On Investigating the Conservative Property of Score-Based Generative Models

Chen-Hao Chao[1], Wei-Fang Sun[1 2], Bo-Wun Cheng[1], and Chun-Yi Lee[1]

[1] Elsa Lab, Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan

[2] NVIDIA AI Technology Center (NVAITC), NVIDIA Corporation, Taiwan

Abstract

Existing Score-Based Models (SBMs) can be categorized into either **Constrained SBMs (CSBMs)** or **Unconstrained SBMs (USBMs)** according to their parameterization approaches. CSBMs assign their predictions as the negative gradients of some scalar-valued energy functions. On the other hand, USBMs employ flexible architectures capable of directly estimating scores without the need to explicitly model energy functions. In this work, we demonstrate that the architectural constraints of CSBMs may limit their modeling abilities. In addition, we show that the failure of USBMs to preserve the property of *conservativeness* may lead to degraded performance in practice. To address the above issues, we propose **Quasi-Conservative Score-Based Models (QCSBMs)** for maintaining the advantages of both CSBMs and USBMs.

Background

Score-Matching Methods

To train a score model $\mathrm{s}(\cdot\,;\theta)$, a feasible approach is to minimize the Explicit Score-Matching (ESM) loss $\mathcal{L}_{\mathrm{ESM}}$ [1], which is represented as follows:

$$\mathcal{L}_{\mathrm{ESM}} = \mathbb{E}_{p(\boldsymbol{x})}\left[\frac{1}{2}\|\mathrm{s}(\boldsymbol{x};\theta) - \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})\|^2\right], \quad (1)$$

where $\boldsymbol{x} \in \mathbb{R}^D$ represents a data sample. Based on Parzen density estimation, an efficient alternative of Eq. (1), called Denoising Score-Matching (DSM) loss [2], is derived for efficiently training $\mathrm{s}(\cdot\,;\theta)$. It is expressed as follows:

$$\mathcal{L}_{\mathrm{DSM}} = \mathbb{E}_{p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})p(\boldsymbol{x})}\left[\frac{1}{2}\|\mathrm{s}(\tilde{\boldsymbol{x}};\theta) - \nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})\|^2\right], \quad (2)$$

where $p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})$ is a Gaussian distribution, σ is the standard deviation of it, and $\nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) = \frac{1}{\sigma^2}(\boldsymbol{x} - \tilde{\boldsymbol{x}})$.

Sampling Process

Given an optimal SBM $\mathrm{s}(\boldsymbol{x};\theta) = \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, Langevin dynamics [3] enables $p(\boldsymbol{x})$ to be iteratively approximated through the following equation:

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha\, \mathrm{s}(\boldsymbol{x}_{t-1};\theta) + \sqrt{2\alpha}\, \boldsymbol{z}_t, \quad (3)$$

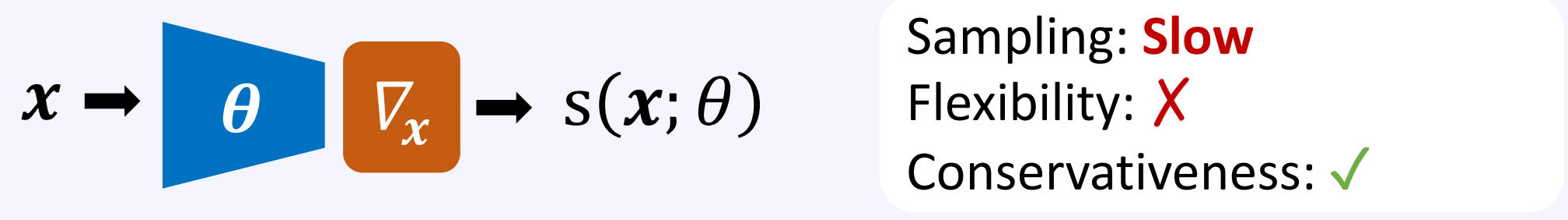
where α is the step size, t is the timestep, and $\boldsymbol{z}_t$ is a noise vector sampled from a Gaussian. Eq. (3) can be extended to a time-inhomogeneous variant by making σ, $\mathrm{s}(\cdot\,;\theta)$, and α continuous on t [4], resulting in a stochastic differential equation (SDE) defined as follows:

$$\mathrm{d}\boldsymbol{x} = [f(\boldsymbol{x},t) - g(t)^2 s(\boldsymbol{x},t;\theta)]\mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}, \quad (4)$$

where $\mathrm{d}t$ is an infinitesimal negative timestep, $\mathrm{d}\bar{\boldsymbol{w}}$ represents the Wiener process, $f(\cdot\,,t)$ and $g(t)$ are the functions representing the drift and diffusion coefficients in an SDE, respectively.

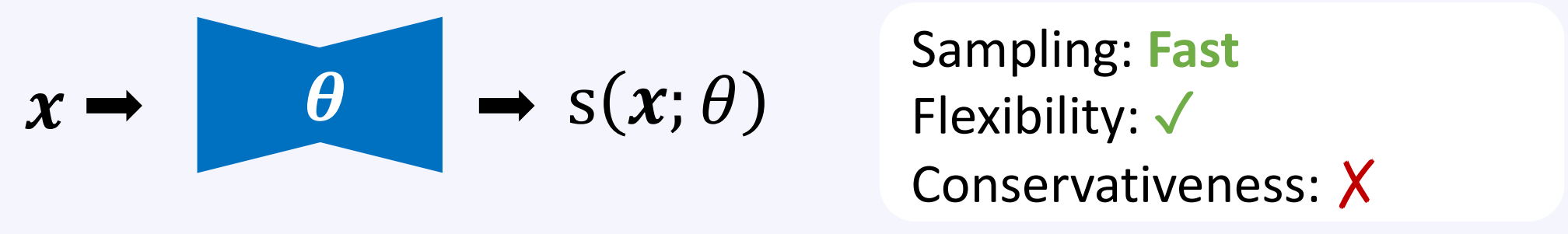
Conservativeness and Rotation Density

- **CSBM**: $\mathrm{s}(\boldsymbol{x};\theta) \triangleq \nabla_{\boldsymbol{x}} f(\boldsymbol{x};\theta)$ for some function $f: \mathbb{R}^D \to \mathbb{R}$.



Sampling: **Slow**
Flexibility: ✗
Conservativeness: ✓

- **USBM**: $\mathrm{s}(\boldsymbol{x};\theta)$ can be any mapping $\mathbb{R}^D \to \mathbb{R}^D$



Sampling: **Fast**
Flexibility: ✓
Conservativeness: ✗

The output of $\mathrm{s}(\cdot\,;\theta)$ is said to be *conservative* if and only if its Jacobian is symmetry [5], which can be expressed as the zero-rotation-density [6] condition as follows:

$$\overline{\mathrm{ROT}}_{\mathrm{ij}}\mathrm{s}(\boldsymbol{x};\theta) \triangleq \frac{\partial \mathrm{s}(\boldsymbol{x};\theta)_i}{\partial \boldsymbol{x}_j} - \frac{\partial \mathrm{s}(\boldsymbol{x};\theta)_j}{\partial \boldsymbol{x}_i} = 0, \qquad 1 \le i,j \le D. \quad (5)$$

where $\overline{\mathrm{ROT}}_{\mathrm{ij}}$ is the rotation density operator [6].

