Diffusion Models: A Comprehensive Survey of Methods and Applications

LING YANG, Peking University, China
ZHILONG ZHANG*, Peking University, China
YANG SONG, OpenAI, USA
SHENDA HONG, Peking University, China
RUNSHENG XU, University of California, Los Angeles, USA
YUE ZHAO, Carnegie Mellon University, USA
WENTAO ZHANG, Peking University, China
BIN CUI, Peking University, China
MING-HSUAN YANG†, University of California at Merced, USA

Diffusion models have emerged as a powerful new family of deep generative models with record-breaking performance in many applications, including image synthesis, video generation, and molecule design. In this survey, we provide an overview of the rapidly expanding body of work on diffusion models, categorizing the research into three key areas: efficient sampling, improved likelihood estimation, and handling data with special structures. We also discuss the potential for combining diffusion models with other generative models for enhanced results. We further review the wide-ranging applications of diffusion models in fields spanning from computer vision, natural language processing, temporal data modeling, to interdisciplinary applications in other scientific disciplines. This survey aims to provide a contextualized, in-depth look at the state of diffusion models, identifying the key areas of focus and pointing to potential areas for further exploration. Github: https://github.com/YangLing0818/Diffusion-Models-Papers-Survey-Taxonomy.

CCS Concepts: • **Computing methodologies** → **Computer vision tasks**; *Natural language generation*; Machine learning approaches.

Additional Key Words and Phrases: Generative Models, Diffusion Models, Score-Based Generative Models, Stochastic Differential Equations

ACM Reference Format:
Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion Models: A Comprehensive Survey of Methods and Applications. 1, 1 (December 2023), 58 pages. https://doi.org/10.1145/3626235

*Contributed equally.
†Wentao Zhang, Bin Cui, and Ming-Hsuan Yang are corresponding authors.

Authors' addresses: Ling Yang, Peking University, China, yangling0818@163.com; Zhilong Zhang, Peking University, China, zhilong.zhang@bjmu.edu.cn; Yang Song, OpenAI, USA, songyang@openai.com; Shenda Hong, Peking University, China, hongshenda@pku.edu.cn; Runsheng Xu, University of California, Los Angeles, USA, rxx3386@ucla.edu; Yue Zhao, Carnegie Mellon University, USA, zhaoy@cmu.edu; Wentao Zhang, Peking University, China, wentao.zhang@pku.edu.cn; Bin Cui, Peking University, China, bin.cui@pku.edu.cn; Ming-Hsuan Yang, University of California at Merced, USA, mhyang@ucmerced.edu.

Manuscript submitted to ACM

Contents

1 INTRODUCTION

Diffusion models [111, 275, 280, 285] have emerged as the new state-of-the-art family of deep generative models. They have broken the long-time dominance of generative adversarial networks (GANs) [88] in the challenging task of image synthesis [60, 111, 280, 285] and have also shown potential in a variety of domains, ranging from computer vision [4, 15, 25, 29, 112, 114, 145, 149, 171, 194, 206, 225, 257, 259, 315, 354, 355, 379, 389], natural language processing [9, 117, 175, 264, 361], temporal data modeling [3, 39, 159, 249, 291, 335], multi-modal modeling [10, 243, 255, 258, 386],

