzmann generator (including generating the initial simulation data) corresponds to ~10^6 steps, but once this is done, statistically independent samples can be generated at no significant cost. For this simple system, the Boltzmann generator is therefore about a factor of 100 more efficient than direct simulation, but much more extreme savings can be observed for complex systems, as shown below.

Thermodynamics of condensed-matter systems

As a second example, we demonstrate that Boltzmann generators can sample high-probability structures and efficiently compute the thermodynamics in crowded condensed-matter systems. We simulated a dense system of two-dimensional particles confined to a box, as suggested in (*25*) (Fig. 3A). Immersed in a fluid is a bistable particle dimer whose open and closed states are separated by a high barrier (Fig. 3B). Opening or closing the dimer directly is not possible due to the high density of the system but requires concerted rearrangement of solvent particles. At close distances, particles repel each other strongly, resulting in a crowded system. Thus, the fraction of low-energy configurations is vanishingly small, and manually designing a sampling method that simultaneously places all 38 particles and achieves low energies appears unfeasible.

We trained a Boltzmann generator to sample one-shot low-energy configurations and used it to compute the free-energy profiles of opening or closing the dimer. Key to treating explicit-solvent systems such as this one is to incorporate the indistinguishability of solvent molecules. If physically identical solvent molecules were to be distinguished, then every exchange of solvent molecule positions due to diffusion would represent a new configuration, resulting in an enormous configuration space even for this 38-particle system. We therefore removed identical-particle permutations from all configurations input into or sampled by the Boltzmann generator by exchanging particle labels to minimize the distance to a reference configuration (supplementary materials).

The training is initialized with examples from separate, disconnected simulations of the open and closed states, but in later stages, training by energy Eq. 1 dominates (supplementary materials, fig. S1, and table S1). The trained Boltzmann generator has learned a transformation of the complex configuration space density to a concentrated, 76-dimensional ball in latent space (Fig. 3C). Indeed, direct sampling from a 76-dimensional Gaussian in latent space and transformation via F_{zx} generates configurations in which all particles are placed without significant clashes and potential energies that overlap with the energy distribution of the unbiased MD trajectories (Fig. 3D). Also, realistic transition states that have not been included in any training data are sampled (Fig. 3D, middle).

To demonstrate the computation of thermodynamic quantities, we perform training by energy (Eq. 1) simultaneously to a range of temperatures (supplementary materials). Although the temperature changes the configuration space distribution in a complex way, it can be modeled as a simple scaling factor in the width of the Gaussian prior distribution in latent space (materials and methods). Then, we sample the Boltzmann generator for a range of temperatures and use simple reweighting to compute the free energies along the dimer distances. As shown in Fig. 3E, these temperature-dependent free energies agree

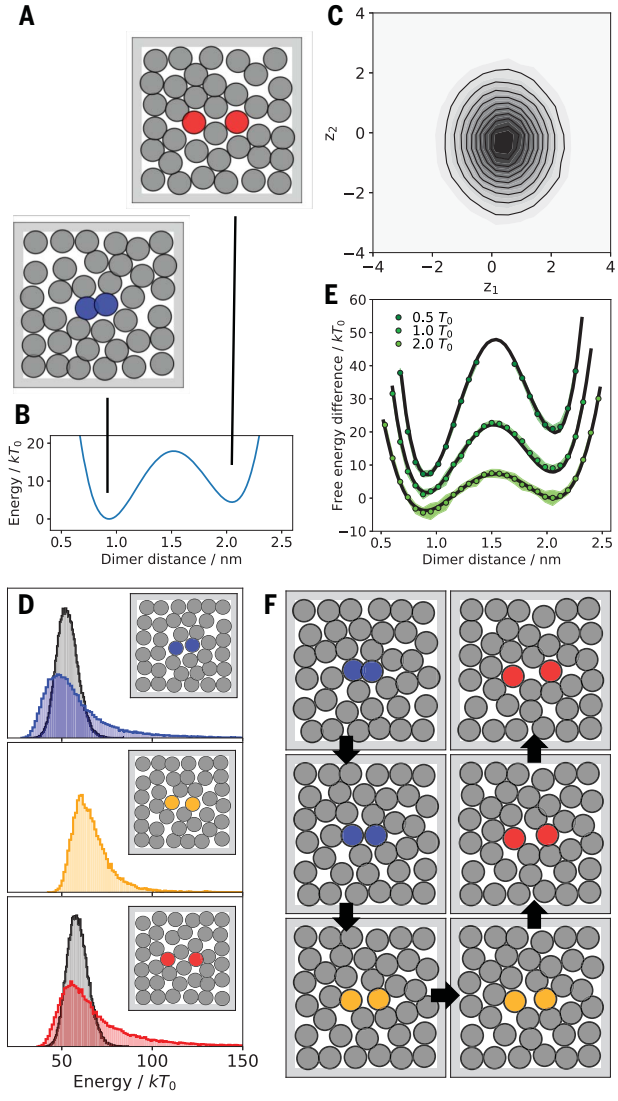


Fig. 3. Repulsive particle system with bistable dimer. (**A**) Closed (blue) and open (red) configurations from MD simulations (input data). (**B**) Bistable dimer potential. (**C**) Distribution of MD simulation data in latent space coordinates z_1, z_2 after training the Boltzmann generator. (**D**) Potential energy distribution from MD (gray) and Boltzmann generator for closed (blue), open (red), and transition configurations (yellow). Insets show one-shot Boltzmann generator samples. (**E**) Free-energy differences as a function of dimer distance and relative temperature sampled with Boltzmann generators (generation and reweighting, green bullets with intervals indicating one standard error from 10 independent repeats) and umbrella sampling (black lines). (**F**) Linear latent space interpolation between the closed and open structures shown in the top row.

precisely with extensive umbrella-sampling simulations that use bias potentials along the dimer distance ((*1*), supplementary materials).

We estimate that the MD simulation needs at least 10^{12} steps to spontaneously sample a single transition from closed to open state and back (supplementary materials), and ~100 such transitions would be needed to compute free-energy differences with the precision of Boltzmann Generators shown in Fig. 3E. The total effort to train the Boltzmann generator is ~3×10^7 energy evaluations, but then statistically independent samples can be drawn in one shot at the entire temperature range trained, resulting in at least seven orders of magnitude speed-up compared with MD.

As above, we perform linear interpolations between the latent space representations of open- and closed-dimer samples. A significant fraction of all pair interpolations results in low-energy pathways. The lowest-energy interpolation of 100 randomly selected pairs of end states is shown in Fig. 3F, representing a physically meaningful rearrangement of the dimer and solvent.

Exploring configuration space

In the previous examples, Boltzmann generators were used to sample known regions of configuration space and compute statistics thereof. Here, we demonstrate that Boltzmann generators can help to explore configuration space. The basic idea is as follows: we construct an exploratory sampling method using an established sampling algorithm in latent space while simultaneously training the Boltzmann generator transformation using the configurations found so far.

We initialize the method with a (possibly small) set of configurations X. Training is done here by minimizing the symmetric loss function $J = J_{KL} + J_{ML}$ (Eqs. 1 and 2). The likelihood loss function J_{ML} is initially biased by the input data, but as X approaches an unbiased Boltzmann sample, the symmetric loss converges to a meaningful distance of probabilities (materials and methods). As an example, here, we use Metropolis Monte Carlo in the Boltzmann generator latent space to update X (materials and methods). The step size is chosen adaptively but reaches the order of the latent space distribution width. Thus, large-scale configuration transitions in physical space can be overcome in a single Monte Carlo step.

We now revisit the three previous examples and initialize X with only a single input configuration from the most stable state (Fig. 4, A, D, and G). The exploration method quickly fills the local metastable states and finds new metastable states within a few 10^5 energy calls, i.e., orders of magnitude faster than direct MD (Fig. 4, B, E, and H). This demonstrates that Boltzmann generators sample new, previously unseen states with a significant probability, and that this ability can be turned into exploring configuration space when past samples are stored and reused for training.

