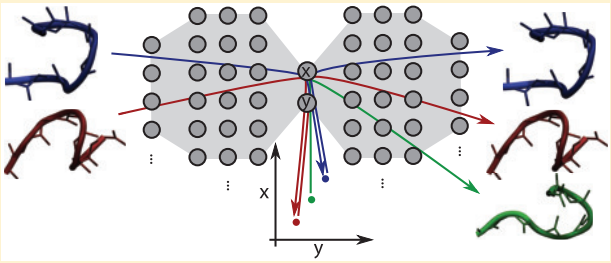
Erschienen in: Journal of Chemical Theory and Computation (JCTC) ; 15 (2019), 2. - S. 1209-1215
https://dx.doi.org/10.1021/acs.jctc.8b00975

EncoderMap: Dimensionality Reduction and Generation of Molecule Conformations

Tobias Lemke and Christine Peter*

Theoretical Chemistry, University of Konstanz, 78547 Konstanz, Germany

ABSTRACT: Molecular simulation is one example where large amounts of high-dimensional (high-d) data are generated. To extract useful information, e.g., about relevant states and important conformational transitions, a form of dimensionality reduction is required. Dimensionality reduction algorithms differ in their ability to efficiently project large amounts of data to an informative low-dimensional (low-d) representation and the way the low and high-d representations are linked. We propose a dimensionality reduction algorithm called EncoderMap that is based on a neural network autoencoder in combination with a nonlinear distance metric. A key advantage of this method is that it establishes a functional link from the high-d to the low-d representation and vice versa. This allows us not only to efficiently project data points to the low-d representation but also to generate high-d representatives for any point in the low-d map. The potential of the algorithm is demonstrated for molecular simulation data of a small, highly flexible peptide as well as for folding simulations of the 20-residue Trp-cage protein. We demonstrate that the algorithm is able to efficiently project the ensemble of high-d structures to a low-d map where major states can be identified and important conformational transitions are revealed. We also show that molecular conformations can be generated for any point or any connecting line between points on the low-d map. This ability of inverse mapping from the low-d to the high-d representation is particularly relevant for the use in algorithms that enhance the exploration of conformational space or the sampling of transitions between conformational states.



1. INTRODUCTION

With modern computational resources the amount and complexity of data that can be generated, processed, and stored grows rapidly. At the same time, we as humans remain unfit to grasp such complex data directly, and one of the key research questions is how to condense massive amounts of data into a human understandable form.

Molecular simulation is one such example where huge data sets are generated.[1] The primary result of such a simulation is typically a trajectory containing atom positions of a given (bio)molecular system for multiple (time) frames. With three spatial coordinates per atom, each frame of the trajectory is a point in a phase space with easily several thousands or millions of dimensions. Not all of these are equally important for mechanistic interpretation. For a protein in aqueous solution it is, for example, possible to describe much of a conformation through the backbone-dihedral angles. With two backbone-dihedral angles Φ and Ψ per amino acid, this still leads to a representation with tens or hundreds of dimensions. To make sense of such data, i.e., to characterize processes such as folding or conformational transitions and to identify relevant (conformational) states, one needs to further reduce the dimensionality.

Besides "simply guessing" relevant individual or collective variables that are well suited to describe processes or distinguish states, a whole variety of systematic dimensionality reduction approaches are available: algorithms such as principle component analysis (PCA),[2] sketch-map,[3,4] diffusion maps,[5] or time-lagged independent component analysis (TICA)[6] have been successfully established in the simulation community. There are different aspects that determine how well a given dimensionality reduction algorithm is suited for the data at hand. For molecular simulation data, three relevant criteria are (1) how informative is the low-d representation, and how well is it able to separate the data points into distinct states, (2) how fast is it, and (3) how are the high-d and low-d representations linked?

An efficient link from the high-d to the low-d representation is important whenever additional data points should be projected to the low-d representation. This is especially important if dimensionality reduction is not performed solely for analysis purposes. For example, biasing of simulations for enhanced sampling purposes requires a link from the high-d to the low-d representation that allows to apply a biasing potential/force defined in the low-d representation.[7,8] Likewise, a link in the opposite direction, i.e., the ability to generate high-d coordinates corresponding to a given point in the low-d representation, is highly desirable, for example, to accelerate/enhance the sampling of phase space by initiating new

simulations from specific regions or states. Such inverse mapping or back-mapping problems from a coarse representation back to a more detailed representation are well-known in scale-bridging between simulation levels,.[9,10]