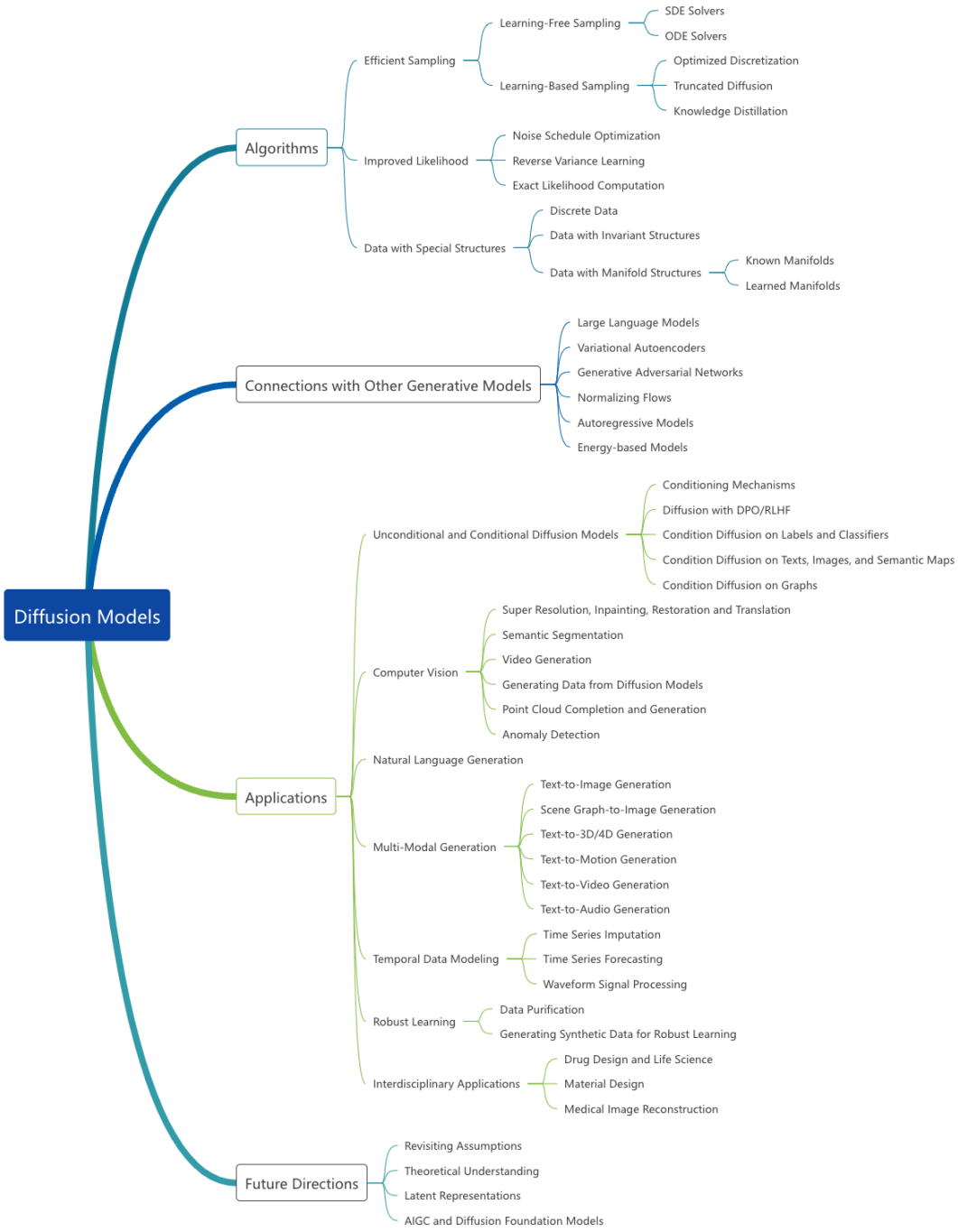

Fig. 1. Taxonomy of diffusion models variants (in Sections 3 to 5), connections with other generative models (in Section 6), applications of diffusion models (in Section 7), and future directions (in Section 8).

robust machine learning [23, 33, 144, 308, 357], to interdisciplinary applications in fields such as computational chemistry [5, 115, 133, 166, 169, 196, 327] and medical image reconstruction [31, 47–49, 53, 202, 230, 284, 328].

Numerous methods have been developed to improve diffusion models, either by enhancing empirical performance [214, 277, 281] or by extending the model's capacity from a theoretical perspective [187, 188, 279, 285, 371]. Over the past two years, the body of research on diffusion models has grown significantly, making it increasingly challenging for new researchers to stay abreast of the recent developments in the field. Additionally, the sheer volume of work can obscure major trends and hinder further research progress. This survey aims to address these problems by providing a comprehensive overview of the state of diffusion model research, categorizing various approaches, and highlighting key advances. We hope this survey to serve as a helpful entry point for researchers new to the field while providing a broader perspective for experienced researchers.

In this paper, we first explain the foundations of diffusion models (Section 2), providing a brief but self-contained introduction to three predominant formulations: denoising diffusion probabilistic models (DDPMs) [111, 275], score-based generative models (SGMs) [280, 281], and stochastic differential equations (Score SDEs) [141, 279, 285]. Key to all these approaches is to progressively perturb data with intensifying random noise (called the "diffusion" process), then successively remove noise to generate new data samples. We clarify how they work under the same principle of diffusion and explain how these three models are connected and can be reduced to one another.

Next, we present a taxonomy of recent research that maps out the field of diffusion models, categorizing it into three key areas: efficient sampling (Section 3), improved likelihood estimation (Section 4), and methods for handling data with special structures (Section 5), such as relational data, data with permutation/rotational invariance, and data residing on manifolds. We further examine the models by breaking each category into more detailed sub-categories, as illustrated in Fig. 1. In addition, we discuss the connections of diffusion models to other deep generative models (Section 6), including variational autoencoders (VAEs) [156, 252], generative adversarial networks (GANs) [88], normalizing flows [62, 64, 226, 254], autoregressive models [302], and energy-based models (EBMs) [165, 283]. By combining these models with diffusion models, researchers have the potential to achieve even stronger performance.

Following that, our survey reviews six major categories of application that diffusion models have been applied to in the existing research (Section 7): computer vision, natural language process, temporal data modeling, multi-modal learning, robust learning, and interdisciplinary applications. For each task, we provide a definition, describe how diffusion models can be employed to address it and summarize relevant previous work. We conclude our paper (Sections 8 and 9) by providing an outlook on possible future directions for this exciting new area of research.

2 FOUNDATIONS OF DIFFUSION MODELS

Diffusion models are a family of probabilistic generative models that progressively destruct data by injecting noise, then learn to reverse this process for sample generation. We present the intuition of diffusion models in Fig. 2. Current research on diffusion models is mostly based on three predominant formulations: denoising diffusion probabilistic models (DDPMs) [111, 214, 275], score-based generative models (SGMs) [280, 281], and stochastic differential equations (Score SDEs) [279, 285]. We give a self-contained introduction to these three formulations in this section, while discussing their connections with each other along the way.

2.1 Denoising Diffusion Probabilistic Models (DDPMs)

A *denoising diffusion probabilistic model* (DDPM) [111, 275] makes use of two Markov chains: a forward chain that perturbs data to noise, and a reverse chain that converts noise back to data. The former is typically hand-designed with

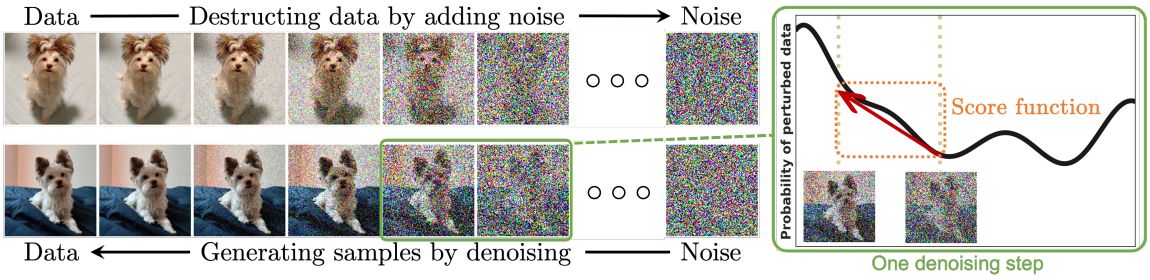

Fig. 2. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise. Each denoising step in the reverse process typically requires estimating the score function (see the illustrative figure on the right), which is a gradient pointing to the directions of data with higher likelihood and less noise.

the goal to transform any data distribution into a simple prior distribution (*e.g.*, standard Gaussian), while the latter Markov chain reverses the former by learning transition kernels parameterized by deep neural networks. New data points are subsequently generated by first sampling a random vector from the prior distribution, followed by ancestral sampling through the reverse Markov chain [158].

