AN EFFICIENT ON-POLICY DEEP LEARNING FRAMEWORK FOR STOCHASTIC OPTIMAL CONTROL

Mengjian Hua
NYU-ECNU Institute of Mathematical Sciences
NYU Shanghai
Shanghai, China 200124
mh5113@nyu.edu

Matthieu Laurière
Shanghai Frontiers Science Center of AI and DL
NYU-ECNU Institute of Mathematical Sciences
NYU Shanghai
Shanghai, China 200124
mathieu.lauriere@nyu.edu

Eric Vanden-Eijnden
Courant Institute of Mathematical Sciences
New York University
New York, NY 10012, USA
eve2@cims.nyu.edu

ABSTRACT

We present a novel on-policy algorithm for solving stochastic optimal control (SOC) problems. By leveraging the Girsanov theorem, our method directly computes on-policy gradients of the SOC objective without expensive backpropagation through stochastic differential equations or adjoint problem solutions. This approach significantly accelerates the optimization of neural network control policies while scaling efficiently to high-dimensional problems and long time horizons. We evaluate our method on classical SOC benchmarks as well as applications to sampling from unnormalized distributions via Schrödinger-Föllmer processes and fine-tuning pre-trained diffusion models. Experimental results demonstrate substantial improvements in both computational speed and memory efficiency compared to existing approaches.

1 INTRODUCTION

Stochastic Optimal Control (SOC) problems (Mortensen, 1989; Fleming & Rishel, 2012) arise across sciences and engineering, from traditional domains like finance and economics (Pham, 2009; Fleming & Stein, 2004; Aghion & Howitt, 1992) and robotics (Theodorou et al., 2011; Pavlov et al., 2018) to emerging applications in sampling complex distributions and simulating rare events (Zhang & Chen, 2022; Holdijk et al., 2023; Hartmann et al., 2013; 2017; Ribera Borrell et al., 2024). Their goal is to optimize a cost function by adding an adjustable drift (the control) to a reference stochastic differential equation (SDE).

While low-dimensional SOC problems can be solved using standard numerical methods for the Hamilton-Jacobi-Bellman equation, these approaches fail in high dimensions. This has motivated recent deep learning (DL) solutions (Han & E, 2016; Han et al., 2018; Huré et al., 2020; Domingo-Enrich et al., 2023; Germain et al., 2021; Hu & Lauriere, 2024) that parameterize the control using neural networks and optimize it via stochastic gradient descent on the SOC objective, evaluated using controlled SDE solutions. This Neural SDE approach (Tzen & Raginsky, 2019; Li et al., 2020), though conceptually simple, requires differentiating through SDE solutions—making it computationally expensive and limiting scalability.

One alternative uses the Girsanov theorem to compute the SOC objective via expectations over a reference process with a control independent from the one being optimized. However, this introduces an exponential weighing factor whose high variance (when the reference control differs significantly from the actual control) again limits scalability.

We propose a new approach based on expressing the gradient of the SOC objective exactly through expectations over the controlled process (on-policy evaluation) without differentiating through process

solutions (simulation-free). This result, first derived by Yang & Kushner (1991), avoids exponential weighing factors. We show this gradient can be computed via automatic differentiation of an alternative objective by selectively detaching parameters from the computational graph.

Specifically, our **main contributions** are:

- We propose an on-policy algorithm for solving generic SOC problems via deep learning that scales to scenarios where Neural SDE methods are computationally intractable. Our approach uses on-policy evaluation of gradients without requiring differentiation through SDE solutions.
- We show how to apply our approach to construct Föllmer processes between a point mass and an unnormalized target distribution, enabling sampling from this target and computing its normalization constant.
- We also show how to fine-tune diffusion-based generative models by solving an SOC problem that transforms an initial sampling SDE into one that samples from a tilted distribution weighted by a reward function.
- Through numerical experiments, we demonstrate significant reductions in computational time and memory usage compared to methods requiring SDE solution differentiation, such as Neural SDE approaches.

1.1 Related Work

Deep learning approaches to SOC have evolved along several directions. Han & E (2016) pioneered learning feedback control functions for high-dimensional problems, inspiring algorithms for backward SDEs and PDEs (Han et al., 2018). Alternative approaches include the algorithms proposed in Ji et al. (2020), dynamic programming methods (Huré et al., 2021; 2020), and stochastic optimal control matching (Domingo-Enrich et al., 2023) based on iterative diffusion optimization (Nüsken & Richter, 2023). These methods either use off-policy learning with high-variance estimators or require costly differentiation through SDE solutions.

The gradient formula we use was originally proposed in Yang & Kushner (1991) in the context of sensitivity analysis in finance (Pham, 2009; Fleming & Stein, 2004; Aghion & Howitt, 1992), with generalizations in Gobet & Munos (2005). While referenced in recent DL works (Mohamed et al., 2020; Li et al., 2020; Lie, 2021; Domingo-Enrich et al., 2023; Ribera Borrell et al., 2024; Domingo-Enrich, 2024), it has not been fully exploited algorithmically. Our approach also connects to Reinforcement Learning (Quer & Borrell, 2024; Domingo-Enrich et al., 2024; Domingo-Enrich, 2024), resembling a continuous-time version of the REINFORCE algorithm (Williams, 1992; 1988; Sutton et al., 1999).

For Föllmer processes (Föllmer, 1986), several deep learning methods have been proposed (Huang et al., 2021; Jiao et al., 2021; Vargas et al., 2023b). The Path Integral Sampler (PIS) (Zhang & Chen, 2022) is closest to our approach, as it performs on-policy minimization of the same SOC objective. However, PIS requires differentiating through controlled processes, while our approach avoids this costly step.

Generative models based on diffusion can be fine-tuned by modifying their drifts based on reward functions (Fan et al., 2024; Clark et al., 2024; Uehara et al., 2024). This task can be formulated as a SOC problem (Domingo-Enrich et al., 2024). Our method offers an efficient solution when the base distribution is a point-mass.

2 Methods

2.1 Problem setup

We consider the stochastic optimal control (SOC) problem:

$$\min_{u\in\mathcal{U}} J(u) \qquad \text{with} \qquad J(u) = \mathbb{E}[\mathcal{J}(u, X^u)], \tag{1}$$

in which, for a generic process $X = (X_t)_{t\in[0,T]}$,

$$\mathcal{J}(u, X) = \int_0^T \left(\tfrac{1}{2}|u_t(X_t)|^2 + f_t(X_t)\right) dt + g(X_T), \tag{2}$$

and $X^u = (X_t^u)_{t\in[0,T]}$ is the solution to the SDE

$$dX_t^u = (b_t(X_t^u) + \sigma_t u_t(X_t^u))\, dt + \sigma_t dW_t, \quad X_0^u \sim \mu_0. \tag{3}$$

In these equations, $X_t^u \in \mathbb{R}^d$ is the system state, $\mathbb{E}$ denotes expectation over the law of X^u, $u : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$ is a closed-loop Markovian control that belongs to some set $\mathcal{U}$ of admissible controls to be specified later, $f : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$ is the state cost, $g : \mathbb{R}^d \to \mathbb{R}^d$ is the terminal cost, $b : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$ is the base drift, $\sigma : [0,T] \to \mathbb{R}^d \times \mathbb{R}^d$ is the volatility matrix, which we assume invertible and independent of the state X_t^u, $(W_t)_{t\in[0,T]}$ is a Wiener process taking values in $\mathbb{R}^d$, and μ_0 is some probability distribution on $\mathbb{R}^d$ for the initial state.

We are interested in solving (1) in situations where the set of admissible controls $\mathcal{U}$ is a rich parametric class, for example made of deep neural networks (DNN). We denote functions in the class by u^θ, where $\theta \in \Theta$ collectively denotes the parameters to be adjusted, e.g. the weights if we use a DNN.

2.2 Reformulation with Girsanov theorem

The key challenge in solving the SOC problem (1) is that a vanilla calculation of the objective's gradient requires to differentiate X^u since it depends on the control u: this requires costly back-propagation through the solutions of the SDE (3). We propose a method to compute the gradient of (1) that avoids this expensive step and refer to it as a *simulation-free* method.