The Metropolis Monte Carlo method causes the sample to converge toward the Boltzmann distribution. However, we do not need to wait for this method to be converged because, with sufficient samples in the states of interest, the equilibrium free energies can be computed by reweighting as in Figs. 2 and 3 above. Although new states are found, the data-based loss J_{ML} may increase and decrease again while the Boltzmann generator transformation is updated to include these new states (Fig. 4, C, F, and I, top row). During training, the energy-based loss J_{KL} decreases steadily until the full Boltzmann distribution is sampled (Fig. 4, C, F, and I, middle row). We also observe that the Metropolis Monte Carlo efficiency, defined by the product of step length and acceptance rate, tends to increase over time, although it may decrease temporarily when more states are found (Fig. 4, C, F, and I, bottom row).

Due to the invertible transformation between latent and configuration space, any sampling method that involves reweighting or Monte Carlo acceptance steps can be reformulated in Boltzmann generator latent space and potentially yield enhanced performance.

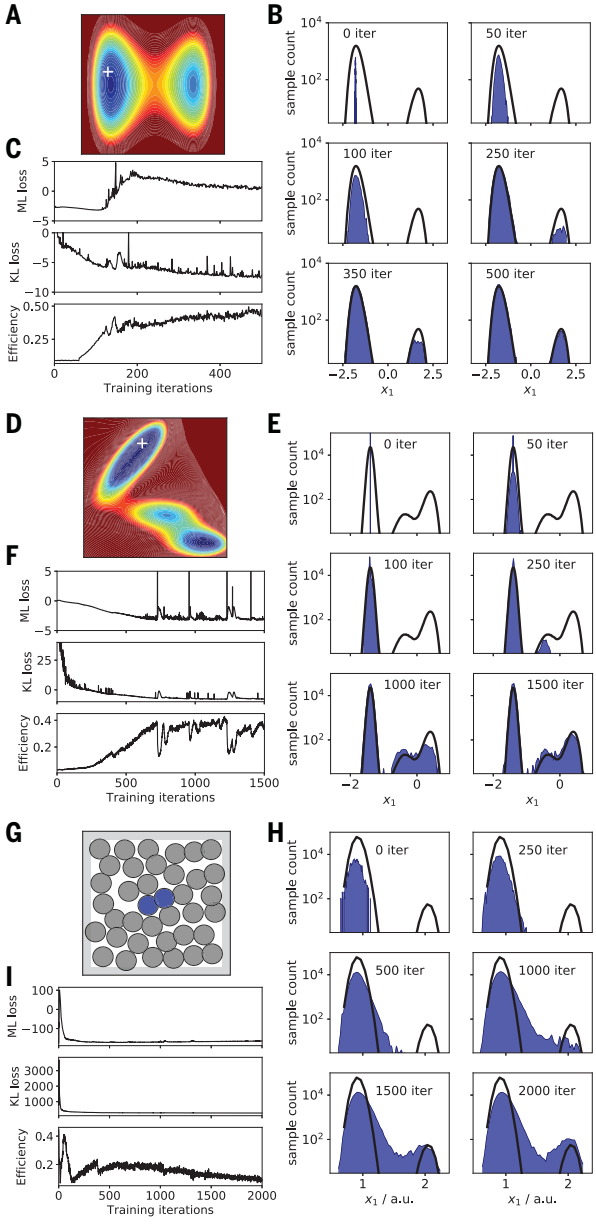


Fig. 4. Exploration with Boltzmann generators from a single snapshot. (**A** to **C**) Double-well potential, (**D** to **F**) Mueller potential, and (**G** and **H**) solvated particle dimer. (A, D, and G) Starting configuration. (C, F, and I) Convergence of the loss terms (J_{ML} and J_{KL}) and the MCMC efficiency (product of step length and acceptance rate). (B, E, and H) Evolution of sample distribution over MCMC iteration. As soon as sufficient density is available in the states of interest, these distributions can be reweighted to equilibrium as in Figs. 2 and 3. iter, iterations; acc, acceptance.

Complex molecules

We demonstrate that Boltzmann generators can generate equilibrium all-atom structures of macromolecules in one shot using the BPTI in an implicit solvent model (Fig. 5 and supplementary materials). To train Boltzmann generators for complex molecular models, we wrapped the energy and force computation functions of the OpenMM simulation software (*26*) in the standard deep learning library Tensorflow (*27*).

Training a Boltzmann generator directly on the Cartesian coordinates resulted in large energies and unrealistic structures with distorted bond lengths and angles. This problem was solved by incorporating the following coordinate transformation in the first layer of the Boltzmann generator that is invertible up to rotation and translation of the molecule (Fig. 5A and materials and methods): the coordinates are split into Cartesian and internal coordinate sets. The Cartesian coordinates include heavy atoms of the backbone and the disulfide bridges. Cartesian coordinates are whitened, i.e., decorrelated and normalized, using a principal component analysis (PCA) of the input data. During whitening, the six degrees of freedom corresponding to global translation and rotation of the molecule are discarded. The remaining side-chain atoms are measured in internal coordinates (bond lengths, angles, and torsions with respect to parent atoms) and subsequently normalized.

After this coordinate transformation, the learning problem is substantially simplified as the transformed input data are already nearly Gaussian distributed. We first demonstrate that Boltzmann generators can learn to sample heterogeneous equilibrium structures when trained with examples from different configurations. To this end, we generated six short simulations of 20 ns each, starting from snapshots of the well-known 1-ms simulation of BPTI produced on the Anton supercomputer (*28*). In the more common situation that no ultralong MD trajectory is available, the Boltzmann generator can be seeded with different crystallographic structures or homology models. To promote simultaneous sampling of high- and low-probability configurations, we included an RC loss using two slow collective coordinates that have been used earlier in the analysis of this BPTI simulation (supplementary materials) (*29*, *30*). Boltzmann generators of BPTIs that do not use RCs are discussed in the subsequent section.

A trained Boltzmann generator with eight invertible blocks can sample all 892 atom positions (2676 dimensions) in one shot and produce locally and globally valid structures (Fig. 5C). The potential energies of samples exhibit significant overlap with the potential energy distribution of MD simulations (Fig. 5D), and thus samples can be reweighted for free-energy calculations. The probability distributions of most bond lengths and valence angles are almost indistinguishable from the distributions of the equilibrium MD simulations (Fig. 5E). The only exception is that the distributions of valence angles involving sulfur atoms are slightly narrower.

The trained Boltzmann generator learns to encode and sample significantly different structures (Fig. 5F). In particular, it generates independent one-shot samples of the near-crystallographic structure "X" (1.4-Å mean backbone RMSD to crystal structure), and the open "O" structure, which involves significant changes in flexible loops and repacking of side chains (Fig. 5. G and H). The X→O transition has been sampled only once in the millisecond Anton trajectory, which is consistent with the observation of a millisecond-timescale "major–minor" state transition observed in nuclear magnetic resonance spectroscopy (*31*). We note that X↔O transition states are not included in the Boltzmann generator training data.

Sampling such a transition and collecting statistics for it is challenging for any existing simulation method. Brute-force MD simulations would require several milliseconds of simulation data. To use enhanced sampling, an order parameter or RC able to drive the complex rearrangement shown in Fig. 5, G and H, would need to be found, but because BPTI has multiple states with lifetimes on the order of ~10 to 100 μs (*28*, *30*), the simulation time required for convergence would still be extensive. The computation of free-energy differences using Boltzmann generators will be discussed in the next section.

Thermodynamics between disconnected states

We develop a reaction-coordinate–free approach to compute free-energy differences from disconnected MD or MCMC simulations in separated states, such as two conformations of a protein. As demonstrated above, this can be achieved by a single Boltzmann generator that simultaneously captures multiple metastable states and maps them to the same latent space Z, where they are connected via the Gaussian prior distribution (Figs. 2, B and H, and 3C). However, a more direct statistical mechanics idea that has been successfully applied to certain simple liquids and solids is to compute free-energy differences by relating to a tractable reference state, e.g., ideal gas or crystal (*32*–*34*). Here, we show that Boltzmann generators can turn this idea into a general method applicable to complex many-body systems.