Motivational Examples

The Influences of Non-Conservativeness on the Sampling Process

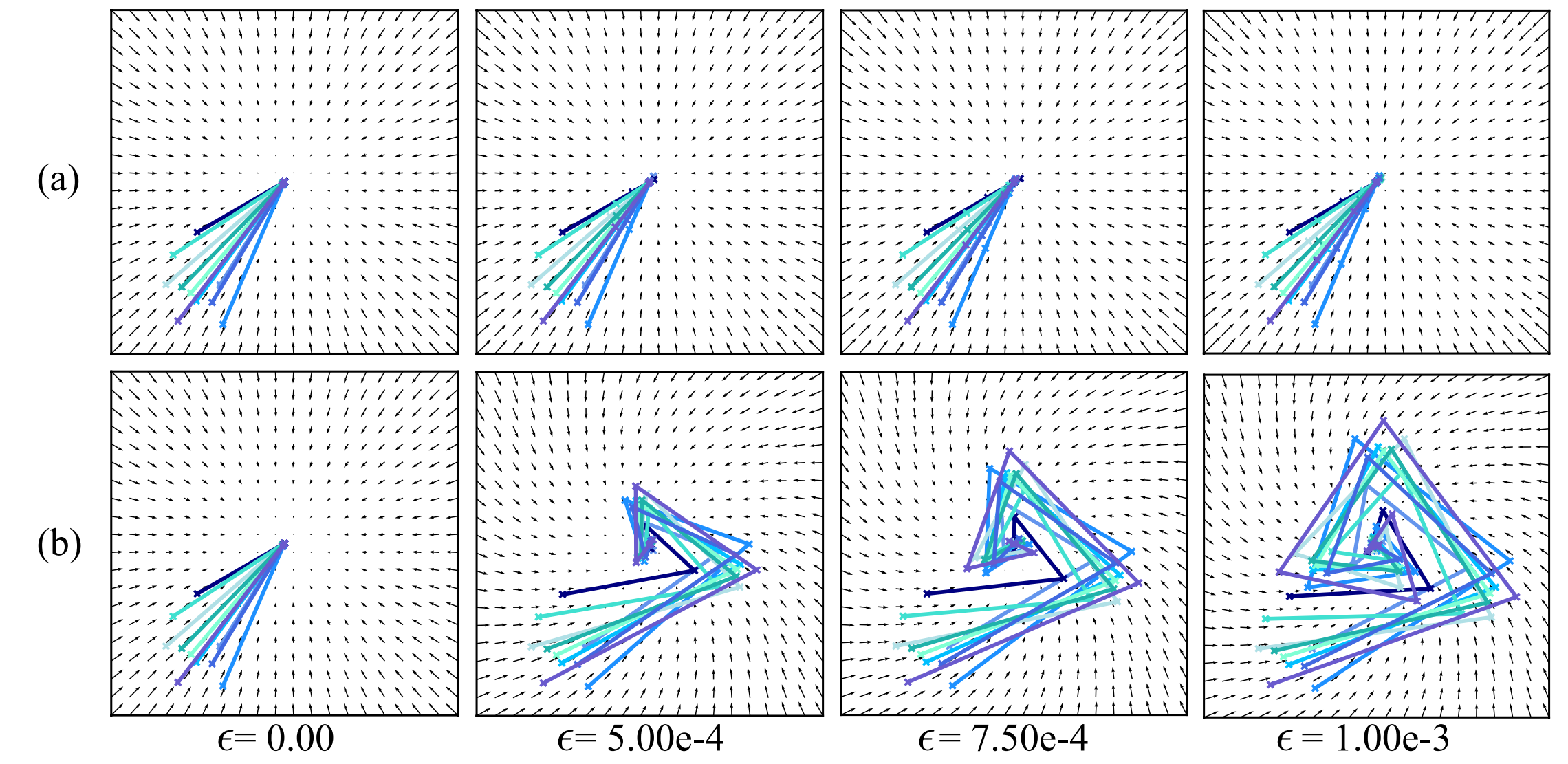
- **Asymmetry Metric (*Asym*)**:

$$\mathbb{E}_{p(\boldsymbol{x})}\left[\frac{1}{2}\sum_{i,j=1}^{D}\left(\overline{\mathrm{ROT}}_{\mathrm{ij}}\mathrm{s}(\boldsymbol{x};\theta)\right)^2\right] = \mathbb{E}_{p(\boldsymbol{x})}\left[\|\mathcal{J} - \mathcal{J}^{\mathrm{T}}\|_F^2\right]. \quad (6)$$

