SAPIENZA
UNIVERSITÀ DI ROMA

On the Spectral Dynamics of Diffusion Models Sampling

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica

Corso di Laurea Magistrale in Computer Science

Candidate

Matteo De Sanctis

ID number 1937858

Thesis Advisor

Prof. Iacopo Masi

Co-Advisor

Dr. Maria Rosaria Briglia

Academic Year 2024/2025

Thesis defended on January 29th 2026
in front of a Board of Examiners composed by:

Prof. Tronci (chairman)
Prof. Arrigoni
Prof. Calamoneri
Prof. Cinelli
Prof. Masi
Prof. Navigli
Prof. Wollan

On the Spectral Dynamics of Diffusion ModelsSampling
Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: desanctis.1937858@studenti.uniroma.it

In deepest appreciation for the unwavering support from my family and the cherished moments shared with my friends.

Abstract

Diffusion models have emerged as state-of-the-art generative models for high-dimensional data, yet the internal structure and dynamical behavior of their sampling process remain only partially understood. This thesis investigates diffusion sampling through the lens of *spectral dynamics*, with a focus on how frequency structure, variance concentration, endpoint estimates and intrinsic dimensionality evolve along the reverse denoising trajectory.

We conduct a systematic spectral and rank-based analysis of images generated at intermediate timesteps, comparing diffusion trajectories against natural-image baselines and examining how model and sampling choices influence spectral signatures. Beyond descriptive metrics, we study the geometry of the sampling process through PCA-based analyses of trajectory subspaces, and we probe the stability of generation via controlled perturbations, adversarial conditioning, and latent-space steering along principal directions.

Our results reveal distinct dynamical regimes in the reverse process, highlight phases where generative decisions become effectively irreversible, and provide evidence of structured low-dimensional behavior emerging during sampling. Collectively, the proposed methodology and findings contribute reproducible tools and conceptual insight for understanding diffusion models as dynamical systems, informing both future model design and evaluation practice.

Contents

List of Figures

Chapter 1

Introduction

Machine learning has recently experienced rapid progress in generative modelling, whose goal, given observed samples $\boldsymbol{x}$ drawn from an (unknown) data distribution of interest, aim to learn an approximate model of the true underlying data distribution $p(\boldsymbol{x})$ so that new, realistic samples can be produced on demand.

Several families of generative models coexist in the literature. These include ***G**enerative **A**dverarial **N**etworks* (GANs), likelihood-based models as autoregressive models, normalizing flows and **V**ariational **A**uto**e**ncoder (VAEs), energy-based models, and the closely related class of score-based methods. At the current state of the art stand ***D**iffusion **M**odels* (DMs), which admit both likelihood-based and score-based interpretations, have proven particularly effective for modeling complex, high-dimensional data. **S**core-based **G**enerative **M**odels (SGMs) in particular construct samples by iteratively transforming noise using estimates of the score of progressively noised versions of the target distribution.

Diffusion models comprise a *forward* corruption process that progressively noises the data distribution reaching a simple prior (isotropic Gaussian), and a *reverse* process that undoes this corruption by simulating an **O**rdinary or **S**tochastic **D**ifferential **E**quations (ODE/SDE) whose drift depends on the score of the noised distribution. The forward dynamics are mathematically analogous to a time-inhomogeneous Brownian motion with controlled variance, and the corresponding reverse-time equations follow from Fokker–Planck / reverse-SDE arguments; in practice the score is approximated by a ***N**eural **N**etwork* (NN) via a denoising score-matching objective [15,41].

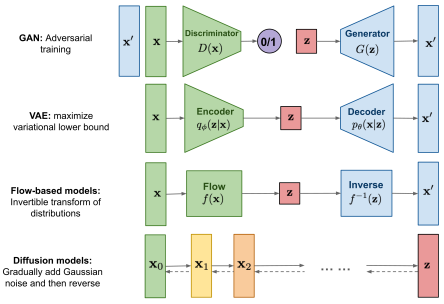


Figure 1.1. Different types of generative models.

Despite strong empirical success, theoretical understanding remains incomplete. Convergence results exist for finite-dimensional settings, but real data are very high dimensional, where sample interpolation is impeded by the curse of dimensionality and concentration phenomena. Using the exact empirical score (closed-form) on a finite training set leads to memorization; by contrast, practical SGMs estimate the score parametrically and benefit from implicit regularization (architecture and training), plus smoothing from the noise schedule and discretization, which promote generalization. A quantitative, principled explanation for how diffusion-based samplers escape memorization and generalize in high dimensions is still lacking.

Furthermore, Diffusion Models remain only partially understood as dynamical systems. While the training objective and asymptotic properties of the forward and reverse processes are well characterized, much less is known about how semantic structure emerges during sampling, how dimensionality evolves over time, and when generative decisions become effectively irreversible.