Recall that the value of the energy-loss function J_{KL} (Eq. 1) estimates the free-energy difference of transforming the Gaussian prior distribution to the generated distribution in configuration space. If we are now given MD data sampled in two or more disconnected states, then we can train independent Boltzmann generators for each of them. The goal here is not to explore configuration space, so training by energy is combined with training by example (Eq. 2) to restrain the generated distribution around the separate states. For each Boltzmann generator, the transformation free energy is computed, e.g., $\langle J^1_{KL} \rangle$ and $\langle J^2_{KL} \rangle$, by sampling from the Gaussian prior distributions and inserting into Eq. 1. The free-energy difference between the two states is directly given as a difference between these two values, $\Delta A_{12} = \langle J^2_{KL} \rangle - \langle J^1_{KL} \rangle$ (Fig. 6A).

We illustrate our method by computing temperature-dependent free-energy differences for the four systems discussed above, each using two completely disconnected MD simulations as input. Because the estimate of the free-energy difference is readily available from the value of the loss function, it can be conveniently tracked for convergence while the Boltzmann generators are trained (Fig. 6B and fig. S3).

For the two-dimensional systems (double well, Mueller potential), exact reference values for the free-energy differences can be computed, and the Boltzmann generator method recovers them accurately with a small statistical uncertainty over the entire temperature range (Fig. 6, C and D) using 10-fold less simulation data and an ~10-fold shorter training time than for the estimates using a single joint Boltzmann generator reported in Fig. 2 (supplementary materials).

For the solvated bistable dimer, we use simulations that are 10-fold shorter than for the single joint Boltzmann generator reported in Fig. 3 and train two independent Boltzmann generators at multiple temperatures. As a reference, three independent umbrella-sampling simulations were conducted at each of five different temperatures. Both predictions of the free-energy difference between open and closed dimer states are consistent and have overall similar uncertainties (Fig. 6E and fig. S3B; note that the uncertainty of umbrella sampling is strongly temperature dependent). Although umbrella sampling is well-suited for this system with a clear RC, the two–Boltzmann-generator method required 50 times fewer energy calls than the umbrella-sampling simulations at five temperatures and yet makes predictions across the full temperature range (supplementary materials).

Finally, we used the same method to predict the temperature-dependent free-energy difference of the "X" and "O" states in the BPTI protein. Sampling this millisecond transition 10 times by brute-force MD would take ~30 years on one of the GTX1080 graphics cards that are used for computations in this paper and would only give us the free-energy difference at a single temperature. Although speeding up this complex transition with enhanced sampling may be possible, engineering a suitable order parameter is a time-consuming trial-and-error task. Training two Boltzmann generators does not require any RC to be defined.

Here, we used the two–Boltzmann-generator method starting from two simulations of 20 ns each, which were started from selected frames of the 1-ms trajectory, but could generally be started from crystallographic structures or homology models. Conducting the MD simulations, training and analyzing the Boltzmann generators used a total of $<3 \times 10^7$ energy calls, which results in a converged free-energy estimate within a total of about 10 hours on a GTX1080 graphics card, i.e., about five orders of magnitude faster than the brute-force approach (Fig. 6F and fig. S3C).

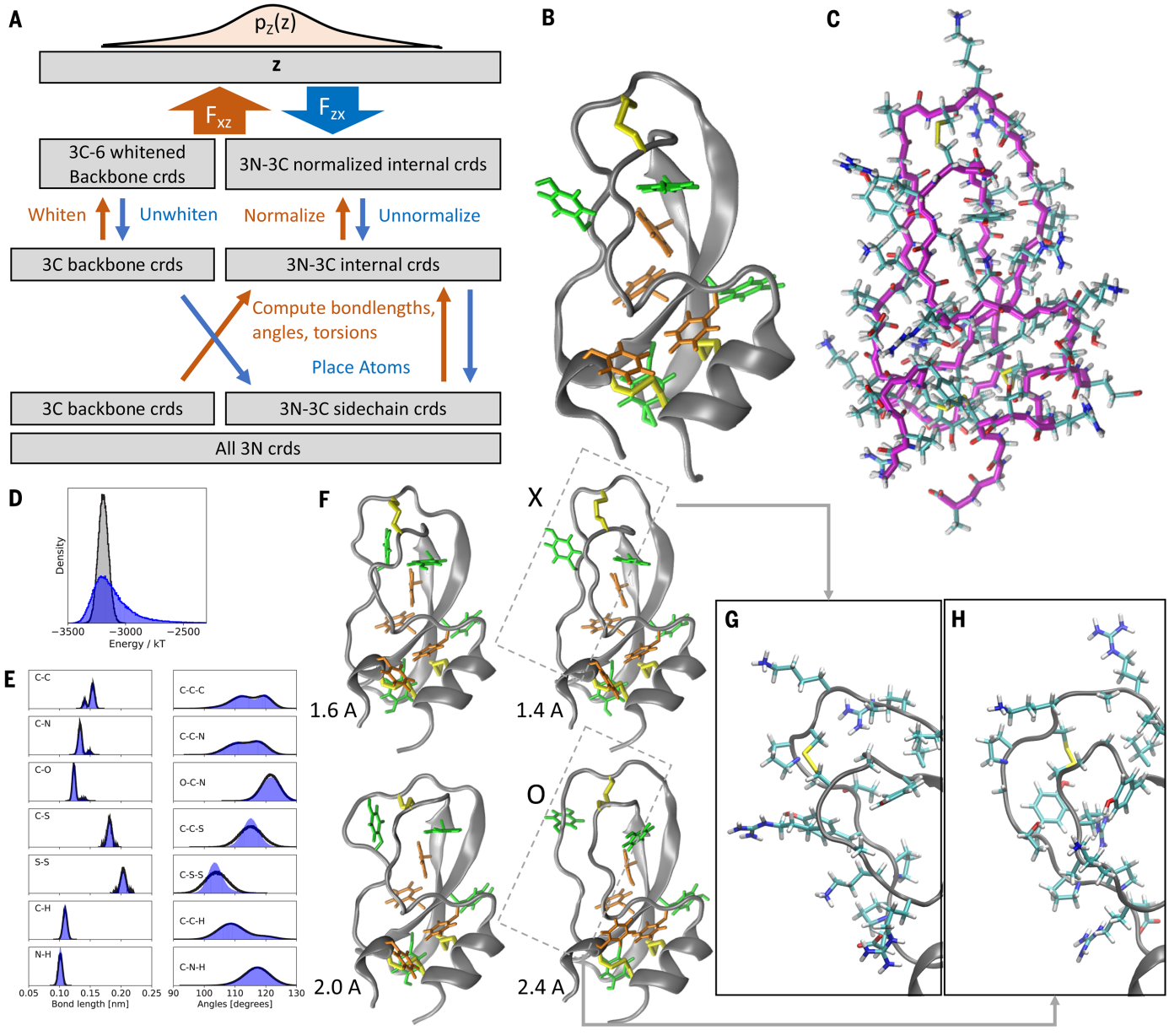


Fig. 5. One-shot sampling of all-atom structures in different conformations of the BPTI protein. (**A**) Boltzmann generator for macromolecules: Backbone atoms are whitened using PCA; side-chain atoms are described in normalized internal coordinates (crds). (**B**) BPTI x-ray crystal structure (PDB: 5PTI). Cysteine disulfide bridges and aromatic residues are shown for orientation. (**C**) One-shot Boltzmann generator sample of all 892 atoms (2670 dimensions) of the BPTI protein similar to the x-ray structure. (**D**) Potential energy distribution from MD simulation (gray) and Boltzmann generator one-shot samples (blue). (**E**) Distribution of bonds and angles compared between MD simulation (black) and Boltzmann generator (blue). (**F**) Representative snapshots of four clusters of structures generated with the Boltzmann generator. Backbone root mean square deviation from the x-ray structure is given below the structure (in angstroms). Marked are the x-ray–like structure "X" and the open structure "O." (**G** and **H**) Magnification of the most variable parts of the Boltzmann-generated samples from the "X" and "O" states. Side chains are shown in atomistic resolution.

