A Data Driven Method for Computing Quasipotentials

Bo Lin

Department of Mathematics

Joint work with Qianxiao Li and Weiqing Ren

Outline

Dynamical system

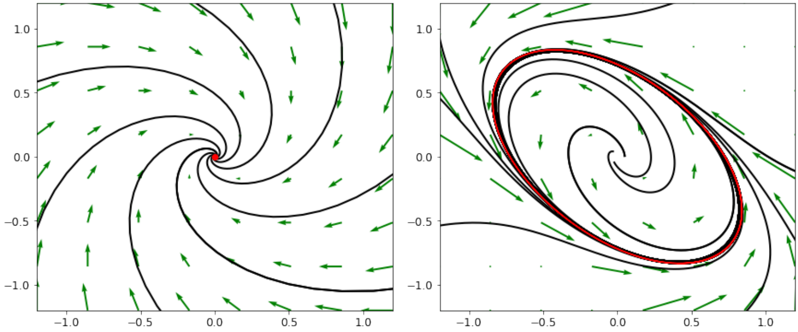
Consider the process $\mathbf{x}_t \in \mathbb{R}^d$ modeled by the stochastic differential equation (SDE):

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t)dt + \sqrt{\epsilon}d\mathbf{W}_t, \quad t > 0. \tag{1}$$

- $\mathbf{f}$: force field.
- $0 < \epsilon \ll 1$: amplitude of the noise.
- $\mathbf{W}_t$: standard Brownian motion.

Attractor

Let A be an attractor of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$.
It can be a stable equilibrium point (*left*) or a limit cycle (*right*).



Quasipotential

The quasipotential with respect to the attractor A is defined as

$$U_A(\mathbf{x}) = \inf_{T>0} \inf_{\varphi} \int_0^T \frac{1}{2} |\dot{\varphi} - \mathbf{f}(\varphi)|^2 \, dt, \tag{2}$$

where φ is a path connecting the attractor A and the state $\mathbf{x}$.

- Quasipotential is defined in the state space. (usually in high-d)
- It is the "energy" needed for the system to transit from A to $\mathbf{x}$ when the noise is small.

Why do we care about quasipotential?

Quasipotential can be used to[1]

- identify the maximum likelihood path from A to another state: the tangent of the path is parallel to $\mathbf{f} + \nabla U_A$.
- estimate the expected exit time τ from A:

$$\lim_{\epsilon \to 0} \epsilon \log \mathbb{E}[\tau] = \min_{\mathbf{x} \in \partial \mathcal{B}(A)} U_A(\mathbf{x}),$$

where $\mathcal{B}(A)$ is the basin of A.

[1] Freidlin and Wentzell (2012)

Previous methods

Mesh-based methods[2]

- compute the quasipotential on 2D or 3D meshes.
- limited to low-d systems.

Path-based methods[3] (minimum action method (MAM), adaptive MAM and geometric MAM)

- give quasipotential along the minimum action path.
- expensive when computing quasipotential landscape for high-d systems.

Curse of dimensionality!

[2] M. K, Cameron (2012); D. Dahiya and M. Cameron (2018); S. Yang, F. P. Samuel, and K. C. Maria (2019).

[3] W. E, W. Ren, and E. Vanden-Eijnden (2004); X. Zhou, W. Ren, and W. E (2008); M. Heymann and E. Vanden-Eijnden (2008).

Characterization of quasipotential

Quasipotential can be characterized by a decomposition of the force field:

$$\mathbf{f}(\mathbf{x}) = -\nabla V(\mathbf{x}) + \mathbf{g}(\mathbf{x}), \quad \text{with } \nabla V(\mathbf{x})^T \mathbf{g}(\mathbf{x}) = 0, \tag{3}$$

where the term $-\nabla V(\mathbf{x})$ is referred to as the potential component of $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ as the rotational component.

- The function $2V$ coincides with the quasipotential up to an additive constant.

Characterization of quasipotential (cont'd)

Theorem 1 (Freidlin and Wentzell)

Suppose the vector field $\mathbf{f}$ *has the orthogonal decomposition and* V *attains its strict local minimum at a point or limit cycle, denoted by* A*. If there is a bounded domain* $\mathcal{D}$ *containing* A *such that*

- V *is continuously differentiable in* $\mathcal{D} \cup \partial\mathcal{D}$*;*
- $V(\mathbf{x}) > V(A)$ *and* $\nabla V(\mathbf{x}) \neq 0$ *for all* $\mathbf{x} \in \mathcal{D} \cup \partial\mathcal{D}$ *and* $\mathbf{x} \notin A$*,*

then the quasipotential of the system with respect to the attractor A *in the set* $\{\mathbf{x} \in \mathcal{D} \cup \partial\mathcal{D} : V(\mathbf{x}) \leq \min_{\mathbf{y} \in \partial\mathcal{D}} V(\mathbf{y})\}$ *coincides with* $2V(\mathbf{x})$ *up to an additive constant.*

Problem setup

Given trajectory data, learn the force field in the form of the orthogonal decomposition.

