Martingale Neural Networks for High-Dimensional Stochastic Optimal Controls

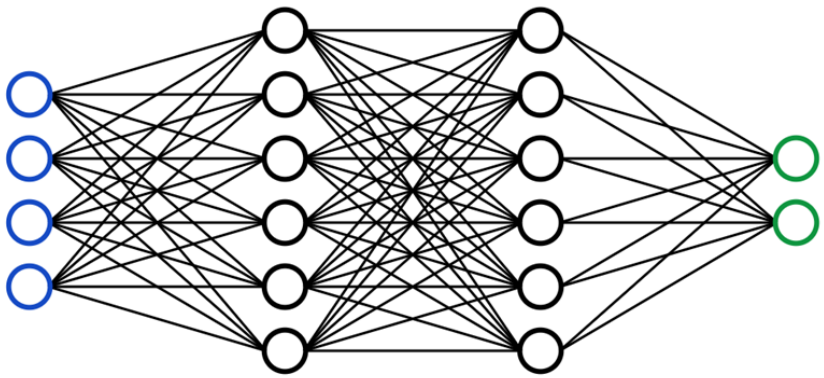
Wei Cai

Department of Mathematics
Southern Methodist University

Extreme and Rare Events: Modeling, Algorithms, and Applications
Sep 26 - 27, 2026
ICERM, Brown University

Joint work with Shuixin Fang (CAS),
Wenzhong Zhang (USTC), and Tao Zhou (CAS)

Neural network as a high-dimensional function



Outline

Outline

State equation:

$$X_t^u = x_0 + \int_0^t \bar{\mu}\left(s, X_s^u, u(s, X_s^u)\right) \mathrm{d}s + \int_0^t \bar{\sigma}\left(s, X_s^u, u(s, X_s^u)\right) \mathrm{d}B_s, \quad t \in [0, T], \tag{1}$$

where u is the admissible **feedback control**

Cost functional:

$$J(u) := \mathbb{E}\left[\int_0^T c\left(s, X_s^u, u(s, X_s^u)\right) \mathrm{d}s + g(X_T^u)\right], \quad u \in \mathcal{U}_{\text{ad}} \tag{2}$$

with $c : [0, T] \times \mathbb{R}^d \times U \to \mathbb{R}$ and $g : \mathbb{R}^d \to \mathbb{R}$.

Stochastic optimal control problem (SOCP):

$$\text{Find } \; u^* \in \mathcal{U}_{\text{ad}} \; \text{ s.t. } \; J\left(u^*\right) = \inf_{u \in \mathcal{U}_{\text{ad}}} J\left(u\right). \tag{3}$$

Value function:

$$v(t,x) := \inf_{u \in \mathcal{U}_{\text{ad}}} \mathbb{E}\left[\int_t^T c\left(s, X_s^u, u(s, X_s^u)\right) \mathrm{d}s + g(X_T^u) \middle| X_t^u = x\right]$$

for $(t,x) \in [0,T] \times \mathbb{R}^d$.

$$\begin{cases} \partial_t v(t,x) + \inf_{\kappa \in U} H\left(t, x, \kappa, \partial_x v(t,x), \partial_{xx}^2 v(t,x)\right) = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d, \\ v(T,x) = g(x), \quad x \in \mathbb{R}^d. \end{cases} \tag{4}$$

Classical **verification theorem** [Yong and Zhou, 1999]:
if $v \in C^{1,2}$, then the optimal control u and its trajectory $X^* := X^u$ satisfy

$$\begin{aligned} & H\left(t, X_t^*, u(t, X_t^*), \partial_x v(t, X_t^*), \partial_{xx}^2 v(t, X_t^*)\right) \\ = & \inf_{\kappa \in U} H\left(t, X_t^*, \kappa, \partial_x v(t, X_t^*), \partial_{xx}^2 v(t, X_t^*)\right), \end{aligned} \tag{5}$$

where H is the Hamiltonian given by

$$H(t,x,\kappa,z,p) := \frac{1}{2} \operatorname{Tr}\left(\bar{\sigma}\bar{\sigma}^\top(t,x,\kappa)p\right) + \bar{\mu}^\top(t,x,\kappa)z + c(t,x,\kappa) \tag{6}$$

for $(t,x,\kappa,z,p) \in [0,T] \times \mathbb{R}^d \times U \times \mathbb{R}^d \times \mathbb{R}^{d \times d}$.

The **curse of dimensionality** lies in

- High dimensionality of $x \in \mathbb{R}^d$ and $U \subset \mathbb{R}^m$;
- Optimization problem $\inf_{\kappa \in U} H$ for each (t, x).

$$\begin{cases} \partial_t v(t,x) + \inf\limits_{\kappa \in U} H\left(t, x, \kappa, \partial_x v(t,x), \partial^2_{xx} v(t,x)\right) = 0, & (t,x) \in [0,T) \times \mathbb{R}^d, \\ v(T,x) = g(x), & x \in \mathbb{R}^d. \end{cases} \tag{7}$$

HJB equations and rare event simulation

[Dupuis, Spiliopoulos, and Wang, 2012],[Vanden-Eijnden and Weare, 2012], etc

Previous neural network results on Stochastic Optimal Controls:

- **Direct optimization** using DNN [Han and E, 2016]
- **Dynamic programming**, Hybrid-now method [Huré et al., 2021].
- **SDE based methods**: Actor-critic method [Zhou, Han, and Lu, 2021]. Stochastic maximum principle (SMP) & deep learning [Ji, Peng, Peng, and Zhang, 2022], [Nüsken and Richter, 2021], [Li, Verma, , and Ruthotto, 2024].

Selected other methods for HJB equations (deterministic and stochastic controls, i.e., hyperbolic vs parabolic PDEs)

Darbon and Osher, Resr. math sci, 2016; Darbon, Meng, JCP, 2021; (SDGD) Hu, Shukla, Karniadakis, and K. Kawaguchi, Neural Networks, 2024; Nakamura-Zimmerer, Gong, Kang, SISC, 2021; Dolgov, Kalise,and Kunisch, SISC, 2021, and others

Related work I: Deep learning for the discrete time SOCP:

$$\min_{u\in\mathcal{U}_{\text{ad}}} J(u), \quad J(u) := \mathbb{E}\left[\sum_{n=0}^{N-1} c_n\left(X_n, u_n\right) + g(X_N)\right],$$
$$X_{n+1} = F(X_n, u_n, \xi_{n+1}), \quad n = 0, 1, \cdots, N-1,$$

where the function F is known; $u_n = u_n(X^n)$ is the state feedback control; $\{\xi_0, \cdots, \xi_{N-1}\}$ are i.i.d. random noise.

Direct optimization based on DNN approximations [Han and E, 2016]:

$$u_n(X_n) \approx u_n(X_n; \theta_n), \qquad \min_{(\theta_n)_{n=0}^{N-1}} \mathbb{E}\left[\sum_{n=0}^{N-1} c_n\left(X_n, u_n\left(X_n; \theta_n\right)\right) + g(X_N)\right].$$

Dynamic programming principle: Let

$$J_n(x;u) := \mathbb{E}\left[\sum_{m=n}^{N-1} c_m\left(X_m, u_m\right) + g(X_N)\middle| X_n = x\right], \quad x \in \mathbb{R}^d.$$

For the optimal control $u^* = (u_n^*)_{n=0}^{N-1}$,

$$J_n\left(x;u^*\right) = \mathbb{E}\left[c_n\left(x, u_n^*(x)\right) + J_{n+1}\left(F(x, u_n^*(x), \xi_{n+1}); u^*\right)\middle| X_n = x\right], \tag{8}$$

$$u_n^*(x) = \arg\min_{\kappa \in U} \mathbb{E}\left[c_n\left(x,\kappa\right) + J_{n+1}\left(F(x,\kappa,\xi_{n+1}); u^*\right)\middle| X_n = x\right]. \tag{9}$$

Hybrid-now method [Huré et al., 2021]: For $n = N-1, N-2, \cdots, 0$,

1. Approximate $x \mapsto u_n^*(x)$ and $x \mapsto J_n(x;u^*)$ by DNNs.
2. Train DNNs by enforcing (8) and (9).

Related work II: Actor-critic method [Zhou, Han, and Lu, 2021].