- **Normalized Asymmetry Metric (*NAsym*)**:

$$\mathbb{E}_{p(\boldsymbol{x})}\left[\|\mathcal{J} - \mathcal{J}^{\mathrm{T}}\|_F^2/(4\|\mathcal{J}\|_F^2)\right], \text{ where } \mathcal{J} = \frac{\partial}{\partial x}s(\boldsymbol{x};\theta). \quad (7)$$

This example inspects the impact of the non-conservativeness of USBMs on the sampling process by comparing the sampling efficiency of a USBM s_U and a CSBM s_C under the same approximation error ϵ, i.e., $\mathcal{L}_{\mathrm{ESM}} = \epsilon$.



ϵ	*Asym*	*NAsym*	Steps
0.00	0.00	0.00	1
5.00e-4	0.00	0.00	2
7.50e-4	0.00	0.00	3
1.00e-3	0.00	0.00	3
1.50e-3	0.00	0.00	4
1.75e-3	0.00	0.00	5
2.25e-3	0.00	0.00	16

ϵ	*Asym*	*NAsym*	Steps
0.00	0.00	0.00	1
5.00e-4	4.41e-7	1.77e-2	4
7.50e-4	6.60e-7	2.65e-2	5
1.00e-3	8.78e-7	3.53e-2	6
1.50e-3	1.31e-6	5.28e-2	57
1.75e-3	1.53e-6	6.15e-2	121
2.25e-3	1.96e-6	7.89e-2	238

Figure 1. The visualized examples of (a) the conservative s_C and (b) the non-conservative s_U under different choices of ϵ. The table on the right-hand side reports the results measured using the *Asym* and *NAsym* metrics as well as the number of sampling steps.

Methodology

Quasi-Conservative Score-Based Models

QCSBMs resort to penalizing the non-conservativeness of USBMs through a regularization loss $\mathcal{L}_{\mathrm{QC}}$. The objective for QCSBMs is expressed as:

$$\mathcal{L}_{Total} = \mathcal{L}_{\mathrm{SM}} + \lambda\, \mathcal{L}_{\mathrm{QC}}, \quad (8)$$

where $\mathcal{L}_{\mathrm{SM}}$ represents the score-matching objective, and λ corresponds to the balancing factor.

Computationally Efficient Formulation of $\mathcal{L}_{\mathrm{QC}}$

In this work, $\mathcal{L}_{\mathrm{QC}}$ is implemented using Eq. (6), and reformulated as:

$$\begin{aligned}\mathcal{L}_{\mathrm{QC}} &= \mathbb{E}_{p(x)}\left[\|\mathcal{J} - \mathcal{J}^{\mathrm{T}}\|_F^2\right] \\ &= \mathbb{E}_{p(x)}[tr(\mathcal{J}\mathcal{J}^{\mathrm{T}}) - tr(\mathcal{J}\mathcal{J})] \\ &= \mathbb{E}_{p(x)}\left[\mathbb{E}_{p_{\mathbf{v}}(v)}[\boldsymbol{v}^{\mathrm{T}}\mathcal{J}\mathcal{J}^{\mathrm{T}}\boldsymbol{v}] - \mathbb{E}_{p_{\mathbf{v}}(v)}[\boldsymbol{v}^{\mathrm{T}}\mathcal{J}\mathcal{J}\boldsymbol{v}]\right] \\ &= \mathbb{E}_{p(x)p_{\mathbf{v}}(v)}[\boldsymbol{v}^{\mathrm{T}}\mathcal{J}\mathcal{J}^{\mathrm{T}}\boldsymbol{v} - \boldsymbol{v}^{\mathrm{T}}\mathcal{J}\mathcal{J}\boldsymbol{v}] \triangleq \tilde{\mathcal{L}}_{\mathrm{QC}}^{est},\end{aligned} \quad (9)$$

where $\mathbf{v}$ is a random projection vector satisfying $\mathbb{E}_{p_{\mathbf{v}}(v)}[\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}] = \mathbf{I}$.