- The force field $\mathbf{f}$ is not explicitly known.
- Data-driven: we learn the force field and the quasipotential from the data.

Method: Parameterization

Orthogonal decomposition:

$$\mathbf{f}(\mathbf{x}) = -\nabla V(\mathbf{x}) + \mathbf{g}(\mathbf{x}), \quad \nabla V(\mathbf{x})^T \mathbf{g}(\mathbf{x}) = 0,$$

- The two components in the decomposition are represented by **neural networks**.

Method: Parameterization (cont'd)

- Parameterization of V:

$$V_\theta(\mathbf{x}) = \hat{V}_\theta(\mathbf{x}) + |\mathbf{x}|^2,$$

$\hat{V}_\theta$: fully connected neural network with activation $\texttt{tanh}$.

- Parameterization of $\mathbf{g}$ by a neural network $\mathbf{g}_\theta$ with continuously differentiable activation (e.g. $\texttt{tanh}(z)$ or $\mathrm{ReLU}^2(z)$).

The parameterized force field is given by

$$\mathbf{f}_\theta(\mathbf{x}) = -\nabla V_\theta(\mathbf{x}) + \mathbf{g}_\theta(\mathbf{x}).$$

Method: Trajectory data

The observation dataset

$$X = \{X_i(t_j), X_i(t_j + \Delta t) : \ i = 1, \dots, N, \ j = 0, \dots, M\}$$

contains N trajectories of the deterministic system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$$

where $X_i(t)$ denotes the i^{th} trajectory. Along each trajectory, $2M + 2$ data points are sampled at the times

$$t_0, t_0 + \Delta t, t_1, t_1 + \Delta t, ..., t_M, t_M + \Delta t,$$

where $t_0 < t_1 < ... < t_M$ and Δt is a small time step.

Method: Loss function

We take the loss function of the form:

$$L = L^{dyn} + \lambda L^{orth}.$$

- L^{dyn} is to reconstruct the dynamics as given by the trajectory data.
- L^{orth} is to impose the orthogonality condition.
- λ is a parameter.

Method: Loss function (cont'd)

$$L^{dyn} = \frac{1}{N(M+1)} \sum_{i=1}^{N} \sum_{j=0}^{M} \bar{h} \left(\frac{1}{\Delta t} \left(\mathcal{I}_{\Delta t}[\mathbf{f}_\theta; X_i(t_j)] - X_i(t_j + \Delta t) \right); \delta_1 \right),$$

- $\mathcal{I}_{\Delta t}$ is a numerical integrator with time step Δt.
- $\bar{h}(\mathbf{e}; \delta_1)$ denotes the mean Huber loss.

$$L^{orth} = \frac{1}{S} \sum_{i=1}^{S} w \left(\frac{\nabla V_\theta(\tilde{X}_i)^T \mathbf{g}_\theta(\tilde{X}_i)}{|\nabla V_\theta(\tilde{X}_i)| \cdot |\mathbf{g}_\theta(\tilde{X}_i)|}; \delta_2 \right),$$

- $w(y; \delta_2) = y^2 I_{y>0} + \delta_2 y^2 I_{y<0}$ with $\delta_2 = \frac{1}{10}$.
- $\tilde{X}_1, \ldots, \tilde{X}_S$ are representative data points sampled from X.

Numerical examples

- Adam optimizer and mini-batch of size 5000.
- The learning rate exponentially decays.
- Two hidden layers in the neural networks.

Numerical example: 3D system

We consider the following system in three-dimensional space

$$\frac{dx}{dt} = -2(x^3 - x) - (y + z),$$
$$\frac{dy}{dt} = -y + 2(x^3 - x),$$
$$\frac{dz}{dt} = -z + 2(x^3 - x).$$

- This system has two stable equilibrium points: $\mathbf{x}_a = (-1, 0, 0)$ and $\mathbf{x}_b = (1, 0, 0)$.
- The quasipotential is given by

$$U(x, y, z) = (1 - x^2)^2 + y^2 + z^2.$$

Numerical example: 3D system

- The two neural networks $\tilde{V}_\theta$: 2-50-50-1 ($\mathtt{tanh}$) and $\mathbf{g}_\theta$: 2-50-50-2 ($\mathtt{tanh}$).
- The dataset contains 2×10^5 data points $(2,000$ trajectories).

