Importance Sampling via Score-based Generative Models

Heasung Kim, Taekyun Lee, Hyeji Kim, and Gustavo de Veciana
The University of Texas at Austin
Austin, TX, USA
heasung.kim@utexas.edu, taekyun.lee@utexas.edu, hyeji.kim@utexas.edu, deveciana@utexas.edu

Abstract

For a given positive loss function, the Optimal Importance Sampling (OIS)—sampling from a probability density function (PDF) proportional to the product of the loss function and the original PDF—serves as a powerful tool in various applications including variance reduction for mean value estimation, root cause analysis, and data augmentation. However, deriving the OIS PDF is generally extremely challenging particularly in high-dimensional spaces with existing methods relying on cross-entropy methods to learn parameterized PDF mimicing OIS PDF. In this paper, we propose a novel importance sampling method which is based solely on the score function of the original PDF and loss function without requiring the score function of OIS PDF. Conceptually, we approximate the data generation process characterized as a backward stochastic differential equation (SDE) associated with the score function of the OIS PDF by using only the score function of the origianl PDF and loss function, without requiring explicit computing of the OIS PDF. We provide a comprehensive analysis of our approach, demonstrating its effectiveness and scalabaility across a wide range of datasets including industrial and natural images, while introducing applications of OIS.

1. Introduction

Many real-world tasks can be modeled as state $\mathbf{x} \in \mathcal{X}$ following Probability Density Function (PDF) $p : \mathcal{X} \mapsto [0, 1]$, and positive loss function $l : \mathcal{X} \mapsto \mathbb{R}_+$ whose expectation $\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[l(\mathbf{x})]$ is what we want to minimize. For example, in autonomous driving, the traffic environment status can be interpreted as $\mathbf{x}$ following the system p and we want to minimize the loss due to traffic accident. In these setup, we often eager focus on the following crucial question.

"What kind of state $\mathbf{x}$ *is the root cause of high loss? / Can we sample state showing high loss function?"*

to figure out the root causes of the high loss of the task or generating data which potentially useful for further training modules with more robustness especially on the states causing high loss. This reminds us a new PDF $q \propto lp$, i.e., the PDF giving more density to sample showing high loss value under the original distribution. Formally, we define q as follows.

Definition 1 (Optimal Importance Sampling PDF)**.** *Given a positive loss function* $l(\cdot) : \mathcal{X} \times \Theta \mapsto \mathbb{R}^+$ and corresponding weighting function

$$q(\mathbf{x}) = \frac{l(\mathbf{x})p(\mathbf{x})}{\int l(\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}} \tag{1}$$

q is called the Optimal Importance Sampling (OIS) PDF for a given loss function and the original PDF.

As q provides various benefits, as mentioned, root causes analysis, data augmentation with high loss samples, and low variance expectation estimation, various existing methods relies on cross entropy methods.

Likelihood-based models either require strong restrictions on the model architecture to ensure a tractable normalizing constant for likelihood computation, or must rely on surrogate objectives to approximate maximum likelihood training (yang blog).

Recent approach to obtain a generative model for a target distribution is training neural networks that can approximate the score function of target distribution, in our case, $\nabla_{\mathbf{x}} q(\mathbf{x})$, and then use it to generate x folowing q through Stochastic Differential Equations (SDE) However, existing methods require the samples sampled from the target distribution, in our case we do not have dataset from q which is our target.

To address these challenges, we devise a novel way to gemerate samples $\mathbf{x}$ follows q without explicitly learn q or $\nabla_{\mathbf{x}} \log q$. For a given dataset following $p(\mathbf{x})$, we propose to learn $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ using existing methods. Then we propose to approximate the data generation process that following $q(\mathbf{x})$ in terms of $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ and $l(\mathbf{x})$. By doing this, we can do any kind of importance sampling without needs of newly trained density model. It also indicates, once we have a score function of the original distribution q, we can do any kind of importance sampling for a differentiable function l.

The main constributions of this paper can be summarized as follows.

- We present a powerful score-based importance sampling method by approximate the data generation process of OIS PDF by using the score function of original PDF and target loss function. To the best of our knowledge, this is the first score-based importance sampling framework not requiring any score function of OIS PDF.
- The proposed method enables us to perform importance sampling for any kind of differentiable loss function, make the algorithm is highly scalable adapted to any kind ot problems.
- Through extensive experiments, we provide various applications of the importance sampling for various dataset among synthetic dataset, industrial datset, and also natural image dataset which enables us to do root cause analysis, data augmentation.

Notation. The symbol d represents an infinitesimal increment. We use $\mathbf{x}$ to denote both a random vector and its realization, where the distinction is clear from context. $\|\cdot\|$ denotes the Euclidean norm for vectors and the operator norm for matrices, as determined by context.

2. Related Work

Importance Sampling For a given loss or target function, the OIS PDF [25] is widely applied in tasks such as variance reduction for expected value computation, data augmentation , root cause analysis, and reliability analysis [2, 9, 15]. However, in practical applications, accurately determining the OIS PDF often poses significant challenges. Conventional approaches typically rely on Cross-Entropy methods [23, 24], which seek to derive an importance sampling PDF from a pre-specified class of distributions by minimizing the Kullback-Leibler (KL) divergence between the candidate PDF and the OIS PDF.

For high-dimensional and complex datasets, particularly those involving natural images with distributions that are not available in closed form, selecting an appropriate class of PDFs is itself a complex problem. Recent advances in generative modeling aim to address these limitations through the use bijective mappings. By applying the change-of-variables formula, the tractable initial distribution is flowed through these mappings, resulting in a valid probability distribution at the end of the process, which is called normalizing flow [14, 19]. In [17], the normalizing flow-based OIS PDF training has been suggested with efficient invertible functions. The normalizing flow-based model is also utilized in [28] with interpretable shape-constrained networks for importance sampling.

In [9] proposed likelihood-free importance weighting method to correct for the bias in generative models by using a probabilistic classifier to distinguish samples from the original distribution and the genrative models.

The conventional and recent existing method largely relies on the shape-constrained density models, or function specific weight estimation. In

Score-based Generative Models In existing method relies on the normalizing flow-based methods. However, the flow based model typically requires shape-constrained density model so that the form of the density function is constrained due to make it tracktable. Recent advancement in generative model is centered around score-based models which tries to learn the score function of target distribution instead of learning pdf.