Zhou, Han, and Lu [2021] focuses the **elliptic problem**

$$\begin{cases} \gamma v(x) - \inf\limits_{\kappa \in U} \left\{ \frac{1}{2} \operatorname{Tr}\left(\bar{\sigma}\bar{\sigma}^\top(x,\kappa)\partial^2_{xx} \right) v(x) + \bar{\mu}^\top(x,\kappa)\partial_x v(x) + f(x,\kappa) \right\} = 0, \quad x \in D, \\ v(x) = g(x), \quad x \in \partial D, \end{cases}$$

where $D \subset \mathbb{R}^d$ and $\gamma > 0$. The dynamic programming reveals that

$$v(x) = \inf_{u \in \mathcal{U}_{\text{ad}}} J(x;u), \quad J(x;u) = \mathbb{E}\left[\int_0^\tau f(X_s^u, u(X_s^u)) e^{-\gamma s} \, \mathrm{d}s + e^{-\gamma\tau} g(X_\tau^u) \,\middle|\, X_0^u = x \right]$$

with $\tau := \inf\{t > 0 : X_t \notin D\}$ and

$$X_t^u = X_0^u + \int_0^t \bar{\mu}(X_s^u, u(s, X_s^u)) \, \mathrm{d}s + \int_0^t \bar{\sigma}(X_s^u, u(s, X_s^u)) \, \mathrm{d}B_s, \quad t \in [0, \infty).$$

Critic: For given $u \in \mathcal{U}_{\text{ad}}$, if $v(x) = J(x;u)$, then Itô's formula implies

$$e^{-\gamma(T\wedge\tau)}v(X^u_{T\wedge\tau}) = v(X^u_0) - \int_0^{T\wedge\tau} f(X^u_s, u(X^u_s))e^{-\gamma s}\,\mathrm{d}s + \int_0^{T\wedge\tau} \partial_x v\sigma(X^u_s, u(X^u_s))e^{-\gamma s}\,\mathrm{d}B_s,$$

which leads to the loss function for v and $z := \partial_x v$ as

$$L_1(v,z;u) := \mathbb{E}\left[\left|\int_0^{T\wedge\tau} f(X^u_s, u(X^u_s))e^{-\gamma s}\,\mathrm{d}s - \int_0^{T\wedge\tau} z\sigma(X^u_s, u(X^u_s))e^{-\gamma s}\,\mathrm{d}B_s + e^{-\gamma(T\wedge\tau)}v(X^u_{T\wedge\tau}) - v(X^u_0)\right|^2\right], \quad X^u_0 \sim \mu.$$

Actor: Given v, the optimal control u can be found by minimizing

$$\tilde{J}(u;v) := \mathbb{E}\left[\int_0^{T\wedge\tau} f(X^u_s, u(X^u_s))e^{-\gamma s}\,\mathrm{d}s + e^{-\gamma\tau}v(X^u_{T\wedge\tau})\right]. \tag{10}$$

Key point: If $v(x) = J(x;u)$ for $x \in \mathbb{R}^d$, then

$$\tilde{J}(u;v) = J(X^u_0;u) \text{ for any } T > 0.$$

Actor-critic method:

1. Introduce neural networks $(u_\alpha, v_\theta, z_\eta) \approx (u, v, \partial_x v)$.
2. **Critics:** Sample the trajectories of X^{u_α} and update

$$(\theta, \eta) \leftarrow (\theta, \eta) - \delta \nabla_{(\theta,\eta)} L_1(v_\theta, z_\eta; u_\alpha)$$

3. **Actor:** Sample the trajectories of X^{u_α} and update

$$\alpha \leftarrow \alpha - \delta \nabla_\alpha \tilde{J}(u_\alpha; v_\theta).$$

4. Repeat the steps 2 and 3 until the training converges.

Related work III: Stochastic maximum principle (SMP) & deep learning [Ji, Peng, Peng, and Zhang, 2022].

SMP: If the tuple (u^*, X^*) is the optimal control and trajectory, then the quadruple (u^*, X^*, p^*, q^*) is the solution to

$$\begin{cases} \mathrm{d}X_t^* = \partial_p H(t, X_t^*, u_t^*, -p_t^*, -q_t^*)\,\mathrm{d}t + \partial_q H(t, X_t^*, u_t^*, -p_t^*, -q_t^*)\,\mathrm{d}B_t, \\ \mathrm{d}p_t^* = \partial_x H(t, X_t^*, u_t^*, -p_t^*, -q_t^*)\mathrm{d}t + q_t^*\ \mathrm{d}B_t, \\ X_0^* = x_0, \quad p_T^* = -\partial_x g(X_T^*), \\ H(t, X_t^*, u_t^*, -p_t^*, -q_t^*) = \min\limits_{\kappa \in U} H(t, X_t^*, \kappa, -p_t^*, -q_t^*) \end{cases} \tag{11}$$

for $t \in [0, T]$, where H is the SMP Hamiltonian.

The **main idea** of Ji, Peng, Peng, and Zhang [2022]:

Algorithm 1: Apply the **DeepBSDE** [E et al., 2017, Han et al., 2018] to the FBSDEs in (11a)-(11c), where

- q_t^* is approximated by a neural network $q_\eta\,(t, X_t^*)$;
- u_t^* is solved by minimizing $H(t, X_t^*, \kappa, -p_t^*, -q_t^*)$ over $\kappa \in U$.

Algorithm 2:

- The condition (11d) is fulfilled by appending a penalty $G(\alpha)$ to the loss function:

$$G(\alpha) := \lambda \int_0^T \mathbb{E}\left[\left|\partial_u H\left(t, X_t^*, u_\alpha(t, X_t^*), -p_t^*, -q_t^*\right)\right|^2\right] \mathrm{d}t, \quad \lambda > 0.$$

  Typical requirement: u is unconstrained and $u \mapsto H$ is convex.

The trajectories of (X^*, p^*) depend on (u_α, q_η), which need to be updated at each training step.

Outline

Derivative-Free Martingale Network (SOC-MartNet)

SOC-MartNet - martingale networks for SOC problem: Using Varadhan martingale problem to convert PDEs problem to a martingale problem of SDEs; sampling SDEs paths to enforce martingale conditions.

Main Aims:

- Addressing cost of updating paths of system SDE under controls
- Avoid computing derivatives of value function $v(t,x)$, i.e. Hessian, in HJB and Optimization of Hamiltonian

SOC-MartNet: Martingale networks for SOC:

$$\boxed{\text{Parabolic PDE}} \qquad \boxed{\inf_{\kappa\in U} H\left(t,x,\kappa,\partial_x v(t,x),\partial^2_{xx} v(t,x)\right)}$$

$$\Downarrow \text{Itô's formula} \qquad \Downarrow \text{Find feedback control}$$

$$\text{Martingale problem} \qquad u(t,x) := \arg\min_{\kappa\in U} H\left(t,x,\kappa,\partial_x v(t,x),\partial^2_{xx} v(t,x)\right)$$

$$\Downarrow \mathbb{E}\left[\cdot|X_t\right] \text{ is a projection} \qquad \Downarrow \text{Neural network } u_\alpha \approx u$$

$$\boxed{\textbf{Galerkin} \text{ Adversarial learning} \;+\; \min_{\alpha} \int_0^T \mathbb{E}\left[\bar{H}(t,X_t,u_\alpha(t,X_t))\right]\,\mathrm{d}t}$$

Using the structure of H, the HJB equation can be rewritten into

$$\partial_t v(t,x) + \inf_{\kappa \in U} \{\mathcal{L}^\kappa v(t,x) + c(t,x,\kappa)\} = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d \tag{12}$$

with

$$\mathcal{L}^\kappa := \bar{\mu}(t,x,\kappa)\partial_x + \frac{1}{2}\,\mathrm{Tr}\left\{\bar{\sigma}\bar{\sigma}^\top(t,x,\kappa)\,\partial^2_{xx}\right\}.$$

By introducing the optimal feedback control u,

$$u(t,x) := \arg\min_{\kappa \in U} \{\mathcal{L}^\kappa v(t,x) + c\,(t,x,\kappa)\}\,, \quad (t,x) \in [0,T) \times \mathbb{R}^d, \tag{13}$$

the HJB equation (12) becomes a **linear** PDE as

$$\partial_t v(t,x) + \mathcal{L}^u v(t,x) + c\,(t,x,u(t,x)) = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d. \tag{14}$$

The linear operator $\partial_t + \mathcal{L}^u$ is involved in the Itô formula.

