Machine-Learned Collective Variables that Represent Molecular Dynamics

Arnav Mehta [1] Roy Lam [2] Mentor: Maria Cameron [3]

[1]University of California, Berkeley [2]Cornell University [3]University of Maryland, College Park

Introduction

Consider the Overdamped Langevin Dynamics $dq = -\nabla V(q)d\tau + \sqrt{2\beta^{-1}}dW_\tau$ that models the dynamics of particles in a potential force field, where $q \in \mathbb{R}^D$ is the position of the particles, V is the potential, β^{-1} is proportional to the temperature, and W_τ is the standard Wiener process. Our goal is to find collective variables $\xi_i(q)$ that approximate dynamics well, where $\xi : \mathbb{R}^D \to \mathbb{R}^d$ is the collective variable map we want to learn. The criterion we use to learn $\xi(q)$ is that its level sets are orthogonal to the manifold on which the dynamics lives.

Summary of Our Approach

- **Step 1** Target Measure Diffusion Map (finding 3 dominant eigenvectors and mapping data points onto the eigenspace)
- **Step 2** Diffusion Net (using NNs to learn the mapping from data space to eigenspace)
- **Step 3** Learn Manifold on the Embedding Space using NNs
- **Step 4** Learn CVs that are Orthogonal to the Manifold
- **Step 5** Compute Transition Rates

Proposed Approach for Learning CVs and Testing Them

Step 1: Target Measure Diffusion Map[1]

Goal: Learn a low-dimensional embedding of the original high-dimensional data space.
Algorithm (given points $\{x_i\}_{i=1}^n$):

- Construct kernel matrix K_ϵ where $(K_\epsilon)_{ij} = k_\epsilon(x_i,\ x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\epsilon}\right)$
- Compute kernel density estimate $q_\epsilon(x_i) = \sum_{j=1}^n (K_\epsilon)_{ij}$
- Construct diagonal matrix $D_{\epsilon,\pi}$ where $(D_{\epsilon,\pi})_{ii} = \pi^{1/2}(x_i) q_\epsilon^{-1}(x_i)$
- Normalization: $K_{\epsilon,\pi} = K_\epsilon D_{\epsilon,\pi}$
- Construct diagonal matrix $\tilde{D}_{\epsilon,\pi}$ where $(\tilde{D}_{\epsilon,\pi})_{ii} = \sum_{j=1}^n (K_{\epsilon,\pi})_{ij}$
- Compute operator for the diffusion process $L_{\epsilon,\pi} = \frac{1}{\epsilon}(\tilde{D}_{\epsilon,\pi}^{-1} K_{\epsilon,\pi} - I)$

To project points from the original high-dimensional space $\mathbb{R}^D$ to the low-dimensional embedding space $\mathbb{R}^d$, we take the first d eigenvectors of $L_{\epsilon,\pi}$. We denote the low-dimensional embedding of each $x_i \in \mathbb{R}^D$ as $\Psi(x_i) \in \mathbb{R}^m$.

Step 2: Diffusion Net[2]

Goal: Learn a neural network (encoder) $f_e : \mathbb{R}^D \to \mathbb{R}^m$ that approximates the function from the original high-dimensional data space to the low-dimensional embedding space given by the Target Measure Diffusion Map. the neural f_e, lets us perform out-of-sample extensions.
Training Input: $\{x_i\}_{i=1}^n \in \mathbb{R}^D$
Target: $\Psi(x_i) \in \mathbb{R}^m$
Loss Function:

$$L = \frac{\theta_1}{2n}\sum_{i=1}^n \|f_e(x_i) - \Psi(x_i)\|_2^2 + \frac{\theta_2}{2}\sum_{l=1}^{M-1}\|W^{(l)}\|_F^2$$

M: number of layers in the neural network
θ_1, θ_2: weights of mean-squared loss & regularization term, respectively, in the loss function.

Step 3: Learning the Manifold on Which the Dynamics Lives

Goal: We decompose our potential as $V(x) = V_0(x) + \frac{1}{\epsilon}V_1(x)$, where $\{x \mid V_1(x) = 0\}$ is the manifold on which the dynamics of the system lives. In other words, V_0, V_1 are the potentials of the 'fast' & 'slow' processes, respectively. We want to learn $\psi : \mathbb{R}^m \to \mathbb{R}$ such that $\psi^2(f_e^1(x), ..., f_e^m(x)) = 0$ and hence our 'fast' process potential $V_1(x) \sim \psi^2(f_e(x))$.
Training Input: $\{f_e(x_i)\}_{i=1}^n \in \mathbb{R}^m$ and regularization point cloud $\{d_j\}_{j=1}^p \in \mathbb{R}^m$ surrounding $\{f_e(x_i)\}_{i=1}^n$
Loss Function:

$$L = \theta_1 \cdot \sum_{i=1}^n \psi^2(f_e(x_i)) + \theta_2 \cdot \sum_{j=1}^p \frac{1}{\psi^2(d_j)}$$

θ_1, θ_2: weights of mean-squared loss for manifold points and point cloud, respectively, in the loss function.