However, those existing score-based generative model methods cannot directly be applied to learn the score function of OIS pdf because the existing method requires the data samples sampled from the target distribution to learn.

3. Importance Sampling via Score-based Generative Models

3.1. Problem Formulation

In this paper, we aim to devise an algorithm that can sample $\mathbf{x}$ follows the OIS PDF $q(\mathbf{x}) = \frac{l(\mathbf{x})p(\mathbf{x})}{\int l(\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}}$ where l is a positive function as $l(\mathbf{x}) \geq m$ and $m > 0$. We consider p which is twice differentiable and q and l is differntiable with respect to $\mathbf{x}$. Reflecting practical scenario we assume that the data instances sampled from the origianl distribution p is available not from q.

3.2. Preliminaries

The recent advancements in generative models are predominantly rooted in score-based generative modeling, where the objective is to estimate the score function of the target data distribution, $\nabla_{\mathbf{x}} \log q(\mathbf{x})$. This is achieved through a parameterized model $s_\theta(\mathbf{x})$, typically implemented as a neural network, with the goal of approximating the true score function, i.e., $s_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log q(\mathbf{x})$. The score function can be learned via score matching techniques [11, 26, 27] if a dataset $\{\mathbf{x}_i\}_{i=1}^N$ sampled from q is available.

A notable innovation in score-based generative modeling is the introduction of score functions for noise-perturbed data, rather than for the original data distribution directly. Learning the score function of the noise-perturbed data not only facilitates image generation via Stochastic Differential Equations (SDEs) but also leads to more robust training of the score function. Specifically, a continuous-time perturbation process can be modeled through the following SDE

$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}\mathbf{w}, \tag{2}$$

where $\mathbf{x}(0) \sim q(\mathbf{x})$, implying that $\mathbf{x}$ at $t = 0$ follows the target distribution we aim to learn. In this context, f represents

the drift coefficient, $\sigma(t)$ denotes the diffusion coefficient, and $\mathrm{d}\mathbf{w}$ indicates Brownian motion. As time t progresses, the distribution of $\mathbf{x}$ evolves, which we denote as $q_t(\mathbf{x})$.

This framework allows for the estimation of the score function of the perturbed data $\mathbf{x}(t)$, which in turn enables the sampling of $\mathbf{x}(0)$ from $q_0 = q$ by using a corresponding backward SDE. The backward SDE is expressed as follows [1]:

$$\mathrm{d}\mathbf{x} = b_1(\mathbf{x}, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}\bar{\mathbf{w}}, \tag{3}$$

where

$$b_1(\mathbf{x}, t) = f(\mathbf{x}, t) - \sigma(t)\sigma(t)^\top \nabla_{\mathbf{x}} \log q_t(\mathbf{x}), \tag{4}$$

with $\mathbf{x}(t) \sim q_t$ and $\mathrm{d}\bar{\mathbf{w}}$ denoting the Brownian motion associated with the reverse-time process.

In this work, we adopt the following drift and diffusion coefficients:

$$f(\mathbf{x}, t) = -\frac{1}{2}\beta(t)\mathbf{x} \quad \text{and} \quad \sigma(t) = \sqrt{\beta(t)} \tag{5}$$

where $\beta(t)$ is a predefined scalar-valued function such that $0 \leq \beta(t) \leq 1$, which is a widely utilized setup for generative modeling [4, 10, 16, 18].

For instance, given an initial sample $\mathbf{x}(T) \sim q_T(\mathbf{x})$, solving SDE (3) yields a sample $\mathbf{x}(0)$ following q_0. Typically, for sufficiently large values of T, $\mathbf{x}(T)$ can be initialized as random Gaussian noise.

Challenges Despite the advancements in score-based generative models, significant challenges remain in effectively learning $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$, which limits their direct application in score-based generative modeling. The key challenges are outlined as follows.

- *Lack of target distribution samples.* Existing score-based generative models rely on a dataset $\{\mathbf{x}_i\}_{i=1}^N$ that follows the target PDF which the model aims to learn, where N denotes the number of data samples. However, in real-world scenarios, it is infeasible to obtain data instances that are directly sampled from the desired target PDF, q, presenting a fundamental obstacle for learning the score function.

  The problem is still presence when training score functions of noise-perturbed samples. Traditional methods condition the score function on time t by adding noise to the input instances. Yet, this approach becomes invalid when there is no dataset available that follows the target distribution, making it difficult to apply these techniques to learn a score function of OIS PDF.
- *Challenges in importance sampling with varied target losses.* Many generative models require importance sampling with respect to different target losses, l. For instance, a generative model tasked with generating human face images may need to perform importance sampling for specific features, such as black hair, and separately for blond hair. Existing approaches, which maximize log-likelihood through cross-entropy methods, learn a density model for each importance weight associated with l. Learning the exact density model q for multiple target losses is not only computationally challenging but also impractical, especially when dealing with a wide range of target losses.

We overcome these challenges by proposing a novel approach that represents the score function of perturbed importance samples in terms of the score function of p_t and the gradient of the loss function l. This enables the model to learn the score function of p_t—a task that existing methods can handle—and subsequently allows direct sampling of $\mathbf{x}$ from the target distribution q_0.

3.3. Proposed method

Our ultimate goal is to represent $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$ in terms of $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ and $\nabla_{\mathbf{x}} \log l(\mathbf{x})$. To achieve this, we begin by examining the form of the importance sampling distribution, $q_t(\mathbf{x})$, for the perturbed importance samples under the SDE defined in (2). Consider the transition probability density function $G : \mathcal{X} \times \mathcal{X} \times \mathcal{T} \mapsto \mathbb{R}_{\geq 0}$, also referred to as the Green's function or fundamental solution of the SDE [3, 6, 7]. This function represents the probability density of the process transitioning from the given initial state $\mathbf{x}(0)$ to the state $\mathbf{x}$ at time t. Specifically, $G(\mathbf{x}, \mathbf{x}(0), t)$ denotes the probability density of transitioning from $\mathbf{x}(0)$ to $\mathbf{x}$ over time t. Using this Green's function, we can express $q_t(\mathbf{x})$ as

$$q_t(\mathbf{x}) = \int G(\mathbf{x}, \mathbf{x}(0), t) q_0(\mathbf{x}(0))\,\mathrm{d}\mathbf{x}(0) \tag{6}$$

$$= \frac{1}{Z_0} \int G(\mathbf{x}, \mathbf{x}(0), t) l(\mathbf{x}(0)) p_0(\mathbf{x}(0))\,\mathrm{d}\mathbf{x}(0) \tag{7}$$

where $q_0 = q$, $p_0 = p$, and Z_0 is the normalization constant, $Z_0 = \int l(\mathbf{x}) p(\mathbf{x})\,\mathrm{d}\mathbf{x}$.