This thesis addresses these questions by studying diffusion models through the lens of spectral dynamics. By tracking how frequency content, variance concentration, and effective rank change across denoising timesteps, we aim to expose the internal organization of the reverse process and to relate recent theoretical predictions about dynamical regimes to measurable behavior in practical models.

1.1 Thesis Outline

This thesis is organized as follows. Chapter 1 introduces diffusion models within the broader landscape of generative modeling and motivates the study of their internal sampling dynamics. Chapter 2 reviews the theoretical background required for the analysis, including variational diffusion models, score-based formulations, guidance mechanisms, and latent diffusion architectures. It directly follows the current literature on the topic [35,15,37,18,22,45,41,42,38,39,40,43,14,26]. Chapter 3 presents the core contributions of the thesis, analyzing the spectral dynamics of the reverse diffusion process, introducing PCA-based diagnostics, and studying the effects of controlled perturbations and latent-space steering. Chapter 4 concludes the thesis by summarizing the main findings, discussing their implications, and outlining directions for future research.

1.2 Contributions

The main contributions of this thesis are as follows:

- A spectral and rank-based analysis of diffusion model sampling dynamics, based on intermediate clean-image predictions rather than final outputs alone.
- An empirical characterization of an effective speciation time during reverse diffusion, identified through abrupt contraction of PCA spectra across timesteps and mode selection.
- A clear distinction between intermediate speciation-related dimensional collapse and late-stage manifold-induced low-rank structure.
- Controlled perturbation experiments demonstrating how the temporal placement of noise and latent interventions determines their semantic impact.
- Latent-space steering methods based on PCA directions that enable targeted semantic injection while preserving image realism.

Chapter 2

Background

Diffusion models currently underpin state-of-the-art text-conditioned image synthesis (e.g., Imagen [29], Dall·E [25], Stable Diffusion [27]) and have been extended to other modalities such as video generation [24].

Conceptually, diffusion models admit two complementary formulations, later derived in this chapter. The variational formulation (DDPM) is derived from a variational lower bound [11], while the score-matching / continuous-time formulation characterizes the model as estimating score functions and reversing an SDE [38]. Subsequent work established that these views correspond to different discretizations and parameterizations of the same reverse-diffusion dynamics [41].

The goal of a **generative model** is, given observed samples **x** from a distribution of interest, to learn to model its true data density $p(x)$. Once learned, this approximate model can generate new sample from the distribution at will; furthermore, under some formulations, it is possible to use the learned model to evaluate the likelihood of observed or sampled data as well.

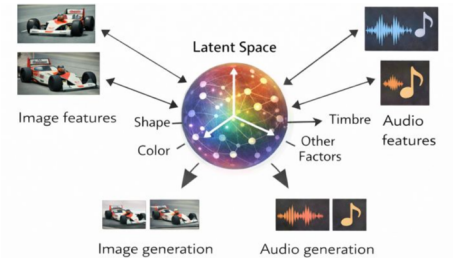


Figure 2.1. Latent features provide a compact representation of the high-level semantic manifold underlying the data.

Many data modalities admit an interpretation via latent variables: observed measurements are (noisy or partial) manifestations of lower-dimensional, higher-level factors (e.g., object shape, color, pose, acoustic timbre). Learning compact latent representations can therefore be viewed as a form of compression that exposes semantically meaningful structure in the data and facilitates generation and downstream inference.

2.1 Likelihood-based Generative Modeling

Let $p(\boldsymbol{x}, \mathbf{z})$ denote a joint distribution over observed variables $\boldsymbol{x}$ and latent variables $\mathbf{z}$. Likelihood-based generative modeling fits a parametric model by maximizing the marginal likelihood of the observed data, $p(\boldsymbol{x})$ of all observed $\boldsymbol{x}$. The marginal likelihood may be recovered by marginalizing out the latent variable:

$$p(\boldsymbol{x}) \;=\; \int p(\boldsymbol{x}, \mathbf{z})\, \mathrm{d}\mathbf{z} \;=\; \int p(\boldsymbol{x} \,|\, \mathbf{z})\, p(\mathbf{z})\, \mathrm{d}\mathbf{z}, \tag{2.1}$$

or equivalently via the chain rule of probability:

$$p(\boldsymbol{x}) \;=\; \frac{p(\boldsymbol{x}, \mathbf{z})}{p(\mathbf{z} \,|\, \boldsymbol{x})}. \tag{2.2}$$

In many practical models the integral in (2.1) (or direct evaluation of (2.2)) is intractable, which motivates approximate inference schemes such as **variational inference** and importance sampling.