Step 4: Learn CVs[3]

Goal: We want to learn CVs $\xi_i(x)$ by imposing the following conditions:
Orthogonality of CVs to V_1 (the fast potential): $\|J(\xi)(x) \cdot \nabla V_1(x)\|_F^2 = 0$
Orthogonality of CVs to each other: $\|J(\xi)(x) \cdot J(\xi)(x)^T - I\|_F^2 = 0$

Neural Network

Training Input: $\{x_i\}_{i=1}^n \in \mathbb{R}^D$
Loss Function:

$$L = \frac{\theta_1}{2n}\sum_{i=1}^n \|J(\xi)(x_i) \cdot \nabla V_1(x_i)\|_F^2 + \frac{\theta_2}{2n}\sum_{i=1}^n \|J(\xi)(x_i) \cdot J(\xi)(x_i)^T - I\|_F^2 + \frac{\theta_3}{2}\sum_{l=1}^{M-1}\|W^{(l)}\|_F^2$$

M: number of layers in the neural network
$\theta_1, \theta_2, \theta_3$: weights of the two orthogonality condition losses & the regularization term, respectively, in the loss function

Step 5: Evaluating CVs[4]

Goal: We evaluate our learned CVs through computing the diffusion tensor, the free energy, and approximating the backward Kolmogorov operator for our SDE. We run dynamics (Euler-Maruyama) with a new potential using CVs to obtain points $\{y_{t_k}\}_{k=1}^n \in \mathbb{R}^D$.
New Potential: $U(y; \kappa, z) = V(y) + \sum_{i=1}^d \frac{\kappa}{2}(\xi_i(y) - z_i)^2$, where κ is a large spring constant.
Dynamics for Computing the Diffusion Tensor: $dy = -\nabla U dt + \sqrt{2\beta^{-1}}dW$
Diffusion Tensor $M(x)$ for $x = \xi(y)$: $M_{ij}(x) = \frac{1}{n}\sum_{k=1}^n \sum_{l=1}^D \frac{\partial \xi_i(y_k)}{\partial y_l}\frac{\partial \xi_j(y_k)}{\partial y_l}$, where y_l are atomic coordinates in the original data space
Free Energy: $\nabla F(x) = \frac{\kappa}{n}\sum_{k=1}^n (x - \xi(y_{t_i}))$
SDE: $dx_t = (-M(x_t)\nabla F(x_t) + \beta^{-1}\nabla \cdot M(x_t))dt + \sqrt{2\beta^{-1}}M(x_t)^{\frac{1}{2}}dW$
Backward Kolmogorov Operator: $\mathcal{L}f = \beta^{-1}e^{\beta F}\nabla \cdot (e^{-\beta F}M\nabla f)$
The dominant eigenvalues of $\mathcal{L}$ approximate the transition rates, which can be compared to true rates to evaluate CVs (what we currently are working on).

Learning CVs for 2D Lennard-Jones System (LJ7)

Background

The 2D LJ7 system is 14-dimensional. There are 7 particles, each of which has two coordinates (x_i, y_i), respectively. Existing literature has proposed two collective variables, $\mu_2(x_1, y_1, ..., x_7, y_7)$ and $\mu_3(x_1, y_1, ..., x_7, y_7)$, corresponding to the second and third central moments of smooth functions approximating the number of nearest neighbors of particle i, to represent the dynamics of the system. We aim to use our proposed approach to improve upon existing CVs for the system.
The system has four distinct metastable states corresponding to four potential energy minima: hexagon (purple), trapezoid (yellow), capped parallelogram 1 (bright green), and capped parallelogram 2 (blue). The colors in brackets correspond to the colors of the four metastable states, respectively, mapped in μ_2 vs μ_3 space, as shown on the right side of Figure 1 below.