For the controlled state process X^u given in (1), the Itô formula leads to

$$v(t, X_t^u) = v(0, X_0) + \int_0^t (\partial_t + \mathcal{L}^u) v(s, X_s^u)\, \mathrm{d}s + \int_0^t \nabla v(s, X_s^u) \sigma\, \mathrm{d}B_s \tag{15}$$

$$= v(0, X_0) - \int_0^t c\,(s, X_s^u, u_s)\ \mathrm{d}s + \int_0^t \nabla v(s, X_s^u) \sigma\, \mathrm{d}B_s \tag{16}$$

which implies a martingale

$$\mathcal{M}_t^{u,v} :=\ v(t, X_t^u) + \int_0^t c\,(s, X_s^u, u_s)\ \mathrm{d}s \tag{17}$$

with $u_s := u(s, X_s^u)$.

Martingale formulation for (14)**:** under some usual conditions,

$$\mathcal{M}^{u,v} \text{ is a } \mathbb{F}^B\text{-martingale} \ \Leftrightarrow\ (\partial_t + \mathcal{L}^u)\, v(t, X_t^u) + c\,(t, X_t^u, u(t, X_t^u)) = 0, \ \text{ a.s..} \tag{18}$$

Increment of the Martingale and PDE residual

Using the Itô formula, we have

$$\begin{aligned}\mathcal{M}_{t+\Delta t}^{u,v}-\mathcal{M}_t^{u,v} =& v(t+h,X_{t+h}^u)-v(t,X_t^u)+\int_t^{t+h} c\left(s,X_s^u,u_s\right)\,\mathrm{d}s\\ =&\int_t^{t+h}(\partial_t+\mathcal{L}^u)v(s,X_s^u)+c\left(s,X_s^u,u_s\right)\mathrm{d}s+\int_t^{t+h}\nabla v(s,X_s^u)\sigma\,\mathrm{d}B_s\\ =&\int_t^{t+h} R(v(s,X_s^u))\,\mathrm{d}s+\int_t^{t+h}\nabla v(s,X_s^u)\sigma\,\mathrm{d}B_s,\end{aligned}$$

For sufficiently small $h>0$, we have $\Delta\mathcal{M}_h^{u,v}(t,x)$ as the Euler-scheme approximation to $\mathcal{M}_{t+h}^{u,v}-\mathcal{M}_t^{u,v}$ with $X_t^u=x$, i.e.,

$$\Delta\mathcal{M}_h^{u,v}(t,x):=v(t+h,x+\xi_h^{t,x,u})-v(t,x)+hc\left(t,x,u(t,x)\right), \tag{19}$$

$$\xi_h^{t,x,u}:=\bar{\mu}\left(t,x,u(t,x)\right)h+\bar{\sigma}\left(t,x,u(t,x)\right)\left(B_{t+h}-B_t\right). \tag{20}$$

Comment: Increment of the martingale is an random difference (material derivative along random direction) approximation of the PDE.

[1] Cai, et al, Deep random difference method for high-dim. quasilinear parabolic PDEs. J. of Comput. Phys, 2026; [2] Xu, Zhang, A deep shotgun method for solving high-dim parabolic PDEs, J. of Sci. Comput. 2025..

How to fulfill the martingale property in parallel in time?

Let X be any uncontrolled **pilot** process s.t. $\Gamma(X_t) = \Gamma(X_t^*) := \text{Support}(X_t^*)$.

The martingale property of $\mathcal{M}^{u,v}$ can be ensured by

$$\mathbb{E}\left[\left(\mathcal{M}_{t+h}^{u,v} - \mathcal{M}_t^{u,v}\right) | X_t^u = x\right] = 0, \quad x \in \Gamma(X_{t_n}^u), \quad t \in [0, T-h]. \tag{21}$$

$$\Uparrow \text{If } \Gamma(X_t^u) \subset \Gamma(X_t),$$

$$\mathbb{E}\left[\left(\mathcal{M}_{t+h}^{u,v} - \mathcal{M}_t^{u,v}\right) | X_t^u = x\right] = 0, \quad x \in \Gamma(X_t), \quad t \in [0, T-h]. \tag{22}$$

$$\Updownarrow \text{Galerkin method}$$

$$\sup_{\rho \in \mathcal{T}} \int_0^{T-h} \mathbb{E}\left[\rho(t, X_t) \Delta \mathcal{M}_h^{u,v}(t, X_t)\right] \, \mathrm{d}t = 0, \quad t \in [0, T-h], \tag{23}$$

- **Path density pdf weighted Weak form of PINN**

How to avoid computing $\mathbb{E}\left[\mathcal{M}_T^{u,v}|\mathcal{F}_t\right]$ for $t \in [0,T]$?

Using the idea of Weak Adversarial Network (WAN) [Zang, Bao, Ye, and Zhou, 2020],

$$\mathbb{E}\left[(\mathcal{M}_{t+\Delta t}^{u,v} - \mathcal{M}_t^{u,v})\middle|X_t\right] = 0, \quad (t,\omega) \in [0, T-\Delta t] \times \Omega, \ \text{a.e.}, \tag{24}$$

$$\iff \int_0^{T-\Delta t} \mathbb{E}\Big[\rho(t,X_t)\mathbb{E}\left[(\mathcal{M}_{t+\Delta t}^{u,v} - \mathcal{M}_t^{u,v})\middle|X_t\right]\Big]\,\mathrm{d}t = 0, \quad \forall \rho \in \mathcal{T} \tag{25}$$

with $\mathcal{T}$ the set of **test functions** defined by

$$\mathcal{T} := \left\{\rho : [0,T] \times \mathbb{R}^d \to \mathbb{R}\middle|\ \rho \text{ is smooth and bounded}\right\}. \tag{26}$$

Since $\rho(t, X_t)$ is $\mathcal{F}^{X_t}$-measurable,

$$(25) \iff \int_0^{T-\Delta t} \mathbb{E}\left[\mathbb{E}\left[\rho(t,X_t)(\mathcal{M}_{t+\Delta t}^{u,v} - \mathcal{M}_t^{u,v})\middle|X_t\right]\right]\,\mathrm{d}t = 0, \quad \forall \rho \in \mathcal{T}, \tag{27}$$

$$\iff \int_0^{T-\Delta t} \mathbb{E}\left[\rho(t,X_t)(\mathcal{M}_{t+\Delta t}^{u,v} - \mathcal{M}_t^{u,v})\right]\,\mathrm{d}t = 0, \quad \forall \rho \in \mathcal{T}, \tag{28}$$

$$\iff \sup_{\rho\in\mathcal{T}}\left|\int_0^{T-\Delta t} \mathbb{E}\left[\rho(t,X_t)\left(\mathcal{M}_{t+\Delta t}^{u,v} - \mathcal{M}_t^{u,v}\right)\right]\,\mathrm{d}t\right|^2 = 0. \tag{29}$$

How to efficiently compute $\inf_{\kappa\in U} H\left(t,x,\kappa,\partial_x v(t,x),\partial^2_{xx} v(t,x)\right)$**?**

It is sufficient to pursue

$$u(t,X_t) = \arg\min_{\kappa\in U} H\left(t,X_t,\kappa,\partial_x v(t,X_t),\partial^2_{xx} v(t,X_t)\right), \quad (t,\omega)\in[0,T]\times\Omega \ \text{ a.e.}, \tag{30}$$

$$\Updownarrow$$

$$u = \arg\min_{\bar{u}:[0,T]\times\mathbb{R}^d\to U} \int_0^T \mathbb{E}\left[H\left(t,X_t,\bar{u}(t,X_t),\partial_x v(t,X_t),\partial^2_{xx} v(t,X_t)\right)\right] \mathrm{d}t,$$

for $\Gamma(X_t)\supset\Gamma(X_t^*) := \text{Support}(X_t^*)$.