2.1.1 Evidence Lower Bound

Direct maximization of the marginal likelihood $p(\boldsymbol{x})$ is typically intractable: it either involves integrating out all latent variables $\boldsymbol{z}$ in Equation 2.1, which is intractable for complex models, or it involves having access to a ground truth latent encoder $p(\boldsymbol{z} \mid \boldsymbol{x})$ in Equation 2.2, not available in closed form. **Variational inference** circumvents these difficulties by introducing a tractable, parameterized proposal (encoder) $q_{\boldsymbol{\phi}}(\mathbf{z} \mid \boldsymbol{x})$ and maximizing a surrogate objective, the ***E**vidence **L**ower **Bo**und* (ELBO):

$$\mathcal{L}(\boldsymbol{x}; \boldsymbol{\phi}) \;=\; \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z} \mid \boldsymbol{x})}\right]. \tag{2.3}$$

The ELBO lower-bounds the evidence, quantified as the log likelihood of the observed data,

$$\log p(\boldsymbol{x}) \;\geq\; \mathcal{L}(\boldsymbol{x}; \boldsymbol{\phi}), \tag{2.4}$$

and admits the exact KL decomposition

$$\log p(\boldsymbol{x}) \;=\; \mathcal{L}(\boldsymbol{x}; \boldsymbol{\phi}) \;+\; \mathrm{KL}\big(q_{\boldsymbol{\phi}}(\mathbf{z} \mid \boldsymbol{x}) \,\|\, p(\mathbf{z} \mid \boldsymbol{x})\big), \tag{2.5}$$

where $\mathrm{KL}(\cdot\|\cdot)$ denotes the Kullback–Leibler divergence. Consequently, maximizing the ELBO with respect to parameters $\boldsymbol{\phi}$ is equivalent to minimizing the posterior KL divergence between the variational encoder and the true posterior; in the ideal limit where the KL vanishes, the ELBO equals the log-evidence. See Appendix A.1 for a derivation and further discussion.

2.1.2 Variational Autoencoders

The Variational Autoencoder (VAE) framework directly maximizes the ELBO ([16,26]). The approach is *variational* because it optimizes a parametric variational posterior $q_{\boldsymbol{\phi}}(\mathbf{z} | \boldsymbol{x})$ (over a chosen family of potential posterior distributions parameterized by $\boldsymbol{\phi}$) to approximate the true posterior $p(\mathbf{z} \,|\, \boldsymbol{x})$. The VAE is called an *autoencoder* because the optimization resembles reconstructing inputs after compressing them through a bottleneck (the latent distribution). This is connection is made explicit deriving the ELBO as:

$$\begin{aligned}
&\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right] + \\
&+ \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] = \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \underbrace{p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})}_{\text{decoder}}\right]}_{\text{reconstruction term}} \underbrace{- \mathrm{KL}(\underbrace{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}_{\text{encoder}} \,||\, p(\boldsymbol{z}))}_{\text{prior matching term}}
\end{aligned} \tag{2.6}$$

the first term is the *reconstruction* term for the decoder (expected log-likelihood under the variational encoder), learned to convert a given latent vector $\boldsymbol{z}$ into an observation $\boldsymbol{x}$, ensuring that the learned distribution is modeling effective latents that the original data can be regenerated from. The second term, measuring how similar the learned variational distribution is to a prior belief held over latent variables, is a *prior-matching* regularizer that prevents the encoder (that transforms inputs into a distribution over possible latents) from collapsing to a delta distribution. Maximizing $\mathcal{L}$ therefore trades off accurate reconstruction against adherence to the prior; it is thus equivalent to maximizing its first term and minimizing its second term.

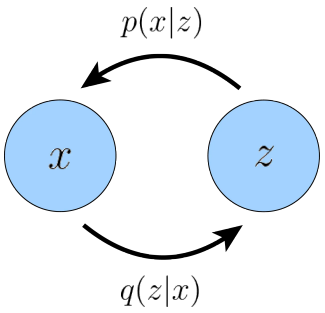


Figure 2.2. Vanilla Variational Autoencoder.

A common and practical choice is a Gaussian encoder with diagonal covariance and a standard normal prior:

$$q_{\boldsymbol{\phi}}(\mathbf{z} \mid \boldsymbol{x}) = \mathcal{N}(\mathbf{z};\, \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}),\, \mathrm{diag}(\boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\boldsymbol{x}))),$$
$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z};\, \mathbf{0}, \mathbf{I}).$$

Then, the KL divergence term of the ELBO in (2.6) admits a closed-form expression, while the expectation in the reconstruction term is approximated by Monte Carlo estimate. Optimization is performed jointly over encoder parameters $\boldsymbol{\phi}$ and decoder parameters $\boldsymbol{\theta}$, the objective can be rewritten as:

$$\begin{aligned} &\arg\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right] - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \,||\, p(\boldsymbol{z})) \\ &\approx \arg\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}^{(l)}) - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \,||\, p(\boldsymbol{z})) \end{aligned} \tag{2.7}$$

where latents $\{\boldsymbol{z}^{(l)}\}_{l=1}^{L}$ are sampled from $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ for every observation $\boldsymbol{x}$ in the dataset. The *reparameterization trick* permits low-variance gradient estimates by expressing a sample from the encoder as

$$\mathbf{z} \;=\; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}) \;+\; \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x}) \odot \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.8}$$

so that gradients propagate through $\boldsymbol{\mu}_{\boldsymbol{\phi}}$ and $\boldsymbol{\sigma}_{\boldsymbol{\phi}}$, optimzing jointly over $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ ($\odot$ representing an element-wise product).

