Stochastic Optimal Control Matching

Carles Domingo-Enrich, Jiequn Han, Brandon Amos, Joan Bruna, Ricky T. Q. Chen

Thomas Mousseau

August 18, 2025

Overview

1. **Setup and Preliminaries**
2. **Stochastic Optimal Control Matching**
3. **Experiments and results**
4. **Conclusion**

Neural ODE: Continuous Normalizing Flows

Continuous Normalizing Flows (CNF)

CNFs model complex distributions by transforming a simple distribution (e.g., Gaussian) through a continuous-time ODE. The transformation is defined by a neural network that learns the dynamics of the flow.

PAS COMPLETE

Key Idea

Instead of discrete steps, CNFs use a continuous-time approach to model the evolution of the distribution, allowing for more flexible and expressive transformations.

Evolution of Generative Models

2020 **DDPM:** Denoising Diffusion Probabilistic Models interpret generation as reversing a discrete noise-adding process, learning to denoise at each step. They produced high-quality samples but required thousands of slow sampling steps.

2021 **Score-based Models:** Score-based generative models extended diffusion to continuous-time SDEs, learning the score function ($\nabla_x \log p_t(x)$) to reverse a stochastic diffusion process. This unified diffusion with stochastic control, allowed probability flow ODEs, and sped up sampling.

2023 **Flow Matching:** Flow matching views generation as learning a deterministic ODE vector field that directly transports a simple distribution (e.g., Gaussian) to data. This removed stochasticity and significantly improved efficiency compared to diffusion/score methods.

What is a Stochastic Control Problem?

A stochastic control problem involves finding an optimal control policy to steer a dynamical system under uncertainty.

Key Components

- **State Process:** $X_t \in \mathbb{R}^d$ (position in state space at time t)
- **Control Process:** $u_t \in \mathbb{R}^d$ (action/decision at time t)
- **Noise Process:** W_t (random disturbances, typically Brownian motion)

Dynamics (SDE)

$$dX_t = f(X_t, u_t, t)dt + g(X_t, t)dW_t \quad (1)$$

Cost Function

$$J(u) = \mathbb{E}\left[\int_0^T L(X_t, u_t, t)dt + \Phi(X_T)\right] \quad (2)$$

The Goal: Finding Optimal Control

Optimal Control u^*

Find the control policy u^* that minimizes the expected cost: $u^* = \arg\min_u J(u)$

Classical Approaches

- **Hamilton-Jacobi-Bellman (HJB) equation:** Partial differential equation approach
- **Dynamic Programming:** Discrete-time recursive approach

Challenge

These classical methods become computationally intractable in high dimensions due to the *curse of dimensionality*.

Reasons behind SOCM (1/2)

Many fundamental tasks in machine learning can be naturally cast as stochastic optimal control problems, highlighting the importance of efficient SOC methods.

Key ML Applications of SOC

- **Reward fine-tuning of diffusion and flow models:** Optimizing generation quality using reward signals
- **Conditional sampling on diffusion and flow models:** Steering generation towards specific conditions or constraints
- **Sampling from unnormalized densities:** Efficiently drawing samples from complex, intractable distributions
- **Importance sampling of rare events in SDEs:** Computing probabilities of low-probability but critical events

Reasons behind SOCM (2/2)

Current SOC methods suffer from optimization challenges that limit their effectiveness.

Current SOC Methods

- Use **adjoint methods** (like CNFs)
- Yield **non-convex** function landscapes
- Difficult optimization with local minima
- Unstable training dynamics

Diffusion Models Success

- Use **least-squares loss**
- Create **convex** functional landscapes
- Stable and reliable optimization
- Excellent empirical performance

SOCM's Innovation

Goal: Develop least-squares loss formulations for SOC problems, combining the expressiveness of stochastic control with the optimization stability of diffusion models.

SOCM in Context: Optimization Landscapes

Task	Non-convex	Least Squares
Generative Modeling	Maximum Likelihood CNFs	Diffusion models and Flow Matching
Stochastic Optimal Control	Adjoint Methods	**Stochastic Optimal Control Matching**

Introducing Stochastic Optimal Control Matching

SOCM offers a more principled, stable, and accurate way to learn generative dynamics by blending stochastic control theory with modern matching-based generative modeling.