Theorem 1

An optimal feedback control u and the value function v satisfying the HJB equation for $t \in [0, T]$ and $x \in \Gamma(X_t)$, can be found by the following two conditions,

$$\boxed{\mathcal{M}_t^{u,v} = \mathbb{E}\left[\mathcal{M}_T^{u,v} | \mathcal{F}_t\right], \quad t \in [0, T],} \tag{31}$$

$$\boxed{\int_0^T \mathbb{E}\left[H_t^{u,v}\right] \mathrm{d}t = \inf_{\bar{u} \in \mathcal{U}_{\mathrm{ad}}} \int_0^T \mathbb{E}\left[H_t^{\bar{u},v}\right] \mathrm{d}t,} \tag{32}$$

How to find the optimal feedback control without computing derivatives in Hamiltonian?

$$\textbf{(optimal control)} \quad \min_{u\in\mathcal{U}_{\text{ad}}} \int_0^T \mathbb{E}\left[\mathcal{L}^u v(t,X_t) + c\left(t,X_t,u(t,X_t)\right)\right] \mathrm{d}t. \tag{33}$$

However,

$$\mathbb{E}\left[\Delta\mathcal{M}_h^{u,v}\left(t,X_t\right)\right] = h\left\{(\partial_t + \mathcal{L}^u)v(t,X_t) + c(t,X_t,u(t,X_t))\right\} + O(h^2)$$

Thus

$$\min_{u\in\mathcal{U}_{\text{ad}}} \int_0^{T-h} \mathbb{E}\left[h^{-1}\Delta\mathcal{M}_h^{u,v}\left(t,X_t\right) - \partial_t v(t,X_t)\right] \mathrm{d}t.$$

i.e. the optimal control can be found from

$$\min_{u\in\mathcal{U}_{\text{ad}}} \mathbb{G}(u,v,1) \tag{34}$$

where

$$\mathbb{G}(u,v,\rho) := \int_0^{T-h} \mathbb{E}\left[\rho(t,X_t)\Delta\mathcal{M}_h^{u,v}\left(t,X_t\right)\right] \mathrm{d}t \tag{35}$$

Derivative-Free SOC-MartNet:

$$(u, v) = \lim_{\lambda \to +\infty} \underset{(\bar{u}, \bar{v}) \in \mathcal{U}_{\text{ad}} \times \mathcal{V}}{\arg\min} \left\{ \sup_{\rho \in \mathcal{T}} \mathbb{L}(\bar{u}, \bar{v}, \rho, \lambda) \right\}, \tag{36}$$

where

$$\begin{aligned} \mathcal{V} &:= \left\{ v : [0, T] \times \mathbb{R}^d \to \mathbb{R} \middle| \; v \in C^{1,2}, \; v(T, x) = g(x), \; \forall x \in \mathbb{R}^d \right\}, \\ \mathbb{L}(u, v, \rho, \lambda) &:= |\mathbb{G}(u, v, 1)|^2 + \lambda |\mathbb{G}(u, v, \rho)|^2, \\ \mathbb{G}(u, v, \rho) &:= \int_0^{T-h} \mathbb{E}\left[\rho(t, X_t) \Delta \mathcal{M}_h^{u,v}(t, X_t) \right] \mathrm{d}t \end{aligned} \tag{37}$$

with $\Delta \mathcal{M}_h^{u,v}(t, x)$ the Euler approximation to $\mathcal{M}_{t+h}^{u,v} - \mathcal{M}_t^{u,v}$ with $X_t^u = x$, i.e.,

$$\Delta \mathcal{M}_h^{u,v}(t, x) := v(t + h, x + \xi_h^{t,x,u}) - v(t, x) + hc(t, x, u(t, x)) \tag{38}$$

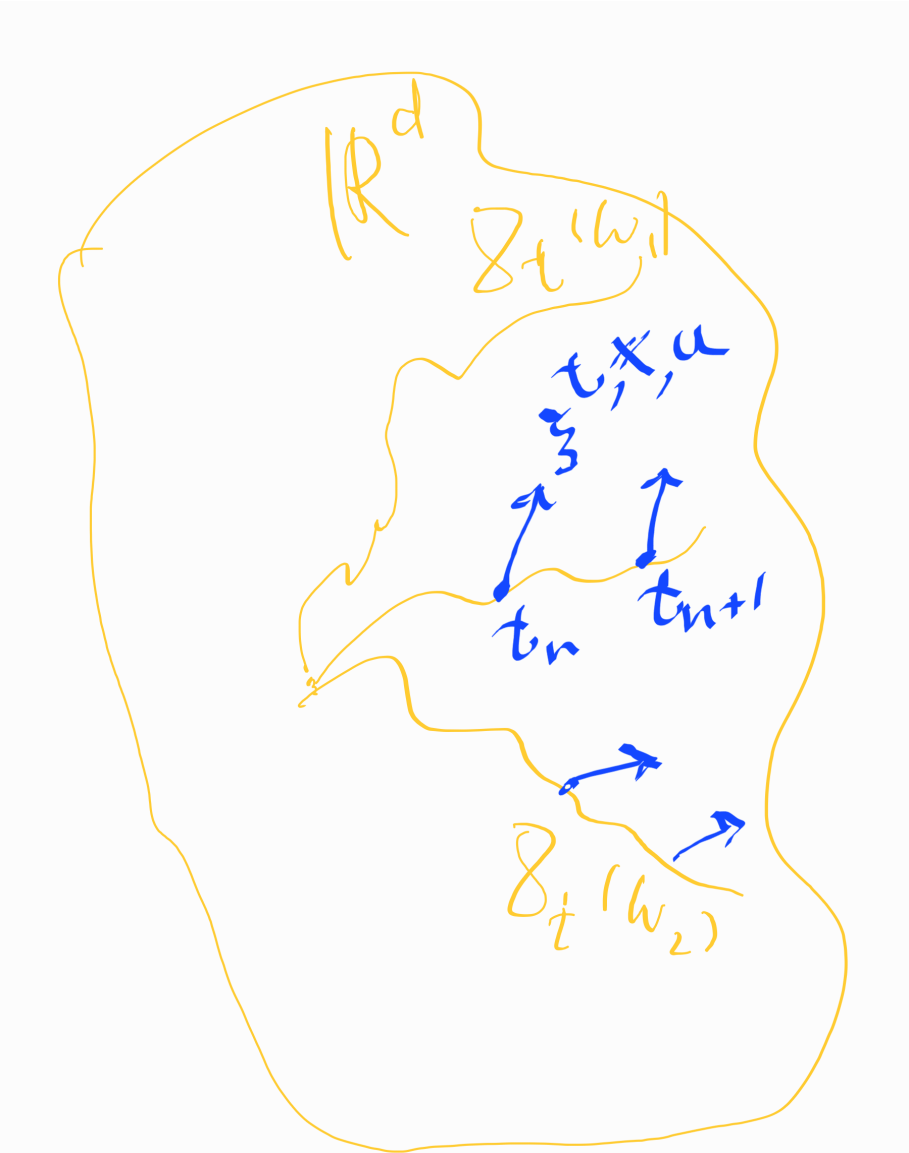
$$\xi_h^{t,x,u} := \bar{\mu}(t, x, u(t, x)) h + \bar{\sigma}(t, x, u(t, x)) (B_{t+h} - B_t). \tag{39}$$

The functions (u, v, ρ) are approximated by neural networks, i.e,

$$(u, v, \rho) \approx (u_\alpha, v_\theta, \rho_\eta) \tag{40}$$

with α, θ and η the parameters.

Schematics of DF SOC-Martnet - Random jumps under control along Pilot Paths



Error bounds for linear PDEs

We consider

$$\begin{cases} \mathcal{D}v(t,x) = f\big(t,x,v(t,x)\big), & (t,x) \in [0,T) \times \mathbb{R}^d, \\ v(T,x) = g(x), & x \in \mathbb{R}^d. \end{cases}$$

Assumptions:

1. Lipschitz continuity of f: $\exists\, C_0 > 0$ s.t.

$$|f(t,x,y_1) - f(t,x,y_2)| \le C_0 \, |y_1 - y_2|, \quad \forall y_1, y_2 \in \mathbb{R}, \quad (t,x) \in [0,T] \times \mathbb{R}^d.$$

2. Linearity of $\mathcal{D}$:

$$\mathcal{D} = \partial_t + \mu^\top \partial_x + \frac{1}{2} \operatorname{Tr}\left\{\sigma\sigma^\top \partial^2_{xx}\right\}, \tag{41}$$

where $\mu = \mu\big(t,x\big)$ and $\sigma = \sigma\big(t,x\big)$ are **independent of** $v(t,x)$.