After training, generation proceeds by sampling $\mathbf{z} \sim p(\mathbf{z})$ and decoding via $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \mathbf{z})$. VAEs are particularly useful when the latent dimension is smaller than the observation dimension: the learned latent space can provide a compact, semantically meaningful representation that supports interpolation, manipulation and controlled generation. This latent-editing capability will be employed later when steering Stable Diffusion latents in our experiments.

2.1.3 Hierarchical VAE

Variational autoencoders naturally extend to multi-level latent hierarchies, yielding ***Hi**erarchical **V**ariational **A**uto**e**ncoder* (HVAEs) [17,36]. In this setting latent variables are organized across T levels, where higher-level latents capture increasingly abstract factors of variation.

To draw a connection with diffusion-style generative chains we focus on a common and

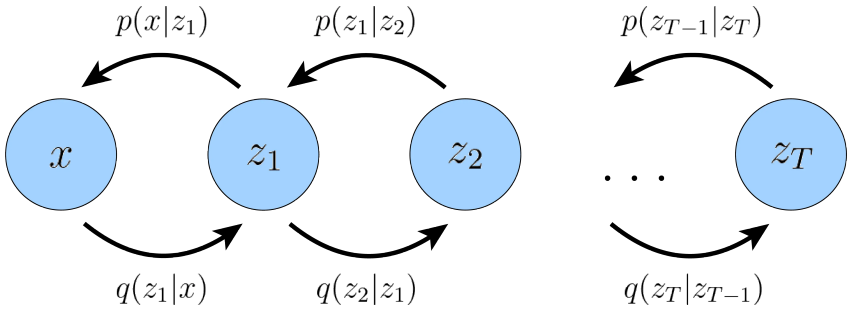


Figure 2.3. A Markovian Hierarchical VAE. The generative process is modeled as a Markov chain, where each latent is generated only from the previous latent.

convenient special case: the ***Markovian HVAE*** (MHVAE). In a MHVAE the generative model defines a top-down Markov chain: the top-level prior $p(\mathbf{z}_T)$ generates $\mathbf{z}_{T-1}$, which in turn generates $\mathbf{z}_{T-2}$, and so forth until $\mathbf{z}_1$ generates the observation $\boldsymbol{x}$. The generative joint density thus factorizes as

$$p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) = p(\boldsymbol{z}_T)p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1} \mid \boldsymbol{z}_t) \tag{2.9}$$

Inference is conventionally performed bottom-up: the variational posterior is chosen to factorize as

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x}) = q_{\boldsymbol{\phi}}(\boldsymbol{z}_1 \mid \boldsymbol{x}) \prod_{t=2}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{z}_t \mid \boldsymbol{z}_{t-1}), \tag{2.10}$$

so that the encoder maps $\boldsymbol{x}$ to $\mathbf{z}_1$ and subsequent levels are inferred conditionally upwards. The ELBO for this hierarchical model follows by the usual variational decomposition:

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})} \right] \tag{2.11}$$

See Appendix A.2 for the full derivation. Substituting (2.9) and (2.10) into Equation 2.11 yields a decomposed objective with interpretable reconstruction and conditional-regularization terms:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})} \right] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{z}_T)p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1} \mid \boldsymbol{z}_t)}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_1 \mid \boldsymbol{x}) \prod_{t=2}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{z}_t \mid \boldsymbol{z}_{t-1})} \right]$$

This objective will be useful for further decomposition into interpretable components when investigating Variational Diffusion Models. Overall, this Markovian hierarchical structure—top-down generative transitions paired with bottom-up inference—establishes a close formal analogy to diffusion-like chains and will be useful when analyzing variational diffusion models and their spectral properties.

2.1.4 Variational Diffusion Models

A **V**ariational **D**iffusion **M**odel (VDM) [35,11,18] can be interpreted as a Markovian HVAE subject to three additional structural constraints. First, the latent variables have the same dimensionality as the observed data. Second, the encoder transitions are not learned, but

are instead specified as linear Gaussian conditionals whose means depend only on the previous latent state. Third, the parameters of these Gaussian encoders are chosen (or learned under suitable parameterizations) such that the terminal latent distribution at timestep T is a standard Gaussian. The Markov property across hierarchical levels is preserved.