Although no reference for this free-energy difference in the given simulation model is known, the temperature profile admits basic consistency checks: The x-ray structure is identified as the most stable structure at temperatures below 330 K. The internal energy and entropy terms of the free-energy difference (Eq. 1) are both positive across all temperatures. Therefore, the free-energy decreases at high temperatures as the entropic stabilization becomes stronger. A higher configurational entropy of the "O" state is consistent with its more open loop structure (compare Fig. 5, G and H) and the higher degree of fluctuations in the "O" state observed by the analysis in (*30*).

Discussion and conclusion

Boltzmann generators can overcome rare event-sampling problems in many-body systems by generating independent samples from different metastable states in one shot. We have demonstrated this for dense and unstructured many-body systems with up to 892 atoms (over 2600 dimensions) that are placed simultaneously, with most samples having globally and locally valid structures and potential energies in the range of the equilibrium distribution. In contrast to other generative neural networks, Boltzmann generators

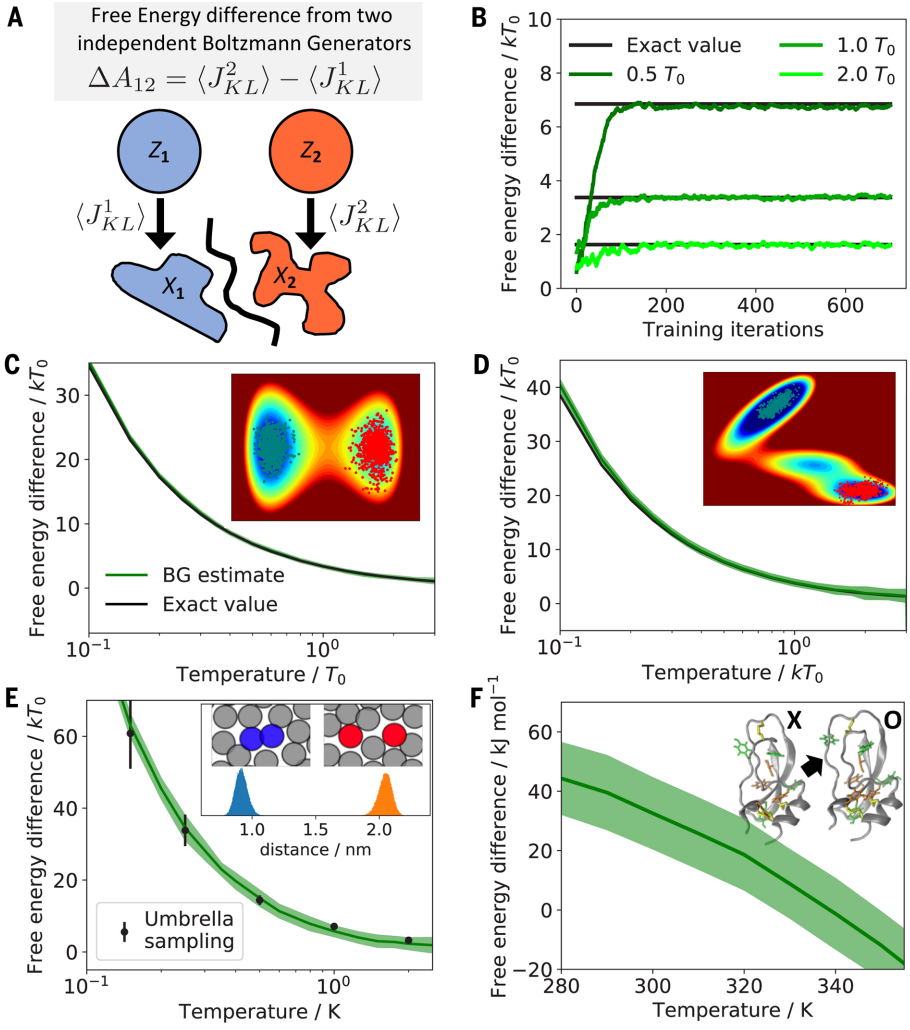


Fig. 6. Thermodynamics between disconnected states by coupling multiple Boltzmann generators. (**A**) Using multiple Boltzmann generators, we can compute free-energy differences between states without requiring RCs using only disconnected MD simulations in each of them. This is possible because each Boltzmann generator estimates the free-energy difference to a common reference state. (**B**) Example for tracking convergence: estimate of free-energy difference for the double well potential (multiple temperatures) as a function of the training iterations of two Boltzmann generators. Convergence plots for the other systems are shown in fig. S3. Results show estimates from two Boltzmann generators with mean and one standard error computed from bootstrapping the converged segment of the free-energy estimate. (**C**) Left-to-right transition in the double well, (**D**) left-to-right transition in the Mueller potential, (**E**) closed-to-open transition in the solvated bistable particle dimer, (**F**) X→O transition in an atomistic model of BPTI.

produce unbiased samples, as the generated probability density is known at each sample point and can be reweighted to the target Boltzmann distribution. This feature directly translates into being able to compute free-energy differences between different metastable states.

In contrast to enhanced sampling methods that directly operate in configuration space, such as umbrella sampling or metadynamics, Boltzmann generators can sample between metastable states without any predefined RC connecting them. This is achieved by learning a coordinate transformation in which different metastable states become neighbors in the transformed space where the sampling occurs. If suitable RCs are known, then these can be incorporated into the training to sample continuous pathways between states, e.g., to compute a continuous free-energy profile along a reaction.

As in many areas of machine learning, the key to success is to choose a representation of the input data that supports the learning problem. For macromolecules, we have found that a successful representation is to describe its backbone in Cartesian coordinates and all atoms that branch off from the backbone in internal coordinates. Additionally, normalizing these coordinates to mean 0 and variance 1 already makes their probability distribution close to a Gaussian normal distribution, thus simplifying the learning problem considerably.

We have shown, in principle, how explicit solvent systems can be treated. For this, it is essential to build the physical invariances into the learning problem. Specifically, we need to account for permutation invariance: when two equivalent solvent molecules exchange positions, the potential energy of the system is unchanged and so is the Boltzmann probability.

We have demonstrated scaling of Boltzmann generators to 1000's of dimensions. Generative networks in other fields have been able to generate photorealistic images with 10^6 dimensions in one shot (*15*). However, for the present application, the statistical efficiency, i.e., the usefulness of these samples to compute equilibrium free energies, will decline with increasing dimension. Scaling to systems with 100,000's of dimensions or more, such as solvated atomistic models of large proteins, can be achieved in different ways. Sampling of the full atomistic system may be addressed with a divide-and-conquer approach: In each iteration of such an approach, one would resample the positions of a cluster of atoms using the sum of potential energies between cluster atoms and all system atoms and then, e.g., perform Monte Carlo steps using these cluster proposals. Alternatively, Boltzmann generators could be used to sample lower-dimensional free-energy surfaces learned from all-atom models (*35*, *36*).

A caveat of Boltzmann generators is that, depending on the training method, they may not be ergodic, i.e., they may not be able to reach all configurations. Here, we have proposed a training method that promotes state space exploration by performing Monte Carlo steps in the Boltzmann generator's latent space while training the network. This may be viewed as a general recipe: The whole plethora of existing sampling algorithms, such as umbrella sampling, metadynamics, and replica exchange, can be reformulated in the latent space of a Boltzmann generator, potentially leading to dramatic performance gains. Any such approach can always be combined with MD or MCMC moves in configuration space to ensure ergodicity.