To further investigate the relationship between q_t and p_t, we consider the SDE of $\mathbf{y} \in \mathcal{X}$, which shares the same drift and diffusion coefficients as in (2), but where the initial distribution is $p_0 = p$ rather than q_0:

$$\mathrm{d}\mathbf{y} = -\frac{1}{2}\beta(t)\mathbf{y}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\mathbf{w}, \tag{8}$$

where $\mathbf{y}(0) \sim p_0$. It should be noted that the corresponding Green's function for this SDE is also G in (6), and the distribution of $\mathbf{y}$ at time t, denoted by p_t, is given as follows

$$p_t(\mathbf{y}) = \int G(\mathbf{y}, \mathbf{y}(0), t) p_0(\mathbf{y}(0))\,\mathrm{d}\mathbf{y}(0). \tag{9}$$

By using (6) and (9), we have

$$q_t(\mathbf{x}) = \frac{p_t(\mathbf{x})}{Z_0} \int l(\mathbf{x}(0)) G(\mathbf{x}, \mathbf{x}(0), t) \frac{p_0(\mathbf{x}(0))}{p_t(\mathbf{x})} \, \mathrm{d}\mathbf{x}(0) \tag{10}$$

$$= \frac{p_t(\mathbf{x})}{Z_0} \mathbb{E}_{\mathbf{x}(0) \sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))] \tag{11}$$

where $p_{0|t}$ is the conditional distribution of the initial state $\mathbf{x}(0)$ for a given state at t.

We can observe that q_t can be represented in terms of p_t and l as in (10) and the score function of OIS can be represented as

$$\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \nabla_{\mathbf{x}} \log \mathbb{E}_{\mathbf{x}(0) \sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))] \tag{12}$$

For practical computation of (12), we consider the first order Taylor approximation of $l(\mathbf{x}(0))$ at $\hat{\mathbf{x}}(0)|_{\mathbf{x},t} = \mathbb{E}_{\mathbf{x}(0) \sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[\mathbf{x}(0)]$ which gives us

$$l(\mathbf{x}(0)) \approx l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) + \nabla l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})^\top (\mathbf{x}(0) - \hat{\mathbf{x}}(0)|_{\mathbf{x},t}) \tag{13}$$

Substituting (13) to (12), we have

$$\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \nabla_{\mathbf{x}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}). \tag{14}$$

Note that $\hat{\mathbf{x}}(0)|_{\mathbf{x},t}$ is the conditional mean of the initial state for a given $\mathbf{x}$ at t and can be obtained through through the Tweedie's approach ([5, 8, 12]) as

$$\hat{\mathbf{x}}(0)|_{\mathbf{x},t} = \frac{1}{\sqrt{\bar{\alpha}(t)}} (\mathbf{x} + (1 - \bar{\alpha}(t)) \nabla_{\mathbf{x}} \log p_t(\mathbf{x})) \tag{15}$$

where

$$\bar{\alpha}(t) = \exp\left(- \int_0^t \beta(s) \, ds \right). \tag{16}$$

By using the chain rule, we can represent $\nabla_{\mathbf{x}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})$ as follows.

$$\nabla_{\mathbf{x}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) = \frac{\left(\mathbf{I} + (1 - \bar{\alpha}(t)) \boldsymbol{\nabla}_{\mathbf{x}}^2 \log p_t(\mathbf{x})\right)}{\sqrt{\bar{\alpha}(t)}} \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) \tag{17}$$

where $\boldsymbol{\nabla}_{\mathbf{x}}^2 \log p_t(\mathbf{x})$ is the Hessian matrix of the log likelihood.

For a small enough $\epsilon > 0$, the first-order Taylor's approximation of the Hessian matrix-vector multiplication can be applied as

$$\boldsymbol{\nabla}_{\mathbf{x}}^2 \log p_t(\mathbf{x}) \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) \tag{18}$$

$$\approx \frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) - \nabla_{\mathbf{x}} \log p_t(\mathbf{x})}{\epsilon} \tag{19}$$

Combining (3), (14), and (18), we finally suggest to use the following approximated score function.

$$\nabla_{\mathbf{x}} \log \tilde{q}_t(\mathbf{x}) \coloneqq \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \frac{\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})}{\sqrt{\bar{\alpha}(t)}} + \frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) - \nabla_{\mathbf{x}} \log p_t(\mathbf{x})}{\epsilon (1 - \bar{\alpha}(t))^{-1} \sqrt{\bar{\alpha}(t)}} \tag{20}$$

whose corresponding backward SDE to sample $\mathbf{x} \sim q_0$ is given as

$$\mathrm{d}\mathbf{x} = \left[-\frac{1}{2} \beta(t) \mathbf{x} - \beta(t) \nabla_{\mathbf{x}} \log q_t(\mathbf{x}) \right] \mathrm{d}t + \sqrt{\beta(t)} \, \mathrm{d}\bar{\mathbf{w}} \tag{21}$$

$$\approx \left[-\frac{1}{2} \beta(t) \mathbf{x} - \beta(t) \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \right] \mathrm{d}t + \sqrt{\beta(t)} \, \mathrm{d}\bar{\mathbf{w}} - \beta(t) \left[\frac{\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})}{\sqrt{\bar{\alpha}(t)}} - \frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}{\epsilon (1 - \bar{\alpha}(t))^{-1} \sqrt{\bar{\alpha}(t)}} + \frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})}{\epsilon (1 - \bar{\alpha}(t))^{-1} \sqrt{\bar{\alpha}(t)}} \right] \mathrm{d}t. \tag{22}$$

This approach provides significant benefits for generating importance samples.

Remark 1 (Training-Free Importance Sampling)**.** The proposed backward SDE in (21) demonstrates that, for a given score function of the original PDF $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ and a differentiable function l, ***importance sampling can be performed without any additional training***. This presents a substantial advantage over existing methods, which require learning the density of q through individual training for each specific l. Furthermore, in modern Computer Vision research, pre-trained score functions are widely available in the public domain and are frequently utilized, especially for natural image generative models [20–22]. This means that, with an appropriately defined differentiable importance measure, $l(\mathbf{x})$, it is possible to directly leverage these pre-trained models without the need for further training of the OIS PDF.