With a slight abuse of notation, we denote both observed data and latent variables by $\boldsymbol{x}_t$, where $t = 0$ corresponds to the data and $t \in [1, T]$ indexes the latent hierarchy. The variational posterior retains the Markov factorization of Eq. 2.10, and can be written as

$$q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) \tag{2.12}$$

In contrast to a generic Markovian HVAE, the encoder transitions are fixed to be linear Gaussian maps. Their mean and variance may either be prescribed as hyperparameters [11] or endowed with a parameterization that permits learning [18]. A common variance-preserving parameterization is

$$\boldsymbol{\mu}_t(\boldsymbol{x}_{t-1}) = \sqrt{\alpha_t}\, \boldsymbol{x}_{t-1}, \qquad \boldsymbol{\Sigma}_t = (1 - \alpha_t)\mathbf{I},$$

where the schedule α_t controls the rate at which noise is injected across the hierarchy. The forward encoder transition therefore takes the form

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1 - \alpha_t)\mathbf{I}) \tag{2.13}$$

Under the third assumption, the top-level prior is fixed as a standard Gaussian, $p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; \mathbf{0}, \mathbf{I})$, and the joint distribution of a MHVAE (Eq. 2.9) factorizes according to the Markovian top-down chain

$$p(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \tag{2.14}$$

Together, these assumptions describe a gradual *noisification* process: an observed sample $\boldsymbol{x}_0$ is progressively corrupted by Gaussian perturbations until, at timestep T, it becomes statistically indistinguishable from pure Gaussian noise. Figure 2.4 provides an illustrative visualization of this forward diffusion process.

In a VDM, the encoder conditionals $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$ are no longer parameterized by $\boldsymbol{\phi}$, since they are fixed as Gaussian transitions. Consequently, learning focuses exclusively on the reverse conditionals $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$, which define the generative process.[1] After optimization, sampling proceeds by drawing Gaussian noise $\boldsymbol{x}_T \sim p(\boldsymbol{x}_T)$ and iteratively applying the denoising transitions $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$ for T steps to obtain a novel $\boldsymbol{x}_0$.

Like any HVAE, a VDM is trained by maximizing the ELBO [35]. One derivation of an ELBO depending only on $\boldsymbol{x}_0$ is reported in App. A.3. However, that formulation is suboptimal in practice, because some consistency terms are expressed as expectations over pairs of random variables $\{\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}\}$. To avoid this, it is preferable to work with an ELBO in which every term is an expectation over at most one latent variable at a time. This can be achieved by rewriting the encoder transitions as $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0)$, where

[1] If the learned reverse transitions $p_{\boldsymbol{\theta}}$ exactly matched the true reverse conditionals, sampling would recover the empirical data distribution. In practice, model approximation error, stochastic optimization, and implicit regularization prevent exact memorization and enable generalization beyond the finite training set.

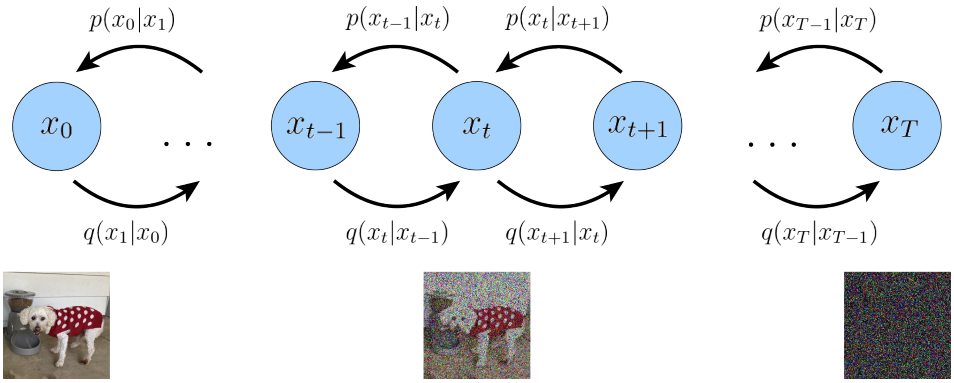


Figure 2.4. Visual representation of a Variational Diffusion Model. An input is steadily noised over time until it becomes identical to Gaussian noise; a diffusion model learns to reverse this process.

the conditioning on $\boldsymbol{x}_0$ is redundant by the Markov property, and then applying Bayes' rule:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)} \tag{2.15}$$

Under this reparameterization, the ELBO takes the form [35,11]

$$\begin{aligned}\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)}\left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\right] &= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)\right]}_{\text{reconstruction term}} - \underbrace{\mathrm{KL}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_0) \,||\, p(\boldsymbol{x}_T))}_{\text{prior matching term}} \\ &- \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathrm{KL}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}\end{aligned} \tag{2.16}$$

See the proof in App.A.4.1. The ELBO can therefore be decomposed as a sum of individual terms that are expectations of at most one random variable at a time. This decomposition has a natural interpretation. The first term is a reconstruction objective, typically estimated by Monte Carlo. The second term enforces prior consistency: under the variance-preserving forward process, $q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)$ matches the standard Gaussian prior exactly, and this term evaluates to zero as it has no trainable parameters. The final summation consists of denoising-matching terms, which train $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$ to approximate the tractable ground-truth reverse transition $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$. Since the latter specifies how to denoise $\boldsymbol{x}_t$ when the clean target $\boldsymbol{x}_0$ is known, the KL divergence is minimized when the learned reverse dynamics faithfully replicate the exact Bayes-optimal denoising step. A schematic visualization of this ELBO decomposition is provided in Fig. 2.5.