Finally, the Boltzmann generators described here learned a system-specific coordinate transformation. The approach would become much more general and efficient if Boltzmann generators could be pretrained on certain building blocks of a molecular system, such as oligopeptides in solvent or a protein, and then reused on a complex system consisting of these building blocks. A promising approach is to involve transferrable featurization methods developed in the context of machine learning for quantum mechanics (*37*–*40*).

In summary, Boltzmann generators represent a powerful approach to addressing the long-standing rare-event sampling problem in many-body

systems and open the door for new developments in statistical mechanics.

Materials and methods

Invertible networks

We used invertible networks with trainable parameters θ to learn the transformation between the Gaussian random variables $\mathbf{z}$ and the Boltzmann-distributed random variables $\mathbf{x}$:

$$\mathbf{z} = F_{xz}(\mathbf{x};\theta)$$

$$\mathbf{x} = F_{zx}(\mathbf{z};\theta).$$

Hence $F_{xz} = F_{zx}^{-1}$. Each transformation has a Jacobian matrix with the pairwise first derivatives of outputs with respect to inputs:

$$\mathbf{J}_{zx}(\mathbf{z};\theta) = \left[\frac{\partial F_{zx}(\mathbf{z};\theta)}{\partial z_1}, ..., \frac{\partial F_{zx}(\mathbf{z};\theta)}{\partial z_n}\right]$$

$$\mathbf{J}_{xz}(\mathbf{x};\theta) = \left[\frac{\partial F_{xz}(\mathbf{x};\theta)}{\partial x_1}, ..., \frac{\partial F_{xz}(\mathbf{x};\theta)}{\partial x_n}\right]$$

The absolute value of the Jacobian's determinant, $|\det \mathbf{J}_{zx}(\mathbf{z};\theta)|$, measures how much a volume element at z is scaled by the transformation. Below, we will omit the symbol θ and use the abbreviations:

$$R_{xz}(\mathbf{x}) = |\det \mathbf{J}_{xz}(\mathbf{x})|$$

$$R_{zx}(\mathbf{z}) = |\det \mathbf{J}_{zx}(\mathbf{z})|$$

We use invertible transformations because they allow us to transform random variables as follows:

$$\begin{aligned} p_X(\mathbf{x}) &= p_Z(\mathbf{z}) R_{zx}(\mathbf{z})^{-1} \\ &= p_Z\Big(F_{xz}(\mathbf{x})\Big) R_{xz}(\mathbf{x}) \end{aligned} \tag{3}$$

$$\begin{aligned} p_Z(\mathbf{z}) &= p_X(\mathbf{x}) R_{xz}(\mathbf{x})^{-1} \\ &= p_X\Big(F_{zx}(\mathbf{z})\Big) R_{zx}(\mathbf{z}) \end{aligned} \tag{4}$$

Trainable invertible layers

We use the RealNVP transformation as trainable part of invertible networks (*20*). The main idea is to split the variables into two channels, $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$, $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)$, and do only trivially invertible operations on each channel, such as multiplication and addition. Additionally, we use arbitrary, noninvertible artificial neural networks S and T that respectively scale and translate the second input channel $\mathbf{x}_2$ using a nonlinear transformation of the first input channel $\mathbf{x}_1$.

$$\begin{aligned} &f_{xz}(\mathbf{x}_1, \mathbf{x}_2) \\ &: \begin{pmatrix} \mathbf{z}_1 = \mathbf{x}_1 \\ \mathbf{z}_2 = \mathbf{x}_2 \odot \exp\Big(S(\mathbf{x}_1;\theta)\Big) + T(\mathbf{x}_1;\theta) \end{pmatrix} \end{aligned} \tag{5}$$

$$\log R_{xz} = \sum_i S_i(\mathbf{x}_1;\theta) \tag{6}$$

$$\begin{aligned} &f_{zx}(\mathbf{z}_1, \mathbf{z}_2) \\ &: \begin{pmatrix} \mathbf{x}_1 = \mathbf{z}_1 \\ \mathbf{x}_2 = \Big(\mathbf{z}_2 - T(\mathbf{x}_1;\theta)\Big) \odot \exp\Big(-S(\mathbf{z}_1;\theta)\Big) \end{pmatrix} \end{aligned} \tag{7}$$

$$\log R_{zx} = -\sum_i S_i(\mathbf{z}_1;\theta) \tag{8}$$

A RealNVP "block" is defined by two stacked RealNVP layers with channels swapped, such that both channels are transformed:

$$(\mathbf{y}_1, \mathbf{y}_2) = f_{xy}(\mathbf{x}_1, \mathbf{x}_2)$$

$$(\mathbf{z}_1, \mathbf{z}_2) = f_{yz}(\mathbf{y}_2, \mathbf{y}_1)$$

Boltzmann generators are built by putting the forward and the inverse of such blocks in parallel that share the same nonlinear transformations T and S and parameters (Fig. 1B).

PCA whitening layer

We define a fixed-parameter layer "W" to transform the input coordinates into whitened principal coordinates. For systems with rototranslationally invariant energy, we first remove global translation and rotation by superimposing each configuration to a reference configuration. We then perform PCA on the N input coordinates X by solving the eigenvalue problem:

$$\frac{1}{N} X^T X \mathbf{R} = \mathbf{R}\Lambda$$

where $\mathbf{R} = [\mathbf{r}_1, ..., \mathbf{r}_N]$ are principal components vectors and $\Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_d)$ their variances. For systems with rototranslationally invariant energy, the six smallest eigenvalues are 0 and are discarded along with the corresponding eigenvectors. The whitening transformation and its inverse are defined by:

$$W(\mathbf{x}) : \mathbf{z} = \Lambda^{-\frac{1}{2}} \mathbf{R}^\top \mathbf{x}$$

$$W^{-1}(\mathbf{z}) : \mathbf{x} = \mathbf{R}\Lambda^{\frac{1}{2}} \mathbf{z}$$

Note that when translation and rotation are removed in the transformation, this layer is only invertible for $\mathbf{x}$ where translation and rotation are removed as well. However, the network is always invertible for the relevant sequence $\mathbf{z} \to \mathbf{x} \to \mathbf{z}$. The Jacobians of W are:

$$\log R_{xz} = -\frac{1}{2}\sum_i \log \lambda_i$$

$$\log R_{zx} = \frac{1}{2}\sum_i \log \lambda_i$$

Mixed coordinate transformation layer

To treat macromolecules, we defined a new transformation layer "M" that transforms into mixed whitened Cartesian and normalized internal coordinates. We first split the coordinates into a Cartesian and an internal coordinate set, $\mathbf{x} \to [\mathbf{x}_C, \mathbf{x}_I]$. $\mathbf{x}_C$ is whitened (see above), $\mathbf{x}_I$ is transformed into internal coordinates (ICs). For every particle i in $\mathbf{x}_I$, we define three "parent" particles, j, k, and l, and the Cartesian coordinates of particles i, j, k, and l are converted into distance, angle and dihedral $(d_{ij}, \alpha_{ijk}, \phi_{ijkl})$. Finally, each IC is normalized by subtracting the mean and dividing by the standard deviation of the corresponding coordinates in the input data (Fig. 5A). PCA whitening and IC normalization are essential for training Boltzmann generators for complex molecules, as this sets large fluctuations of the whole molecule on the same scale as small vibrations of stiff coordinates such as bond lengths. We briefly call the transformation to normalized internal coordinates $I(\mathbf{x})$.

The inverse transformation is straightforward: The Cartesian set is first restored by applying W^{-1}. Then, the particles in the internal coordinate unnormalized and then placed in a valid sequence, i.e., first particles I, whose parent particles are all in the Cartesian set, then particles with parents that have just been placed, etc. As for the W layer, the M layer is invertible up to global translation and rotation of the molecule that may have been removed during whitening. Additionally, we prevent noninvertibility in dihedral space by avoiding the generation of angle values outside the range $[-\pi, \pi]$ (supplementary materials).

The Jacobians of the M layer are computed using Tensorflow's automatic differentiation methods.