Formally, given a data distribution $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, the forward Markov process generates a sequence of random variables $\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_T$ with transition kernel $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$. Using the chain rule of probability and the Markov property, we can factorize the joint distribution of $\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_T$ conditioned on $\mathbf{x}_0$, denoted as $q(\mathbf{x}_1, \dots, \mathbf{x}_T \mid \mathbf{x}_0)$, into

$$q(\mathbf{x}_1, \dots, \mathbf{x}_T \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}). \tag{1}$$

In DDPMs, we handcraft the transition kernel $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ to incrementally transform the data distribution $q(\mathbf{x}_0)$ into a tractable prior distribution. One typical design for the transition kernel is Gaussian perturbation, and the most common choice for the transition kernel is

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}), \tag{2}$$

where $\beta_t \in (0, 1)$ is a hyperparameter chosen ahead of model training. We use this kernel to simply our discussion here, although other types of kernels are also applicable in the same vein. As observed by Sohl-Dickstein et al. (2015) [275], this Gaussian transition kernel allows us to marginalize the joint distribution in Eq. (1) to obtain the analytical form of $q(\mathbf{x}_t \mid \mathbf{x}_0)$ for all $t \in \{0, 1, \cdots, T\}$. Specifically, with $\alpha_t \coloneqq 1 - \beta_t$ and $\bar{\alpha}_t \coloneqq \prod_{s=0}^{t} \alpha_s$, we have

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}). \tag{3}$$

Given $\mathbf{x}_0$, we can easily obtain a sample of $\mathbf{x}_t$ by sampling a Gaussian vector $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and applying the transformation

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}. \tag{4}$$

When $\bar{\alpha}_T \approx 0$, $\mathbf{x}_T$ is almost Gaussian in distribution, so we have $q(\mathbf{x}_T) \coloneqq \int q(\mathbf{x}_T \mid \mathbf{x}_0) q(\mathbf{x}_0) \mathrm{d}\mathbf{x}_0 \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$.

Intuitively speaking, this forward process slowly injects noise to data until all structures are lost. For generating new data samples, DDPMs start by first generating an unstructured noise vector from the prior distribution (which is typically trivial to obtain), then gradually remove noise therein by running a learnable Markov chain in the reverse time direction. Specifically, the reverse Markov chain is parameterized by a prior distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ and a

learnable transition kernel $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$. We choose the prior distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ because the forward process is constructed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$. The learnable transition kernel $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ takes the form of

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \tag{5}$$

where θ denotes model parameters, and the mean $\mu_\theta(\mathbf{x}_t, t)$ and variance $\Sigma_\theta(\mathbf{x}_t, t)$ are parameterized by deep neural networks. With this reverse Markov chain in hand, we can generate a data sample $\mathbf{x}_0$ by first sampling a noise vector $\mathbf{x}_T \sim p(\mathbf{x}_T)$, then iteratively sampling from the learnable transition kernel $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ until $t = 1$.

Key to the success of this sampling process is training the reverse Markov chain to match the actual time reversal of the forward Markov chain. That is, we have to adjust the parameter θ so that the joint distribution of the reverse Markov chain $p_\theta(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T) \coloneqq p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ closely approximates that of the forward process $q(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T) \coloneqq q(\mathbf{x}_0) \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ (Eq. (1)). This is achieved by minimizing the Kullback-Leibler (KL) divergence between these two:

$$\mathrm{KL}(q(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T) \,||\, p_\theta(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)) \tag{6}$$

$$\overset{(i)}{=} -\mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)} [\log p_\theta(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)] + \text{const} \tag{7}$$

$$\overset{(ii)}{=} \underbrace{\mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)} \left[-\log p(\mathbf{x}_T) - \sum_{t=1}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)}{q(\mathbf{x}_t \mid \mathbf{x}_{t-1})} \right]}_{\coloneqq -L_{\mathrm{VLB}}(\mathbf{x}_0)} + \text{const} \tag{8}$$

$$\overset{(iii)}{\geq} \mathbb{E} [-\log p_\theta(\mathbf{x}_0)] + \text{const}, \tag{9}$$

where (i) is from the definition of KL divergence, (ii) is from the fact that $q(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)$ and $p_\theta(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T)$ are both products of distributions, and (iii) is from Jensen's inequality. The first term in Eq. (8) is the variational lower bound (VLB) of the log-likelihood of the data $\mathbf{x}_0$, a common objective for training probabilistic generative models. We use "const" to symbolize a constant that does not depend on the model parameter θ and hence does not affect optimization. The objective of DDPM training is to maximize the VLB (or equivalently, minimizing the negative VLB), which is particularly easy to optimize because it is a sum of independent terms, and can thus be estimated efficiently by Monte Carlo sampling [212] and optimized effectively by stochastic optimization [286].