Note that in both Eq. A.3.1 and Eq. 2.16 the only structural assumption is Markovianity; therefore, the formulae apply to any arbitrary MHVAE. Moreover, setting $T = 1$ recovers the standard ELBO for a vanilla VAE, as in Eq. 2.6.

In the ELBO of Eq. 2.16, the dominant optimization cost again resides in the summation term. In a general MHVAE, each KL divergence $\mathrm{KL}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))$ is difficult to minimize because it requires simultaneously learning both the encoder and

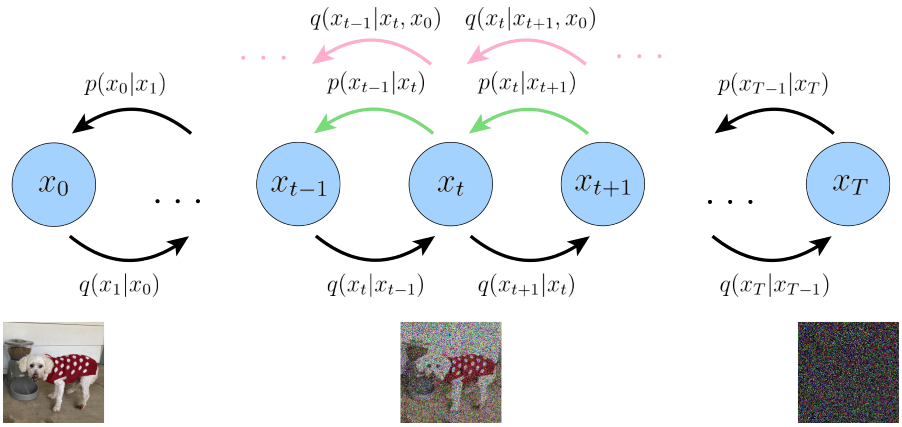


Figure 2.5. A VDM can also be optimized by learning the denoising step for each individual latent by matching it with a tractably computed ground-truth denoising step. This is denoted visually by matching the distributions represented by the green arrows with those of the pink arrows.

decoder for arbitrary posteriors. In a VDM, however, the Gaussian transition assumption renders these terms analytically tractable. By Bayes' rule,

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}$$

Since, by assumption, $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I})$ (Eq. 2.13), what remains is to obtain closed-form expressions for $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ and $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)$.

Because the encoder transitions form a linear Gaussian chain, the reparameterization trick gives

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I}) \tag{2.17}$$

and analogously for $\boldsymbol{x}_{t-1}$ from $\boldsymbol{x}_{t-2}$. The form of $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ can be recursively derived, unrolling the above transitions yields

$$\begin{aligned} \boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) &= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0 && (2.18) \\ &\sim \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I}) && (2.19) \end{aligned}$$

with derivations provided in App. A.4.2. The resulting Gaussian form also provides the parameterization of $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)$. Substituting these expressions into the Bayes expansion gives

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \propto \mathcal{N}(\boldsymbol{x}_{t-1}; \underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)}) \tag{2.20}$$

(see derivations in App.A.4.3) showing that $\boldsymbol{x}_{t-1} \sim q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$ is normally distributed at every step with mean $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ and variance $\boldsymbol{\Sigma}_q(t) = \sigma_q^2(t)\mathbf{I}$, where $\sigma_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$ (following Equation 2.20)

To match the learned reverse transition $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$ to this ground-truth denoising step $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$, we model the former as a Gaussian $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, t) = \mathcal{N}(\boldsymbol{x}_{t-1};\, \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t))$, and set $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t) = \boldsymbol{\Sigma}_q(t)$, as the noising scheduler (α terms) is fixed, leaving the network to learn only the mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$. Since $\boldsymbol{x}_0$ is unavailable at sampling time, differently than $q(\cdot|\cdot, \boldsymbol{x}_0)$, conditioning on t provides information about the current noise level.

Recalling the denoising-matching term in Eq. 2.16, $\mathrm{KL}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))$, maximizing the ELBO corresponds to minimizing this KL term.
For two Gaussians with identical covariances, optimizing the KL reduces to minimizing the difference between the means of the two distributions (see App. A.4.4):

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left[\left\| \boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q \right\|_2^2 \right] \tag{2.21}$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ and $\boldsymbol{\mu}_q = \boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$, which from our derived Equation 2.20, takes the form:

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)\boldsymbol{x}_0}{1 - \bar{\alpha}_t}, \tag{2.22}$$

so $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ can match it closely by setting it to the following form:

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1 - \bar{\alpha}_t} \tag{2.23}$$

where $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ is a neural predictor of $\boldsymbol{x}_0$. The resulting objective becomes

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1 - \alpha_t)^2}{(1 - \bar{\alpha}_t)^2} \left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2 \right] \tag{2.24}$$

(see App.A.4.5) so training a VDM reduces to learning to predict $\boldsymbol{x}_0$, the original ground truth image, from an arbitrarily noisy sample of it [11]. Finally, minimizing the full summation in Eq. 2.16 can be approximated by sampling a single timestep per iteration:

$$\begin{aligned} &\arg\min_{\boldsymbol{\theta}} \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[\mathrm{KL}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \right] \\ &= \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{t \sim U\{2,T\}} \left[\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[\frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1 - \alpha_t)^2}{(1 - \bar{\alpha}_t)^2} \left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2 \right] \right] \right] \end{aligned} \tag{2.25}$$

which is optimized via Monte Carlo sampling over datapoints and timesteps (single datapoint and single timestep per iteration).