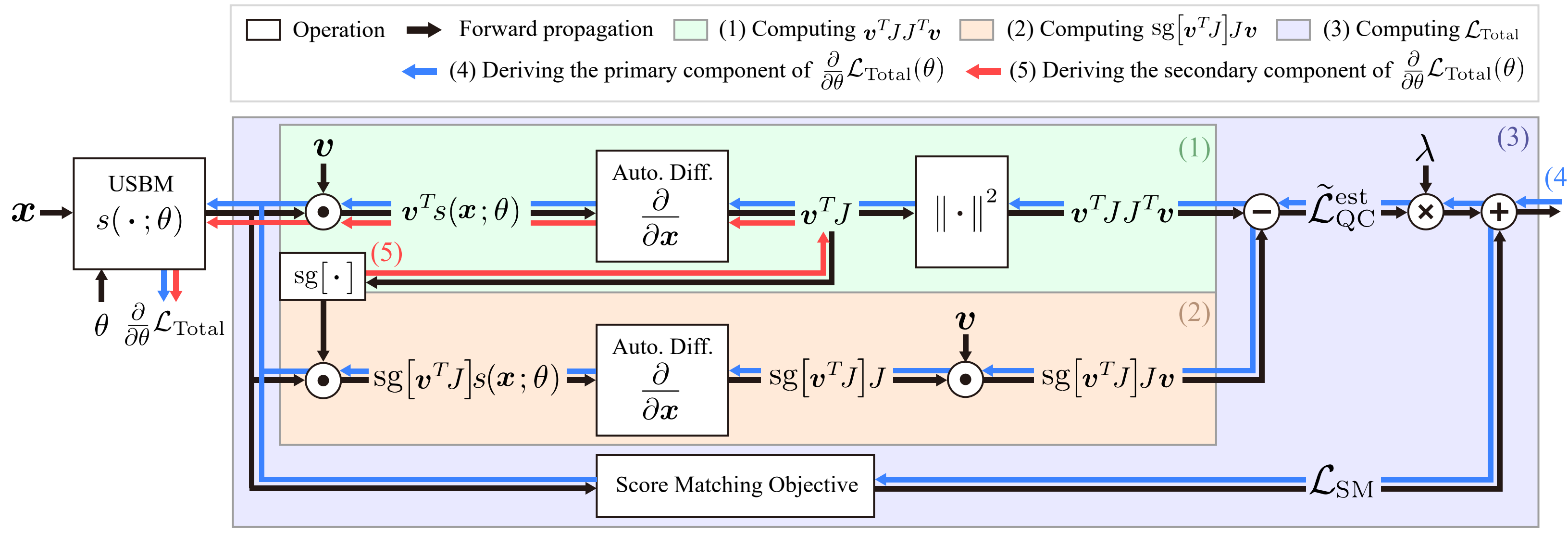

Operation
Forward propagation
(1) Computing v^TJJ^Tv
(2) Computing $\mathrm{sg}[v^TJ]Jv$
(3) Computing $\mathcal{L}_{\mathrm{Total}}$
(4) Deriving the primary component of $\frac{\partial}{\partial\theta}\mathcal{L}_{\mathrm{Total}}(\theta)$
(5) Deriving the secondary component of $\frac{\partial}{\partial\theta}\mathcal{L}_{\mathrm{Total}}(\theta)$
x
USBM
$s(\cdot\,;\theta)$
θ
$\frac{\partial}{\partial\theta}\mathcal{L}_{\mathrm{Total}}$
v
$v^Ts(x;\theta)$
Auto. Diff.
$\frac{\partial}{\partial x}$
v^TJ
$\|\cdot\|^2$
v^TJJ^Tv
$\tilde{\mathcal{L}}_{\mathrm{QC}}^{\mathrm{est}}$
λ
$\mathcal{L}_{\mathrm{Total}}$
sg[·]
(1)
(2)
(3)
(4)
(5)
$\mathrm{sg}[v^TJ]s(x;\theta)$
$\mathrm{sg}[v^TJ]J$
$\mathrm{sg}[v^TJ]Jv$
Score Matching Objective
$\mathcal{L}_{\mathrm{SM}}$

Figure 2. The computational graph of $\mathcal{L}_{Total}$ in QCSBMs. The 'Auto. Diff.' blocks represent the operation of differentiating $\boldsymbol{u}^T s(\boldsymbol{x};\theta)$ with respect to $\boldsymbol{x}$, where $\boldsymbol{u}$ is a constant vector with respect to $\boldsymbol{x}$. $sg[\cdot]$ represents the stop-gradient operator. The entire training procedure is divided into five steps, denoted as Steps (1)~(5), respectively. Steps (1)~(3) describe the forward propagation process of calculating $\mathcal{L}_{Total}$, which is depicted by the black arrows. Steps (4) and (5) correspond to the backward propagation processes of the two gradient components comprising $\frac{\partial}{\partial\theta}\mathcal{L}_{Total}(\theta)$, which are named the primary and secondary components, and are depicted as the blue and red arrows, respectively.