The various dimensionality reduction algorithms exhibit particular individual strengths (and weaknesses) with regard to the above criteria. Although the expressiveness of the low-d representation is a vague and hard to define criterion, it is clear that linear techniques such as PCA lack the capability to unravel inherently nonlinear features of the data as found, for example, in the folding of proteins.[11] Here nonlinear techniques such as sketch-map[3] have proven to be more suitable. Sketch-map is a multidimensional-scaling like algorithm,[12] which aims for a reproduction of the (high-d) pairwise distances between data points in the low-d representation. The main idea behind sketch-map is that not all of these pairwise distances are equally important. If one imagines the high-d molecule conformations to be associated in clusters connected in a spider-web like fashion, then small distances originate from fluctuations inside such clusters. Long distances originate from the relative position of these clusters in high-d space but cannot be well represented in low-d. Intermediate distances contain information about neighboring clusters and are therefore most important to capture the connectivity of a high-d network. Sketch-map therefore transforms the pairwise distances with sigmoid functions to place the focus on intermediate distances where these sigmoid functions are tuned to be most steep. A low-d representation is obtained by minimizing the following cost function:

$$C_{\text{sketch}} = \frac{1}{m} \sum_{i \neq j} [\text{SIG}_{\text{h}}(R_{ij}) - \text{SIG}_{\text{l}}(r_{ij})]^2 \tag{1}$$

where m is the number of pairwise distances, R_{ij} are the pairwise distances in the high-d space, r_{ij} are the pairwise distances in the low-d space, and SIG_{h} and SIG_{l} are sigmoid functions of the following form:

$$\text{SIG}_{\sigma,a,b}(r) = 1 - (1 + (2^{a/b} - 1)(r/\sigma)^a)^{-b/a} \tag{2}$$

where σ defines the location of the inflection point of the sigmoid and a and b determine how quickly the function approaches 0 and 1, respectively. Minimizing this cost function is computationally expensive and scales unfavorably because the number of pairwise distances grows quadratically with the number of data points. The sketch-map algorithm overcomes this scaling problem by performing the dimensionality reduction with a representative subset of the data, so-called landmarks. The remaining data points are then projected into the resulting low-d map. Note that the inverse mapping, i.e., the reconstruction of high-d coordinates given a point in the low-d representation is not easily possible because no functional relationship between high-d (atomistic) and low-d sketch-map coordinates exists. In summary, sketch-map is a nonlinear method that can nicely capture quite complex network like features in a low-d representation, however, it has limitations in the efficiency of finding the low-d representation and projecting data in and out of the low-d representation.

Neural network autoencoders,[13] which have just recently come to the attention of the simulation community,[8,14−17] are quite the opposite. Autoencoders have their strengths where sketch-map has its weaknesses. The main concept of an autoencoder is to train an artificial neural network to reproduce its inputs. This seemingly pointless undertaking becomes useful by introducing a bottleneck in the middle of the neural network. The autoencoder can then be viewed as consisting of two parts. One encoder part that is trained to encode as much information as possible into a low-d representation and a decoder part that is trained to reconstruct the initial high-d input from this low-d representation. Autoencoders are usually trained by gradient descent with mini-batches of data. In other words, creating the low-d representation has a very favorable linear scaling with the number of data points used. It is therefore not necessary to select landmarks. Instead, large amounts of data can be used to obtain the low-d representation. Also, due to the recent boom in machine learning, there are highly optimized computational libraries like Torch,[18] Theano,[19] or TensorFlow[20] available that make the training procedure fast and efficient. Once the autoencoder is trained, the encoder part of the network is a differentiable mathematical function that directly maps high-d inputs to the low-d space. This makes projecting additional points into the low-d representation very efficient and, for example, allows to use the low-d representation for enhanced sampling with biasing techniques[8] like umbrella sampling[21] or metadynamics.[22,23] With the decoder part of the network an autoencoder also provides a mathematical function to reconstruct a high-d point given any point in the low-d representation. A drawback of autoencoders, however, is how data are expressed in the low-d representation. During the training process the low-d representation is optimized in such a way that the decoder part of the network can reproduce the data most accurately. This is not necessarily the most intuitive or informative way to present the data to humans.[24] Note that distances between data points are not necessarily preserved in the low-d space.

Here we propose a dimensionality reduction algorithm called EncoderMap that combines the advantages of autoencoders and the sketch-map cost function. The autoencoder provides the efficient functional links between the high-d and low-d representations and the nonlinear distance metric-based cost function forces the autoencoder to arrange points in the low-d representation in a meaningful way. In the following we explain the method and show and discuss results for two example data sets from molecular dynamics simulations: The small peptide aspartic acid heptamer (Asp-7) and the mini protein Trp-cage (PDB 1L2Y).[25] The method, however, is not limited to such data.

2. ENCODERMAP

EncoderMap unites the advantages of two methods in a single algorithm. It combines a neural network autoencoder[13] with the cost function of sketch-map.[3] The autoencoder used in this work consists of several fully connected layers: an input layer, several hidden layers, a bottleneck layer with few neurons, further hidden layers, and an output layer with the same number of neurons as in the input layer (see Figure 1). In a fully connected layer the activation of each neuron is calculated as a weighted sum of the activations of all neurons in the previous layer to which an activation function is applied. This can efficiently be noted and calculated in the form of a matrix multiplication:

$$\mathbf{a}_l = f_l(\mathbf{W}_l \mathbf{a}_{l-1} + \mathbf{b}_l) \tag{3}$$

where $\mathbf{a}_l$ are the activations of layer l, f_l is the activation function, $\mathbf{W}_l$ is a matrix of weights, $\mathbf{a}_{l-1}$ are the activations of