Remark 2 (High Scalability and Applicability)**.** The proposed algorithm requires only the score function of the original PDF and the differentiable function l, allowing the use of a wide range of importance measures, particularly those implemented using differentiable neural networks. For instance, consider a neural autoencoder consisting of an encoder and decoder, which can be employed to assess the compression effectiveness for a given sample. If the objective is to prioritize samples likely to experience higher distortion in a compression task, the autoencoder can serve as the function l, enabling the sampling of instances that are

more prone to high distortion. This approach is readily applicable to a variety of tasks, such as importance sampling for high-quality images by giving greater weight to better quality samples, or sampling images based on stylistic similarity to a reference image by placing more emphasis on samples that align closely with the desired style.

Computational complexity It is important to highlight that the additional computational complexity introduced by this approach, compared to the original score-based sampling that computes $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, is minimal. The computation of $\hat{\mathbf{x}}(0)|_{\mathbf{x},t}$ involves only a simple linear transformation of the input $\mathbf{x}$ and the precomputed score function, as described in (15), and is typically handled internally in most existing methods based on the Euler–Maruyama scheme [13] for solving the backward SDE. Aside from this negligible cost, the only other potential overhead is calculating the derivative of the loss function, $\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})$, which can be efficiently computed through a single gradient backpropagation step with respect to the precomputed realization of $\hat{\mathbf{x}}(0)|_{\mathbf{x},t}$. This additional computation might be far less burdensome than methods that require learning an exact OIS PDF for each individual l.

3.4. Technical Analysis

The following Theorem 1 shows that the Euclidean norm gap between the approximated score function and the ground truth score function is upper bounded, with its proof provided in Appendix 5.1.

Assumption 1 (Lipschitz continuity)**.** $l(\mathbf{x})$ is L_1-Lipschitz continuous and $\nabla_{\mathbf{x}} \log p(\mathbf{x}_t)$ is L_2-Lipschitz continuous. $\boldsymbol{\nabla}^2_{\mathbf{x}} \log p(\mathbf{x}_t)$ is L_3. Recall that a function l is L-Lipschitz continuous if $\|l(x) - l(y)\| \leq L\|x - y\|$ with x and y in the domain of l.

Assumption 2 (Bounded norm)**.** For all $t \in [0, T]$, we have $\|\nabla_{\mathbf{x}} \mathbb{E}_{\mathbf{x}(0)\sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))]\| \leq \delta_1$, $\|\nabla_{\mathbf{x}} l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})\| \leq \delta_2$, and $\|\nabla_{\mathbf{x}} \log l(\mathbf{x})\| \leq \delta_l$.

Theorem 1 (Score function gap)**.** *Suppose that Assumptions 1-2 hold. For a given $\epsilon > 0$, the gap between the true score function of OIS PDF of the perturbed sample, $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$ and the approximated score function is upper-bounded as* $\|\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) - \nabla_{\mathbf{x}} \log \tilde{q}_t(\mathbf{x})\| \leq \frac{\delta_1+\delta_2}{m} + \frac{L_3 \delta_l^2 \epsilon}{2}$.

Remark 3 (Bounded score function gap)**.** The gap between the true score function and the proposed approximation shows bounded norm gap, especially in terms of the minimum value of the positive function l as $m = \min_{\mathbf{x}} l(\mathbf{x})$ and ϵ for the Hessian-vector matrix multiplication.

Remark 4 (Unbiased estimation for $t \to 0$)**.** As $t \to 0$, we can see that

$$\nabla_{\mathbf{x}} \log q_0(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_0(\mathbf{x}) + \nabla_{\mathbf{x}} \log l(\mathbf{x}) \tag{23}$$

as $\bar{\alpha}(t) \to 0$.

4. Experiments

In this section, we provide various applications of our methods. We first show that the proposed method effectively perform importance sampling by using synthetic spiral-shaped density. Through this, we shows that the proposed method can sample the instances on the valid manifold effectively.

4.1. Spiral Dataset

4.2. CSI: Root Cause Analysis via Importance Sampling

4.3. CelebA: Importance Sampling over Neural Classifier / Descriminator

4.4. Weibull distribution

4.5. Rare probability measure

4.6. Style Transfer: Importance Sampling over Style Vector Similarity

5. Conclusion

This simple idea shows good results.

References

[1] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. 3

[2] A Antoniou. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017. 2

[3] James V Beck, Kevin D Cole, A Haji-Sheikh, and Bahman Litkouhl. *Heat conduction using Green's function*. Taylor & Francis, 1992. 3

[4] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. 3

[5] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations*, 2022. 4

[6] Dean G Duffy. *Green's functions with applications*. Chapman and Hall/CRC, 2015. 3

[7] Eleftherios N Economou. *Green's functions in quantum physics*. Springer Science & Business Media, 2006. 3

[8] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. 4

[9] Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. Bias correction of learned generative models using likelihood-free importance weighting. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3

[11] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005. 2

[12] Kwanyoung Kim and Jong Chul Ye. Noise2score: tweedie's approach to self-supervised image denoising without clean images. *Advances in Neural Information Processing Systems*, 34:864–874, 2021. 4

[13] Peter E Kloeden, Eckhard Platen, Peter E Kloeden, and Eckhard Platen. *Stochastic differential equations*. Springer, 1992. 5

[14] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020. 2

[15] Guosheng Li, Weiwei Cai, Lei Tian, Leping Yang, Weihua Zhang, and Zeping Wu. Efficient augmented radial basis function for reliability analysis with dynamic importance sampling. *Engineering Optimization*, pages 1–20, 2024. 2

[16] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022. 3

[17] Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. Neural importance sampling. *ACM Transactions on Graphics (ToG)*, 38(5):1–19, 2019. 2

[18] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 3

[19] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 2

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 4

[21] Litu Rout, Yujia Chen, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Beyond first-order tweedie: Solving inverse problems using latent diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9472–9481, 2024.