Ho et al. (2020) [111] propose to reweight various terms in L_{VLB} for better sample quality and noticed an important equivalence between the resulting loss function and the training objective for noise-conditional score networks (NCSNs), one type of *score-based generative models*, in Song and Ermon [280]. The loss in [111] takes the form of

$$\mathbb{E}_{t \sim \mathcal{U}[\![1,T]\!], \mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left[\lambda(t) \, \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2 \right] \tag{10}$$

where $\lambda(t)$ is a positive weighting function, $\mathbf{x}_t$ is computed from $\mathbf{x}_0$ and $\boldsymbol{\epsilon}$ by Eq. (4), $\mathcal{U}[\![1, T]\!]$ is a uniform distribution over the set $\{1, 2, \cdots, T\}$, and $\boldsymbol{\epsilon}_\theta$ is a deep neural network with parameter θ that predicts the noise vector $\boldsymbol{\epsilon}$ given $\mathbf{x}_t$ and t. This objective reduces to Eq. (8) for a particular choice of the weighting function $\lambda(t)$, and has the same form as the loss of denoising score matching over multiple noise scales for training score-based generative models [280], another formulation of diffusion models to be discussed in the next section.

2.2 Score-Based Generative Models (SGMs)

At the core of score-based generative models [280, 281] is the concept of *(Stein) score* (a.k.a., score or score function) [126]. Given a probability density function $p(\mathbf{x})$, its score function is defined as the gradient of the log probability density $\nabla_{\mathbf{x}} \log p(\mathbf{x})$. Unlike the commonly used *Fisher score* $\nabla_\theta \log p_\theta(\mathbf{x})$ in statistics, the Stein score considered here is a function of the data $\mathbf{x}$ rather than the model parameter θ. It is a vector field that points to directions along which the probability density function has the largest growth rate.

The key idea of score-based generative models (SGMs) [280] is to perturb data with a sequence of intensifying Gaussian noise and jointly estimate the score functions for all noisy data distributions by training a deep neural network model conditioned on noise levels (called a noise-conditional score network, NCSN, in [280]). Samples are generated by chaining the score functions at decreasing noise levels with score-based sampling approaches, including Langevin Monte Carlo [96, 137, 227, 280, 285], stochastic differential equations [136, 285], ordinary differential equations [141, 188, 279, 285, 371], and their various combinations [285]. Training and sampling are completely decoupled in the formulation of score-based generative models, so one can use a multitude of sampling techniques after the estimation of score functions.

With similar notations in Section 2.1, we let $q(\mathbf{x}_0)$ be the data distribution, and $0 < \sigma_1 < \sigma_2 < \cdots < \sigma_t < \cdots < \sigma_T$ be a sequence of noise levels. A typical example of SGMs involves perturbing a data point $\mathbf{x}_0$ to $\mathbf{x}_t$ by the Gaussian noise distribution $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_0, \sigma_t^2 I)$. This yields a sequence of noisy data densities $q(\mathbf{x}_1), q(\mathbf{x}_2), \cdots, q(\mathbf{x}_T)$, where $q(\mathbf{x}_t) \coloneqq \int q(\mathbf{x}_t) q(\mathbf{x}_0) \mathrm{d}\mathbf{x}_0$. A noise-conditional score network is a deep neural network $\mathbf{s}_\theta(\mathbf{x}, t)$ trained to estimate the score function $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$. Learning score functions from data (a.k.a., score estimate) has established techniques such as score matching [126], denoising score matching [245, 246, 304], and sliced score matching [282], so we can directly employ one of them to train our noise-conditional score networks from perturbed data points. For example, with denoising score matching and similar notations in Eq. (10), the training objective is given by

$$\mathbb{E}_{t\sim\mathcal{U}[\![1,T]\!],\mathbf{x}_0\sim q(\mathbf{x}_0),\mathbf{x}_t\sim q(\mathbf{x}_t|\mathbf{x}_0)}\left[\lambda(t)\sigma_t^2 \left\|\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) - \mathbf{s}_\theta(\mathbf{x}_t, t)\right\|^2\right] \tag{11}$$

$$\overset{(i)}{=} \mathbb{E}_{t\sim\mathcal{U}[\![1,T]\!],\mathbf{x}_0\sim q(\mathbf{x}_0),\mathbf{x}_t\sim q(\mathbf{x}_t|\mathbf{x}_0)}\left[\lambda(t)\sigma_t^2 \left\|\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t \mid \mathbf{x}_0) - \mathbf{s}_\theta(\mathbf{x}_t, t)\right\|^2\right] + \text{const} \tag{12}$$

$$\overset{(ii)}{=} \mathbb{E}_{t\sim\mathcal{U}[\![1,T]\!],\mathbf{x}_0\sim q(\mathbf{x}_0),\mathbf{x}_t\sim q(\mathbf{x}_t|\mathbf{x}_0)}\left[\lambda(t) \left\|-\frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t} - \sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)\right\|^2\right] + \text{const} \tag{13}$$

$$\overset{(iii)}{=} \mathbb{E}_{t\sim\mathcal{U}[\![1,T]\!],\mathbf{x}_0\sim q(\mathbf{x}_0),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\lambda(t) \left\|\boldsymbol{\epsilon} + \sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)\right\|^2\right] + \text{const}, \tag{14}$$

where (i) is derived by [304], (ii) is from the assumption that $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathbf{N}(\mathbf{x}_t; \mathbf{x}_0, \sigma_t^2 \mathbf{I})$, and (iii) is from the fact that $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}$. Again, we denote by $\lambda(t)$ a positive weighting function, and "const" a constant that does not depend on the trainable parameter θ. Comparing Eq. (14) with Eq. (10), it is clear that the training objectives of DDPMs and SGMs are equivalent, once we set $\boldsymbol{\epsilon}_\theta(\mathbf{x}, t) = -\sigma_t \mathbf{s}_\theta(\mathbf{x}, t)$. Moreover, one can generalize the score matching with higher order. High-order derivatives of data density provide additional local information about the data distribution. Meng et al. [209] proposes a generalized denoising score matching method to efficiently estimate the high-order score function. The proposed model can improve the mixing speed of Langevin dynamics and thus the sampling efficiency of diffusion models.