To this end, we first use the Girsanov theorem to reformulate the problem so that the control u appears explicitly in the objective:

Lemma 1. *Given a reference control $v \in \mathcal{U}$, the objective in (1) can be expressed as*

$$J(u) = \mathbb{E}\left[\mathcal{J}(u, X^v) M(u,v)\right], \tag{4}$$

where $X^v = (X_t^v)_{t\in[0,T]}$ solves the SDE (3) with u replaced by v, $M(u,v)$ is the Girsanov factor

$$M(u,v) = \exp\left(-\int_0^T (v_t(X_t^v) - u_t(X_t^v)) \cdot dW_t - \frac{1}{2}\int_0^T |v_t(X_t^v) - u_t(X_t^v)|^2\, dt \right), \tag{5}$$

and the expectation $\mathbb{E}$ in (4) is taken over the law of X^v.

The result follows directly from the Girsanov change of measure formula between the law of X^u and X^v. For a proof, see e.g. Karatzas & Shreve (1991).

Expression (4) presents $J(u)$ as an off-policy objective, making u explicit since the process X^v is independent of this control. This eliminates the need to differentiate through state process trajectories when computing gradients. However, empirically evaluating $J(u)$ and its gradient using (4) with finite samples from X^v yields estimators whose variance depends heavily on the reference control v. This suggests keeping v close to u. Next we show that we can use (4) to evaluate the gradient using the controlled process itself.

2.3 Gradient computation

Our method builds on a gradient formula for parametrized controls $u = u^\theta$, originally derived in Yang & Kushner (1991) and also presented in Ribera Borrell et al. (2024):

Proposition 1. *Let u^θ with $\theta \in \Theta$ be a parametric realization of a control in $\mathcal{U}$ and denote $L(\theta) \equiv J(u^\theta)$ the objective (1) viewed as a function of θ. Then*

$$\partial_\theta L(\theta) = \mathbb{E}\left[\int_0^T u_t^\theta(X_t^\theta) \cdot \partial_\theta u_t^\theta(X_t^\theta) dt\right] + \mathbb{E}\left[\mathcal{J}(u^\theta, X^\theta) \int_0^T \partial_\theta u_t^\theta(X_t^\theta) \cdot dW_t\right] \tag{6}$$

where $\partial_\theta u_t^\theta(X_t^\theta)$ denotes $\partial_\theta u_t^\theta(x)$ evaluated at $x = X_t^\theta$ and $X^\theta = (X_t^\theta)_{t\in[0,T]} \equiv (X_t^{u^\theta})_{t\in[0,T]}$ solves the SDE

$$dX_t^\theta = \left(b_t(X_t^\theta) + \sigma_t u_t^\theta(X_t^\theta)\right) dt + \sigma_t dW_t, \quad X_0^\theta \sim \mu_0, \tag{7}$$

and the expectation $\mathbb{E}$ in (6) is taken over the law of X^θ.

For completeness, we give the proof of this proposition in Appendix A. Expression (6) eliminates the need to differentiate through state trajectories $(X_t^\theta)_{t\in[0,T]}$. While it requires an invertible, control-independent volatility σ_t, the formula can be extended to control-dependent volatilities using Malliavin calculus (Gobet & Munos, 2005).

Algorithm 1 Simulation-Free On-Policy Training

Initialize: n walkers, K time steps, model parameters θ for u^θ, gradient descent optimizer
repeat
  Set $\bar{\theta} = \text{stopgrad}(\theta)$
  Randomize time grid: $t_1, \dots, t_K \sim \text{Uniform}(0, T)$
  Add $t_0 = 0$, $t_K = T$, and sort such that $0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$
  Set $\Delta t_k = t_{k+1} - t_k$
  for each walker $i = 1, \dots, n$ **do**
    Set $x_0^i \sim \mu_0$
    for $k = 0, \dots, K-1$ **do**
      $\Delta W_k^i = \sqrt{\Delta t_k}\, \zeta_k^i$, where $\zeta_k^i \sim N(0, \text{Id})$
      $x_{t_{k+1}}^i = x_{t_k}^i + u_{t_k}^{\bar{\theta}}(x_{t_k}^i)\Delta t_k + \sigma_{t_k} \Delta W_k^i$
    end for
    $A_K^i = \sum_{k=1}^K \frac{1}{2} |u_{t_k}^\theta(x_{t_k}^i)|^2 \Delta t_k$
    $\bar{B}_K^i = \sum_{k=1}^K \left(\frac{1}{2} |u_{t_k}^{\bar{\theta}}(x_{t_k}^i)|^2 + f_{t_k}(x_{t_k}^i) \right) \Delta t_k$
    $C_K^i = \sum_{k=1}^K u_{t_k}^\theta(x_{t_k}^i) \cdot \Delta W_k^i$
  end for
  Compute: $\hat{L}_n(\theta, \bar{\theta}) = n^{-1} \sum_{i=1}^n \left[A_K^i + \left(\bar{B}_K^i + g(x_{t_K}^i) \right) C_K^i \right]$.
  Compute $\partial_\theta \hat{L}(\theta, \bar{\theta})\big|_{\bar{\theta}=\theta}$ and take a step of gradient descent to update θ.
until converged

2.4 Alternative objective for implementation

Equation (6) can be implemented to directly estimate the gradient of the objective $L(\theta) = J(u^\theta)$ by estimating the expectation empirically over an ensemble of independent realizations of the SDE (7). Alternatively, we can use automatic differentiation of an alternative objective (Ribera Borrell et al., 2024; Domingo-Enrich, 2024):

Proposition 2. *We have*

$$\partial_\theta L(\theta) = \partial_\theta \hat{L}(\theta, \bar{\theta})\big|_{\bar{\theta}=\theta}, \tag{8}$$

where we defined

$$\hat{L}(\theta, \bar{\theta}) = \mathbb{E}\left[\int_0^T \tfrac{1}{2} |u_t^\theta(X_t^{\bar{\theta}})|^2 dt \right] + \mathbb{E}\left[\mathcal{J}(u^{\bar{\theta}}, X^{\bar{\theta}}) \int_0^T u_t^\theta(X_t^{\bar{\theta}}) \cdot dW_t \right] \tag{9}$$

in which $X^{\bar{\theta}} = (X_t^{\bar{\theta}})_{t \in [0,T]}$ solves (7) with u^θ replaced by $u^{\bar{\theta}}$ and the expectation $\mathbb{E}$ in (6) is taken over the law of $X^{\bar{\theta}}$.

The proof of this proposition is immediate by direct calculation so we omit it for the sake of brevity.

The gradient $\partial_\theta \hat{L}(\theta, \bar{\theta})\big|_{\bar{\theta}=\theta}$ can be computed via automatic differentiation by using $\bar{\theta} = \text{stopgrad}(\theta)$. This avoids differentiating through $\bar{X} \equiv X^{\bar{\theta}}$ while maintaining an on-policy objective. The expectation can be estimated empirically using samples from the SDE (7), as detailed in Algorithm 1.

2.5 Application to sampling via construction of a Föllmer process

By definition, the Föllmer process that samples a given target probability distribution μ is the process $(Y_t^u)_{t \in [0,1]}$ that uses the optimal control u obtained by solving

$$\min_{u \in \mathcal{U}} \mathbb{E} \int_0^1 \tfrac{1}{2} |u_t(Y_t^u)|^2 dt \tag{10}$$

where

$$dY_t^u = u_t(Y_t^u)dt + dW_t, \quad Y_0^u \sim \delta_0, \quad Y_1^u \sim \mu. \tag{11}$$

This problem is a special case of the Schrödinger bridge problem (Léonard, 2014) when the base distribution is the Dirac delta distribution δ_0, i.e. the point mass at $x = 0$.