Key Novel Contributions

1. **Controlled Stochastic Process:** Views the generation process as a controlled stochastic process bridging a simple distribution to data.
2. **Least-Squares Matching:** Learning the control via least-squares matching, a stable and convex regression objective.
3. **Joint Optimization:** Optimizing control and variance-reducing reparameterization matrices simultaneously, for efficient learning.
4. **Path-wise Reparameterization:** Introducing a path-wise reparameterization trick, boosting gradient estimation quality.

The SOCM Framework

SOCM Loss Function

The Stochastic Optimal Control Matching objective is defined as:

$$\mathcal{L}_{SOCM}(u, M) := \mathbb{E}\left[\frac{1}{T}\int_0^T \|u(X_t^v, t) - w(t, v, X^v, B, M_t)\|^2 dt \times \alpha(v, X^v, B)\right] \tag{3}$$

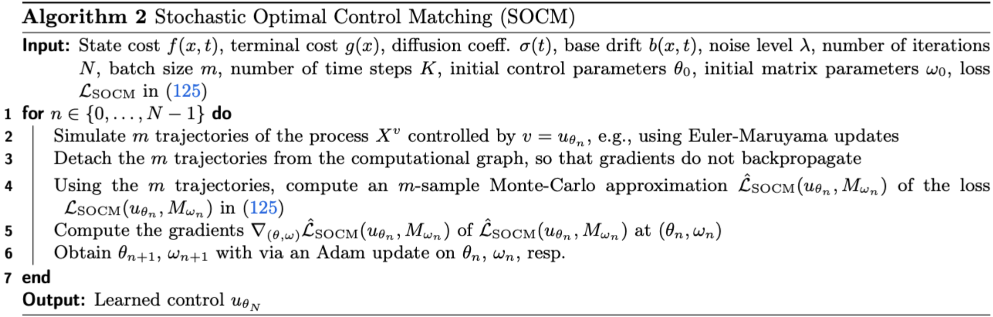
Where:

- X^v is the process controlled by v:

$$dX_t^v = (b(X_t^v, t) + \sigma(t)v(X_t^v, t))dt + \sqrt{\lambda}\sigma(t)dB_t \tag{4}$$

  with $X_0^v \sim p_0$
- $u(X_t^v, t)$ is the control policy being learned
- $w(t, v, X^v, B, M_t)$ is the target matching function
- $\alpha(v, X^v, B)$ is a weighting function

SOCM Algorithm

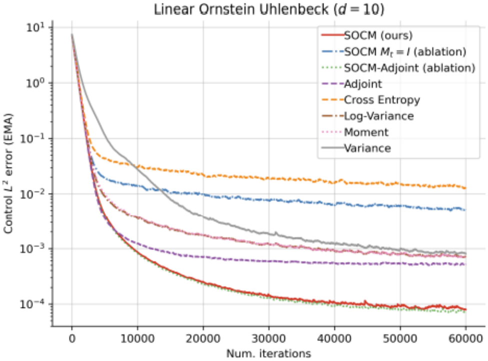


Algorithm 2 Stochastic Optimal Control Matching (SOCM)