Error metric: At each time step t_n, define the mean squared error by

$$\mathbb{M}_n^2 \left[v - v_\theta\right] := \int_{\mathbb{R}^d} \left|v(t_n,x) - v_\theta(t_n,x)\right|^2 \mathcal{P}_{X_n}(x) \, \mathrm{d}x, \tag{42}$$

where $x \mapsto \mathcal{P}_{X_n}(x)$ is the probability density function of X_n generated by $\mathcal{D}$, i.e.,

$$X_{n+1} = X_n + \mu(t_n, X_n)\, h + \sigma(t_n, X_n)\, \sqrt{h}\, \xi_{n+1}, \quad \xi_{n+1} \sim \mathrm{N}(0, I_q). \tag{43}$$

Theorem 2 (Cai-Fang-Zhou 2026)

Suppose Assumptions 1-2 hold, $v \in C_b^{2,4}$, *and* $0 < h < \min\{1, (2\sqrt{2}C_0)^{-1}\}$. *Then*

$$\max_{0\le n\le N-1} \mathbb{M}_n^2 \left[v - v_\theta\right] \le C\left(\mathbb{M}_N^2 \left[v - v_\theta\right] + L_{\mathrm{rdm},\pi}(v_\theta) + h^2\right) \tag{44}$$

for some constant $C > 0$ *independent of the time step size* h, *where*

- $\mathbb{M}_N^2 \left[v - v_\theta\right]$ *is due to the residual of the terminal condition;*
- $L_{\mathrm{rdm},\pi}(v_\theta)$ *is induced by the residual of the Random difference method, i.e.,*

$$L_{\mathrm{rdm},\pi}(v_\theta) := \frac{T}{N} \sum_{n=0}^{N-1} \int_{\mathbb{R}^d} \left|\xi R(t_n, x, \xi; v_\theta)\right|^2 \mathcal{P}_{X_n}(x)\, \mathrm{d}x; \tag{45}$$

- h^2 *is due to the Random difference method approximation* $\mathcal{D}_h v(t,x) = \mathcal{D}v(t,x) + \mathcal{O}(h)$.

• **Cai W, Fang S, Zhou T. (2026)** Deep random difference method for high dimensional quasilinear parabolic PDEs. J. of Comput. Phys, Vol. 555, 114767.

The derivative free SOC-MartNet (36) enjoys the following features:

- Its loss function (37) is free of computing $\partial_x v$ and $\partial^2_{xx} v$.
- The pilot diffusion process X for exploration is not necessarily dependent on $\mathcal{L}v(t,x)$, providing more flexibility to design the sample distribution.

The price of the derivative-free formulation:

- The random jumps $\xi_h^{t,x,u}$ in (38) need to be updated along with the optimal control, which can not be pre-calculated before the training as in the original SOC-MartNet [Cai et al, SISC, 2025].

Outline

Time partition:

$$\pi_N := \{t_0, t_1, \cdots, t_N\} \text{ s.t. } 0 = t_0 < t_1 < t_2 < \cdots < t_n < t_{n+1} < \cdots < t_N = T. \quad (46)$$

$$\Delta t_n := t_{n+1} - t_n, \quad \Delta B_{n+1} := B_{n+1} - B_n.$$

Numerical approximations:

1. Apply the **Euler** approximation to pilot paths $(X_{t_n})_{n=0}^N$:

$$X_{n+1} = X_n + \mu(t_n, X_n)\Delta t_n + \sigma(t_n, X_n)\Delta B_{n+1}, \quad n = 0, 1, 2, \cdots, N-1. \quad (47)$$

2. Apply the **trapezoid formula** to the time integral $\int_{t_n}^{t_{n+1}} H_s^{u,v}\, \mathrm{d}s$:

$$\mathcal{M}_{t_{n+1}}^{u_\alpha, v_\theta} - \mathcal{M}_{t_n}^{u_\alpha, v_\theta} \approx \Delta\mathcal{M}_{n+1}^{\alpha,\theta}, \quad n = 0, 1, \cdots, N-1,$$

$$\Delta\mathcal{M}_{n+1}^{\alpha,\theta} := v_\theta(t_{n+1}, \mathbf{X}_{\mathbf{n+1}}^{\mathbf{u}_\alpha}) - v_\theta(t_n, X_n) + \frac{1}{2}\left(c_n^{\alpha,\theta} + c_{n+1}^{\alpha,\theta}\right)\Delta t_n, \quad (48)$$

$$c_n^{\alpha,\theta} := c\left(t_n, X_n, u_\alpha(t_n, X_n)\right), \quad c_{n+1}^{\alpha,\theta} := c\left(t_{n+1}, \mathbf{X}_{\mathbf{n+1}}^{\mathbf{u}_\alpha}, u_\alpha(t_{n+1}, \mathbf{X}_{\mathbf{n+1}}^{\mathbf{u}_\alpha})\right) \quad (49)$$

3. Approximate $\mathbb{E}[\cdot]$ by the **Monte-Carlo** method based on the i.i.d. samples

$$\{(X_n^{(m)}, c_n^{\alpha,\theta,(m)}, c_{n+1}^{\alpha,\theta,(m)}, \Delta\mathcal{M}_n^{\alpha,\theta,(m)})\}_{n=0}^N, \quad m = 1, 2, \cdots, M. \quad (50)$$

Mini-batch loss function:

$$L(\alpha, \theta, \eta, \lambda; A_1, A_2) := |G(\alpha, \theta, 1; A_1)|^2 + \lambda |G(\alpha, \theta, \eta; A_2)|^2 , \tag{51}$$

$$G(\alpha, \theta, \eta; A) := \frac{1}{|A|} \sum_{(n,m)\in A} \rho_\eta(t_n, X_n^{(m)}) \Delta \mathcal{M}_{n+1}^{\alpha,\theta,(m)} \Delta t_n, \tag{52}$$

where $A \subset \{0, 1, \cdots, N\} \times \{1, 2, \cdots, M\}$ is updated at each optimization step.