While the minimization problem in (10) differs from a standard SOC problem (1) due to its terminal condition $Y^u_{t=1} \sim \mu$, it can be reformulated as a SOC problem (Léonard, 2014; Chen et al., 2014)—an insight exploited by Zhang & Chen (2022). The connection is as follows:

Proposition 3. *Assume that μ is absolutely continuous with respect of the Lebesgue measure and let its probability density function be $\rho(x) = Z^{-1}e^{-U(x)}$ where $U : \mathbb{R}^d \to \mathbb{R}$ is a known potential and $Z = \int_{\mathbb{R}^d} e^{-U(x)}dx < \infty$ is an unknown normalization factor. Consider the SOC problem using the objective*

$$J(u) = \mathbb{E}\left[\int_0^1 \tfrac{1}{2}|u_t(X^u_t)|^2 dt - \tfrac{1}{2}|X^u_1|^2 + U(X^u_1)\right], \tag{12}$$

where $(X^u_t)_{t\in[0,T]}$ solves the SDE

$$dX^u_t = u_t(X^u_t)dt + dW_t, \qquad X^u_{t=0} \sim \delta_0. \tag{13}$$

Then the process $(X^u_t)_{t\in[0,1]}$ obtained by using the optimal control minimizing (12) in the SDE (13) is the Föllmer process that satisfies $X^u_{t=1} \sim \mu$.

We omit the proof of this proposition since it is a special case of Proposition 5 established below.

Our approach can solve the SOC problem in Proposition 3, providing a simulation-free implementation of the Path Integral Sampler (PIS) (Zhang & Chen, 2022). Section 3 demonstrates the computational advantage of our approach through examples.

Since we replaced the terminal constraint in SDE (13) with a terminal cost in (12), $X^u_{t=1} \sim \mu$ is not guaranteed for suboptimal controls. However, as noted in Zhang & Chen (2022), we can still compute unbiased expectations over μ for any control through Girsanov reweighting:

Proposition 4. *Consider the process $(X^u_t)_{t\in[0,T]}$ obtained by solving the SDE (13) with any (not necessary optimal) control u. Then, given any suitable test function $h : \mathbb{R}^d \to \mathbb{R}$, we have*

$$\int_{\mathbb{R}^d} h(x)\mu(dx) = Z^{-1}\mathbb{E}\left[h(X^u_T)M(u)\right], \qquad Z = \int_{\mathbb{R}^d} e^{-U(x)}dx = \mathbb{E}\left[M(u)\right], \tag{14}$$

where we defined

$$M(u) = (2\pi)^{d/2}\exp\left(-\int_0^1 \tfrac{1}{2}|u_t(X^u)|^2 dt - \int_0^1 u_t(X^u_t)\cdot dW_t + \tfrac{1}{2}|X^u_1|^2 - U(X^u_1)\right). \tag{15}$$

In addition $M(u) = Z$ iff u is the optimal control minimizing the SOC problem with objective (12).

We also omit the proof of this proposition since it is a special case of Proposition 6 established below.

2.6 Application to fine-tuning

The approach in Section 2.5 can be adapted to fine-tune generative models. Suppose that the drift b in the SDE

$$dY_t = b_t(Y_t)dt + \sigma_t dW_t, \qquad Y_0 \sim \delta_0, \tag{16}$$

has been tailored in such a way that $Y_{t=T} \sim \nu$ where ν is a given probability distribution. Learning such a b can for instance be done using the framework of score-based diffusion models (Song et al., 2021) or stochastic interpolants (Albergo & Vanden-Eijnden, 2022; Albergo et al., 2023) tailored to building Föllmer processes (Chen et al., 2024). Assume that we would like to fine-tune this diffusion so that it samples instead the probability distribution

$$\mu(dx) = Z^{-1}e^{r(x)}\nu(dx), \quad Z = \int_{\mathbb{R}^d} e^{r(x)}\nu(dx), \tag{17}$$

obtained by tilting ν by the reward function $r : \mathbb{R}^d \to \mathbb{R}$ (assuming that this tilted measure is normalizable, i.e. $Z < \infty$). Such problems arise in the context of image generation where they have received a lot of attention lately. Our next result shows that it can be cast into a SOC problem.

Proposition 5. *Consider the SOC problem (1) with zero running cost, $f = 0$, and terminal cost set to minus the reward function, $g = -r$, in the objective (10). Assume also that the drift b and the volatility σ used in the SDE (3) are the same as those used in the SDE (16) that guarantees that $Y_{t=T} \sim \nu$. Then the solutions of the SDE (3) solved with the optimal u minimizing this SOC problem and $X^u_{t=0} = 0$ are such that $X^u_{t=T} \sim \mu$.*

The proof of this proposition is given in Appendix A. Note that Proposition 3 follows from Proposition 5 as a special case if we set $b_t(x) = 0$, $\sigma_t = 1$, and $T = 1$, in which case $\nu = N(0, \mathrm{Id})$, and we can set $r(x) = -U(x) + \frac{1}{2}|x|^2$ to target $\mu(dx) = Z^{-1}e^{-U(x)}dx$.

The SOC problem in Proposition 5 can again be solved in a simulation-free way using our approach, thereby offering a simple alternative to the Adjoint Matching method proposed in Domingo-Enrich et al. (2024). Since in practice the learned control will be imperfect, we will need to reweigh the samples to get unbiased estimates of expectations over them. It can be done using this result:

Proposition 6. *Let X^u solve SDE (3) starting from the initial condition $X^u_{t=0} = 0$ with an arbitrary (not necessarily optimal) u and with the drift b and the volatility σ that guarantee that the solutions the SDE (3) satisfy $Y_{t=T} \sim \nu$. Then given any suitable test function $h : \mathbb{R}^d \to \mathbb{R}$, we have*

$$\int_{\mathbb{R}^d} h(x)\mu(dx) = Z^{-1}\mathbb{E}\left[h(X^u_T)M_r(u)\right], \qquad Z = \int_{\mathbb{R}^d} e^{r(x)}\nu(dx) = \mathbb{E}\left[M_r(u)\right] \tag{18}$$

where we defined

$$M_r(u) = \exp\left(-\int_0^T \tfrac{1}{2}|u_t(X^u)|^2 dt - \int_0^T u_t(X^u_t)\cdot dW_t + r(X^u_T)\right) \tag{19}$$

and the expectation is taken over the law of X^u. In addition, $M_r(u) = Z$ iff u is the optimal control specified in Proposition 5.

The proof of this proposition is given in Appendix A. Proposition 4 follows from Proposition 6 as a special case if we set $b_t(x) = 0$, $\sigma_t = 1$, $T = 1$, and $r(x) = -U(x) + \frac{1}{2}|x|^2$.

3 Experiments

We test our method on two applications: sampling from unnormalized distributions via Schrödinger-Föllmer processes and fine-tuning pre-trained diffusion models. In Appendix C, we also report the performance of our approach on classical SOC benchmarks involving linear Ornstein-Ulhenbeck processes with linear and quadratic costs, that are amenable to exact solution for benchmarking.

3.1 Sampling from an unnormalized distribution

We test our method on Neal's funnel distribution in $d = 10$ dimensions, which can be sampled by solving the SOC problem formulated in Sec.2.5. The distribution is defined by $x_0 \sim N(0, \sigma_0)$ and $x_{1:9}|x_0 \sim N(0, e^{x_0}\mathrm{Id})$. Previously examined by Zhang & Chen (2022) using the Path Integral Sampler (PIS), this distribution becomes exponentially difficult to sample as σ_0 increases: negative x_0 values produce exponentially small spreads in $x_{1:9}$, while positive values yield exponentially large spreads. We examine cases with $\sigma_0 = 1$ and $\sigma_0 = 3$.

Following Zhang & Chen (2022), we parameterize the control as

$$u^\theta_t(x) = \mathrm{NN}^\theta_1(t, x) + \mathrm{NN}^\theta_2(t) \times \nabla \log \rho(x),$$

where $\nabla \log \rho(x)$ is the score of the funnel distribution density. For each network NN^θ_1 and NN^θ_2, we encode the scalar t into 128 dimensions using Fourier positional encoding, process it through two fully connected layers with 64 hidden units, and separately process x through a two-layer MLP to obtain 64-dimensional features. The concatenated features feed into a three-layer network for the final output. To increase difficulty, the last linear layers of both networks are initialized to zero, yielding $u^\theta_t(x) = 0$ initially. Optimization uses Adam (Kingma, 2014) with learning rate $5 \cdot 10^{-3}$.