Training and using Boltzmann generators

The Boltzmann generator is trained by minimizing a loss functional of the following form:

$$J = w_{ML} J_{ML} + w_{KL} J_{KL} + w_{RC} J_{RC}. \tag{9}$$

where the terms represent ML ("training by example"), KL ("training by energy"), and RC optimization and the w's control their weights. Below, we will derive these terms in detail.

We call the "exact" distributions μ and the generated distributions q. In particular, $\mu_Z(\mathbf{z})$ is the Gaussian prior distribution from which we sample latent space variables and $q_X(\mathbf{x})$ is the distribution that results from the network transformation F_{zx}. Likewise, $\mu_X(\mathbf{x}) \propto \exp(-u(\mathbf{x}))$ is the Boltzmann distribution in configuration space and $q_Z(\mathbf{z})$ is the distribution that results from the network transformation F_{xz}:

$$\mu_Z(\mathbf{z}) \xrightarrow{F_{zx}} q_X(\mathbf{x})$$

$$\mu_X(\mathbf{x}) \xrightarrow{F_{xz}} q_Z(\mathbf{z})$$

A special case is to use Boltzmann generators to sample from the Boltzmann distribution of the canonical ensemble. Other ensembles can be modeled by incorporating the choice of ensemble into the reduced potential (*41*). The Boltzmann distribution has the form:

$$\mu_X(\mathbf{x}) = Z_X^{-1} \mathrm{e}^{-\beta U(\mathbf{x})} \tag{10}$$

where $\beta^{-1} = k_B T$ with Boltzmann constant k_B and temperature T. When we only have one temperature, we define the reduced energy:

$$u(\mathbf{x}) = \frac{U(\mathbf{x})}{k_B T}$$

To work with multiple temperatures $(T^1, \ldots, T^K)$, we define a reference temperature T^0 and reduced energy $u^0(\mathbf{x}) = U(\mathbf{x})/k_B T^0$. The reduced energies are then obtained by scaling with the relative temperature $\tau_k = T^k/T^0$:

$$u^k(\mathbf{x}) = \frac{T^0}{T^k} u^0(\mathbf{x}) = \frac{u^0(\mathbf{x})}{\tau_k}.$$

To generate the prior distribution, we sample the input in $\mathbf{z}$ from the isotropic Gaussian distribution:

$$\mu_Z^k(\mathbf{z}) = \mathcal{N}(0, \sigma_k^2 \mathbf{I}) = Z_Z^{-1} \mathrm{e}^{-\frac{1}{2}\mathbf{z}^2/\sigma_k^2}, \qquad (11)$$

with normalization constant Z_Z. This corresponds to the prior energy of a harmonic oscillator:

$$u_Z^k(\mathbf{z}) = -\log \mu_Z^k(\mathbf{z}) = \frac{1}{2\sigma_k^2}\mathbf{z}^2 + \text{const.} \qquad (12)$$

Thus, the variance takes the same role as the relative temperature. We (arbitrarily) choose variance 1 for the standard temperature and obtain:

$$\sigma_k^2 = \tau_k.$$

Latent KL divergence

The KL divergence measures the difference between two distributions q and p:

$$\mathrm{KL}(q\|p) = \int q(\mathbf{x})[\log q(\mathbf{x}) - \log p(\mathbf{x})]\mathrm{d}\mathbf{x},$$

$$= -H_q - \int q(\mathbf{x}) \log p(\mathbf{x}) \mathrm{d}\mathbf{x},$$

where H_q is the entropy of distribution q. Here, we minimize the difference between the probability densities predicted by the Boltzmann generator and the respective reference distribution. Using Eqs. 3, 4, and 10, the KL divergence in latent space is:

$$\begin{aligned}\mathrm{KL}_\theta[\mu_Z\|q_Z] &= -H_Z - \int \mu_Z(\mathbf{z}) \log q_Z(\mathbf{z};\theta)\mathrm{d}\mathbf{z}, \\ &= -H_Z - \int \mu_Z(\mathbf{z}) \Big[\log \mu_X\Big(F_{zx}(\mathbf{z};\theta)\Big) \\ &\qquad + \log R_{zx}(\mathbf{z};\theta)\Big]\mathrm{d}\mathbf{z}, \\ &= -H_Z + \log Z_X + \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}\Big[u\Big(F_{zx}(\mathbf{z};\theta)\Big) \\ &\qquad - \log R_{zx}(\mathbf{z};\theta)\Big]\end{aligned}$$

Here, θ are the trainable neural network parameters. Because H_Z and Z_X are constants in θ, the KL loss is given by:

$$J_{KL} = \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}\Big[u\Big(F_{zx}(\mathbf{z};\theta)\Big) - \log R_{zx}(\mathbf{z};\theta)\Big] \qquad (13)$$

Practically, each training batch samples points $\mathbf{z} \sim q_Z(\mathbf{z})$ from a normal distribution, transforms them via F_{zx}, and evaluates Eq. 13. As shown in the supplementary materials, the KL loss can be rewritten to:

$$J_{KL} = U - H_X + H_Z \qquad (14)$$

which is, up to the constant H_Z, equal to the free energy of the generated distribution with internal energy U and entropic factor H_X.

We can extend Eq. 13 to simultaneously train at multiple temperatures, obtaining:

$$J_{KL}^{T^1,\ldots,T^K} = \sum_{k=1}^{K} \mathbb{E}_{\mathbf{z}\sim\mu_Z^k(\mathbf{z})}\Big[u^k\Big(F_{zx}(\mathbf{z};\theta)\Big) - \log R_{zx}(\mathbf{z};\theta)\Big]$$

The KL divergence $\mathrm{KL}_\theta[\mu_Z\|q_Z]$ is also minimized in probability density distillation used in different contexts, e.g., in the training of recent audio generation networks (*16*).

Reweighting and interpretation of latent KL as reweighting loss

A simple way to compute quantitative statistics using Boltzmann generators is to use reweighting of probability densities by assigning the statistical weight $w_X(\mathbf{x})$ to each generated configuration $\mathbf{x}$. Using Eqs. 3 and 4, we obtain:

$$w_X(\mathbf{x}) = \frac{\mu_X(\mathbf{x})}{q_X(\mathbf{x})} = \frac{q_Z(\mathbf{z})}{\mu_Z(\mathbf{z})} \qquad (15)$$

$$\propto \mathrm{e}^{-u_X\left(F_{zx}(\mathbf{z})\right) + u_Z(\mathbf{z}) + \log R_{zx}(\mathbf{z};\theta)}$$

Equilibrium expectation values can then be computed as:

$$\mathbb{E}[O] \approx \frac{\sum_{i=1}^{N} w_X(\mathbf{x}) O(\mathbf{x})}{\sum_{i=1}^{N} w_X(\mathbf{x})} \qquad (16)$$

All free-energy profiles shown in Figs. 2 and 3 and fig. S2 were computed by $-k_B T \log p(R(\mathbf{x}))$ where $p(R(\mathbf{x}))$ is a probability density computed from a weighted histogram of the coordinate $R(\mathbf{x})$ using the weighted expectation (Eq. 16). Histogram bins with weights worth less than 0.01 samples are discarded to avoid making unreliable predictions.