[22] Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alex Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 4

[23] Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1:127–190, 1999. 2

[24] Reuven Y Rubinstein. Combinatorial optimization, cross-entropy, ants and rare events. *Stochastic optimization: algorithms and applications*, pages 303–363, 2001. 2

[25] Reuven Y Rubinstein and Dirk P Kroese. *Simulation and the Monte Carlo method*. John Wiley & Sons, 2016. 2

[26] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 2

[27] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011. 2

[28] Zhengliang Xiang, Xuhui He, Yunfeng Zou, and Haiquan Jing. An importance sampling method for structural reliability analysis based on interpretable deep generative network. *Engineering with Computers*, 40(1):367–380, 2024. 2

Importance Sampling via Score-based Generative Models

Supplementary Material

5.1. Proof of Theorem 1

Proof. We first explore the Jensen gap The gap between (12) and (14).

$$\|\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) - \nabla_{\mathbf{x}} \log \tilde{q}_t(\mathbf{x})\| = \|\nabla_{\mathbf{x}} \log \mathbb{E}_{\mathbf{x}(0)\sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))] - \nabla_{\mathbf{x}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})\| \tag{24}$$

$$= \left\| \frac{\nabla_{\mathbf{x}} \mathbb{E}_{\mathbf{x}(0)\sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))]}{\mathbb{E}_{\mathbf{x}(0)\sim p_{0|t}(\mathbf{x}(0)|\mathbf{x})}[l(\mathbf{x}(0))]} - \frac{\nabla_{\mathbf{x}} l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})}{l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})} \right\| \tag{25}$$

Deriving a General Error Bound

We aim to derive a general error bound for approximating the Hessian-vector product $H\mathbf{v}$ using finite differences of the gradient. Given a twice differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, where $\nabla f(\mathbf{x})$ is the gradient and $H = \nabla^2 f(\mathbf{x})$ is the Hessian at point $\mathbf{x}$, we approximate $H\mathbf{v}$ using:

$$H\mathbf{v} \approx \frac{\nabla f(\mathbf{x} + h\mathbf{v}) - \nabla f(\mathbf{x})}{h}, \tag{26}$$

where h is a small scalar step size.

Using Taylor's Theorem with Remainder

We express the score function at $\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})$ using Taylor's theorem with the integral form of the remainder

$$\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}))$$
$$= \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \int_0^{\epsilon} \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x} + s\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})) \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})\, \mathrm{d}s. \tag{27}$$

We then have

$$\frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})) - \nabla_{\mathbf{x}} \log p_t(\mathbf{x})}{\epsilon} - \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x}) \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})$$
$$= \frac{1}{\epsilon} \int_0^{\epsilon} \left(\boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x} + s\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})) - \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x}) \right) \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})\, \mathrm{d}s. \tag{28}$$

Considering the norm of the first term of (28), we have

$$\left\| \frac{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} + \epsilon \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})) - \nabla_{\mathbf{x}} \log p_t(\mathbf{x})}{\epsilon} - \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x}) \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) \right\| \tag{29}$$

$$\leq \frac{1}{\epsilon} \int_0^{\epsilon} \left\| \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x} + s\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})) - \boldsymbol{\nabla}^2_{\mathbf{x}} \log p_t(\mathbf{x}) \right\| \cdot \left\| \nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t}) \right\| \mathrm{d}s. \tag{30}$$

$$\leq \frac{1}{\epsilon} \int_0^{\epsilon} L_p s \|\nabla_{\hat{\mathbf{x}}(0)|_{\mathbf{x},t}} \log l(\hat{\mathbf{x}}(0)|_{\mathbf{x},t})\|^2 \mathrm{d}s \leq \frac{L_3 \delta_l^2 \epsilon}{2}. \tag{31}$$

where (31), by Lipschitz property of the Hessian matrix, we have

Combining (31), we have

$$\|\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) - \nabla_{\mathbf{x}} \log \tilde{q}_t(\mathbf{x})\| \leq \frac{\delta_1 + \delta_2}{m} + \frac{L_3 \delta_l^2 \epsilon}{2}. \tag{32}$$

We express the gradient at $\mathbf{x} + h\mathbf{v}$ using Taylor's theorem with the integral form of the remainder:

$$\nabla f(\mathbf{x} + h\mathbf{v}) = \nabla f(\mathbf{x}) + \int_0^h \nabla^2 f(\mathbf{x} + t\mathbf{v})\mathbf{v}\, dt. \tag{33}$$

Subtracting $\nabla f(\mathbf{x})$ from both sides and dividing by h:

$$\frac{\nabla f(\mathbf{x}+h\mathbf{v})-\nabla f(\mathbf{x})}{h}=\frac{1}{h}\int_0^h \nabla^2 f(\mathbf{x}+t\mathbf{v})\mathbf{v}\,dt. \tag{34}$$

We can rewrite the right-hand side as:

$$\frac{1}{h}\int_0^h \left[\nabla^2 f(\mathbf{x})+\left(\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\right)\right]\mathbf{v}\,dt. \tag{35}$$

Simplifying:

$$\frac{\nabla f(\mathbf{x}+h\mathbf{v})-\nabla f(\mathbf{x})}{h}=\nabla^2 f(\mathbf{x})\mathbf{v}+\frac{1}{h}\int_0^h \left(\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\right)\mathbf{v}\,dt. \tag{36}$$

Defining the Error Term

Define the error $\mathbf{E}$ as:

$$\mathbf{E}=\frac{\nabla f(\mathbf{x}+h\mathbf{v})-\nabla f(\mathbf{x})}{h}-\nabla^2 f(\mathbf{x})\mathbf{v}. \tag{37}$$

Substituting from the previous expression:

$$\mathbf{E}=\frac{1}{h}\int_0^h \left(\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\right)\mathbf{v}\,dt. \tag{38}$$

Bounding the Error Using Lipschitz Continuity of the Hessian

Assume that the Hessian $\nabla^2 f$ is Lipschitz continuous with Lipschitz constant L, that is:

$$\|\nabla^2 f(\mathbf{y})-\nabla^2 f(\mathbf{x})\|\le L\|\mathbf{y}-\mathbf{x}\|,\quad \forall \mathbf{y},\mathbf{x}\in\mathbb{R}^n. \tag{39}$$

For $\mathbf{y}=\mathbf{x}+t\mathbf{v}$:

$$\|\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\|\le Lt\|\mathbf{v}\|. \tag{40}$$