3.2 Fine-tuning on the φ^4 model

We sample the φ^4 model in $d = 2$ spacetime dimensions, a statistical lattice field theory where field configurations $\varphi \in \mathbb{R}^{L\times L}$ represent the lattice state (L denotes spatiotemporal extent). This model poses sampling challenges due to its phase transition from disorder to full order, during which neighboring sites develop strong correlations in sign and magnitude Vierhaus (2010); Albergo et al.

Algorithm	MMD ↓
Our Method	$\mathbf{0.043 \pm 0.001}$
PIS (Zhang & Chen, 2022)	$\mathbf{0.048 \pm 0.001}$
FAB (Midgley et al., 2022)	$\mathbf{0.032 \pm 0.000}$
GMMVI (Arenz et al., 2022)	$\mathbf{0.031 \pm 0.000}$
DDS (Vargas et al., 2023a)	0.172 ± 0.031
AFT (Arbel et al., 2021)	0.159 ± 0.010
CRAFT (Arbel et al., 2021)	0.115 ± 0.003
CMCD-KL (Nusken et al., 2024)	0.095 ± 0.003
NETS-AM (Albergo & Vanden-Eijnden, 2025)	$\mathbf{0.041 \pm 0.001}$

Table 1: **Funnel Distribution Example**: Performance of our method measured by MMD (Maximum Discrepancy Distance) from the true distribution. Benchmarking is quoted from comparative results of Blessing et al. (2024) and Albergo & Vanden-Eijnden (2025) for reproducibility. Following Blessing et al. (2024), we compute the maximum mean discrepancy (MMD) between 10000 samples from the model and 10000 samples from the target and compare to the numbers reported by Blessing et al. (2024) and Albergo & Vanden-Eijnden (2025).

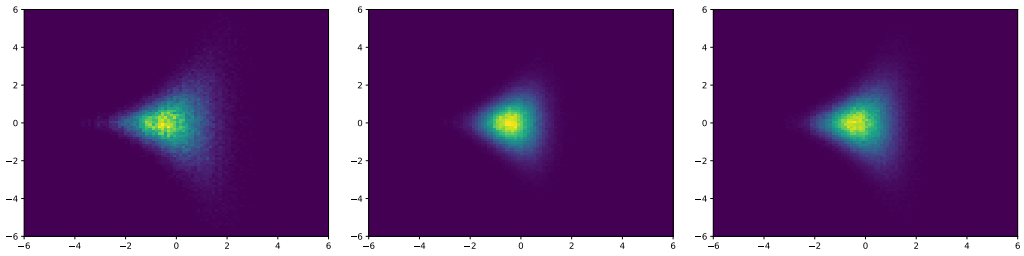

Figure 1: **Funnel distribution example** ($\sigma = 1$)**:** The samples from the funnel distribution (left panel), Path Integral Samplers (middle panel) and our method (right panel). To plot the samples of the ten-dimensional funnel distribution in 2D, we use the independence of its coordinates $\{x_1, \cdots, x_9\}$, squeeze these nine dimensions into one coordinate, and keep the first dimension x_0.

(2019). Using the framework from Sec. 2.6, we fine-tune a Gaussian distribution to match the fully ordered φ^4 model.

The φ^4 model is specified by the following probability density function (PDF)

$$\rho(\varphi) = Z^{-1} e^{-E(\varphi)} \tag{20}$$

where $Z = \int_{\mathbb{R}^{L\times L}} e^{-E(\varphi)} d\varphi$ is a normalization constant and E is an energy function defined as

$$E(\varphi) = \frac{1}{2}\alpha \sum_{a\sim b} |\varphi(a) - \varphi(b)|^2 + \frac{1}{2}\beta \sum_a |\varphi(a)|^2 + \frac{1}{4}\gamma \sum_a |\varphi(a)|^4. \tag{21}$$

where $a, b \in [0, \ldots, L-1]^2$ denote the discrete positions on a 2-dimensional lattice of size $L \times L$, $a \sim b$ denotes neighboring sites on the lattice, and we assume periodic boundary conditions; $\alpha > 0$, $\beta \in \mathbb{R}$ and $\gamma > 0$ are parameters. We will sample PDF (20) by fine-tuning a reference SDE whose time $t = 1$ solutions sample the Gaussian PDF

$$\rho_0(\varphi) = Z_0^{-1} e^{-E_0(\varphi)},$$

where $Z_0 = \int_{\mathbb{R}^{L\times L}} e^{-E_0(\varphi)} d\varphi$ and

$$E_0(\varphi) = \frac{1}{2}\alpha \sum_{a\sim b} |\varphi(a) - \varphi(b)|^2 + \frac{1}{2}\beta_0 \sum_a |\varphi(a)|^2, \tag{22}$$

with $\alpha > 0$ as in (21) and $\beta_0 > 0$. Writing $E_0(\varphi) = \frac{1}{2}\varphi^T C^{-1} \varphi$ shows that $\rho_0(\varphi)$ is a zero-mean Gaussian with covariance C (see Appendix B). Thus, solutions to

$$d\varphi_t^0 = C^{-1/2} dW_t, \qquad \varphi_{t=0}^0 = 0 \tag{23}$$

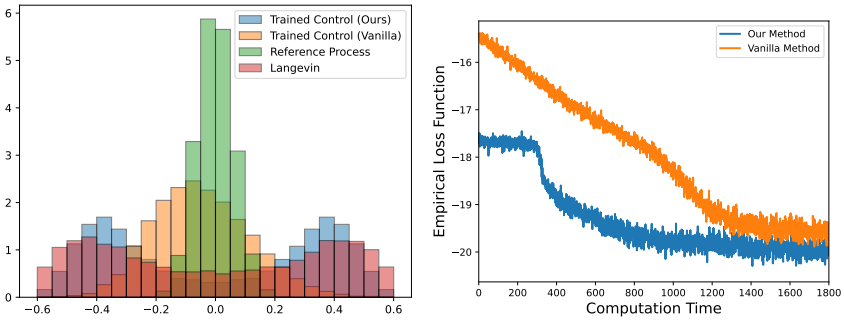

Figure 2: **Top Panel**: Histograms of the average magnetization of 10000 lattice configurations, sampled with the trained control of our method, the vanilla method, Langevin dynamics with $E(\varphi)$, and the reference PDF $\rho_0(\varphi)$. We use samples obtained by running Langvein dynamics with the target potential $E(\varphi)$ as the ground-truth. (See Appendix B for more details about how to sample with Langevin dynamics). Note that the results of our method is closer to the Langevin target than the vanilla method, which has adopts a much smaller model and fails to capture the statistics. **Bottom Panel**: The empirical loss function over GPU compute time. Our method enables a much larger model under the same memory budget and therefore presents a much better learning curve as compared to the vanilla method.

satisfy $\varphi^0_{t=1} \sim \rho_0(\varphi)$, and we take it as reference process to fine-tune.

Specifically, we consider the solution u to the optimal control problem with the objective

$$\min_u \mathbb{E}\left[\frac{1}{2}\int_0^1 \sum_a |u(t,a,\varphi^u_t)|^2 dt + U(\varphi^u_T)\right], \tag{24}$$

where $\varphi^u_t(a)$ solves the SDE

$$d\varphi^u_t(a) = C^{-1}(a)u(t,a,\varphi^u_t)dt + C^{-1/2}(a)dW_t(a) \tag{25}$$

with $\varphi^u_{t=0}(a) = 0, U(\varphi^u_T) = E(\varphi^u_T) - E_0(\varphi^u_T)$ and $T = 1$. By Proposition 3, if we use the optimal control in (44), we have that $\varphi^u_{t=1}(a)$ samples the PDF (20).