Learned CVs

Following our approach aforementioned, we learned 2 CVs $\xi(x)$ with $\xi : \mathbb{R}^{14} \to \mathbb{R}^2$. On the left side of Figure 1, we map data points $\{x_i\}_{i=1}^n \in \mathbb{R}^{14}$ in the 2D CV space. The right side of Figure 1 is the reference figure for the colors of four metastable states in μ_2 vs μ_3 space. Notice that the four metastable states are clearly separated by the 2CVs, while separate clusters with the same color represent different particle configurations with the same metastable state.

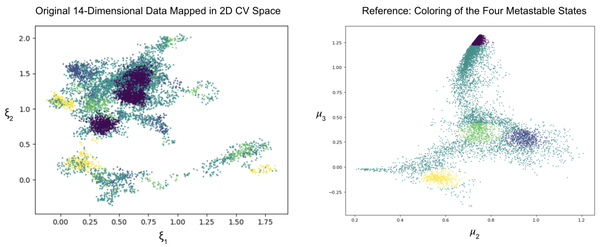

Figure 1. 2D machine learned Collective Variables, ξ_1 and ξ_2, for LJ7

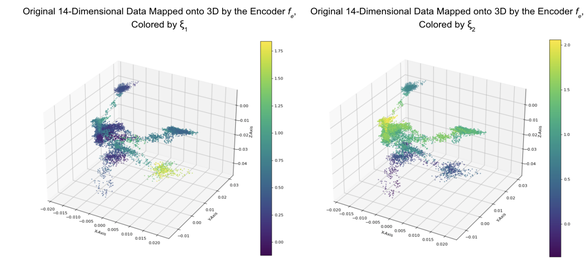

Figure 2. Encoder-Transformed Points Colored by ξ_1 (CV1) and ξ_2 (CV2), Respectively

In the above figure, we see that the coloring by ξ_1 and ξ_2 on the encoder transformed manifold are approximately orthogonal to each other, satisfying one of the criteria of Step 4.

Learning CVs for the Butane System

We use the atomic coordinates of the 4 carbon atoms as input, $x \in \mathbb{R}^{12}$, obtained through enhanced sampling methods (metadynamics, delta-net) using the openMM library.
Physically motivated CVs The dihedral angle of the butane system is considered a representative 1D collective variable. We use this to visual validation our machine learned CVs.
CV dimensions : The manifold obtained in step 3 suggests that the dynamics of the system are represented by a 2D variable, hence we learn $\xi : \mathbb{R}^{12} \to \mathbb{R}^2$, using the above methodology.

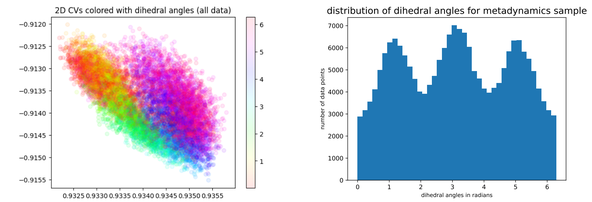

Figure 3. 2D machine learned collective variables ξ_1 and ξ_2, for butane

The distribution of the dihedrals suggest that the meta-stable states coincide with the angles: $\left\{\frac{\pi}{3}, \pi, \frac{5\pi}{3}\right\}$. We can see that the machine learned collective variables separate these meta-stable states. Further, if we color the manifold learned from step 3 with our CVs we observe that they are clearly orthogonal, confirming their uniqueness.

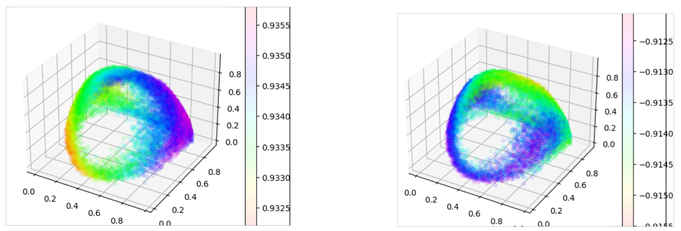

Figure 4. Encoder-Transformed Points Colored by ξ_1 (CV1) and ξ_2 (CV2), Respectively

References

[1] Banisch, Trstanova, Bittracher, Klus, Koltai, 2018, Diffusion maps tailored to arbitrary non-degenerate Ito processes

[2] Mishne, Shaham, Cloninger, Cohen, 2015, Diffusion Nets

[3] Legoll, Lelievre, 2009, Effective dynamics using conditional expectations

[4] Maragliano, Fischer, Vanden-Eijnden, Ciccotti, 2006, String method in collective variables: Minimum free energy paths and isocommittor surfaces

arnavmehta@berkeley.edu

NSF REU grant DMS-2149913

yrl4@cornell.edu