Now, we can bound the norm of the error $\mathbf{E}$:

$$\|\mathbf{E}\|=\left\|\frac{1}{h}\int_0^h \left(\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\right)\mathbf{v}\,dt\right\| \tag{41}$$

$$\le\frac{1}{h}\int_0^h \left\|\nabla^2 f(\mathbf{x}+t\mathbf{v})-\nabla^2 f(\mathbf{x})\right\|\cdot\|\mathbf{v}\|\,dt \tag{42}$$

$$\le\frac{1}{h}\int_0^h Lt\|\mathbf{v}\|^2\,dt \tag{43}$$

$$=\frac{L\|\mathbf{v}\|^2}{h}\int_0^h t\,dt \tag{44}$$

$$=\frac{L\|\mathbf{v}\|^2}{h}\left[\frac{t^2}{2}\Big|_0^h\right] \tag{45}$$

$$=\frac{Lh\|\mathbf{v}\|^2}{2}. \tag{46}$$

General Error Bound

Therefore, the general error bound is:

$$\|\mathbf{E}\| \leq \frac{Lh\|\mathbf{v}\|^2}{2}. \tag{47}$$

This bound shows that the error in approximating $H\mathbf{v}$ using finite differences of the gradient is proportional to the step size h, the Lipschitz constant L, and the square of the norm of $\mathbf{v}$.

$$\mathbf{x}_t = x + \int_0^t b_1(\mathbf{x}_s, s)\,\mathrm{d}s + \int_0^t \sigma(s)\,\mathrm{d}\mathbf{w}_s, \tag{48}$$

$$\mathbf{y}_t = x + \int_0^t b_2(\mathbf{y}_s, s)\,\mathrm{d}s + \int_0^t \sigma(s)\,\mathrm{d}\mathbf{w}_s. \tag{49}$$

By Minkowski inequality, we have

$$\|\mathbf{x}_t - \mathbf{y}_t\|_p \le \left\| \int_0^t b_1(\mathbf{x}_s, s) - b_2(\mathbf{y}_s, s)\,\mathrm{d}s \right\|_p \tag{50}$$

$$\le \left\| \int_0^t b_1(\mathbf{x}_s, s) - b_1(x, 0)\,\mathrm{d}s \right\|_p + t\Delta b(x) \tag{51}$$

$$+ \left\| \int_0^t b_2(\mathbf{x}_s, s) - b_2(x, 0)\,\mathrm{d}s \right\|_p \tag{52}$$

□

Suppose Assumption 1 holds and $\mathbf{x}$ and $\mathbf{y}$ be the solution to the two SDEs (**??**) and (**??**), respectively. Consider a initial state x for both random process. Then we have the following almost surely.

$$\|\mathbf{x}_t - \mathbf{y}_t\|_p \le C. \tag{53}$$

Proof. Based on the integral representation of the solutions, we have

$$\mathbf{x}_t = x + \int_0^t b_1(\mathbf{x}_s, s)\,\mathrm{d}s + \int_0^t \sigma(s)\,\mathrm{d}\mathbf{w}_s, \tag{54}$$

$$\mathbf{y}_t = x + \int_0^t b_2(\mathbf{y}_s, s)\,\mathrm{d}s + \int_0^t \sigma(s)\,\mathrm{d}\mathbf{w}_s. \tag{55}$$

By Minkowski inequality, we have

$$\|\mathbf{x}_t - \mathbf{y}_t\|_p \le \left\| \int_0^t b_1(\mathbf{x}_s, s) - b_2(\mathbf{y}_s, s)\,\mathrm{d}s \right\|_p \tag{56}$$

$$\le \left\| \int_0^t b_1(\mathbf{x}_s, s) - b_1(x, 0)\,\mathrm{d}s \right\|_p + t\Delta b(x) \tag{57}$$

$$+ \left\| \int_0^t b_2(\mathbf{x}_s, s) - b_2(x, 0)\,\mathrm{d}s \right\|_p \tag{58}$$

□

- $X_t \in \mathbb{R}^n$ is the state of the process at time t.
- $b(X_t, t)$ is the drift term, which determines the deterministic part of the evolution.
- $\sigma(X_t, t)$ is the diffusion coefficient, a matrix-valued function that governs the stochastic dynamics.
- W_t is an m-dimensional Wiener process.

6. Infinitesimal Generator of the SDE

The infinitesimal generator A of a diffusion process described by the SDE above is defined as:

$$Af(x) = \lim_{t \to 0} \frac{\mathbb{E}_x[f(X_t)] - f(x)}{t}, \tag{59}$$

for a sufficiently smooth function $f : \mathbb{R}^n \to \mathbb{R}$. This can be further expressed as:

$$Af(x) = \sum_{i=1}^{n} b_i(x, t) \frac{\partial f}{\partial x_i}(x) + \frac{1}{2} \sum_{i,j=1}^{n} (D(x, t))_{ij} \frac{\partial^2 f}{\partial x_i \partial x_j}(x), \tag{60}$$

where $D(x, t) = \sigma(x, t)\sigma^T(x, t)$ is the covariance matrix of the process.

7. Conditional Diffusion Process and Measure Transformation

Let $(X_t, \mathcal{F}_t, P^T_{x,y})$ be a conditional diffusion process conditioned on reaching a terminal value y at time T. The infinitesimal generator A of this process is expressed as:

$$A = L_b + \nabla_g \log p(z, T-t, y) \nabla_g, \tag{61}$$

where:
- L_b is the infinitesimal generator of the original SDE.
- ∇_g denotes the gradient in the g-norm.
- $\log p(z, T-t, y)$ is the log of the transition density function conditioned on reaching y at time T.

8. Measure Transformation and New Probability Measure

Define a measurable vector field $c(x)$ on $\mathbb{R}^n$ as:

$$c(x) = \sum_i c_i(x) \frac{\partial}{\partial x_i}. \tag{62}$$

This vector field c is used to introduce a new probability measure Q_x, defined as:

$$\left.\frac{dQ_x}{dP_x}\right|_{\mathcal{F}_t} = \exp\left(\int_0^t \langle c(X_s), dM_s\rangle_g - \frac{1}{2}\int_0^t |c|^2_g(X_s)\, ds\right), \tag{63}$$

where $\langle c(X_s), dM_s\rangle_g$ is the inner product of $c(X_s)$ with the martingale dM_s under the g-metric, and $|c|^2_g(X_s)$ is the squared norm of c.