Using Eq. 15, it can be shown that minimization of the KL divergence (Eq. 13) is equivalent to maximizing the sample weights:

$$\begin{aligned}&\min \mathrm{KL}_\theta[\mu_Z\|q_Z] \\ &\quad = \min \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}[\log \mu_Z(\mathbf{z}) \ - \log q_Z(\mathbf{z};\theta)] \\ &\quad = \max \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}[\log w_X(\mathbf{x}|\mathbf{z})]\end{aligned}$$

Configuration KL divergence and ML

Likewise, we can express the KL divergence in $\mathbf{x}$ space where we compute the divergence between the probability of generated samples with their Boltzmann weight. Using Eqs. 3, 4, and 11:

$$\begin{aligned}\mathrm{KL}_\theta[\mu_X\|q_X] &= H_X - \int \mu_X(\mathbf{x}) \log q_X(\mathbf{x};\theta)\mathrm{d}\mathbf{x} \\ &= H_X - \int \mu_X(\mathbf{x}) \Big[\log \mu_Z\Big(F_{xz}(\mathbf{x};\theta)\Big) \\ &\qquad + \log R_{xz}(\mathbf{z};\theta)\Big]\mathrm{d}\mathbf{x}. \\ &= H_X + \log Z_Z + \mathbb{E}_{\mathbf{x}\sim\mu(\mathbf{x})}\Big[\frac{1}{\sigma^2}\|F_{xz}(\mathbf{x};\theta)\|^2 \\ &\qquad - \log R_{xz}(\mathbf{x};\theta)\Big]\end{aligned}$$

This loss is difficult to evaluate because we cannot sample from $\mu(\mathbf{x})$ a priori. However, we can approximate the configuration KL divergence by starting from a sample $\rho(\mathbf{x})$, resulting in:

$$\begin{aligned}J_{ML} &= -\mathbb{E}_{\mathbf{x}\sim\rho(\mathbf{x})}[\log q_X(\mathbf{x};\theta)] \\ &= \mathbb{E}_{\mathbf{x}\sim\rho(\mathbf{x})}\Big[\frac{1}{\sigma^2}\|F_{xz}(\mathbf{x};\theta)\|^2 - \log R_{xz}(\mathbf{x};\theta)\Big]\end{aligned}$$

J_{ML} is the negative log-likelihood, i.e., minimizing it maximizes the likelihood of the sample $\rho(\mathbf{x})$ in the Gaussian prior density.

Symmetric divergence

The two KL divergences above can be naturally combined to the symmetric divergence

$$\mathrm{KL}_{\mathrm{sym}} = \frac{1}{2}\mathrm{KL}[\mu_X\|q_X] + \frac{1}{2}\mathrm{KL}[\mu_Z\|q_Z]$$

which corresponds, up to an additive constant, to a Jensen–Shannon divergence that uses the geometric mean of $m = \sqrt{q_X q_Z}$ instead of the arithmetic mean.

Reaction coordinate loss

In some applications, we do not want to sample from the Boltzmann distribution but promote the sampling of high-energy states in a specific direction of configuration space, for example, to compute a free-energy profile along a predefined RC $r(\mathbf{x})$ (Fig. 2, E and K). This is achieved by adding the RC loss to the minimization problem:

$$\begin{aligned}J_{RC} &= \int p\Big(r(\mathbf{x})\Big) \log p\Big(r(\mathbf{x})\Big)\ \mathrm{d}r(\mathbf{x}) \\ &= \mathbb{E}_{\mathbf{x}\sim q_X(\mathbf{x})} \log p\Big(r(\mathbf{x})\Big)\end{aligned}$$

To implement this loss, the function r is a user input, minimum and maximum bounds are given, and $p(r(\mathbf{x}))$ is computed as a batchwise kernel density estimate between the bounds.

Adaptive sampling and training

We define the following adaptive sampling method that trains a Boltzmann generator while simultaneously using it to propose new samples. The method has a sample buffer X that stores a predefined number of $\mathbf{x}$ samples. This number is chosen such that low-probability states of interest still have a chance to be part of the buffer when it represents an equilibrium sample. For the examples in Fig. 4, it was chosen

to be 10,000 (double well, Mueller potential) and 100,000 (solvated particle dimer). X can be initialized with any candidates for configurations; in the examples in Fig. 4, it was initialized with only one configuration (copied to all elements of X); for the particle dimer system, we additionally added small Gaussian noise with standard deviation 0.05 nm to avoid the initial Boltzmann generator overfitting on a single point. The Boltzmann generator was initially trained by example, minimizing J_{ML}, using batch-size 128 and 20, 20, and 200 iterations for double well, Mueller potential, and particle dimer, respectively. We then iterated the following adaptive sampling and training loop using batch-size 1000 for all examples.

1. Sample batch $\{\mathbf{x}_1, ..., \mathbf{x}_B\}$ from X.
2. Update Boltzmann generator parameters θ by training on batch.
3. For each $\mathbf{x}$ in batch, propose a Metropolis Monte Carlo step in latent space with step size s:

$$\mathbf{z}' = T_{zx}(\mathbf{x}) + s\mathcal{N}(0, \mathbf{I}).$$

4. Accept or reject proposal with probability $\min\{1, \exp(-\Delta E)\}$ using:

$$\Delta E = u\Big(T_{zx}(\mathbf{z}')\Big) - u(\mathbf{x}) - \log R_{zx}(\mathbf{z}'; \theta) + \log R_{xz}(\mathbf{x}; \theta)$$