Experiments

Experiments on Real-World Datasets

For a fair evaluation, we follow the approach discussed in [7], and implement a USBM s_U and a CSBM s_C using neural networks f consisting of the same number of parameters as follows:

$$s_U(\boldsymbol{x},t;\theta_U) \triangleq \frac{1}{\sigma_t}(\boldsymbol{x} - f(\boldsymbol{x},t;\theta_U)), \;\; s_C(\boldsymbol{x},t;\theta_C) \triangleq \frac{\partial}{\partial \boldsymbol{x}}\frac{-\|\boldsymbol{x} - f(\boldsymbol{x},t;\theta_C)\|^2}{2\sigma_t}, \quad (10)$$

where σ_t represents the noise scale at timestep t, and θ_U and θ_C are the parameters of s_U and s_C, respectively.

- **Evaluation Metrics:** Negative Log Likelihood (NLL), Fréchet Inception Distance (FID), Inception Score (IS), Asym, NAsym, Precision, Recall, and NFE.
- **Methods: (a)** U-NCSN++ [4];
  (b) C-NCSN++ (i.e., U-NCSN++ constructed using Eq. (10));
  (c) QC-NCSN++ (i.e., U-NCSN++ regularized using Eq. (9)).

	CIFAR-10			ImageNet-32x32		
Method	NLL	*Asym*	*NAsym*	NLL	*Asym*	*NAsym*
C-NCSN++	5.91	**0.00**	**0.00**	5.10	**0.00**	**0.00**
U-NCSN++	3.46	1.88 e8	1.90 e-3	3.96	2.05 e7	7.17 e-4
QC-NCSN++	**3.38**	3.49 e7	8.41 e-4	**3.83**	1.13 e7	5.47 e-4
	CIFAR-100			SVHN		
Method	NLL	*Asym*	*NAsym*	NLL	*Asym*	*NAsym*
C-NCSN++	5.34	**0.00**	**0.00**	5.00	**0.00**	**0.00**
U-NCSN++	3.50	2.98 e8	2.25 e-3	2.15	3.06 e7	6.54 e-4
QC-NCSN++	**3.44**	9.31 e7	1.44 e-3	**2.01**	1.69 e7	4.80 e-4

Table 2. The NLL, *Asym*, and *NAsym* of C-NCSN++, U-NCSN++ [4], and QC-NCSN++ (Ours) evaluated on the CIFAR-10, CIFAR-100, ImageNet-32x32, and SVHN datasets.