9. Cameron-Martin Formula and New SDE

By the Cameron-Martin formula, the new process $(X_t, \mathcal{F}_t, Q_x)$ becomes an $L_b + c$-diffusion process, which has the following SDE:

$$dX_t = (b(X_t, t) + c(X_t))\, dt + \sigma(X_t, t)\, dW^Q_t, \tag{64}$$

where W^Q_t is a Wiener process under the new measure Q_x. The transition density of the modified process is denoted as $p_{b+c}(x, t, y)$.

Derivation of x_t for the SDE

Consider the following stochastic differential equation (SDE):

$$dx_t = -\frac{1}{2}\beta(t)x_t\,dt + \sqrt{\beta(t)}\,dw_t, \tag{65}$$

where w_t is a standard Brownian motion and $\beta(t)$ is a time-dependent function.

Step 1: Rearrange the SDE

The SDE can be rewritten as:

$$dx_t = \left(-\frac{1}{2}\beta(t)x_t\right)dt + \sqrt{\beta(t)}\,dw_t. \tag{66}$$

This SDE has a time-dependent drift term $-\frac{1}{2}\beta(t)x_t$ and a time-dependent diffusion coefficient $\sqrt{\beta(t)}$.

Step 2: Solve the SDE Using the Integrating Factor Method

We can solve this SDE using the integrating factor method. Consider the change of variables:

$$y_t = f(t)x_t, \tag{67}$$

where $f(t)$ is a deterministic function of t chosen to eliminate the time-dependent drift term. We choose $f(t)$ such that:

$$df(t) = \frac{1}{2}\beta(t)f(t)\,dt \quad \Rightarrow \quad f(t) = \exp\left(\frac{1}{2}\int_0^t \beta(s)\,ds\right). \tag{68}$$

Transforming the SDE using $y_t = f(t)x_t$, we get:

$$dy_t = f(t)\sqrt{\beta(t)}\,dw_t. \tag{69}$$

Step 3: Solve for y_t

Integrating both sides from 0 to t:

$$y_t = y_0 + \int_0^t \exp\left(\frac{1}{2}\int_0^s \beta(u)\,du\right)\sqrt{\beta(s)}\,dw_s. \tag{70}$$

Since $y_0 = f(0)x_0 = x_0$, we have:

$$y_t = x_0 + \int_0^t \exp\left(\frac{1}{2}\int_0^s \beta(u)\,du\right)\sqrt{\beta(s)}\,dw_s. \tag{71}$$

Step 4: Solve for x_t

Recall that $y_t = f(t)x_t$, so:

$$x_t = \frac{y_t}{f(t)}. \tag{72}$$

Substitute $f(t) = \exp\left(\frac{1}{2}\int_0^t \beta(s)\,ds\right)$:

$$x_t = \exp\left(-\frac{1}{2}\int_0^t \beta(s)\,ds\right)\left(x_0 + \int_0^t \exp\left(\frac{1}{2}\int_0^s \beta(u)\,du\right)\sqrt{\beta(s)}\,dw_s\right). \tag{73}$$

Define $\bar{\alpha}(t)$ as:

$$\bar{\alpha}(t) = \exp\left(-\int_0^t \beta(s)\,ds\right). \tag{74}$$

Thus, the solution can be rewritten as:

$$x_t = \sqrt{\bar{\alpha}(t)}x_0 + \sqrt{\bar{\alpha}(t)} \int_0^t \frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\, dw_s. \tag{75}$$

Step 2.3: Simplifying the Integral Using the Definition of $\bar{\alpha}(t)$

Let's simplify the integral term in the solution using the definition of $\bar{\alpha}(t)$. Recall that:

$$\bar{\alpha}(t) = \exp\left(-\int_0^t \beta(s)\, ds\right). \tag{76}$$

For the integral term in our solution:

$$\sqrt{\bar{\alpha}(t)} \int_0^t \frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\, dw_s, \tag{77}$$

the variance is given by:

$$\text{Var}\left(\sqrt{\bar{\alpha}(t)} \int_0^t \frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\, dw_s\right) = \bar{\alpha}(t) \int_0^t \left(\frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\right)^2 ds. \tag{78}$$

Substitute $\bar{\alpha}(s)$:

$$\bar{\alpha}(s) = \exp\left(-\int_0^s \beta(u)\, du\right). \tag{79}$$

Thus, the integral becomes:

$$\bar{\alpha}(t) \int_0^t \frac{\beta(s)}{\bar{\alpha}(s)}\, ds = \bar{\alpha}(t) \int_0^t \beta(s) \exp\left(\int_0^s \beta(u)\, du\right) ds. \tag{80}$$

Recognizing that $\exp\left(\int_0^s \beta(u)\, du\right) = \frac{1}{\bar{\alpha}(s)}$, we obtain:

$$\bar{\alpha}(t) \int_0^t \beta(s) \frac{1}{\bar{\alpha}(s)}\, ds = \bar{\alpha}(t) \left(\frac{1}{\bar{\alpha}(t)} - 1\right). \tag{81}$$

Since $\frac{1}{\bar{\alpha}(t)} = \exp\left(\int_0^t \beta(s)\, ds\right)$, the integral simplifies to:

$$\bar{\alpha}(t) \left(\frac{1}{\bar{\alpha}(t)} - 1\right) = 1 - \bar{\alpha}(t). \tag{82}$$

This shows that the variance of the integral term is $1 - \bar{\alpha}(t)$.

Step 5: Derivation of $\sqrt{1 - \bar{\alpha}(t)}$

The variance calculation shows that the noise term's contribution has variance $1 - \bar{\alpha}(t)$. Therefore, the integral term can be represented as:

$$\sqrt{\bar{\alpha}(t)} \int_0^t \frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\, dw_s = \sqrt{1 - \bar{\alpha}(t)}\, z, \tag{83}$$

where $z \sim \mathcal{N}(0, 1)$ is a standard normal random variable.

Step 6: Introducing $z \sim \mathcal{N}(0, 1)$

The term $z \sim \mathcal{N}(0, 1)$ arises because:

1. The stochastic integral $\int_0^t \frac{\sqrt{\beta(s)}}{\sqrt{\bar{\alpha}(s)}}\, dw_s$ is itself normally distributed, with mean 0 and variance $1 - \bar{\alpha}(t)$. 2. Any linear combination of independent Gaussian variables is itself Gaussian. Therefore, the stochastic integral can be written as $\sqrt{1 - \bar{\alpha}(t)}\, z$ for some standard normal random variable $z \sim \mathcal{N}(0, 1)$. 3. This form allows us to capture the effect of the noise term in a simplified manner, with z representing the randomness of the Brownian motion over time.