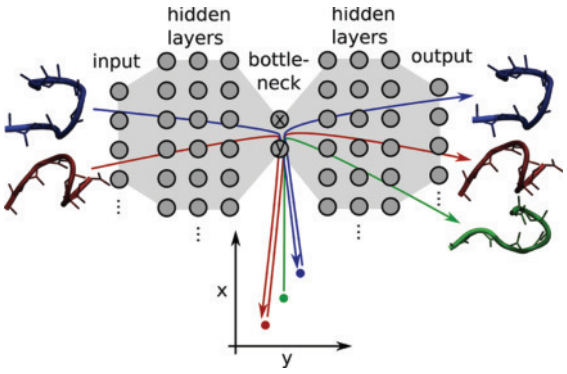


Figure 1. Example autoencoder architecture with two bottleneck neurons. This results in a projection of high-d data, e.g., molecule conformations, to points in a 2-d space. The decoder part (right-hand side) can generate output conformations for 2-d data points for which no corresponding input exists (green point/conformation).

the previous layer, and $\mathbf{b}_l$ are bias values added. The activation function is important to add nonlinearity to the network, and thus the flexibility to fit nonlinear features. The weights and biases are the parameters that are tuned during the training procedure to minimize a cost function. For an autoencoder this cost function naturally involves some distance measure between the inputs and outputs of the neural network:

$$C_{\text{auto}} = \frac{1}{n}\sum_{i=1}^{n} \text{dist}(\mathbf{x}_i, \tilde{\mathbf{x}}_i) \tag{4}$$

where n is the number of training examples, x_i is the vector of inputs for the ith training example, $\tilde{\mathbf{x}}_i$ is the vector of outputs of the autoencoder and dist is some distance metric. With this cost function, the network is trained to reproduce the inputs while it is forced to encode the information in a few dimensions due to the bottleneck in the network. How the information is encoded, however, is very arbitrary. To make this low-d representation better defined and interpretable, EncoderMap combines the autoencoder cost function (eq 4) with the sketch-map cost function (eq 1):

$$C = k_a C_{\text{auto}} + k_s C_{\text{sketch}} + \text{REG} \tag{5}$$

The sketch-map cost function is based on the pairwise distances between data points in the high-d input space and the low-d space given by the bottleneck layer. The constants k_a and k_s allow us to shift the priority between minimizing the autoencoder and the sketch-map cost and REG is a regularization term. Regularization is used to adjust the complexity or roughness of a neural network to prevent overfitting. Here we used the following regularization term:

$$\text{REG} = \frac{k_{l2}}{2}\sum_i w_i^2 + \frac{k_c}{n}\sum_n \sum_j a_{b,j,n}^2 \tag{6}$$

where the first term restricts the weights w_i of the network and the second term restricts the activation in the bottleneck layer a_b to prevent the low-d representation to be shifted toward large numbers. k_{l2} and k_c are constants to scale the influence of these terms. The network is trained using the Adam optimizer.[26] As the training is done with batches of data, no selection of landmarks is necessary. Large amounts of data can be used for training one batch at a time.

In the following examples we use the backbone dihedral angles, Φ and Ψ, as input for EncoderMap as they are

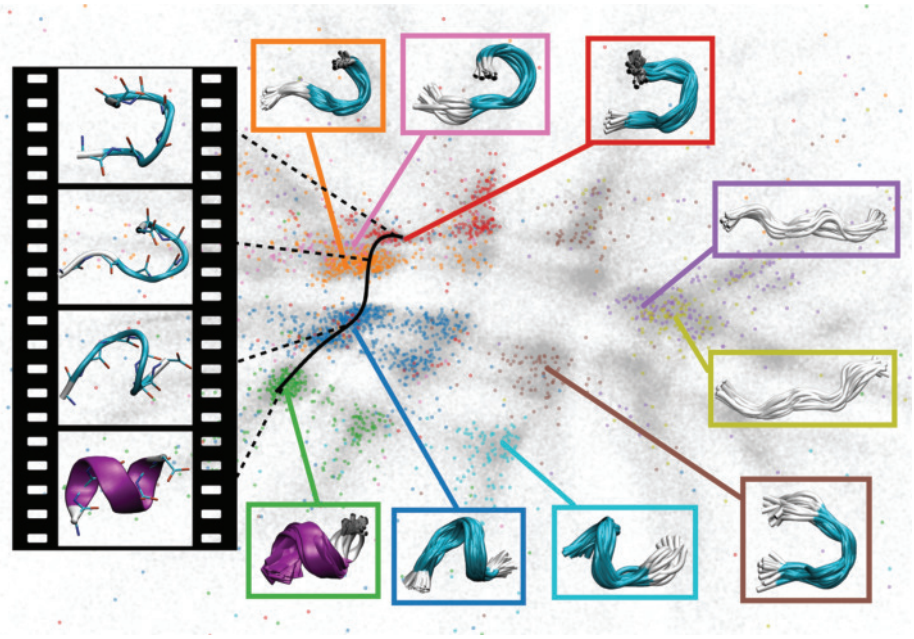


Figure 2. 2-d EncoderMap of Asp-7. Colored dots represent conformations from the nine largest clusters in an rmsd-based clustering.[29] Superimposed molecule conformations from these clusters are shown in the respectively colored frames. All other conformations are represented as gray dots. The black line shows the path along which molecule conformations were generated with the decoder part of the network. Some snapshots of these generated structures are shown in the film strip on the left. The complete sequence of all generated conformations can be found in Movie S1. A comparison of this EncoderMap projection with projections from other dimensionality reduction techniques can be found in Figure S1. Further analysis based on the Asp-7 data set are shown in Figures S2, S3, and S4.