For sample generation, SGMs leverage iterative approaches to produce samples from $\mathbf{s}_\theta(\mathbf{x}, T), \mathbf{s}_\theta(\mathbf{x}, T-1), \cdots, \mathbf{s}_\theta(\mathbf{x}, 0)$ in succession. Many sampling approaches exist due to the decoupling of training and inference in SGMs, some of which are discussed in the next section. Here we introduce the first sampling method for SGMs, called annealed Langevin

dynamics (ALD) [280]. Let N be the number of iterations per time step and $s_t > 0$ be the step size. We first initialize ALD with $\mathbf{x}_T^{(N)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, then apply Langevin Monte Carlo for $t = T, T-1, \cdots, 1$ one after the other. At each time step $0 \leq t < T$, we start with $\mathbf{x}_t^{(0)} = \mathbf{x}_{t+1}^{(N)}$, before iterating according to the following update rule for $i = 0, 1, \cdots, N-1$:

$$\begin{aligned} \boldsymbol{\epsilon}^{(i)} &\leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \mathbf{x}_t^{(i+1)} &\leftarrow \mathbf{x}_t^{(i)} + \frac{1}{2} s_t \mathbf{s}_\theta(\mathbf{x}_t^{(i)}, t) + \sqrt{s_t}\boldsymbol{\epsilon}^{(i)}. \end{aligned}$$

The theory of Langevin Monte Carlo [227] guarantees that as $s_t \to 0$ and $N \to \infty$, $\mathbf{x}_0^{(N)}$ becomes a valid sample from the data distribution $q(\mathbf{x}_0)$.

2.3 Stochastic Differential Equations (Score SDEs)

DDPMs and SGMs can be further generalized to the case of infinite time steps or noise levels, where the perturbation and denoising processes are solutions to stochastic differential equations (SDEs). We call this formulation *Score SDE* [285], as it leverages SDEs for noise perturbation and sample generation, and the denoising process requires estimating score functions of noisy data distributions.

Score SDEs perturb data to noise with a diffusion process governed by the following stochastic differential equation (SDE) [285]:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \tag{15}$$

where $\mathbf{f}(\mathbf{x}, t)$ and $g(t)$ are diffusion and drift functions of the SDE, and $\mathbf{w}$ is a standard Wiener process (a.k.a., Brownian motion). The forward processes in DDPMs and SGMs are both discretizations of this SDE. As demonstrated in Song et al. (2020) [285], for DDPMs, the corresponding SDE is:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}dt + \sqrt{\beta(t)}\mathrm{d}\mathbf{w} \tag{16}$$

where $\beta(\frac{t}{T}) = T\beta_t$ as T goes to infinity; and for SGMs, the corresponding SDE is given by

$$\mathrm{d}\mathbf{x} = \sqrt{\frac{\mathrm{d}[\sigma(t)^2]}{\mathrm{d}t}}\mathrm{d}\mathbf{w}, \tag{17}$$

where $\sigma(\frac{t}{T}) = \sigma_t$ as T goes to infinity. Here we use $q_t(\mathbf{x})$ to denote the distribution of $\mathbf{x}_t$ in the forward process.

Crucially, for any diffusion process in the form of Eq. (15), Anderson [6] shows that it can be reversed by solving the following reverse-time SDE:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g(t)^2\nabla_{\mathbf{x}} \log q_t(\mathbf{x})\right]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \tag{18}$$

where $\bar{\mathbf{w}}$ is a standard Wiener process when time flows backwards, and $\mathrm{d}t$ denotes an infinitesimal negative time step. The solution trajectories of this reverse SDE share the same marginal densities as those of the forward SDE, except that they evolve in the opposite time direction [285]. Intuitively, solutions to the reverse-time SDE are diffusion processes that gradually convert noise to data. Moreover, Song et al. (2020) [285] prove the existence of an ordinary differential equation (ODE), namely the *probability flow ODE*, whose trajectories have the same marginals as the reverse-time SDE. The probability flow ODE is given by:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}} \log q_t(\mathbf{x})\right]\mathrm{d}t. \tag{19}$$

Both the reverse-time SDE and the probability flow ODE allow sampling from the same data distribution as their trajectories have the same marginals.

Once the score function at each time step t, $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$, is known, we unlock both the reverse-time SDE (Eq. (18)) and the probability flow ODE (Eq. (19)) and can subsequently generate samples by solving them with various numerical techniques, such as annealed Langevin dynamics [280] (*cf*., Section 2.2), numerical SDE solvers [136, 285], numerical ODE solvers [141, 188, 277, 285, 371], and predictor-corrector methods (combination of MCMC and numerical ODE/SDE solvers) [285]. Like in SGMs, we parameterize a time-dependent score model $\mathbf{s}_\theta(\mathbf{x}_t, t)$ to estimate the score function by generalizing the score matching objective in Eq. (14) to continuous time, leading to the following objective:

$$\mathbb{E}_{t\sim\mathcal{U}[0,T],\mathbf{x}_0\sim q(\mathbf{x}_0),\mathbf{x}_t\sim q(\mathbf{x}_t|\mathbf{x}_0)}\left[\lambda(t)\left\|\mathbf{s}_\theta(\mathbf{x}_t,t)-\nabla_{\mathbf{x}_t}\log q_{0t}(\mathbf{x}_t\mid\mathbf{x}_0)\right\|^2\right], \tag{20}$$

where $\mathcal{U}[0,T]$ denotes the uniform distribution over $[0,T]$, and the remaining notations follow Eq. (14).