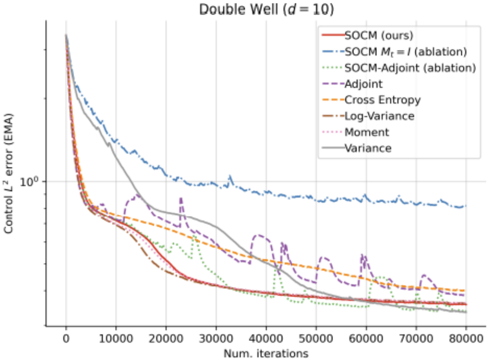
```
Input: State cost $f(x, t)$, terminal cost $g(x)$, diffusion coeff. $\sigma(t)$, base drift $b(x, t)$, noise level $\lambda$, number of iterations $N$, batch size $m$, number of time steps $K$, initial control parameters $\theta_0$, initial matrix parameters $\omega_0$, loss $\mathcal{L}_{\mathrm{SOCM}}$ in (125)
for $n \in \{0, \ldots, N-1\}$ do
    Simulate $m$ trajectories of the process $X^v$ controlled by $v = u_{\theta_n}$, e.g., using Euler-Maruyama updates
    Detach the $m$ trajectories from the computational graph, so that gradients do not backpropagate
    Using the $m$ trajectories, compute an $m$-sample Monte-Carlo approximation $\hat{\mathcal{L}}_{\mathrm{SOCM}}(u_{\theta_n}, M_{\omega_n})$ of the loss $\mathcal{L}_{\mathrm{SOCM}}(u_{\theta_n}, M_{\omega_n})$ in (125)
    Compute the gradients $\nabla_{(\theta,\omega)}\hat{\mathcal{L}}_{\mathrm{SOCM}}(u_{\theta_n}, M_{\omega_n})$ of $\hat{\mathcal{L}}_{\mathrm{SOCM}}(u_{\theta_n}, M_{\omega_n})$ at $(\theta_n, \omega_n)$
    Obtain $\theta_{n+1}$, $\omega_{n+1}$ with via an Adam update on $\theta_n$, $\omega_n$, resp.
end
Output: Learned control $u_{\theta_N}$
```

Figure: Stochastic Optimal Control Matching (SOCM) Algorithm

Experimental Results (1/2)



Linear Ornstein Uhlenbeck ($d = 10$)



Double Well ($d = 10$)

Experimental Results (2/2)

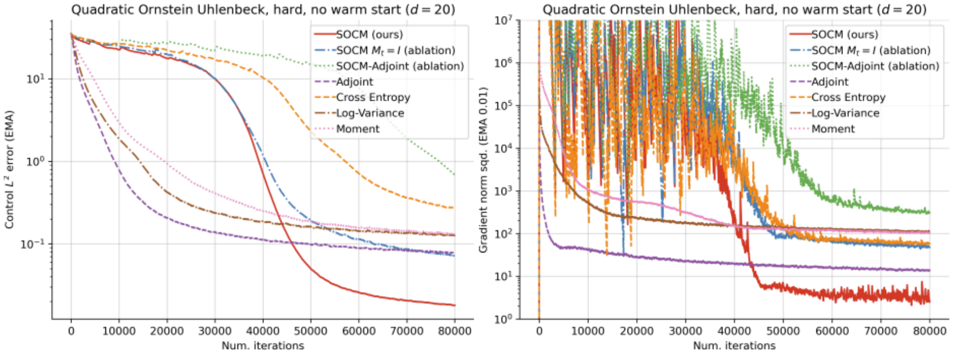
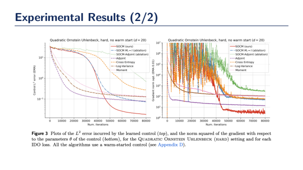


Figure 3 Plots of the L^2 error incurred by the learned control (*top*), and the norm squared of the gradient with respect to the parameters θ of the control (*bottom*), for the Quadratic Ornstein Uhlenbeck (hard) setting and for each IDO loss. All the algorithms use a warm-started control (see Appendix D).



Experimental Results (2/2)

Figure 3 Plots of the L^2 error incurred by the learned control (*top*), and the norm squared of the gradient with respect to the parameters θ of the control (*bottom*), for the Quadratic Ornstein Uhlenbeck (hard) setting and for each IDO loss. All the algorithms use a warm-started control (see Appendix D).

At the end of training, SOCM obtains the lowest L2 error, improving over all existing methods by a factor of around ten. The two SOCM ablations come in second and third by a substantial difference, which underlines the importance of the path-wise reparameterization trick.

Conclusion

Conclusion

The main roadblock when we try to apply SOCM to more challenging problems is that the variance of the factor alpha(v, Xv, B) explodes when f and/or g are large, or when the dimension d is high. The control L2 error for the SOCM and cross-entropy losses remains high and fluctuates heavily due to the large variance of alpha The large variance of alpha is due to the mismatch between the probability measures induced by the learned control and the optimal control. Similar problems are encountered in out-of-distribution generalization for reinforcement learning, and some approaches may be carried over from that area (Munos et al., 2016).

References