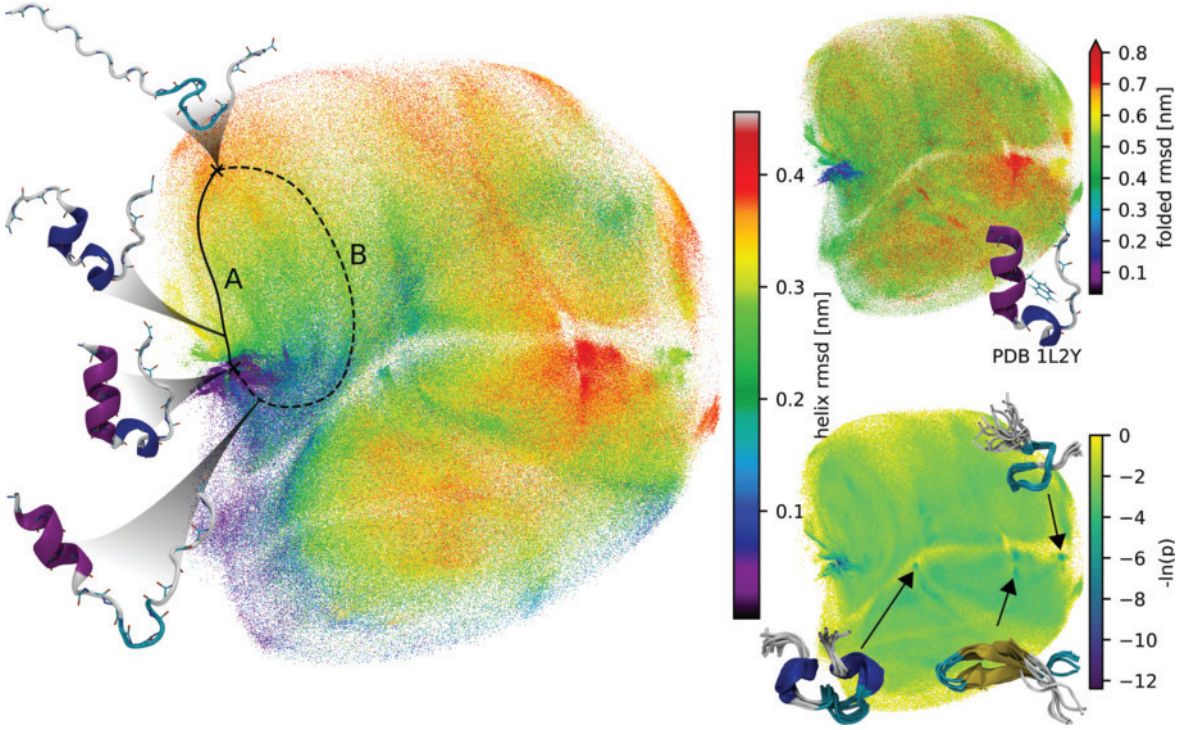


Figure 3. EncoderMap of Trp-cage in different colorings. The top-right map is colored according to the C_α rmsd compared to the shown native structure. The bottom-right map is a 2-d-histogram of the map with 300 × 300 bins colored according to the negative natural logarithm of the bin counts. Superimposed structures from selected dense regions outside the main folded region are shown. The map on the left is colored according to the C_α rmsd of residues 2−8 compared to an α-helical structure. Molecule conformations where generated for points on the two paths shown as solid and dashed lines. Selected generated conformations are shown in the plot and all generated conformations can be found in Movie S3 and Movie S4.

rotationally and translationally invariant and independent from each other. At the same time, they describe the conformation of the peptide backbone almost completely and allow simple reconstruction of a backbone conformation starting with an extended conformation by rotation around the respective dihedral axis. The periodicity of these inputs, however, adds some complexity to the approach. To incorporate periodicity into the network we map all dihedral angles to their sin and cos values before passing them to the first hidden layer. This transforms each angle into two nonperiodic Cartesian coordinates of a point on a unit circle. Pairs of activations of the output layer are then also treated as such Cartesian unit circle coordinates and are mapped back to an angle. Also, the distance metric used for the cost function has to respect periodicity. Here, we use the smallest euclidean distance in a periodic space. With these adjustments EncoderMap works well with periodic inputs like dihedral angles as we show in the following examples.