Numerical Results: We tested various setups and report the results of some of them here. With the same setup as described in Albergo & Vanden-Eijnden (2025), we choose $L = 16$, $\alpha = 2.0, \beta = -2.0, \gamma = 3.2$ so that the lattice system is at the phase transition and in the ordered phase.

Compared to the experiments done in 3.1, this fine-tuning task has a much higher dimensionality and therefore requires more expressive neural networks for effective training. For such situations, the vanilla method no longer scales efficiently. We performed experiments comparing our method to the vanilla method *while maintaining the same memory constraints*. Our method significantly reduces memory usage compared to the vanilla approach, which relies on storing the entire computational graph during SDE integration. This reduction allows our method to train models with a much larger number of parameters, resulting in superior performance. The results of these experiments are summarized in Figure 2.

4 Conclusion

We have introduced a simulation-free on-policy approach to SOC problems: we simulate trajectories using the actual control but detach this control for the computational graph when computing the gradient of the objective. This yields an efficient, scalable method for training deep neural networks to learn feedback controls, outperforming traditional vanilla approaches. We demonstrated its effectiveness across SOC and sampling applications, including Föllmer processes and diffusion model fine-tuning.

REPRODUCIBILITY STATEMENT

All experiments done in this work rely on simply feed-forward neural networks, and can be done locally on a single GPU. Details for the network sizes are given in each experimental subsection.

ACKNOWLEDGEMENTS

We thank Michael Albergo, Joan Bruna, and Carles Domingo-Enrich for helpful discussions. EVE is supported by the National Science Foundation under Awards DMR1420073, DMS-2012510, and DMS-2134216, by the Simons Collaboration on Wave Turbulence, Grant No. 617006, and by a Vannevar Bush Faculty Fellowship.

REFERENCES

Philippe Aghion and Peter Howitt. A model of growth through creative destruction. *Econometrica*, 60(2):323–351, 1992.

M. S. Albergo, G. Kanwar, and P. E. Shanahan. Flow-based generative models for markov chain monte carlo in lattice field theory. *Phys. Rev. D*, 100:034515, Aug 2019. doi: 10.1103/PhysRevD.100.034515. URL https://link.aps.org/doi/10.1103/PhysRevD.100.034515.

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

Michael S. Albergo and Eric Vanden-Eijnden. Nets: A non-equilibrium transport sampler, 2025. URL https://arxiv.org/abs/2410.02711.

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions, 2023. URL https://arxiv.org/abs/2303.08797.

Michael Arbel, Alex Matthews, and Arnaud Doucet. Annealed flow transport monte carlo. In *International Conference on Machine Learning*, pp. 318–330. PMLR, 2021.

Oleg Arenz, Philipp Dahlinger, Zihan Ye, Michael Volpp, and Gerhard Neumann. A unified perspective on natural gradient variational inference with gaussian mixture models. *arXiv preprint arXiv:2209.11533*, 2022.

Denis Blessing, Xiaogang Jia, Johannes Esslinger, Francisco Vargas, and Gerhard Neumann. Beyond elbos: A large-scale evaluation of variational methods for sampling. *arXiv preprint arXiv:2406.07423*, 2024.

Yifan Chen, Mark Goldstein, Mengjian Hua, Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Probabilistic forecasting with stochastic interpolants and föllmer processes, 2024. URL https://arxiv.org/abs/2403.13724.

Yongxin Chen, Tryphon Georgiou, and Michele Pavon. On the relation between optimal transport and schrödinger bridges: A stochastic control viewpoint, 2014. URL https://arxiv.org/abs/1412.4430.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *The Twelfth International Conference on Learning Representations*, 2024.

Carles Domingo-Enrich. A taxonomy of loss functions for stochastic optimal control, 2024. URL https://arxiv.org/abs/2410.00345.

Carles Domingo-Enrich, Jiequn Han, Brandon Amos, Joan Bruna, and Ricky TQ Chen. Stochastic optimal control matching. *arXiv preprint arXiv:2312.02027*, 2023.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control, 2024. URL https://arxiv.org/abs/2409.08861.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Wendell H Fleming and Raymond W Rishel. *Deterministic and stochastic optimal control*, volume 1. Springer Science & Business Media, 2012.

Wendell H Fleming and Jerome L Stein. Stochastic optimal control, international finance and debt. *Journal of Banking & Finance*, 28(5):979–996, 2004. ISSN 0378-4266. doi: https://doi.org/10.1016/S0378-4266(03)00138-9. URL https://www.sciencedirect.com/science/article/pii/S0378426603001389.

H Föllmer. Time reversal on wiener space. *Stochastic Processes—Mathematics and Physics*, pp. 119–129, 1986.

Maximilien Germain, Huyên Pham, and Xavier Warin. Neural networks-based algorithms for stochastic control and PDEs in finance. *Machine Learning And Data Sciences For Financial Markets: A Guide To Contemporary Practices*, 2021.

Emmanuel Gobet and Rémi Munos. Sensitivity analysis using itô–malliavin calculus and martingales, and application to stochastic optimal control. *SIAM Journal on control and optimization*, 43(5): 1676–1713, 2005.

Jiequn Han and Weinan E. Deep learning approximation for stochastic control problems. *arXiv preprint arXiv:1611.07422*, 2016.

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

Carsten Hartmann, Ralf Banisch, Marco Sarich, Tomasz Badowski, and Christof Schütte. Characterization of rare events in molecular dynamics. *Entropy*, 16(1):350–376, 2013.

Carsten Hartmann, Lorenz Richter, Christof Schütte, and Wei Zhang. Variational characterization of free energy: Theory and algorithms. *Entropy*, 19(11), 2017. ISSN 1099-4300. doi: 10.3390/e19110626. URL https://www.mdpi.com/1099-4300/19/11/626.

Lars Holdijk, Yuanqi Du, Ferry Hooft, Priyank Jaini, Bernd Ensing, and Max Welling. Stochastic optimal control for collective variable free sampling of molecular transition paths, 2023. URL https://arxiv.org/abs/2207.02149.

Ruimeng Hu and Mathieu Lauriere. Recent developments in machine learning methods for stochastic control and games. *To appear in Numerical Algebra, Control and Optimization (arXiv preprint arXiv:2303.10257)*, 2024.

Jian Huang, Yuling Jiao, Lican Kang, Xu Liao, Jin Liu, and Yanyan Liu. Schrödinger-Föllmer sampler: sampling without ergodicity. *arXiv preprint arXiv:2106.10880*, 2021.

Côme Huré, Huyên Pham, and Xavier Warin. Deep backward schemes for high-dimensional nonlinear pdes. *Mathematics of Computation*, 89(324):1547–1579, 2020.

Côme Huré, Huyên Pham, Achref Bachouch, and Nicolas Langrené. Deep neural networks algorithms for stochastic control problems on finite horizon: convergence analysis. *SIAM Journal on Numerical Analysis*, 59(1):525–557, 2021.

Shaolin Ji, Shige Peng, Ying Peng, and Xichuan Zhang. Three algorithms for solving high-dimensional fully coupled fbsdes through deep learning. *IEEE Intelligent Systems*, 35(3):71–84, 2020.

Yuling Jiao, Lican Kang, Yanyan Liu, and Youzhou Zhou. Convergence analysis of Schrödinger-Föllmer sampler without convexity. *arXiv preprint arXiv:2107.04766*, 2021.

Ioannis Karatzas and Steven E. Shreve. *Brownian Motion and Stochastic Calculus*, volume 113 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2 edition, 1991. ISBN 0-387-97655-8.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Christian Léonard. A survey of the schrödinger problem and some of its connections with optimal transport. *Discrete & Continuous Dynamical Systems-A*, 34(4):1533–1574, 2014.

Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David K Duvenaud. Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*, pp. 1–28. PMLR, 2020.

Han Cheng Lie. Fréchet derivatives of expected functionals of solutions to stochastic differential equations, 2021. URL https://arxiv.org/abs/2106.09149.

Laurence Illing Midgley, Vincent Stimper, Gregor NC Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. *arXiv preprint arXiv:2208.01893*, 2022.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020. URL http://jmlr.org/papers/v21/19-346.html.