The relative root mean square error (rRMSE) and the relative mean absolute error (rMAE) for the learned quasipotential $U_\theta(\mathbf{x})$ are 0.0037 and 0.0017, respectively.

Numerical example: 3D system

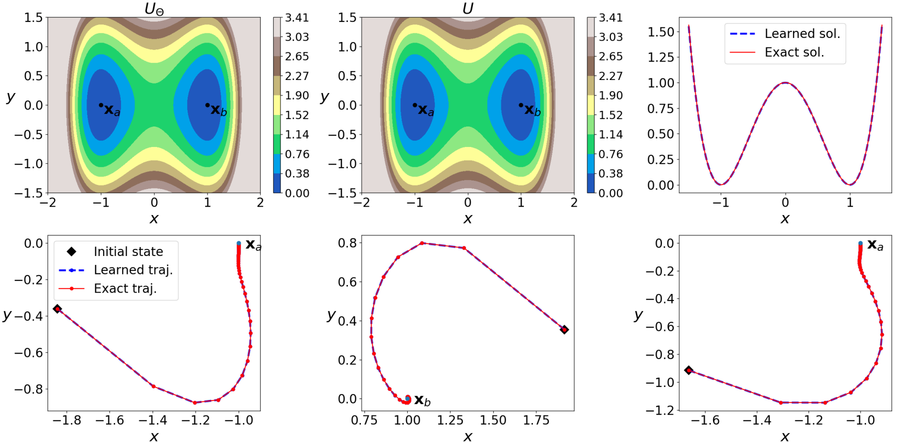


Figure: **Upper**: U_θ (left) and exact quasipotential U (middle) with $z = 0$ and along the line $y = z = 0$ (right). **Lower**: Trajectories of the learned and the original dynamics.

Numerical example: High-d system from discretized PDE

Consider the Ginzburg-Landau equation

$$\begin{cases} u_t = \delta u_{xx} - \delta^{-1} V'(u), \quad x \in [0,1], \\ u(0,t) = u(1,t) = 0, \\ u(x,0) = u^0(x) \end{cases}$$

where $V(u) = \frac{1}{4}(1-u^2)^2$ is double-well potential and $\delta = 0.1$.

Numerical example: High-d system

By discretizing the interval $[0, 1]$ with a uniform mesh, we obtain a high-dimensional system

$$\frac{du_i}{dt} = \delta \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} - \delta^{-1} V'(u_i), \quad 1 \leq i \leq I - 1,$$

with $u_0 = u_I = 0$. The state of the system is denoted by

$$\mathbf{u} = (u_1, \ldots, u_{I-1}).$$

The quasipotential is given by

$$E_h[\mathbf{u}] = \sum_{i=1}^{I} \frac{1}{2} \delta \left(\frac{u_i - u_{i-1}}{h} \right)^2 + \delta^{-1} V(u_i).$$

Numerical example: High-d system

- Take $I = 51$.
- The two neural networks $\tilde{V}_\theta$: 50-100-100-1 ($\mathtt{tanh}$) and $\mathbf{g}_\theta$: 50-100-100-50 (ReLU^2).
- The dataset contains 2×10^6 data points ($10,000$ trajectories).

Numerical example: High-d system

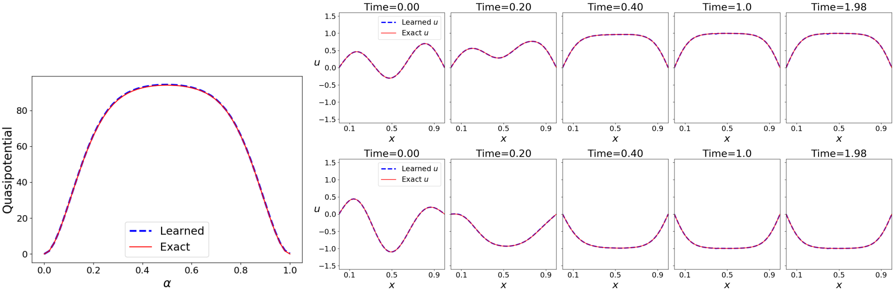


Figure: **Left**: U_θ and U along the MEP. **Right**: Two trajectories from learned dynamics vs original dynamics.

Summary

- We proposed a method for computing the quasipotential and at the same time learning the dynamics from the trajectory data.
- The method is **data-driven**.
- It is an efficient method to map the **landscape** of the quasipotential in high dimensions.