3. RESULTS AND DISCUSSION

3.1. Example 1: Asp-7. For a first test of EncoderMap, we chose an example of medium complexity. Aspartic acid heptamer with its 12 backbone dihedrals has a dimensionality well beyond what is directly graspable and is still a fairly simple molecule. From our previous studies of this system,[27,28] we know that it exhibits a variety of states and only a low fraction of completely folded helical structures. The data set used consists of 500 000 molecule conformations from a 5 μs molecular dynamics simulation. The EncoderMap was obtained on the basis of the 12 dihedral angles of each conformation. This takes around 3 min on a desktop computer (i7-8700K GTX 1080) with our code provided on https://github.com/AG-Peter/EncoderMap. Details about the EncoderMap training procedure are given in the Method Details section. The resulting 2d EncoderMap is shown in Figure 2. All peptide conformations of the data set are represented as gray dots. For comparison we performed the C_α root-mean-square deviation (rmsd) based gromos clustering[29] with a cutoff of 0.1 nm on a subset of 10 000 conformations. The conformations assigned to the nine largest clusters are represented with dots colored according to their cluster numbers. Overlain conformations from each cluster are shown in the accordingly colored frames. Figure 2 shows that EncoderMap nicely separates the conformations from different clusters. Some clusters such as the blue or green ones are further separated into subclusters. Furthermore, clusters with similar structures are located in proximity to each other and connections between these clusters become visible. For example, the green cluster that represents the helical structure is placed close to the two blue clusters where fragments of the helix are already present. One of the big advantages of the EncoderMap algorithm is that it not only allows us to efficiently generate expressive low-d representations of large data sets but also the reconstruction of high-d data based on any point in the low-d map. This works also for new points that were not part of the training data. To illustrate this feature, 450 equally spaced points on the black path shown in Figure 2 were fed into the decoder part of the autoencoder. The decoder yields 12 dihedral angles for each of these 2-d points. On the basis of

these dihedral angles backbone conformations can be reconstructed by simple rotation around the respective dihedral axis. Selected generated conformations are shown in the filmstrip depicted in Figure 2. The generated conformations nicely resemble the conformations of the clusters through which the path was drawn, showing that the decoder predicts realistic conformations for regions where reference data exist. Between the clusters, in regions where only few conformations are present in the original data set, the neural network smoothly interpolates the conformations. This is demonstrated in Movie S1 where all generated conformations on the black path are shown. Note that this is not meant to imply that a path that is drawn along the 2-d map necessarily corresponds to real transition paths in the simulations. Nevertheless, it demonstrates the ability of EncoderMap to generate conformations that connect given data points.

While 2D projections are most practical for visualization purposes, it is possible to create projections of any dimensionality with EncoderMap. Movie S2 shows a 3d EncoderMap of the asp-7 data set. Projections with more than 3 dimensions are impractical for visualization but might still be useful for, e.g., density-based clustering algorithms. Here, the dimensionality affects the separability of states into well identifiable clusters, but a too high dimensionality will again make the density distribution very sparse and clustering problematic. Figure S4 compares projections with different dimensionalities based on the obtained values for the autoencoder and sketch-map part of the cost function. Next, we test the EncoderMap algorithm with a more complex example.

3.2. Example 2: Trp-cage. Trp-cage (PDB 1L2Y) is a well studied 20 residue protein.[25,30] The folded structure of Trp-cage consists of an N-terminal α-helical part and a hydrophobic core around the tryptophan residue. Several computational studies have revealed two folding pathways for the Trp-cage.[31−34] One path where the helix folds after the hydrophobic core is formed and one path where the helix forms before the hydrophobic core. The data set used here consists of 1.5 million Trp-cage conformations from a temperature replica exchange simulation with 40 replicas in a range of temperatures from 300 to 570 K and a total simulated time of 3 μs. Conformations from all temperature replicas are included in the data set (a separate analysis of the lowest replica only can be found in Figure S5). In complete analogy to the Asp-7 example an EncoderMap based on the 38 backbone dihedrals of Trp-cage was obtained. Figure 3 shows the resulting 2-d map in different colorings (further colorings based on the different dihedrals are available in Figure S6). The top-right map is colored according to the C_α rmsd compared to the native folded (PDB 1L2Y) structure. This coloring reveals an area of folded structures in the left part of the map surrounded by a broad area of unfolded structures. The map in the bottom-right shows the density of points on a logarithmic scale. As to be expected, the area of fully folded structures has the highest density of points and the surrounding unfolded or partially folded region is less densely populated. Several regions of increased density can be found outside of the folded area. The overlays of conformations corresponding to three of these regions (highlighted with arrows in the figure) show that the EncoderMap is well suited for the identification of distinct (albeit rarely occurring) clusters of conformations. The map in the left is colored according to the C_α rmsd of residues 2−8 compared to an α-helical structure. This map shows that the folded area exhibits low helix rmsd values since residues 2−8 form a helix in the folded structure. The coloring reveals that this helix is also present in parts of the unfolded/partially folded regions. Approaching the area of fully folded conformations from the bottom leads through unfolded/partially folded structures where the helix is already formed. In contrast, approaching the fully folded area from the top leads through unfolded structures where no helix is present. To further illustrate theses different routes toward the folded structure we use the high-d data generation capabilities of EncoderMap. A total of 450 equidistant points on two connecting lines from a given unfolded structure to the native structure (indicated in black in Figure 3) were fed into the decoder part of the network. Some of the generated conformations are presented in Figure 3; all generated conformations along the two paths are shown in Movie S3 and Movie S4. On path A we find structures close to the folded area where the hydrophobic core is present but the helix region is unfolded. On path B, in contrast, we find structures where the helix is folded but the hydrophobic core is not collapsed. Note that while these paths do not correspond to real transitions found in the simulation, they do identify approaches toward the folded structure that are in good agreement with the known Trp-cage folding pathways from the literature.[31−34]