Richard E Mortensen. Stochastic optimal control: Theory and application (robert f. stengel), 1989.

Nikolas Nusken, Francisco Vargas, Shreyas Padhy, and Denis Blessing. Transport meets variational inference: Controlled monte carlo diffusions. In *The Twelfth International Conference on Learning Representations: ICLR 2024*, 2024.

Nikolas Nüsken and Lorenz Richter. Solving high-dimensional hamilton-jacobi-bellman pdes using neural networks: perspectives from the theory of controlled diffusions and measures on path space, 2023. URL https://arxiv.org/abs/2005.05409.

NG Pavlov, S Koptyaev, GV Lihachev, AS Voloshin, AS Gorodnitskiy, MV Ryabko, SV Polonsky, and ML Gorodetsky. Narrow-linewidth lasing and soliton kerr microcombs with ordinary laser diodes. *Nature Photonics*, 12(11):694–698, 2018.

Huyên Pham. *Continuous-time stochastic control and optimization with financial applications*, volume 61. Springer Science & Business Media, 2009.

Jannes Quer and Enric Ribera Borrell. Connecting stochastic optimal control and reinforcement learning, 2024. URL https://arxiv.org/abs/2211.02474.

Enric Ribera Borrell, Jannes Quer, Lorenz Richter, and Christof Schütte. Improving control based importance sampling strategies for metastable diffusions via adapted metadynamics. *SIAM Journal on Scientific Computing*, 46(2):S298–S323, 2024. doi: 10.1137/22M1503464.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Evangelos Theodorou, Freek Stulp, Jonas Buchli, and Stefan Schaal. An iterative path integral stochastic optimal control approach for learning robotic tasks. *IFAC Proceedings Volumes*, 44(1): 11594–11601, 2011.

Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control, 2024.

Ramon Van Handel. Stochastic calculus, filtering, and stochastic control. *Course notes., URL http://www. princeton. edu/rvan/acm217/ACM217. pdf*, 14, 2007.

Francisco Vargas, Will Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. *arXiv preprint arXiv:2302.13834*, 2023a.

Francisco Vargas, Andrius Ovsianas, David Fernandes, Mark Girolami, Neil D Lawrence, and Nikolas Nüsken. Bayesian learning via neural schrödinger–föllmer flows, 2023b.

Ingmar Vierhaus. *Simulation of ϕ^4 theory in the strong coupling expansion beyond the Ising Limit*. PhD thesis, Humboldt University of Berlin, 07 2010.

Ronald J. Williams. Toward a theory of reinforcement-learning connectionist systems. Technical Report NU-CCS-88-3, Northeastern University, College of Computer Science, 1988.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Jichuan Yang and Harold J. Kushner. A monte carlo method for sensitivity analysis and parametric optimization of nonlinear stochastic systems. *SIAM Journal on Control and Optimization*, 29(5): 1216–1249, 1991. doi: 10.1137/0329064.

Qinsheng Zhang and Yongxin Chen. Path integral sampler: A stochastic control approach for sampling. In *International Conference on Learning Representations*, 2022.

A Proofs of Propositions 1, 5 and 6

Proof of Proposition 1. Equation (6) follows from (4) by a direct calculation in which we first evaluate the gradient of the objective $L(\theta) = J(u^\theta)$ with the fixed reference control v, which gives:

$$\begin{aligned}\partial_\theta L(\theta) = \mathbb{E}\left(\left[\int_0^T u_t^\theta(X_t^v)\cdot\partial_\theta u_t^\theta(X_t^v)dt\right]M(u^\theta,v)\right)\\ + \mathbb{E}\left[\left(\int_0^T\left(\tfrac{1}{2}|u_t^\theta(X_t^\theta)|^2 + f_t(X_t^\theta)\right)dt + g(X_T^v)\right)\right.\\ \left.\times\left(\int_0^T\partial_\theta u_t^\theta(X_t^v)\cdot dW_t + \int_0^T(v_t(X_t^v)-u^\theta(X_t^v))\cdot\partial_\theta u_t^\theta(X_t^v)\cdot dW_t\right)M(u^\theta,v)\right].\end{aligned} \tag{26}$$

Because (26) holds for any v, we can now evaluate it at $v = u^\theta$. Since $M(u^\theta, u^\theta) = 1$, this gives (6). □

Proof of Proposition 5. It is well-known (Léonard, 2014; Chen et al., 2014) that the SOC problem specified in the proposition can be cast into solving the pair of partial differential equations

$$\partial_t\mu_t = -\nabla\cdot(b_t\mu_t) - \nabla\cdot(D_t\nabla\phi_t\mu_t) + \tfrac{1}{2}\nabla\cdot(D_t\nabla\mu_t), \qquad \mu_0 = \delta_0 \tag{27}$$

$$\partial_t\phi_t = -b_t\cdot\nabla\phi_t - \tfrac{1}{2}\nabla\phi_t\cdot D_t\nabla\phi_t - \tfrac{1}{2}\nabla\cdot(D_t\nabla\phi_t), \qquad \phi_T = r \tag{28}$$

where $D_t = D_t^T = \sigma_t\sigma_t^T$, μ_t is the distribution of X_t^u, and the potential ϕ_t gives the optimal control via $u_t = \sigma_t^T\nabla\phi_t$. We also know that the distribution ν_t of Y_t solves the Fokker-Planck equation

$$\partial_t\nu_t = -\nabla\cdot(b_t\nu_t) + \tfrac{1}{2}\nabla\cdot(D_t\nabla\nu_t), \qquad \nu_0 = \delta_0 \tag{29}$$

We will prove that $X_T^u \sim \mu$ by establishing that $\mu_t(dx) = Z^{-1}e^{\phi_t(x)}\nu_t(dx)$ since this will imply that $\mu_T = \mu$ because $\nu_T = \nu$ by definition and $\phi_T = r$ by construction, so that $\mu_T = Z^{-1}e^{\phi_T}\nu_T =$

$Z^{-1}e^r\nu = \mu$. Let $\hat{\mu}_t(dx) = Z^{-1}e^{\phi_t(x)}\nu_t(dx)$. Then we have

$$\begin{aligned}
\partial_t\hat{\mu}_t &= Z^{-1}\left(\partial_t\phi_t e^{\phi_t}\nu_t + e^{\phi_t}\partial_t\nu_t\right),\\
-\nabla\cdot(b_t\hat{\mu}_t) &= Z^{-1}\left(-e^{\phi_t}\nabla\cdot(b_t\nu_t) - b_t\cdot\nabla\phi_t e^{\phi_t}\nu_t\right),\\
-\nabla\cdot(D_t\nabla\phi_t\hat{\mu}_t) &= Z^{-1}\left(-e^{\phi_t}\nabla\cdot(D_t\nabla\phi_t\nu_t) - e^{\phi_t}\nabla\phi_t\cdot D_t\nabla\phi_t\nu_t\right)\\
&= Z^{-1}\left(-e^{\phi_t}\nabla\cdot(D_t\nabla\phi_t)\nu_t - e^{\phi_t}\nabla\phi_t\cdot D_t\nabla\nu_t - e^{\phi_t}\nabla\phi_t\cdot D_t\nabla\phi_t\nu_t\right),\\
\tfrac{1}{2}\nabla\cdot(D_t\nabla\hat{\mu}_t) &= Z^{-1}\left(\tfrac{1}{2}e^{\phi_t}\nabla\cdot(D_t\nabla\nu_t) + e^{\phi_t}\nabla\phi_t\cdot D_t\nabla\nu_t\right.\\
&\quad\left.+\tfrac{1}{2}e^{\phi_t}\nabla\phi_t\cdot D_t\nabla\phi_t\nu_t + \tfrac{1}{2}e^{\phi_t}\nabla\cdot(D_t\nabla\phi_t)\nu_t\right).
\end{aligned} \tag{30}$$

Inserting these expressions in the left-hand side and the right-hand side of (27) and using (29), several terms cancel we are left with

$$\partial_t\phi_t e^{\phi_t}\nu_t = -b_t\cdot\nabla\phi_t e^{\phi_t}\nu_t - \tfrac{1}{2}\nabla\phi_t\cdot D_t\nabla\phi_t e^{\phi_t}\nu_t - \tfrac{1}{2}e^{\phi_t}\nabla\cdot(D_t\nabla\phi_t)\nu_t. \tag{31}$$