Subsequent research on diffusion models focuses on improving these classical approaches (DDPMs, SGMs, and Score SDEs) from three major directions: faster and more efficient sampling, more accurate likelihood and density estimation, and handling data with special structures (such as permutation invariance, manifold structures, and discrete data). We survey each direction extensively in the next three sections (Sections 3 to 5). In Table 1, we list the three types of diffusion models with more detailed categorization, corresponding articles and years, under continuous and discrete time settings.

3 DIFFUSION MODELS WITH EFFICIENT SAMPLING

Generating samples from diffusion models typically demands iterative approaches that involve a large number of evaluation steps. A great deal of recent work has focused on speeding up the sampling process while also improving quality of the resulting samples. We classify these efficient sampling methods into two main categories: those that do not involve learning (learning-free sampling) and those that require an additional learning process after the diffusion model has been trained (learning-based sampling).

3.1 Learning-Free Sampling

Many samplers for diffusion models rely on discretizing either the reverse-time SDE present in Eq. (18) or the probability flow ODE from Eq. (19). Since the cost of sampling increases proportionally with the number of discretized time steps, many researchers have focused on developing discretization schemes that reduce the number of time steps while also minimizing discretization errors.

3.1.1 SDE Solvers. The generation process of DDPM [111, 275] can be viewed as a particular discretization of the reverse-time SDE. As discussed in Section 2.3, the forward process of DDPM discretizes the SDE in Eq. (16), whose corresponding reverse SDE takes the form of

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)(\mathbf{x}_t - \nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t))\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w} \tag{21}$$

Song et al. (2020) [285] show that the reverse Markov chain defined by Eq. (5) amounts to a numerical SDE solver for Eq. (21).

Noise-Conditional Score Networks (NCSNs) [280] and Critically-Damped Langevin Diffusion (CLD) [66] both solve the reverse-time SDE with inspirations from Langevin dynamics. In particular, NCSNs leverage annealed Langevin dynamics (ALD, *cf*., Section 2.2) to iteratively generate data while smoothly reducing noise level until the generated

Table 1. Three types of diffusion models are listed with corresponding articles and years, under continuous and discrete settings.

Primary	Secondary	Tertiary	Article	Year	Setting
Efficient Sampling	Learning-Free Sampling	SDE Solvers	Song et al. [285]	2020	Continuous
			Dockhorn et al. [66]	2021	Continuous
			Jolicoeur et al. [137]	2021	Continuous
			Jolicoeur et al. [136]	2021	Continuous
			Chuang et al. [48]	2022	Continuous
		ODE Solvers	Song et al. [280]	2019	Continuous
			Karras et al. [141]	2022	Continuous
			Liu et al. [181]	2021	Continuous
			Song et al. [277]	2020	Continuous
			Zhang et al. [372]	2022	Continuous
			Karras et al. [141]	2022	Continuous
			Lu et al. [188]	2022	Continuous
			Zhang et al. [371]	2022	Continuous
	Learning-Based Sampling	Optimized Discretization	Watson et al. [313]	2021	Discrete
			Watson et al. [312]	2021	Discrete
			Dockhorn et al. [67]	2021	Continuous
		Knowledge Distillation	Salimans et al. [260]	2021	Discrete
			Luhman et al. [190]	2021	Discrete
			Meng et al. [205]	2022	Discrete
		Truncated Diffusion	Lyu et al. [199]	2022	Discrete
			Zheng et al. [381]	2022	Discrete
Improved Likelihood	Noise Schedule Optimization	Noise Schedule Optimization	Nichol et al. [214]	2021	Discrete
			Kingma et al. [154]	2021	Discrete
			Huang et al. [124]	2024	Discrete
			Yang et al. [351]	2024	Discrete
	Reverse Variance Learning	Reverse Variance Learning	Bao et al.[13]	2021	Discrete
			Nichol et al. [214]	2021	Discrete
	Exact Likelihood Computation	Exact Likelihood Computation	Song et al. [279]	2021	Continuous
			Huang et al. [119]	2021	Continuous
			Song et al. [285]	2020	Continuous
			Lu et al. [187]	2022	Continuous
Data with Special Structures	Manifold Structures	Learned Manifolds	Vahdat et al. [299]	2021	Continuous
			Yang et al. [346]	2024	Discrete
			Ramesh et al. [243]	2022	Discrete
			Rombach et al. [255]	2022	Discrete
		Known Manifolds	Bortoli et al. [56]	2022	Continuous
			Huang et al. [118]	2022	Continuous
	Data with Invariant Structures	Data with Invariant Structures	Niu et al. [219]	2020	Discrete
			Jo et al. [134]	2022	Continuous
			Shi et al. [267]	2022	Continuous
			Xu et al. [333]	2021	Discrete
	Discrete Data	Discrete Data	Meng et al. [204]	2022	Discrete
			liu et al. [184]	2023	Continuous
			Sohl et al. [275]	2015	Discrete
			Austin et al. [9]	2021	Discrete
			Xie et al. [326]	2022	Discrete
			Gu et al. [98]	2022	Discrete
			Campbell et al. [30]	2022	Continuous

data distribution converges to the original data distribution. Although the sampling trajectories of ALD are not exact solutions to the reverse-time SDE, they have the correct marginals and hence produce correct samples under the assumption that Langevin dynamics converges to its equilibrium at every noise level. The method of ALD is further improved by Consistent Annealed Sampling (CAS) [137], a score-based MCMC approach with better scaling of time steps and added noise. Inspired by statistical mechanics, CLD proposes an augmented SDE with an auxiliary velocity term resembling underdamped Langevin diffusion. To obtain the time reversal of the extended SDE, CLD only needs to

learn the score function of the conditional distribution of velocity given data, arguably easier than learning scores of data directly. The added velocity term is reported to improve sampling speed as well as quality.