4. CONCLUSIONS

We have introduced EncoderMap, a dimensionality reduction algorithm that combines a neural network autoencoder with the nonlinear-distance-metric based cost function of sketch-map. The presented molecular example systems described in dihedral space demonstrate EncoderMaps ability to represent complex high-d data in an informative low-d form. Similar data points are aggregated in clusters, and important proximity relations between these clusters are nicely captured. The method is not limited to dihedrals but can be used with different kinds of data such as Cartesian coordinates or distance based measures describing molecular conformations.[35] Even with large data sets of more then a million points, creating an EncoderMap is very efficient and takes only few minutes on a common desktop computer. Additional points can be projected to the map with a differentiable function that the algorithm yields. This allows for efficient projection of additional points to the map and allows us to combine EncoderMap with enhanced sampling schemes that require biasing potentials defined in a low dimensional space. With EncoderMap, however, it is not only possible to project additional high-d data to the map. Our examples show that it is also possible to generate reasonable high-d points for any points in the low-d map. In that sense the EncoderMap can be used as an interesting type of molecular model. A model that is not defined as a high-d energy function or force-field that requires simulations to obtain probable conformations, but a model where conformations can directly be generated from selected points in the low-d map. This inverse-mapping ability opens up new avenues toward the use of EncoderMap for enhanced sampling. Search schemes where new simulations are initiated from sparsely populated or transitional regions in conformational space could benefit from generated structures slightly extrapolated away from low density regions in the probability density distribution, path sampling schemes could benefit from EncoderMap's ability to generate plausible paths between known strucures.

5. METHOD DETAILS

5.1. EncoderMap Parameters. EncoderMap has multiple parameters that can be tuned. For both examples in this paper, however, we use the exact same parameters, which indicates that these might be a good starting point for a broad variety of molecular systems described in dihedral space.

The autoencoder used in this work consists of 9 fully connected layers: an input layer, 3 hidden layers with 128 neurons each, a bottleneck layer with 2 neurons, another 3 hidden layers with 128 neurons each, and an output layer with the same number of neurons as in the input layer. We use tanh for all hidden layers and the identity function for all other layers as activation function. The exact number of neurons in the hidden layers is not important. There have to be enough neurons to give the network the required complexity for the task. A larger number of neurons is, except for computational efficiency, reasons usually not a problem if regularization is used to adjust the roughness of the representation. The used cost function parameters from eq 5 and eq 6 are $k_a = 1$, $k_s = 500$, $k_{l2} = 0.001$, and $k_c = 0.0001$. Figure S2 shows the influence of different k_a and k_s settings. The large k_s value chosen here emphasizes the minimization of the sketch-map part of the cost function. To ensure that accurate generation of high-d data are still given, it is helpful to analyze the deviation between inputs and outputs of the autoencoder, as shown in Figure S7. The regularization parameters can be tuned on the basis of the training and validation cost or to obtain a desired roughness of the map. A smooth map might highlight the most important differences in the data set while a rougher map might give more detail. To obtain the training and validation cost, we have performed 5-fold cross validation. The results of this cross validation study for different neural network settings are shown in Figure S3.

The used sketch-map cost function parameters from eq 2 are $\sigma = 4.5$, $a = 12$, $b = 6$ for the sigmoid applied to the high-d distances, and $\sigma = 1$, $a = 2$, $b = 6$ for the low-d distances. The sketch-map literature[3] gives more details on how to select these parameters. The cost function is minimized in 100 000 steps with batches of 256 training points by the Adam optimizer[26] with a learning rate of 0.001 and exponential decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as implemented in TensorFlow 1.8.[20]

5.2. Simulation Details Asp-7. The Asp-7 simulation was performed with Gromacs 5.1.3[36] and the GROMOS 54A7[37] force field with SPC/E water.[38] Under periodic boundary conditions, 3610 water molecules and one asp-7 molecule where all residues were selected to be in their uncharged state were simulated. The temperature was kept at 300 K with stochastic velocity rescaling.[39] The pressure was adjusted to 1 bar in with a Berendsen barostat[40] in a preceding equilibration simulation. In the production simulation no barostat was used. The leapfrog algorithm was used to integrate the equations of motion with a time step of 2 fs. Long range interactions were calculated with the particle mesh Ewald method[41] with a grid spacing of 0.12 nm and a pme-order of 4. All interactions were truncated to 1.4 nm. All bonds were constrained using the linear constraints solver algorithm[42] with an eighth-order expansion.