If we divide both sides of this equation by $e^{\phi_t}\nu_t$, we recover (28). This shows that $\hat{\mu}_t(dx) = Z^{-1}e^{\phi_t(x)}\nu_t(dx)$ is indeed a solution to the PDE (27). To show that it is the solution, it remains to establish that it satisfies the initial condition in (27). To this end, notice first that, since $\nu_0 = \delta_0$, we have

$$\hat{\mu}_0(dx) = Z^{-1}e^{\phi_0(x)}\nu_0(dx) = Z^{-1}e^{\phi_0(0)}\delta_0(dx) \tag{32}$$

Second, since $\hat{\mu}_t$ satisfies (27), we must have $\int_{\mathbb{R}^d}\mu_t(dx) = 1$ for all $t\in[0,1]$. As a result, we conclude that $e^{\phi_0(0)} = Z$, which means that $\hat{\mu}_0 = \delta_0 = \mu_0$. Since the solution pair (μ_t,ϕ_t) to (27)-(28) is unique, we must have $\mu_t = \hat{\mu}_t = Z^{-1}e^{\phi_t}\nu_t$ and hence $\mu_T = Z^{-1}e^{\phi_T}\nu_T = Z^{-1}e^r\nu$. □

Note that it is key that $\mu_0 = \nu_0 = \delta_0$ (more generally δ_{x_0} for some $x_0\in\mathbb{R}^d$). If the base distribution used to generate initial data in the SDEs (3) and (16) are not atomic at $x = 0$, the statement of Proposition 5 does not hold anymore, because the second equality in (32) fails. That is, our framework only allows to fine-tune generative models that use a Dirac delta distribution as base distribution.

Proof of Proposition 6. By direct application of Girsanov theorem, we have

$$\mathbb{E}_{X^u}\left[h(X_T^u)M_r(u)\right] = \mathbb{E}_Y\left[h(Y_T)e^{r(Y_T)}\right] = \int_{\mathbb{R}^d}h(x)e^{r(x)}\nu(dx), \tag{33}$$

where Y_t solves (16) and we used $Y_T\sim\nu$ to get the second equality. Multiplying both sides of (33) by Z^{-1} we deduce

$$Z^{-1}\mathbb{E}_{X^u}\left[h(X_T^u)M_r(u)\right] = Z^{-1}\int_{\mathbb{R}^d}h(x)e^{r(x)}\nu(dx) = \int_{\mathbb{R}^d}h(x)\mu(dx), \tag{34}$$

which gives the first equation in (18). Setting $h = 1$ in (33) we deduce that

$$\mathbb{E}_{X^u}\left[M_r(u)\right] = \int_{\mathbb{R}^d}e^{r(x)}\nu(dx) = Z \tag{35}$$

which gives the second equation in eq. (18). To establish that $M_r(u) = Z$ *iff* u is the optimal control minimizing the SOC problem specified in Proposition 5, notice that $X_T^u\sim\mu$ *iff* u is this optimal control. Assuming that this is the case, the first equation in eq. (18) implies that

$$\mathbb{E}_{X^u}\left[h(X_T^u)\right] = Z^{-1}\mathbb{E}_{X^u}\left[h(X_T^u)M_r(u)\right] \tag{36}$$

for all suitable test function h. This can only hold if $M_r(u) = Z$. □

B Fourier Representation of the φ^4 Model

We define the discrete Fourier transform with the following:

$$\hat{\varphi}(k) = L^{-d/2}\sum_a e^{2i\pi k\cdot a/L}\varphi(a) \quad\Leftrightarrow\quad \varphi(a) = L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{\varphi}(k) \tag{37}$$

where $a, k \in [0, \dots, L-1]^d$, we can write the energy (22) as

$$E_0(\varphi) = \hat{E}_0(\hat{\varphi}) \equiv \frac{1}{2}\sum_k \hat{M}(k)|\hat{\varphi}(k)|^2, \qquad \hat{M}(k) = 2\alpha\left(d - \sum_{\hat{e}} \cos(2\pi k \cdot \hat{e}/L)\right) + \beta_0, \quad (38)$$

where $\hat{e}$ denotes the d basis vectors on the lattice. Therefore ρ_0 is a Gaussian density with covariance

$$\int_{\mathbb{R}^{L^d}} \varphi(a)\varphi(b)\rho_0(\phi)d\phi = C(a-b), \qquad C(a) = L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{M}^{-1}(k). \quad (39)$$

Let the inverse discrete Fourier transform defined by

$$\varphi_t^0(a) = L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{\varphi}_t^0(k) \quad (40)$$

where $\varphi_t^0(k)$ solves

$$d\hat{\varphi}_t^0(k) = \hat{M}^{-1/2}(k)d\hat{W}_t(k), \qquad \hat{\varphi}_{t=0}^0(k) = 0 \quad (41)$$

in which $\hat{W}$ is the Fourier transform of a Wiener process W with covariance $\mathbb{E}[W_t(a)dW_s(b)] = \delta_{a,b}\min(t,s)$. Then $\varphi_{t=1}^0 \sim \rho_0$. Let also

$$\begin{aligned} U(\varphi) &= \frac{1}{2}(\beta-\beta_0)\sum_a |\varphi(a)|^2 + \frac{1}{4}\gamma\sum_a |\varphi(a)|^4 \\ &= \hat{U}(\hat{\varphi}) = \frac{1}{2}(\beta-\beta_0)\sum_k |\hat{\varphi}(k)|^2 + \frac{1}{4}\gamma\sum_a \left| L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{\varphi}(k)\right|^4 \end{aligned} \quad (42)$$

where φ and $\hat{\varphi}$ are Fourier transform pairs as defined in (37): the last term can be implemented via $\sum_a (\text{ifft}(\hat{\varphi}))^4(a)$. Then, we can derive the Fourier representation of the optimal control problem with objective (24) and controlled process (25):

$$\min_{\hat{u}} \mathbb{E}[\frac{1}{2}\int_0^1 \sum_k |\hat{u}(t,k,\hat{\varphi}_t^u)|^2 dt + \hat{U}(\hat{\varphi}_T^u)] \quad (43)$$

where $\hat{\varphi}_t^u(k)$ solves the controlled process

$$d\hat{\varphi}_t^u(k) = \hat{M}^{-1}(k)\hat{u}(t,k,\hat{\varphi}_t^u)dt + \hat{M}^{-1/2}(k)d\hat{W}_t(k), \qquad \hat{\varphi}_{t=0}^u(k) = 0 \quad (44)$$

Then by Proposition 3, if we use the optimal control in (25), we have that

$$\varphi_{t=1}^u(a) = L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{\varphi}_{t=1}^u(k) \quad (45)$$

sample the PDF (20).

Sampling using the Langevin SDE: To obtain the ground-truth samples from the φ^4 model, one option is to use the SDE

$$d\hat{\varphi}_t(k) = -\hat{M}(k)\hat{\varphi}_t(k)dt - (\beta-\beta_0)\hat{\varphi}_t(k)dt - \gamma\widehat{\varphi_t^3}(k)dt + \sqrt{2}d\hat{W}_t(k). \quad (46)$$

where we denote

$$\widehat{\varphi_t^3}(k) = L^{-d/2}\sum_a e^{2i\pi k\cdot a/L}\left(L^{-d/2}\sum_k e^{-2i\pi k\cdot a/L}\hat{\varphi}_t(k)\right)^3 \quad (47)$$

which can be implenemted via $\text{fft}((\text{ifft}(\hat{\varphi}_t))^3)$. This SDE may be quite stiff, however, a problem that can be alleviated by changing the mobility and using instead

$$d\hat{\varphi}_t(k) = -\hat{\varphi}_t(k)dt - (\beta-\beta_0)\hat{M}^{-1}(k)\hat{\varphi}_t(k)dt - \gamma\hat{M}^{-1}(k)\widehat{\varphi_t^3}(k)dt + \sqrt{2}\hat{M}^{-1/2}(k)d\hat{W}_t(k). \quad (48)$$

The discretized version of this equation reads

$$\begin{aligned} \hat{\varphi}_{t_{n+1}}(k) = \hat{\varphi}_{t_n}(k) &- \Delta t_n\left(\hat{\varphi}_{t_n}(k) + (\beta-\beta_0)\hat{M}^{-1}(k)\hat{\varphi}_{t_n}(k) + \gamma\hat{M}^{-1}(k)\widehat{\varphi_{t_n}^3}(k)\right) \\ &+ \sqrt{2\Delta t_n}\hat{M}^{-1/2}(k)\hat{\eta}_n(k), \end{aligned} \quad (49)$$

where $\hat{\eta}_n$ is the Fourier transform of $\eta_n \sim N(0, \text{Id})$.