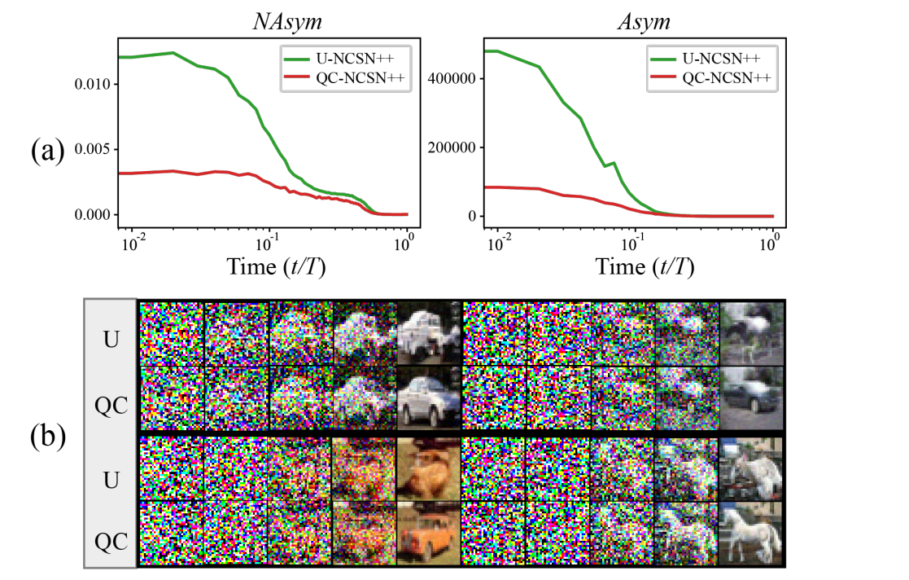


Figure 3. (a) The results of *Asym* and *NAsym* under different timestep t on CIFAR-10. (b) Examples generated by U-NCSN++ [4] and QC-NCSN++ (Ours) with the same random seeds.

Method	NFE	FID (↓)	IS (↑)	Prec. (↑)	Rec. (↑)
CIFAR-10					
C-NCSN++	1,000	10.97	8.58	0.61	0.58
U-NCSN++		2.50	9.58	**0.67**	**0.60**
QC-NCSN++		**2.48**	**9.70**	**0.67**	**0.60**
ImageNet-32x32					
C-NCSN++	1,000	28.97	8.58	**0.61**	0.45
U-NCSN++		19.82	9.89	0.60	**0.52**
QC-NCSN++		**19.62**	**9.94**	**0.61**	**0.52**

Method	NFE	FID (↓)	IS (↑)	Prec. (↑)	Rec. (↑)
CIFAR-100					
C-NCSN++	1,000	17.59	8.38	0.60	0.54
U-NCSN++		2.54	9.63	0.60	0.66
QC-NCSN++		**2.45**	**9.75**	**0.61**	**0.67**
SVHN					
C-NCSN++	1,000	24.71	2.66	**0.61**	0.46
U-NCSN++		14.34	3.10	0.60	**0.67**
QC-NCSN++		**13.88**	**3.12**	**0.61**	**0.67**

Table 3. The sampling performance and NFE of C-NCSN++, U-NCSN++ [4], and QC-NCSN++ (Ours) with the Predictor-Corrector (PC) sampler. The arrow symbols ↑ / ↓ indicate that higher / lower values correspond to better performance, respectively.