The reverse diffusion method proposed in [285] discretizes the reverse-time SDE in the same way as the forward one. For any one-step discretization of the forward SDE, one may write the general form below:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{f}_i(\mathbf{x}_i) + \mathbf{g}_i \mathbf{z}_i, \quad i = 0, 1, \cdots, N-1 \tag{22}$$

where $\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{f}_i$ and $\mathbf{g}_i$ are determined by drift/diffusion coefficients of the SDE and the discretization scheme. Reverse diffusion proposes to discretize the reverse-time SDE similarly to the forward SDE, *i.e.*,

$$\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{f}_{i+1}(\mathbf{x}_{i+1}) + \mathbf{g}_{i+1}\mathbf{g}_{i+1}^t \mathbf{s}_{\theta^*}(\mathbf{x}_{i+1}, t_{i+1}) + \mathbf{g}_{i+1}\mathbf{z}_i \quad i = 0, 1, \cdots, N-1 \tag{23}$$

where $\mathbf{s}_{\theta^*}(\mathbf{x}_i, t_i)$ is the trained noise-conditional score model. Song et al. (2020) [285] prove that the reverse diffusion method is a numerical SDE solver for the reverse-time SDE in Eq. (18). This process can be applied to any types of forward SDEs, and empirical results indicate this sampler performs slightly better than DDPM [285] for a particular type of SDEs called the VP-SDE.

Jolicoeur-Martineau et al. (2021) [136] develop an SDE solver with adaptive step sizes for faster generation. The step size is controlled by comparing the output of a high-order SDE solver versus the output of an low-order SDE solver. At each time step, the high- and low-order solvers generate new sample $\mathbf{x}'_{\text{high}}$ and $\mathbf{x}'_{\text{low}}$ from the previous sample $\mathbf{x}'_{prev}$ respectively. The step size is then adjusted by comparing the difference between the two samples. If $\mathbf{x}'_{\text{high}}$ and $\mathbf{x}'_{\text{low}}$ are similar, the algorithm will return $\mathbf{x}'_{\text{high}}$ and then increase the step size. The similarity between $\mathbf{x}'_{\text{high}}$ and $\mathbf{x}'_{\text{low}}$ is measured by:

$$E_q = \left\| \frac{\mathbf{x}'_{\text{low}} - \mathbf{x}'_{\text{high}}}{\delta(\mathbf{x}', \mathbf{x}'_{\text{prev}})} \right\|^2 \tag{24}$$

where $\delta(\mathbf{x}'_{\text{low}}, \mathbf{x}'_{\text{prev}}) \coloneqq \max(\epsilon_{abs}, \epsilon_{rel} \max(|\mathbf{x}'_{\text{low}}|, |\mathbf{x}'_{\text{prev}}|))$, and ϵ_{abs} and ϵ_{rel} are absolute and relative tolerances.

The predictor-corrector method proposed in [285] solves the reverse SDE by combining numerical SDE solvers ("predictor") and iterative Markov chain Monte Carlo (MCMC) approaches ("corrector"). At each time step, the predictor-corrector method first employs a numerical SDE solver to produce a coarse sample, followed by a "corrector" that corrects the sample' marginal distribution with score-based MCMC. The resulting samples have the same time-marginals as solution trajectories of the reverse-time SDE, *i.e.*, they are equivalent in distribution at all time steps. Empirical results demonstrate that adding a corrector based on Langevin Monte Carlo is more efficient than using an additional predictor without correctors [285]. Karras et al. (2022) [141] further improve the Langevin dynamics corrector in [285] by proposing a Langevin-like "churn" step of adding and removing noise, achieving new state-of-the-art sample quality on datasets like CIFAR-10 [161] and ImageNet-64 [58].

3.1.2 ODE solvers. A large body of works on faster diffusion samplers are based on solving the probability flow ODE (Eq. (19)) introduced in Section 2.3. In contrast to SDE solvers, the trajectories of ODE solvers are deterministic and thus not affected by stochastic fluctuations. These deterministic ODE solvers typically converge much faster than their stochastic counterparts at the cost of slightly inferior sample quality.

Denoising Diffusion Implicit Models (DDIM) [277] is one of the earliest work on accelerating diffusion model sampling. The original motivation was to extend the original DDPM to non-Markovian case with the following Markov

chain

$$q(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_0) \tag{25}$$

$$q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1} | \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \mathbf{I}) \tag{26}$$

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) \coloneqq \sqrt{\overline{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \overline{\alpha}_{t-1} - \sigma_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\overline{\alpha}_t}\mathbf{x}_0}{\sqrt{1 - \overline{\alpha}_t}} \tag{27}$$

This formulation encapsulates DDPM and DDIM as special cases, where DDPM corresponds to setting $\sigma_t^2 = \frac{\hat{\beta}_{t-1}}{\hat{\beta}_t}\beta_t$ and DDIM corresponds to setting $\sigma_t^2 = 0$. DDIM learns a Markov chain to reverse this non-Markov perturbation process, which is fully deterministic when $\sigma_t^2 = 0$. It is observed in [141, 188, 260, 277] that the DDIM sampling process amounts to a special discretization scheme of the probability flow ODE. Inspired by an analysis of DDIM on a singleton dataset, generalized Denoising Diffusion Implicit Models (gDDIM) [372] proposes a modified parameterization of the score network that enables deterministic sampling for more general diffusion processes, such as the one in Critically-Damped Langevin Diffusion (CLD) [66]. PNDM [181] proposes a pseudo numerical method to generate sample along a specific manifold in $\mathcal{R}^N$. It uses numerical solver with nonlinear transfer part to solve differential equation on manifolds and then generates sample, which encapsulates DDIM as a special case.