High efficiency comes from

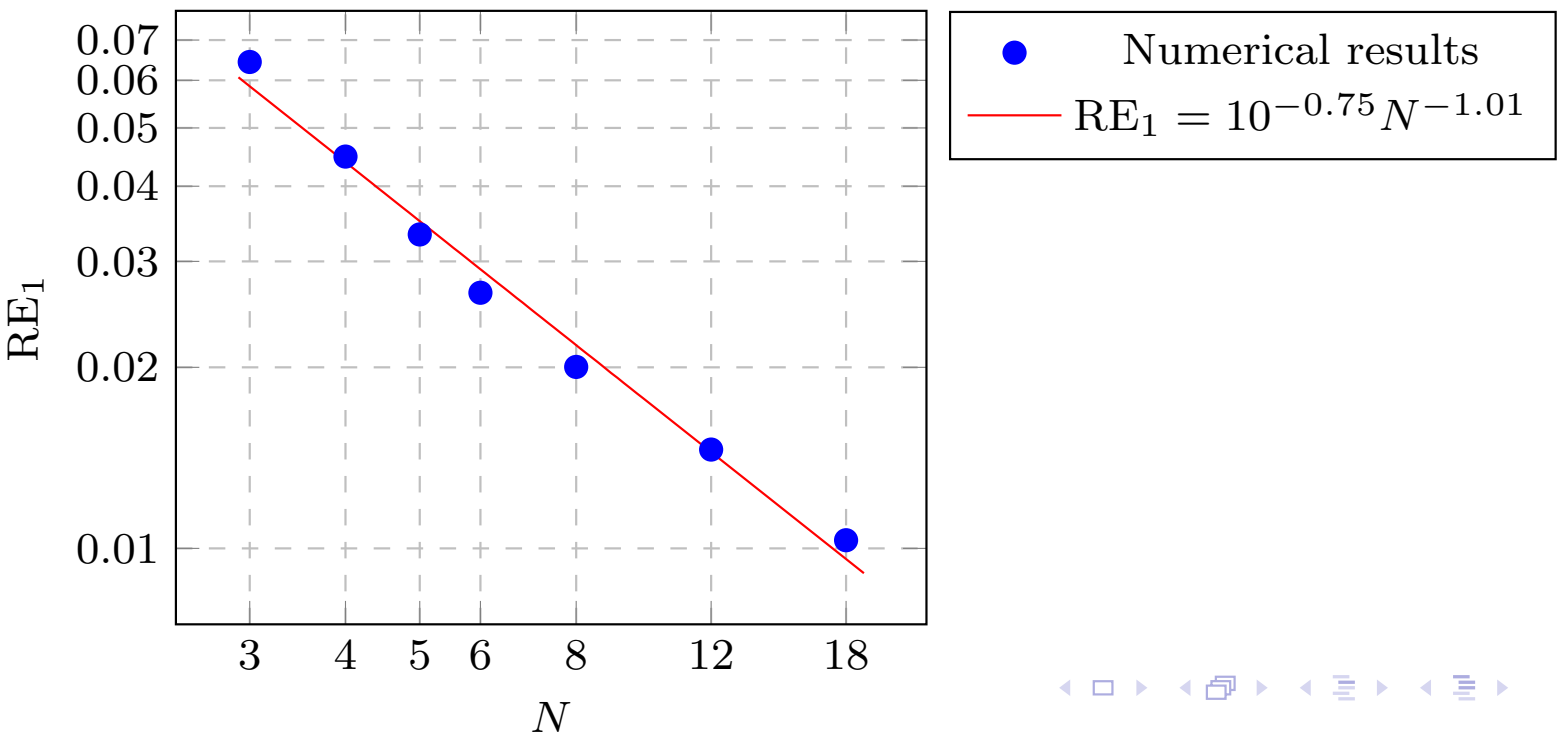
- **Offline** computation for pilot paths $\{X_n^{(m)}\}_{n=0}^N$,
- **Parallel** computation for $\sum_{(n,m)\in A} \cdots$.

We consider an Allen-Cahn-type equation

$$\partial_t v(t,x) + \mathcal{L}v(t,x) + f\left(t, x, v(t,x), \partial_x v(t,x), \partial^2_{xx} v(t,x)\right) = 0 \tag{53}$$

$$\mathcal{L} = \sum_{i=1}^{d} \sin(2x_i)\partial_{x_i} + \frac{1}{2}\sum_{i=1}^{d} (1 + 0.5\sin(5t + x_i))^2 \partial^2_{x_i}, d = 100 \tag{54}$$

$$f(t, x, v, \partial_x v, \partial^2_{xx} v) = v - v^3 + \bar{f}(t, x),$$



HJB equation [Bachouch et al., 2022, Section 3.1]:

$$\begin{cases} (\partial_t + \Delta_x)\, v(t,x) + \inf\limits_{\kappa \in \mathbb{R}^d} \left(2\kappa^\top \partial_x v(t,x) + |\kappa|^2 \right) = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d, \\ v(T,x) = 1 + g(x), \quad x \in \mathbb{R}^d, \end{cases} \tag{55}$$

The HJB equation (55) is associated with the SOCP:

$$u = \underset{\kappa \in \mathcal{U}_{\text{ad}}}{\arg\min}\, J(\kappa), \quad J(\kappa) := 1 + \mathbb{E}\left[\int_0^T |\kappa_s|^2 \, \mathrm{d}s + g\left(X_T^\kappa\right) \right], \tag{56}$$

$$\mathcal{U}_{\text{ad}} = \left\{ \kappa : [0,T] \times \Omega \to \mathbb{R}^d : \kappa \text{ is } \mathbb{F}^B\text{-adapted} \right\}, \tag{57}$$

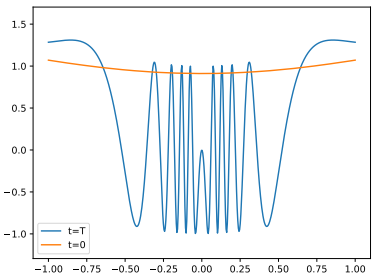
$$X_t^\kappa = X_0 + \int_0^t 2\kappa_s \, \mathrm{d}s + \int_0^t \sqrt{2}\, \mathrm{d}B_s, \quad t \in [0,T]. \tag{58}$$

In (55), $\inf_{\kappa \in \mathbb{R}^d}(\cdots)$ is obtained by enforcing

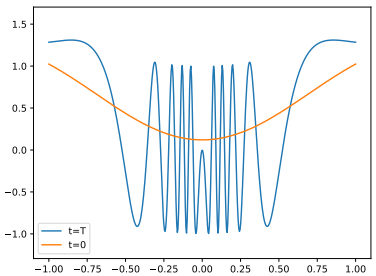
$$\int_0^T \mathbb{E}\left[H_t^{u,v} \right] \mathrm{d}t = \inf_{\bar{u} \in \mathcal{U}_{\text{ad}}} \int_0^T \mathbb{E}\left[H_t^{\bar{u},v} \right] \mathrm{d}t. \tag{59}$$

Consider the HJB equation (55) with an oscillatory terminal function

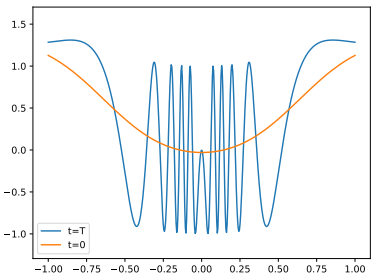
$$g(x) := \frac{1}{d}\sum_{i=1}^{d}\left\{\sin(x_i - \frac{\pi}{2}) + \sin\left(\left(\delta + x_i^2\right)^{-1}\right)\right\}, \quad x \in \mathbb{R}^d, \quad \delta = \pi/10. \tag{60}$$



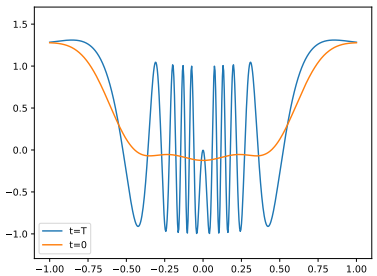
(a) $T = 1$



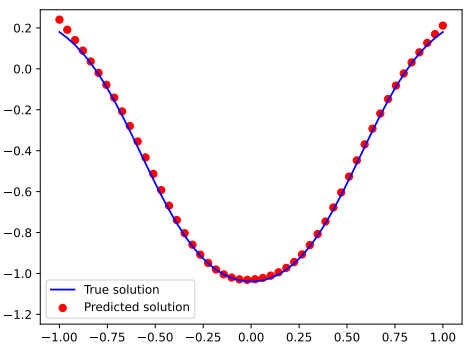
(b) $T = 0.1$



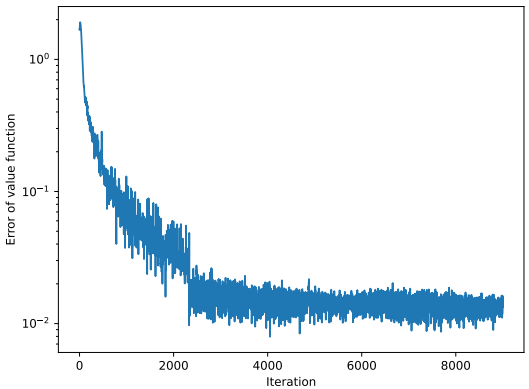
(c) $T = 0.05$



(d) $T = 0.01$



(e)



(f)

Figure 2: Numerical results of derivative-free SOC-MartNet for the $d = 10,000$ HJB equation (55) with oscillatory terminal function (60). The running time is 2814 seconds.