5.3. Simulation Details Trp-Cage. The Trp-cage simulations were performed with Gromacs 2016.3[36] and the Amber99SB-ildn force field with 3000 tip3p water molecules.[43] An equilibration run was performed starting form the folded PDB 1L2Y structure where the temperature was adjusted to 300 K with stochastic velocity rescaling[39] and the pressure was adjusted to 1 bar with a Berendsen barostat.[40] Interactions, constraints, and integrator were set in analogy to the asp-7 simulation; however, interactions were truncated at 1 nm. Based on the equilibrated folded structure, a replica exchange with 40 replicas between 300 and 570 K was set up. Exchange attempts between neighboring replicas were made every 500 simulation steps.

AUTHOR INFORMATION

Corresponding Author

E-mail: Christine.Peter@uni-konstanz.de.

ORCID

Tobias Lemke: 0000-0002-0593-2304

Christine Peter: 0000-0002-1471-5440

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We thank Oleksandra Kukharenko for many helpful discussions. Financial support by the DFG (SFB1214) is gratefully acknowledged. This work was performed with computational resources at the Zentrum für Datenverarbeitung of the University of Tübingen funded within the bwHPC program by the state of Baden-Württemberg and the DFG through grant no INST 37/935-1 FUGG

REFERENCES

(1) Dror, R. O.; Dirks, R. M.; Grossman, J.; Xu, H.; Shaw, D. E. Biomolecular simulation: a computational microscope for molecular biology. *Annu. Rev. Biophys.* **2012**, *41*, 429−452.

(2) Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **1901**, *2*, 559−572.

(3) Ceriotti, M.; Tribello, G. A.; Parrinello, M. Simplifying the representation of complex free-energy landscapes using sketch-map. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 13023−13028.

(4) Kukharenko, O.; Sawade, K.; Steuer, J.; Peter, C. Using dimensionality reduction to systematically expand conformational sampling of intrinsically disordered peptides. *J. Chem. Theory Comput.* **2016**, *12*, 4726−4734.

(5) Coifman, R. R.; Lafon, S.; Lee, A. B.; Maggioni, M.; Nadler, B.; Warner, F.; Zucker, S. W. Geometric diffusions as a tool for harmonic

analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, *102*, 7426−7431.
(6) Pérez-Hernández, G.; Paul, F.; Giorgino, T.; De Fabritiis, G.; Noé, F. Identification of slow molecular order parameters for Markov model construction. *J. Chem. Phys.* **2013**, *139*, No. 015102.
(7) Tribello, G. A.; Ceriotti, M.; Parrinello, M. Using sketch-map coordinates to analyze and bias molecular dynamics simulations. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 5196.
(8) Chen, W.; Tan, A. R.; Ferguson, A. L. Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design. *J. Chem. Phys.* **2018**, *149*, No. 072312.
(9) Peter, C.; Kremer, K. Multiscale simulation of soft matter systems−from the atomistic to the coarse-grained level and back. *Soft Matter* **2009**, *5*, 4357−4366.
(10) Das, P.; Frewen, T. A.; Kevrekidis, I. G.; Clementi, C. *Coping with Complexity: Model Reduction and Data Analysis*; Springer, 2011; pp 113−131.
(11) Das, P.; Moll, M.; Stamati, H.; Kavraki, L. E.; Clementi, C. Low-dimensional, freeenergy landscapes of protein-folding reactions by nonlinear dimensionality reduction. *Proc. Natl. Acad. Sci. U. S. A.* **2006**, *103*, 9885−9890.
(12) Cox, T. F.; Cox, M. A. *Multidimensional scaling*; Chapman and hall /CRC, 2000.
(13) Hinton, G. E.; Salakhutdinov, R. R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504−507.
(14) Wehmeyer, C.; Noé, F. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **2018**, *148*, 241703.
(15) Sultan, M. M.; Wayment-Steele, H. K.; Pande, V. S. Transferable neural networks for enhanced sampling of protein dynamics. *J. Chem. Theory Comput.* **2018**, *14*, 1887−1894.
(16) Ribeiro, J. M. L.; Bravo, P.; Wang, Y.; Tiwary, P. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *J. Chem. Phys.* **2018**, *149*, No. 072301.
(17) Hernández, C. X.; Wayment-Steele, H. K.; Sultan, M. M.; Husic, B. E.; Pande, V. S. Variational encoding of complex dynamics. *Phys. Rev. E: Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top.* **2018**, *97*, No. 062412.
(18) Collobert, R.; Bengio, S.; Marithoz, J. Torch: A Modular Machine Learning Software Library 2002.
(19) Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints abs/1605.02688*, 2016.
(20) Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Goodfellow, I.; Harp, A.; Irving, G.; Isard, M.; Jia, Y.; Jozefowicz, R.; Kaiser, L.; Kudlur, M.; Levenberg, J.; Mané, D.; Monga, R.; Moore, S.; Murray, D.; Olah, C.; Schuster, M.; Shlens, J.; Steiner, B.; Sutskever, I.; Talwar, K.; Tucker, P.; Vanhoucke, V.; Vasudevan, V.; Viégas, F.; Vinyals, O.; Warden, P.; Wattenberg, M.; Wicke, M.; Yu, Y.; Zheng, X. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*; TensorFlow, 2015; https://www.tensorflow.org/.
(21) Torrie, G. M.; Valleau, J. P. Nonphysical sampling distributions in Monte Carlo freeenergy estimation: Umbrella sampling. *J. Comput. Phys.* **1977**, *23*, 187−199.
(22) Huber, T.; Torda, A. E.; Van Gunsteren, W. F. Local elevation: a method for improving the searching properties of molecular dynamics simulation. *J. Comput.-Aided Mol. Des.* **1994**, *8*, 695−708.
(23) Laio, A.; Parrinello, M. Escaping free-energy minima. *Proc. Natl. Acad. Sci. U. S. A.* **2002**, *99*, 12562−12566.
(24) Ayinde, B. O.; Zurada, J. M. Deep learning of constrained autoencoders for enhanced understanding of data. *arXiv preprint arXiv:1802.00003*, 2018.
(25) Neidigh, J. W.; Fesinmeyer, R. M.; Andersen, N. H. Designing a 20-residue protein. *Nat. Struct. Biol.* **2002**, *9*, 425.
(26) Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* 2014,.
(27) Lemke, T.; Peter, C. Neural Network Based Prediction of Conformational Free EnergiesA New Route toward Coarse-Grained Simulation Models. *J. Chem. Theory Comput.* **2017**, *13*, 6213−6221.
(28) Lemke, T.; Peter, C.; Kukharenko, O. Efficient Sampling and Characterization of Free Energy Landscapes of Ion−Peptide Systems. *J. Chem. Theory Comput.* **2018**, *14*, 5476.
(29) Daura, X.; Gademann, K.; Jaun, B.; Seebach, D.; Van Gunsteren, W. F.; Mark, A. E. Peptide folding: when simulation meets experiment. *Angew. Chem., Int. Ed.* **1999**, *38*, 236−240.
(30) Qiu, L.; Pabit, S. A.; Roitberg, A. E.; Hagen, S. J. Smaller and faster: The 20-residue Trp-cage protein folds in 4 μs. *J. Am. Chem. Soc.* **2002**, *124*, 12952−12953.
(31) Juraszek, J.; Bolhuis, P. Sampling the multiple folding mechanisms of Trp-cage in explicit solvent. *Proc. Natl. Acad. Sci. U. S. A.* **2006**, *103*, 15859−15864.
(32) Kim, S. B.; Dsilva, C. J.; Kevrekidis, I. G.; Debenedetti, P. G. Systematic characterization of protein folding pathways using diffusion maps: Application to Trp-cage miniprotein. *J. Chem. Phys.* **2015**, *142*, No. 085101.
(33) Paschek, D.; Hempel, S. Garćıa, A. E. Computing the stability diagram of the Trp-cage miniprotein. *Proc. Natl. Acad. Sci. U. S. A.* **2008**, *105*, 17754−17759.
(34) Deng, N.-j.; Dai, W.; Levy, R. M. How kinetics within the unfolded state affects protein folding: An analysis based on Markov state models and an ultra-long MD trajectory. *J. Phys. Chem. B* **2013**, *117*, 12787−12799.
(35) Berg, A.; Kukharenko, O.; Scheffner, M.; Peter, C. Towards a molecular basis of ubiquitin signaling: A dual-scale simulation study of ubiquitin dimers. *PLoS Comput. Biol.* **2018**, *14*, No. e1006589.
(36) Abraham, M. J.; Murtola, T.; Schulz, R.; Páll, S.; Smith, J. C.; Hess, B.; Lindahl, E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* **2015**, *1*, 19−25.
(37) Schmid, N.; Eichenberger, A. P.; Choutko, A.; Riniker, S.; Winger, M.; Mark, A. E.; van Gunsteren, W. F. Definition and testing of the GROMOS force-field versions 54A7 and 54B7. *Eur. Biophys. J.* **2011**, *40*, 843.
(38) Berendsen, H.; Grigera, J.; Straatsma, T. The missing term in effective pair potentials. *J. Phys. Chem.* **1987**, *91*, 6269−6271.
(39) Bussi, G.; Donadio, D.; Parrinello, M. Canonical sampling through velocity rescaling. *J. Chem. Phys.* **2007**, *126*, No. 014101.
(40) Berendsen, H. J.; Postma, J. v.; van Gunsteren, W. F.; DiNola, A.; Haak, J. Molecular dynamics with coupling to an external bath. *J. Chem. Phys.* **1984**, *81*, 3684−3690.
(41) Essmann, U.; Perera, L.; Berkowitz, M. L.; Darden, T.; Lee, H.; Pedersen, L. G. A smooth particle mesh Ewald method. *J. Chem. Phys.* **1995**, *103*, 8577−8593.
(42) Hess, B.; Bekker, H.; Berendsen, H. J.; Fraaije, J. G. LINCS: a linear constraint solver for molecular simulations. *J. Comput. Chem.* **1997**, *18*, 1463−1472.
(43) Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.* **1983**, *79*, 926−935.