Through extensive experimental investigations, Karras et al. (2022) [141] show that Heun's 2^{nd} order method [8] provides an excellent trade off between sample quality and sampling speed. The higher-order solver leads to smaller discretization error at the cost of one additional evaluation of the learned score function per time step. Heun's method generates samples of comparable, if not better quality than Euler's method with fewer sampling steps.

Diffusion Exponential Integrator Sampler [371] and DPM-solver [188] leverage the semi-linear structure of probability flow ODE to develop customized ODE solvers that are more efficient than general-purpose Runge-Kutta methods. Specifically, the linear part of probability flow ODE can be analytically computed, while the non-linear part can be solved with techniques similar to exponential integrators in the field of ODE solvers. These methods contain DDIM as a first-order approximation. However, they also allow for higher order integrators, which can produce high-quality samples in just 10 to 20 iterations—far fewer than the hundreds of iterations typically required by diffusion models without accelerated sampling.

3.2 Learning-Based Sampling

Learning-based sampling is another efficient approach for diffusion models. By using partial steps or training a sampler for the reverse process, this method achieves faster sampling speeds at the expense of slight degradation in sample quality. Unlike learning-free approaches that use handcrafted steps, learning-based sampling typically involves selecting steps by optimizing certain learning objectives.

3.2.1 Optimized Discretization. Given a pre-trained diffusion model, Watson et al. (2021) [313] put forth a strategy for finding the optimal discretization scheme by selecting the best K time steps to maximize the training objective for DDPMs. Key to this approach is the observation that the DDPM objective can be broken down into a sum of individual terms, making it well suited for dynamic programming. However, it is well known that the variational lower bound used for DDPM training does not correlate directly with sample quality [294]. A subsequent work, called Differentiable Diffusion Sampler Search [312], addresses this issue by directly optimizing a common metric for sample quality called the Kernel Inception Distance (KID) [22]. This optimization is feasible with the help of reparameterization [156, 252]

and gradient rematerialization. Based on truncated Taylor methods, Dockhorn et al. (2022) [67] derive a second-order solver for accelerating synthesis by training a additional head on top of the first-order score network.

3.2.2 Truncated Diffusion. One can improve sampling speed by truncating the forward and reverse diffusion processes [199, 381]. The key idea is to halt the forward diffusion process early on, after just a few steps, and to begin the reverse denoising process with a non-Gaussian distribution. Samples from this distribution can be obtained efficiently by diffusing samples from pre-trained generative models, such as variational autoencoders [156, 252] or generative adversarial networks [88].

3.2.3 Knowledge Distillation. Approaches that use knowledge distillation [190, 205, 260] can significantly improve the sampling speed of diffusion models. Specifically, in Progressive Distillation [260], the authors propose distilling the full sampling process into a faster sampler that requires only half as many steps. By parameterizing the new sampler as a deep neural network, authors are able to train the sampler to match the input and output of the DDIM sampling process. Repeating this procedure can further reduce sampling steps, although fewer steps can result in reduced sample quality. To address this issue, the authors suggest new parameterizations for diffusion models and new weighting schemes for the objective function.

4 DIFFUSION MODELS WITH IMPROVED LIKELIHOOD

As discussed in Section 2.1, the training objective for diffusion models is a (negative) variational lower bound (VLB) on the log-likelihood. This bound, however, may not be tight in many cases [154], leading to potentially suboptimal log-likelihoods from diffusion models. In this section, we survey recent works on likelihood maximization for diffusion models. We focus on three types of methods: noise schedule optimization, reverse variance learning, and exact log-likelihood evaluation.

4.1 Noise Schedule Optimization

In the classical formulation of diffusion models, noise schedules in the forward process are handcrafted without trainable parameters. By optimizing the forward noise schedule jointly with other parameters of diffusion models, one can further maximize the VLB in order to achieve higher log-likelihood values [154, 214].

The work of iDDPM [214] demonstrates that a certain cosine noise schedule can improve log-likelihoods. Specifically, the cosine noise schedule in their work takes the form of

$$\bar{\alpha}_t = \frac{h(t)}{h(0)}, \quad h(t) = \cos\left(\frac{t/T + m}{1+m} \cdot \frac{\pi}{2}\right)^2 \tag{28}$$

where $\bar{\alpha}_t$ and β_t are defined in Eqs. (2) and (3), and m is a hyperparameter to control the noise scale at $t = 0$. They also propose a parameterization of the reverse variance with an interpolation between β_t and $1 - \bar{\alpha}_t$ in the log domain.

In Variational Diffusion Models (VDMs) [154], authors propose to improve the likelihood of continuous-time diffusion models by jointly training the noise schedule and other diffusion model parameters to maximize the VLB. They parameterize the noise schedule using a monotonic neural network $\gamma_\eta(t)$, and build the forward perturbation process according to $\sigma_t^2 = \text{sigmoid}(\gamma_\eta(t))$, $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\bar{\alpha}_t \mathbf{x}_0, \sigma_t^2 \mathbf{I})$, and $\bar{\alpha}_t = \sqrt{(1-\sigma_t^2)}$. Moreover, authors prove that the VLB for data point $\mathbf{x}$ can be simplified to a form that only depends on the signal-to-noise ratio $\mathrm{R}(t) \coloneqq \frac{\bar{\alpha}_t^2}{\sigma_t^2}$. In particular, the L_{VLB} can be decomposed to

$$L_{VLB} = -\mathbb{E}_{\mathbf{x}_0} \mathrm{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \mid\mid p(\mathbf{x}_T)) + \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1} \log p(\mathbf{x}_0|\mathbf{x}_1) - L_D, \tag{29}$$