HJB equation without explicit $\inf_u H$

We consider the following HJB equation:

$$\partial_t v + \inf_{\kappa \in \mathbb{R}^d} \left\{ (b+\kappa)^\top \partial_x v + \frac{1}{2}|\kappa|^2 + \epsilon \sin(\mathbf{1}_d^\top \kappa) \right\} + \frac{1}{8}\Delta_x v = 0 \tag{61}$$

When $\epsilon = 0$, the value function $v(t,x)$ admits a closed-form solution given by

$$v(t,x) = -\frac{1}{4} \ln \left(\mathbb{E}\left[\exp\left(-4v(T, X_T^0)\right) \middle| X_t^0 = x \right] \right) \tag{62}$$

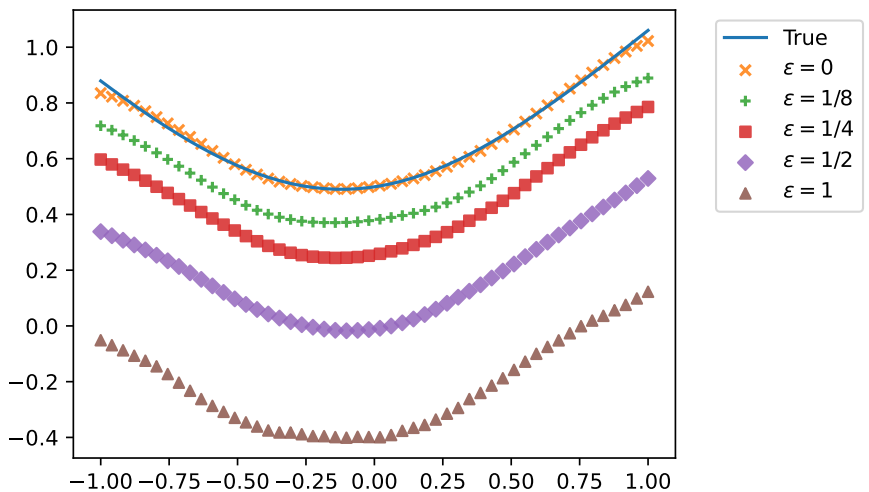


Figure 3: Numerical results of $\mathrm{alg}_a mnet for s \mapsto v(0, s\mathbf{1}_d)$, $s \in [-1,1]$ from (61) with $d = 1,000$ and $\varepsilon = 1, 1/2, 1/4, 1/8, 0$ (from bottom to top). The top blue curve without markers represents the reference solution of (61) with $\varepsilon = 0$.

Convergence in control (LQ Control Problem)

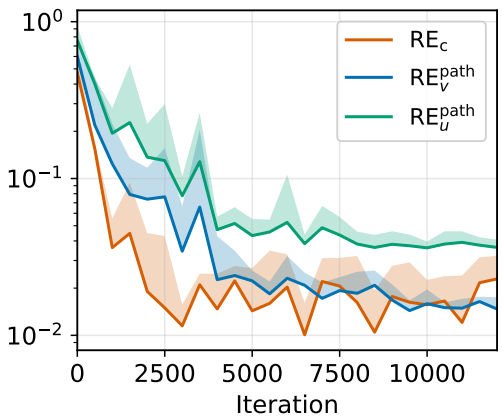
A LQ control problem (i.e.,$c(t,x,\kappa)=\kappa^2$) with the drift and diffusion coefficients

$$\mu_i(t,x,\kappa)=x_{i+1}-x_i+2q_i\kappa,\quad \sigma(t,x,\kappa)=\sqrt{2}QQ^\top,\quad (t,x,\kappa)\in[0,T]\times\mathbb{R}^d\times U,$$
$$x_{d+1}:=x_1,\quad q:=(q_i)_{i=1}^d=0.5\times(\sqrt{2},\sqrt{2},0,0,\cdots,0)^\top\in\mathbb{R}^d,\quad T=1,$$

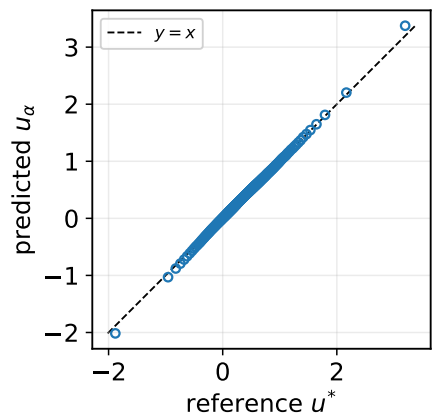
where the optimal feedback control and the value function are given by

$$u^*(t,x)=-q^\top\nabla v(t,x),\quad v(t,x)=\frac{1}{2}\log\left(1+C_{T-t}\right)+\frac{\left(q^\top e^{(T-t)L}x-1\right)^2}{2\left(1+C_{T-t}\right)},\qquad (63)$$

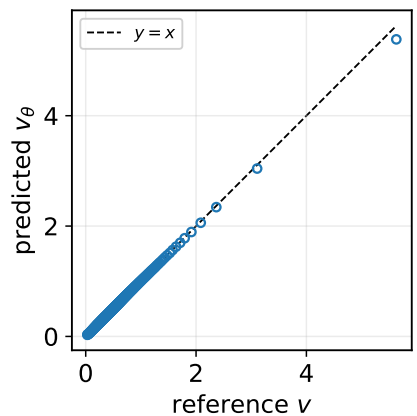
where $C_{T-t}:=2\int_0^{T-t}|q^\top e^{rL}Q|^2\,\mathrm{d}r$



(a) REs vs Iter.



(b) QQ plot of u_α vs u^*



(c) QQ plot of v_θ vs v

Figure 4: Results for the SOCP in $\mathrm{sec}_k 1 pairwith d = 10^4$. In (a), shading denotes the five-run mean plus twice the SD. In (b) and (c), the quantile-quantile (QQ) plots compare u_α and v_θ.

$d = 10^5$ dim. quasilinear PDEs with Allen-Cahn-type source term

$$\mathcal{D}v(t,x) = f(t,x,v(t,x)), \quad (t,x) \in [0,T] \times \mathbb{R}^d \tag{64}$$

$$\mathcal{D} = \partial_t + \frac{v^2}{2}\sum_{i=1}^{d} \partial_{x_i}^2, \quad f(t,x,v) = v - v^3 + Q(t,x); \tag{65}$$

$Q(t,x)$ and the terminal conditions $v(T,x) = g(x)$ are chosen for an exact solution

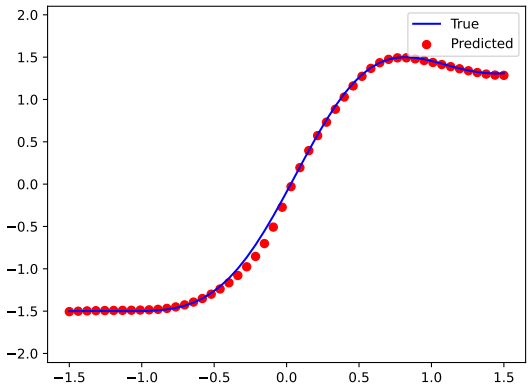
$$v(t,x) = V\big((t-0.5)\mathbf{1}_d + x\big), \quad (t,x) \in [0,T] \times \mathbb{R}^d, \tag{66}$$

where $\mathbf{1}_d := (1,1,\cdots,1)^\top \in \mathbb{R}^d$, and V is given by

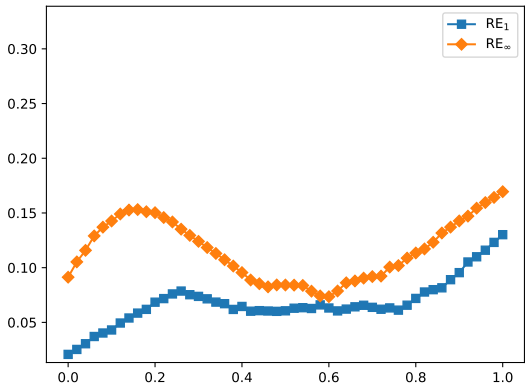
$$V(x) := \sum_{i=1}^{d-1} c_i K(x_i, x_{i+1}) + c_d K(x_d, x_1) \tag{67}$$

with

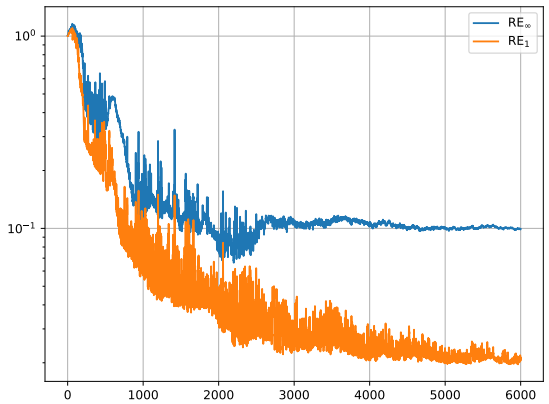
$$c_i := (1.5 - \cos(i\pi/d))/d, \quad K(x_i,x_j) := \sin\big(x_i + \cos(x_j) + x_j\cos(x_i)\big).$$



(a) $s \mapsto v(0, s\mathbf{1}_d)$



(b) $t \mapsto \mathrm{RE}(t)$



(c) RE vs Iter.