QCSBM Implemented as a One-Layered Autoencoder

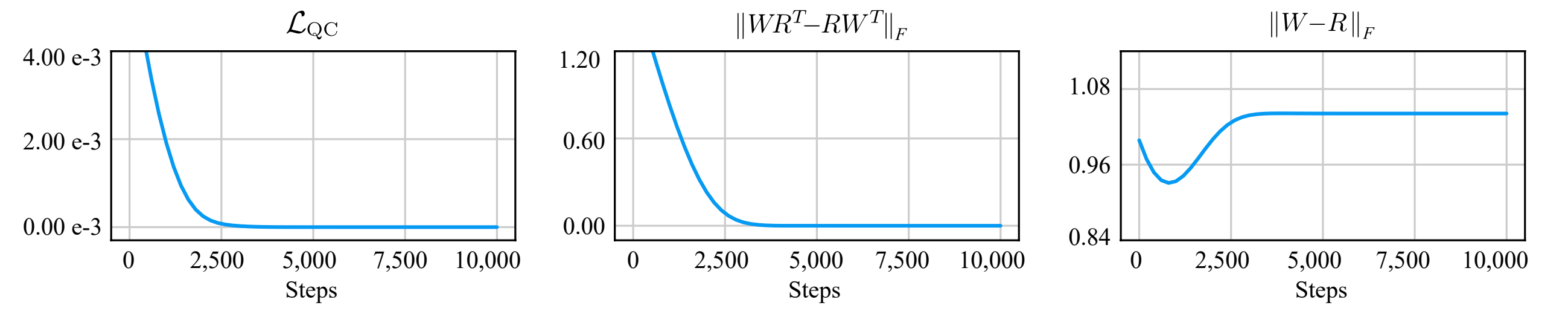


Figure 4. The trends of $\|WR^T - RW^T\|_F^2$ and $\|W - R\|_F^2$ during the optimization process of $\mathcal{L}_{\mathrm{QC}}$. The 'steps' on the x-axes refer to the training steps.

A line of research [2,5,8] focuses on SBMs represented as $\mathrm{s}(\boldsymbol{x};\theta) = \mathbf{R}h(\mathbf{W}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{b}) + \boldsymbol{c}$, where h is an activation function, $\mathbf{R}, \mathbf{W} \in \mathbb{R}^{D\times H}$ are the weights, H represents the hidden dimension, and $\boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}^D$ are the biases. As proved in [5], the output vector field of $\mathrm{s}(\boldsymbol{x};\theta)$ is conservative if and only if $\mathbf{R}\mathbf{W}^{\mathrm{T}} = \mathbf{W}\mathbf{R}^{\mathrm{T}}$. Instead of restricting the weights of $\mathrm{s}(\boldsymbol{x};\theta)$ to be 'tied,' i.e., $\mathbf{R} = \mathbf{W}$, QCSBMs indirectly learn to satisfy the conservativeness condition through minimizing $\mathcal{L}_{\mathrm{QC}}$.

References

[1] A. Hyvärinen. Estimation of Non-Normalized Statistical Models by Score Matching, *JMLR,* 2005.
[2] P. Vincent. A Connection between Score Matching and Denoising Autoencoders, *Neural computation,* 2011.
[3] Roberts *et al*. Optimal Scaling of Discrete Approximations to Langevin Diffusions, *Journal of the Royal Statistical Society,* 1998.
[4] Song *et al*. Score-Based Generative Modeling through Stochastic Differential Equations, *ICLR,* 2021.
[5] Im *et al*. Conservativeness of Untied Auto-Encoders, *AAAI,* 2016.
[6] Glötzl *et al*. Helmholtz Decomposition and Rotation Potentials in n-Dimensional Cartesian Coordinates, 2021.
[7] Salimans *et al*. Should EBMs Model the Energy or the Score?, *The Energy Based Models Workshop at ICLR*, 2021.
[8] Kamyshanska *et al*. On Autoencoder Scoring, *ICML,* 2013.

Acknowledgement

TWCC NVIDIA. NSTC

Questions?

lance_chao@gapp.nthu.edu.tw
cylee@cs.nthu.edu.tw