Three Equivalent Interpretations

As shown previously, training a Variational Diffusion Model (VDM) is equivalent to learning a neural network that predicts the clean image $\boldsymbol{x}_0$ from its noisy counterpart $\boldsymbol{x}_t$ and the associated timestep t. However, $\boldsymbol{x}_0$ admits two additional equivalent parameterizations, which lead to alternative but mathematically consistent interpretations of the learning objective.

Prediction of the Source Noise. From the reparameterization expression in Eq. 2.18, we can rearrange

$$\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{\alpha}_t}} \tag{2.26}$$

Substituting this into the closed-form posterior mean $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ yields [11]

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0$$

Proof in AppA.4.6. Hence we may parameterize the model mean as

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \tag{2.27}$$

which leads to the optimization problem

$$= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} \left[\|\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\|_2^2\right] \tag{2.28}$$

In this view, the (neural network) model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ learns to predict the source noise $\boldsymbol{\epsilon}_0 \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$ that generated $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$. The formulation is equivalent, but empirically, noise–prediction often yields improved optimization stability and sample quality [11,30].

Prediction of the Score Function. A third interpretation follows from Tweedie's formula [8], stating that the true mean of an exponential family distribution, given samples drawn from it, can be estimated by the maximum likelihood estimate of the samples (empirical mean) plus some correction term involving the score of the estimate. In the case of just one observed sample, the empirical mean is just the sample itself. For a Gaussian variable $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$,

$$\mathbb{E}[\boldsymbol{\mu}_z \mid \mathbf{z}] = \mathbf{z} + \boldsymbol{\Sigma}_z \nabla_{\mathbf{z}} \log p(\mathbf{z}). \tag{2.29}$$

Applying to predict the true posterior mean of $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$ (Eq. 2.19), hence $\boldsymbol{\mu}_{x_t} = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0$, yields

$$\begin{aligned} \mathbb{E}\left[\boldsymbol{\mu}_{x_t} \mid \boldsymbol{x}_t\right] = \boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) \quad &\Rightarrow \quad \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 = \boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) \\ \Rightarrow \quad \boldsymbol{x}_0 = \frac{\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla \log p(\boldsymbol{x}_t)}{\sqrt{\bar{\alpha}_t}}. & \end{aligned} \tag{2.30}$$

Substituting this expression into $\boldsymbol{\mu}_q$ gives

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla \log p(\boldsymbol{x}_t)$$

and therefore we may parameterize the approximate denoising transition mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ as

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \tag{2.31}$$

(proof in App.A.4.7) leading to the optimization objective

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{\alpha_t} \left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \nabla \log p(\boldsymbol{x}_t)\|_2^2 \right] \tag{2.32}$$

(See proof in App.A.4.8, linking diffusion training to denoising score matching.) Here, $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ is a neural network that learns to predict the score function $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)$, which is the gradient of $\boldsymbol{x}_t$ in data space, for any arbitrary noise level t.

Explicit **Equivalence of the Three Views** in App.A.4.9.

2.2 Score-based Generative Models

Although diffusion models are often introduced through discrete-time variational formulations, an equivalent and complementary perspective emerges from score-based modeling and stochastic differential equations. In the previous section, we showed that a Variational Diffusion Model can be trained by learning a neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ that predicts the score function $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)$. This derivation followed from an application of Tweedie's formula, but the argument alone does not immediately clarify the intuition behind the score function or why it is a meaningful modeling target.
To develop this intuition, we turn to Score-Based Generative Models [38,41,39], whose formulation is mathematically equivalent to that of VDMs, allowing us to switch between the two viewpoints as convenient.

Connection to Energy-Based Models. Consider the class of energy-based models [20,40], in which a probability density is expressed as

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})} \tag{2.33}$$

where $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is an arbitrarily flexible, parameterizable function called the *energy function*, typically parameterized by a neural network, and $Z_{\boldsymbol{\theta}}$ is the normalizing constant ensuring $\int p_{\boldsymbol{\theta}}(\boldsymbol{x})\, d\boldsymbol{x} = 1$. Maximum-likelihood training of such models/distributions is generally intractable, since computing $Z_{\boldsymbol{\theta}} = \int e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})} d\boldsymbol{x}$ is expensive or impossible for high-dimensional, flexible energy functions.