Figure 5: Numerical results with $d = 10^5$. The network width for v_θ is set to $W = 10010$, and the number of time steps is $N = 32$. The runtime is 3112 seconds at the 6000-th iteration.

W. Cai, S. Fang, T. Zhou. Deep random difference method for high dimensional quasilinear parabolic partial differential equations. J. of Comput. Phys, Vol. 555, 2026, 114767.

Outline

Table 1: Comparison of PDE and SDE-based and hybrid DNNs for solving high-dimensional quasi-linear parabolic equation.

DNN sol. $u_\theta(t, \mathbf{x})$	**Strategy A: PDE based DNNs**	**Strategy B: SDE based DNNs**	**Strategy C: Random differences**
Math. behind the loss	PDE residuals at $\mathbf{t} = \{t_j\}, \mathbf{X} = \{\mathbf{x}_i\}$, $R(u_\theta, \mathbf{t}, \mathbf{X})$ and BC	Probabilistic form for PDE solution and BC	Itô formula (**??**) applied on diffusion paths and BC
Sampling strategy	Monte Carlo sampling of domain by a given distribution or SDEs	Random walk by stochastic systems with the PDE operator generator	Random walk by stochastic systems with the PDE operator generator
∇, $\nabla\nabla^T$	SDGD, HTE, RS	Itô formula	Itô formula
Training aims	Reducing PDE residuals of DNNs at sampled locations in strong form in l_2 or l_∞ $\min_\theta \|\|R(u_\theta, \mathbf{t}, \mathbf{X})\|\|$	Enforcing the nonlinear Feynman-Kac formula for PDE solution and FBSDEs $u_\theta(t, X_t) = Y_t$, $\nabla u_\theta(t, X_t) = Z_t$	Reducing the PDE residual of neural network in $\pi(X_t)$-weighted Galerkin weak form $\min_\theta \max_\eta$ $(\rho_\eta, R(u_\theta, t, X_t))_{\pi(X_t)}$

Pros	Parallel, any PDEs, error estimate for elliptic PDEs, SDGD-PINN/RS-PINN for derivatives	Hessian matrix-free, system path-importance sampling, parallel between SDE paths	Total derivative-free, path-importance sampling, parallel within each path, first order accuracy in Δt, reduced loss variance.
Cons	Simple PINNs require derivatives, lack sampling distribution	need paths, not parallel within a path, only parabolic PDEs	need paths, test network ρ_η, min-max saddle point, only parabolic PDEs
Rep. works	PINN, DeepRitz, DGM, SDGD, HTE, RS-PINN	DeepBSDE, FBSNN, Actor-Critic, Diffusion, Feynman-Kac, Picard Iter, Diffusion MC	Weak form using test function: DeepMartNet, DRDM and, strong PINN-form: Shotgun, DGM, RS-PINN

• **W.Z. Zhang, Z.Y. Hu, W. Cai, G. Karniadakis, Deep Neural networks for solving high-dimensional parabolic partial differential equations, arXiv:2601.13256, January, 2026**.

Conclusions and future work

Conclusions:

$$\left.\begin{array}{l}\text{Martingale Formulation}\\ \text{Adversarial Learning}\\ \text{Maximum Principle}\end{array}\right\} \Longrightarrow \boxed{\text{SOC-MartNet}} \Longrightarrow \left\{\begin{array}{l}\text{SOCP}\\ \text{HJB equation}\\ \text{Parabolic equation}\end{array}\right.$$

- **Desirable features:** high efficiency and ultra-high dimensional applicability.

Future works:

- Extension to the degenerated HJB equation without classical solution $v \in C^{1,2}$;
- Theoretical analysis for SOCP (random difference operator approach).
- Rare events and committer functions.

References I

- **W. Cai, S. Fang, W. Zhang, T. Zhou, Martingale deep learning for very high dimensional stochastic optimal controls (2024). arXiv:2408.14395, under revision for Journal publication, 2026.**
- W. Cai, A. He, D. Margolis. DeepMartNet–A Martingale Based Deep Neural Network Learning Method for Dirichlet BVPs and Eigenvalue Problems of Elliptic PDEs in R^d. SIAM Journal on Scientific Computing, 48 (2026), pp. C25–C50.
- W. Cai, S. Fang, T. Zhou, SOC-MartNet: A martingale neural networkforthe hamilton-jacobi-bellman equation without explicit $inf_{u\in U}H$ in stochastic optimal controls, SIAM J. Sci. Comput. 47 (4) (2025) C795–C819.
- W. Cai, S. Fang, T. Zhou. Deep random difference method for high dimensional quasilinear parabolic partial differential equations. Journal of Computational Physics, Volume 555, 2026, 114767.
- W.Z. Zhang, Z.Y. Hu, W. Cai, G. Karniadakis, Deep Neural networks for solving high-dimensional parabolic partial differential equations, arXiv:2601.13256, 2026.
- W. Cai. Deterministic, Stochastic, and Deep Learning Methods for Computational Electromagnetics, Second Edition, Springer, 2025.

Other References

A. Bachouch, C. Huré, N. Langrené, and H. Pham. Deep neural networks algorithms for stochastic control problems on finite horizon: numerical applications. *Methodol. Comput. Appl. Probab.*, 24(1): 143–178, 2022. ISSN 1387-5841,1573-7713. doi: 10.1007/s11009-019-09767-9. URL https://doi.org/10.1007/s11009-019-09767-9.

P. Dupuis, K. Spiliopoulos, and H. Wang. Importance sampling for multiscale diffusions. *Multiscale Model. Simul.*, 10(1):1–27, 2012. ISSN 1540-3459,1540-3467. doi: 10.1137/110842545.

W. E, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Commun. Math. Stat.*, 5(4):349–380, 2017. ISSN 2194-6701,2194-671X. doi: 10.1007/s40304-017-0117-6. URL https://doi.org/10.1007/s40304-017-0117-6.

J. Han and W. E. Deep learning approximation for stochastic control problems, 2016. URL https://arxiv.org/abs/1611.07422.

J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

C. Huré, H. Pham, A. Bachouch, and N. Langrené. Deep neural networks algorithms for stochastic control problems on finite horizon: convergence analysis. *SIAM J. Numer. Anal.*, 59(1):525–557, 2021. ISSN 0036-1429,1095-7170. doi: 10.1137/20M1316640. URL https://doi.org/10.1137/20M1316640.

S. Ji, S. Peng, Y. Peng, and X. Zhang. Solving stochastic optimal control problem via stochastic maximum principle with deep learning method. *J. Sci. Comput.*, 93(1):Paper No. 30, 28, 2022. ISSN 0885-7474,1573-7691. doi: 10.1007/s10915-022-01979-5. URL https://doi.org/10.1007/s10915-022-01979-5.

X. Li, D. Verma, , and L. Ruthotto. A neural network approach for stochastic optimal control. *SIAM Journal on Sci. Comput.*, 46:C535–C556, 2024.

N. Nüsken and L. Richter. Solving high-dimensional hamilton–jacobi–bellman pdes using neural networks: perspectives from the theory of controlled diffusions and measures on path space. *Partial differential equations and applications*, 2(4):48, 2021.

E. Vanden-Eijnden and J. Weare. Rare event simulation of small noise diffusions. *Commun. Pure Appl. Math.*, 65(12):1770–1803, 2012. ISSN 0010-3640. doi: 10.1002/cpa.21428.

J. Yong and X. Y. Zhou. *Stochastic controls*, volume 43 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1999. ISBN 0-387-98723-1. doi: 10.1007/978-1-4612-1466-3. URL

Thank you!