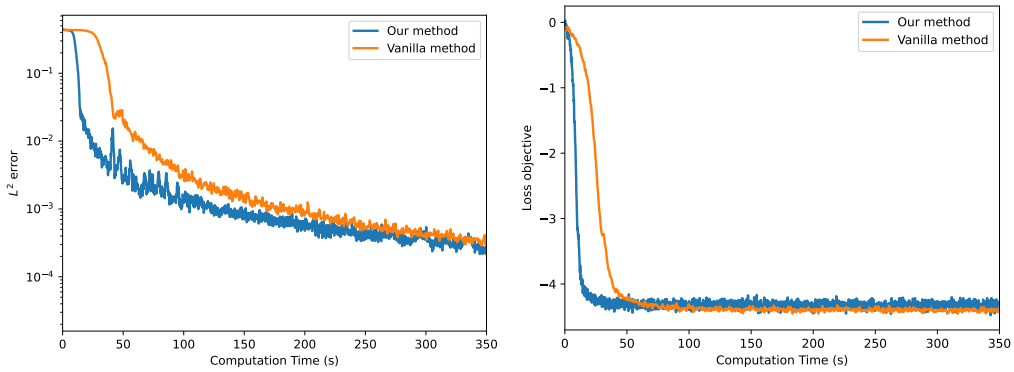

Figure 3: **Linear Ornstein-Uhlenbeck Example**: Our method outperforms the vanilla method in terms of convergence rate measured by the squared L^2 error (top panel) and the training loss (bottom panel).

Model	Memory Cost (GB)	Back-Prop Runtime (s)
Our Method	**0.962** ± 0.001	**0.003** ± 0.000
Vanilla Method	2.590 ± 0.001	0.177 ± 0.006

Table 2: **Linear Ornstein-Uhlenbeck Example**: Comparison between our method and the vanilla method in terms of the GPU memory usage and runtime for one back-propagation pass. Here, we use a mini-batch size of 5k and 256 time steps.

C Additional Numerical Examples

C.1 Linear Ornstein-Uhlenbeck Example

We consider the SOC problem in (1) with $f = 0$, $g(x) = \gamma \cdot x$, $\sigma_t = \sigma = cst$, $\lambda = 1$ and $b_t(x) = Ax$, where $\gamma \in \mathbb{R}^d$ and $\sigma, A \in \mathbb{R}^d \times \mathbb{R}^d$. This example was proposed by Nüsken & Richter (2023) and its optimal control can be calculated analytically:

$$u_t^*(x) = -\sigma_0^\top \exp(A^\top (T-t))\gamma.$$

We set $d = 20$ with initial samples $X_0 \sim \mathcal{N}(0, \frac{1}{2}\mathrm{Id})$. The control $u_t(x)$ is parameterized using a fully connected MLP with 4 layers of 128 hidden dimensions, initialized using PyTorch defaults. Optimization uses Adam (Kingma, 2014) with learning rate $3 \cdot 10^{-4}$ and cosine annealing. For comparison, we implement the vanilla method requiring SDE differentiation under identical conditions. Performance is evaluated using the squared L^2 error between learned and true controls.

$$E = \int_0^T \mathbb{E}\left[|u_t(X_t^*) - u_t^*(X_t^*)|^2\right] dt \tag{50}$$

where X^* is generated with the optimal control u^* and the expectation is estimated via Monte-Carlo sampling over 256 trajectories.

The numerical results are shown in Figure 3 . Compared with the vanilla method, our method achieves comparable accuracy faster and at lower memory cost (see Table 2 for a detailed comparison in terms of memory cost and computational time).

C.2 Quadratic Ornstein-Uhlenbeck Example

Next, we consider a more complicated case where the SOC objective includes a quadratic running cost: $f(x) = x^T P x$, $g(x) = x^T Q x$, $b_t(x) = Ax$, $\sigma_t = \sigma_0$, where $P, Q, A \in \mathbb{R}^d \times \mathbb{R}^d$. This type of SOC problems are often referred to as linear quadratic regulator (LQR) and they have closed-form analytical solution (Van Handel, 2007, Chapter 7):

$$u_t^*(x) = -2\sigma_0^\top F_t x,$$

Model	Memory Cost (GB)	Back-Prop Runtime (s)
Our Method	**1.260** ± 0.001	**0.0034** ± 0.0003
Vanilla Method	3.590 ± 0.001	0.195 ± 0.003

Table 3: **Quadratic Ornstein-Uhlenbeck Example**: Comparison between our method and the vanilla method in terms of the GPU memory usage and runtime for one back-propagation pass. Here, we use a mini-batch size of 512 and 256 time steps.

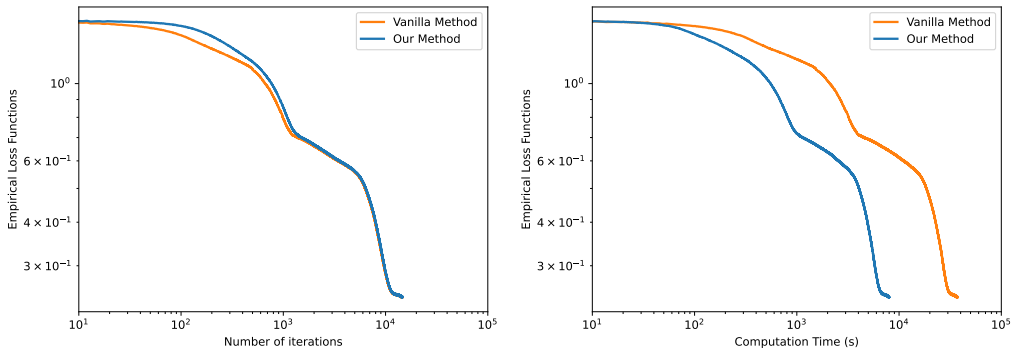

Figure 4: **Quadratic Ornstein-Uhlenbeck Example**: L_2^2 error in terms of the number of training iterations (left panel) and the GPU compute time (right panel).

where F_t solves the Riccati equation

$$\frac{dF_t}{dt} + A^\top F_t + F_t A - 2|\sigma_0^\top F_t|^2 + P = 0$$

with the final condition $F_T = Q$. We consider this example investigated by Domingo-Enrich et al. (2023) with the following configuration:

$$d = 400,\ A = I,\ P = I,\ Q = 0.5I,\ \sigma_0 = I,\ \lambda = 1,\ T = 10,\ X_0 \sim N(0, \tfrac{1}{2}\mathrm{I}).$$

However, compared to Domingo-Enrich et al. (2023), we scale the dimensions from $d = 20$ to $d = 400$ and time horizon from $T =$ to $T = 10$, significantly increasing the task complexity. The neural network parameterization and initialization for $u_t(x)$ follow Sec. C.1. Table 3 compares memory consumption and computational cost between our method and the vanilla approach, while Figure 4 contrasts their L^2 accuracy and training time. With equal computation time, our method achieves better L^2 accuracy and maintains identical learning curves but with faster execution. Notably, our method demonstrates superior scalability as it converges in under 3 hours (< 10000 seconds) while the vanilla approach requires over 14 hours (> 50000 seconds).