Step 7: Final Solution of x_t

The final solution of the SDE is:

$$x_t = \sqrt{\bar{\alpha}(t)}x_0 + \sqrt{1-\bar{\alpha}(t)}z, \tag{84}$$

where $z \sim \mathcal{N}(0,1)$.

This form clearly separates the deterministic part, weighted by $\sqrt{\bar{\alpha}(t)}$, from the stochastic part, weighted by $\sqrt{1-\bar{\alpha}(t)}$.

10. Example: Wirelss Network

In this section, we present a mathematical formulation of a wireless network simulation where the goal is to compute the Signal-to-Noise Ratio (SNR) in a differentiable manner using PyTorch. The location information of the users is denoted as $\boldsymbol{x}$ and is normalized within $[-1, 1]$. We describe the process of mapping this normalized data to actual coordinates, computing the SNR considering interference from other base stations, and applying a differentiable threshold function.

11. User and Base Station Locations

11.1. User Locations

We consider N users in the network. Each user's location is represented by a two-dimensional coordinate. The location information is given by the vector:

$$\boldsymbol{x} = [x_1, x_2, \ldots, x_{2N}]^\top \in \mathbb{R}^{2N}, \tag{85}$$

where each element $x_i \in [-1, 1]$. We reshape $\boldsymbol{x}$ into a matrix to represent individual user coordinates:

$$\boldsymbol{X} = \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \\ \vdots & \vdots \\ x_{2N-1} & x_{2N} \end{bmatrix} \in \mathbb{R}^{N\times 2}. \tag{86}$$

To map the normalized locations to actual coordinates within the area $[0, 100] \times [0, 100]$, we use the following transformation:

$$\boldsymbol{U} = (\boldsymbol{X} + 1) \times 50, \tag{87}$$

where $\boldsymbol{U} \in \mathbb{R}^{N\times 2}$ represents the actual user coordinates.

11.2. Base Station Locations

Assume there are M base stations located at known positions:

$$\boldsymbol{B} = \begin{bmatrix} b_{1x} & b_{1y} \\ b_{2x} & b_{2y} \\ \vdots & \vdots \\ b_{Mx} & b_{My} \end{bmatrix} \in \mathbb{R}^{M\times 2}. \tag{88}$$

For example, with $M = 3$, the base stations are positioned at:

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 100 & 0 \\ 50 & 100 \end{bmatrix}. \tag{89}$$

12. Distance Computation

The Euclidean distance between each user and each base station is calculated as:

$$d_{ij} = \|\boldsymbol{u}_i - \boldsymbol{b}_j\|_2 = \sqrt{(u_{ix} - b_{jx})^2 + (u_{iy} - b_{jy})^2}, \tag{90}$$

where $\boldsymbol{u}_i$ is the coordinate of user i and $\boldsymbol{b}_j$ is the coordinate of base station j.

13. Signal and Interference Power

13.1. Path Loss Model

We employ a simplified path loss model where the received power decreases with distance:

$$P_r = \frac{P_t}{(d^\alpha + \epsilon)}, \tag{91}$$

where:

- P_t is the transmitted power.
- d is the distance between the transmitter and receiver.
- α is the path loss exponent.
- ϵ is a small constant to prevent division by zero.

13.2. Serving Base Station

Each user is connected to the nearest base station. The index of the serving base station for user i is determined by:

$$k_i = \arg\min_j \{d_{ij} \mid j = 1, 2, \ldots, M\}. \tag{92}$$

13.3. Received Signal Power

The received signal power for user i from its serving base station is:

$$P_{\text{signal},i} = \frac{P_t}{(d_{ik_i}^\alpha + \epsilon)}. \tag{93}$$

13.4. Interference Power

The interference power for user i from other base stations is:

$$P_{\text{interference},i} = \sum_{\substack{j=1 \\ j \neq k_i}}^{M} \frac{P_t}{(d_{ij}^\alpha + \epsilon)}. \tag{94}$$

13.5. Noise Power

We assume a constant noise power N_0.

14. Signal-to-Noise Ratio (SNR)

The SNR for user i is computed as:

$$\text{SNR}_i = \frac{P_{\text{signal},i}}{N_0 + P_{\text{interference},i}}. \tag{95}$$

15. Differentiable Threshold Function

To determine whether the SNR exceeds a certain threshold γ, we use a differentiable approximation of the step function via the sigmoid function:

$$\sigma_i = \sigma(k(\text{SNR}_i - \gamma)) = \frac{1}{1 + e^{-k(\text{SNR}_i - \gamma)}}, \tag{96}$$

where:

- $\sigma(\cdot)$ is the sigmoid function.
- k is a positive constant that controls the steepness of the sigmoid curve.
  As $k \to \infty$, the sigmoid function approaches a step function.

16. Conclusion

We have formulated the SNR computation in a wireless network setting with N users and M base stations, where user locations are given by the vector $\boldsymbol{x}$. By mapping the normalized location information to actual coordinates and using differentiable functions, we can compute the SNR and apply thresholding in a way that is fully differentiable. This approach allows for the integration of the wireless network simulation into gradient-based optimization and learning frameworks.

17. Discussion / Idea Note

New Idea: CSI Generator. CSI Compression: what kind of CSI is hard to be compressed?

Experiments:

- Celeb A: Geneder
- CSI: Compression
- MNIST: exclusive generation
- Synthetic setup: like weibull distribution
- Rare Probability measure

While CFL has the potential to deliver tremendous advantages, algorithmic developments have proved to be remarkably challenging due to the need to jointly cluster and train over distributed data. Recent theoretically grounded CFL algorithms group clients based on the models which currently provide each the best performance [**?** **?**], or recursively perform bipartitioning when clients' gradients differ on a converged global model [**?**]. Unfortunately, the state-of-the-art CFL algorithms face difficulties in various relevant settings including e.g., linear regression tasks (See **??**). This raises the following question:

"How do we generate samples causing high loss while the sample is also plausible?"

This is very important problem for digital twin, Root cause analysis.

1. one can just derive a large number of samples and just measure the performance and get the samples... however, if the probability of such samples are super low, then it takes huge computation power.

2. Then, can we just do some reverse engineering baesd on the loss function? No, because, if the probability of such sample is near zero, those kind of samples does not need to be considered.

See the samples.

To this end, we can say we are interested in $g(\mathbf{x}) \propto$