5. For the accepted samples, replace $\mathbf{x}$ by $\mathbf{x}' = T_{zx}(\mathbf{z}')$.

REFERENCES AND NOTES

1. G. M. Torrie, J. P. Valleau, Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *J. Comput. Phys.* **23**, 187–199 (1977). doi: 10.1016/0021-9991(77)90121-8
2. H. Grubmüller, Predicting slow structural transitions in macromolecular systems: Conformational flooding. *Phys. Rev. E* **52**, 2893–2906 (1995). doi: 10.1103/PhysRevE.52.2893; pmid: 9963736
3. A. Laio, M. Parrinello, Escaping free-energy minima. *Proc. Natl. Acad. Sci. U.S.A.* **99**, 12562–12566 (2002). doi: 10.1073/pnas.202427399; pmid: 12271136
4. J. Hénin, G. Fiorin, C. Chipot, M. L. Klein, Exploring Multidimensional Free Energy Landscapes Using Time-Dependent Biases on Collective Variables. *J. Chem. Theory Comput.* **6**, 35–47 (2010). doi: 10.1021/ct9004432; pmid: 26614317
5. C. C. Wang, J. Bomback, W. T. Donlon, C. R. Huo, J. James, Optical third-harmonic generation in reflection from crystalline and amorphous samples of silicon. *Phys. Rev. Lett.* **57**, 1647–1650 (1986). doi: 10.1103/PhysRevLett.57.1647; pmid: 10033507
6. K. Hukushima, K. Nemoto, Exchange Monte Carlo Method and Application to Spin Glass Simulations. *J. Phys. Soc. Jpn.* **65**, 1604–1608 (1996). doi: 10.1143/JPSJ.65.1604
7. E. Marinari, G. Parisi, Simulated Tempering: A New Monte Carlo Scheme. *Europhys. Lett.* **19**, 451–458 (1992). doi: 10.1209/0295-5075/19/6/002
8. J. G. Kirkwood, Statistical Mechanics of Fluid Mixtures. *J. Chem. Phys.* **3**, 300–313 (1935). doi: 10.1063/1.1749657
9. D. Frenkel, B. Smit, *Understanding Molecular Simulation* (Academic, 2001).
10. P. V. Klimovich, M. R. Shirts, D. L. Mobley, Guidelines for the analysis of free energy calculations. *J. Comput. Aided Mol. Des.* **29**, 397–411 (2015). doi: 10.1007/s10822-015-9840-9; pmid: 25808134
11. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**, 436–444 (2015). doi: 10.1038/nature14539; pmid: 26017442
12. Z. Zhu, M. E. Tuckerman, S. O. Samuelson, G. J. Martyna, Using novel variable transformations to enhance conformational sampling in molecular dynamics. *Phys. Rev. Lett.* **88**, 100201 (2002). doi: 10.1103/PhysRevLett.88.100201; pmid: 11909330
13. I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative adversarial networks," in *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems*, arXiv:1406.2661 (2014).
14. D. P. Kingma, M. Welling, "Auto-encoding variational Bayes," in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, arXiv:1312.6114 (2014).
15. T. Karras, T. Aila, S. Laine, J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proceedings of the 7nd International Conference on Learning Representations (ICLR)*, arXiv:1710.10196 (2018).
16. A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, D. Hassabis, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, arXiv:1711.10433 (2018).
17. R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, A. Aspuru-Guzik, Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **4**, 268–276 (2018). doi: 10.1021/acscentsci.7b00572; pmid: 29532027
18. E. G. Tabak, E. Vanden-Eijnden, Density estimation by dual ascent of the log-likelihood. *Commun. Math. Sci.* **8**, 217–233 (2010). doi: 10.4310/CMS.2010.v8.n1.a11
19. L. Dinh, D. Krueger, Y. Bengio, NICE: Non-linear independent components estimation, arXiv:1410.8516 (2015).
20. L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using Real NVP, arXiv:1605.08803 (2016).
21. D. J. Rezende, S. Mohamed, Variational inference with normalizing flows, arXiv:1505.05770 (2015).
22. D. P. Kingma, P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *NIPS'18 Proceedings of the 31st International Conference on Neural Information Processing Systems*, arXiv:1807.03039 (2018).
23. W. Grathwohl, R. T. Q. Chen, J. Bettencourt, I. Sutskever, D. Duvenaud, FFJORD: Free-form continuous dynamics for scalable reversible generative models, arXiv:1810.01367 (2018).
24. P. G. Bolhuis, D. Chandler, C. Dellago, P. L. Geissler, Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annu. Rev. Phys. Chem.* **53**, 291–318 (2002). doi: 10.1146/annurev.physchem.53.082301.113146; pmid: 11972010
25. J. P. Nilmeier, G. E. Crooks, D. D. L. Minh, J. D. Chodera, Nonequilibrium candidate Monte Carlo is an efficient tool for equilibrium simulation. *Proc. Natl. Acad. Sci. U.S.A.* **108**, E1009–E1018 (2011). doi: 10.1073/pnas.1106094108; pmid: 22025687
26. P. Eastman, M. S. Friedrichs, J. D. Chodera, R. J. Radmer, C. M. Bruns, J. P. Ku, K. A. Beauchamp, T. J. Lane, L.-P. Wang, D. Shukla, T. Tye, M. Houston, T. Stich, C. Klein, M. R. Shirts, V. S. Pande, OpenMM 4: A Reusable, Extensible, Hardware Independent Library for High Performance Molecular Simulation. *J. Chem. Theory Comput.* **9**, 461–469 (2013). doi: 10.1021/ct300857j; pmid: 23316124
27. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, Tensorflow: Large-scale machine learning on heterogeneous systems, arXiv:1603.04467 (2016).
28. D. E. Shaw *et al.*, Atomic-level characterization of the structural dynamics of proteins. *Science* **330**, 341–346 (2010). doi: 10.1126/science.1187409; pmid: 20947758
29. G. Pérez-Hernández, F. Paul, T. Giorgino, G. De Fabritiis, F. Noé, Identification of slow molecular order parameters for Markov model construction. *J. Chem. Phys.* **139**, 015102 (2013). doi: 10.1063/1.4811489; pmid: 23822324
30. M. K. Scherer *et al.*, PyEMMA 2: A Software Package for Estimation, Validation, and Analysis of Markov Models. *J. Chem. Theory Comput.* **11**, 5525–5542 (2015). doi: 10.1021/acs.jctc.5b00743; pmid: 26574340
31. M. J. Grey, C. Wang, A. G. Palmer 3rd, Disulfide bond isomerization in basic pancreatic trypsin inhibitor: Multisite chemical exchange quantified by CPMG relaxation dispersion and chemical shift modeling. *J. Am. Chem. Soc.* **125**, 14324–14335 (2003). doi: 10.1021/ja0367389; pmid: 14624581
32. D. Frenkel, A. J. C. Ladd, New Monte Carlo method to compute the free energy of arbitrary solids. Application to the fcc and hcp phases of hard spheres. *J. Chem. Phys.* **81**, 3188–3193 (1984). doi: 10.1063/1.448024
33. W. G. Hoover, F. H. Ree, Melting Transition and Communal Entropy for Hard Spheres. *J. Chem. Phys.* **49**, 3609–3617 (1968). doi: 10.1063/1.1670641
34. F. M. Ytreberg, D. M. Zuckerman, *J. Chem. Phys.* **124**, 104105 (2006). doi: 10.1063/1.2174008; pmid: 16542066
35. M. Chen, T.-Q. Yu, M. E. Tuckerman, Locating landmarks on high-dimensional free energy surfaces. *Proc. Natl. Acad. Sci. U.S.A.* **112**, 3235–3240 (2015). doi: 10.1073/pnas.1418241112; pmid: 25737545
36. J. Wang, S. Olsson, C. Wehmeyer, A. Pérez, N. E. Charron, G. de Fabritiis, Frank Noé, C. Clementi, Machine learning of coarse-grained molecular dynamics force fields. *ACS Central Sci.* **5**, 755–767 (2019). doi: 10.1021/acscentsci.8b00913; pmid: 31139712
37. J. Behler, M. Parrinello, Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **98**, 146401 (2007). doi: 10.1103/PhysRevLett.98.146401; pmid: 17501293
38. M. Rupp, A. Tkatchenko, K.-R. Müller, O. A. von Lilienfeld, Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.* **108**, 058301 (2012). doi: 10.1103/PhysRevLett.108.058301; pmid: 22400967
39. K. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, K.-R. Müller, SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. *Adv. Neural Inf. Process. Syst.* **30**, 992–1002 (2007).
40. J. S. Smith, O. Isayev, A. E. Roitberg, ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **8**, 3192–3203 (2017). doi: 10.1039/C6SC05720A
41. M. R. Shirts, J. D. Chodera, Statistically optimal analysis of samples from multiple equilibrium states. *J. Chem. Phys.* **129**, 124105 (2008). doi: 10.1063/1.2978177; pmid: 19045004
42. F. Noé, S. Olsson, J. Köhler, H. Wu, Boltzmann generators – Sampling equilibrium states of many-body systems with deep learning, Zenodo (2019); doi: 10.5281/zenodo.3242635

ACKNOWLEDGMENTS

We are grateful to C. Clementi (Rice University), B. Husic, M. Sadeghi, M. Hoffmann (FU Berlin), and P. Shanahan (MIT) for valuable comments and discussions. **Funding:** We acknowledge funding from the European Commission (ERC CoG 772230 "ScaleCell"), the Deutsche Forschungsgemeinschaft (CRC1114/A04, GRK2433 DAEDALUS), the MATH+ Berlin Mathematics Research Center (AA1x8, EF1x2), and the Alexander von Humboldt Foundation (postdoctoral fellowship to S.O.). **Author contributions:** F.N., S.O., and J.K. designed and conducted research. F.N., J.K., and H.W. developed theory. F.N., S.O., and J.K. developed computer code. F.N. and S.O. wrote the paper. **Competing interests:** The authors declare no competing interests. **Data and materials availability:** Data and computer code for generating results of this paper are available at Zenodo (*42*).

SUPPLEMENTARY MATERIALS

science.sciencemag.org/content/365/6457/eaaw1147/suppl/DC1
Supplementary Text
Figs. S1 to S3
Table S1
References (*43*–*49*)

29 November 2018; accepted 19 July 2019
10.1126/science.aaw1147

Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning

Frank Noé, Simon Olsson, Jonas Köhler and Hao Wu

Science **365** (6457), eaaw1147.
DOI: 10.1126/science.aaw1147

Efficient sampling of equilibrium states

Molecular dynamics or Monte Carlo methods can be used to sample equilibrium states, but these methods become computationally expensive for complex systems, where the transition from one equilibrium state to another may only occur through rare events. Noé *et al.* used neural networks and deep learning to generate distributions of independent soft condensed-matter samples at equilibrium (see the Perspective by Tuckerman). Supervised training is used to construct invertible transformations between the coordinates of the complex system of interest and simple Gaussian coordinates of the same dimensionality. Thus, configurations can be sampled in this simpler coordinate system and then transformed back into the complex one using the correct statistical weighting.

Science, this issue p. eaaw1147; see also p. 982