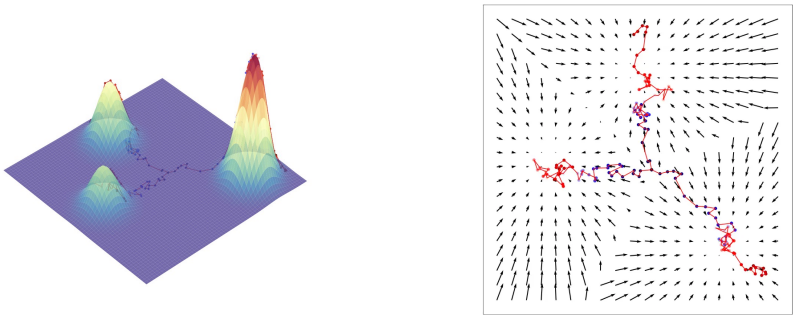
A natural way to avoid handling the normalizing constant is to model instead the *score function* $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ by using a neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x})$. This is motivated by the fact that differentiating the logarithm of (2.33) with respect to $\boldsymbol{x}$ gives

$$\begin{aligned} \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \nabla_{\boldsymbol{x}} \log\left(\frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})}\right) = \nabla_{\boldsymbol{x}} \log \frac{1}{Z_{\boldsymbol{\theta}}} + \nabla_{\boldsymbol{x}} \log e^{-f_{\boldsymbol{\theta}}(\boldsymbol{x})} \\ &= -\nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}) \end{aligned} \tag{2.34}$$

which depends only on the energy function and not on its normalizing constant. One may therefore either parameterize $f_{\boldsymbol{\theta}}$ and obtain the score via differentiation, or, more commonly, directly parameterize the score using a neural network $\mathbf{s}_{\boldsymbol{\theta}}(\boldsymbol{x})$. The learning objective becomes minimization of the Fisher divergence between the model score and the true score:

$$\mathbb{E}_{p(\boldsymbol{x})} \left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla \log p(\boldsymbol{x})\|_2^2 \right] \tag{2.35}$$

Interpretation of the Score. For any point $\boldsymbol{x}$, the gradient $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ indicates the direction in data space along which the likelihood increases most rapidly. The score function therefore defines a vector field over the data domain, whose trajectories flow toward regions of higher probability mass, i.e., toward the modes of the distribution. This geometric interpretation is illustrated by the vector field in Fig. 2.6.



(a) Sampling trajectories on a three-dimensional contour

(b) Sampling trajectories against the ground-truth score function

Figure 2.6. Visualization of three Langevin dynamics sampling trajectories for a mixture of Gaussians, all initialized at the same point. Stochastic noise enables exploration of different modes, whereas deterministic score following would converge to the same mode in every run.

By learning the score function of the true data distribution, we can generate samples by initializing at an arbitrary point in the data space and iteratively moving in the direction of increasing likelihood until a mode is reached. This procedure corresponds to Langevin dynamics, which is defined as

$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \epsilon \nabla \log p(\boldsymbol{x}_i) + \sqrt{2\epsilon}\boldsymbol{z}_i, \quad i = 0, 1, ..., K$$

where $\boldsymbol{x}_0$ is sampled from a prior distribution (e.g., uniform), and $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is an injected noise term. The noise prevents trajectories from collapsing deterministically onto a single mode, allowing samples to explore the neighborhood of high-density regions and producing diversity. It also introduces stochasticity into an otherwise deterministic score field, which is particularly important when initialization occurs in regions that lie between multiple modes. A schematic illustration of Langevin sampling and the role of the noise term is shown in the figure above.

Note that the objective in Equation 2.35 assumes access to the ground-truth score function, which is unavailable for complex distributions such as those underlying natural images. Fortunately, alternative estimators based on *score matching* [14,31,42,43] enable minimization of the Fisher divergence without explicitly knowing the true score, and can be optimized using stochastic gradient descent.

Taken together, representing a probability distribution through its score function and generating samples using Markov Chain Monte Carlo procedures such as Langevin dynamics constitutes the framework of *Score-based Generative Modeling* [38,41,39].

Limitations of Vanilla Score Matching. As discussed by *Generative Modeling by Estimating Gradients of the Data Distribution* (2020), vanilla score matching exhibits three key shortcomings. First, the score function is ill-defined when $\boldsymbol{x}$ lies on a low-dimensional manifold embedded in a high-dimensional ambient space: points off the manifold have probability zero, making $\log p(\boldsymbol{x})$ undefined. This is particularly problematic for natural

images, whose distribution is known to concentrate on a low-dimensional manifold.

Second, the learned score will be inaccurate in low-density regions. Because the objective in Equation 2.35 is an expectation over $p(\boldsymbol{x})$, the model receives little to no learning signal for rarely observed samples. This is concerning because sampling is initialized from random noise, which almost surely lies in low-density regions; the resulting generative trajectory may therefore follow a poorly estimated score and require many iterations to reach a realistic sample, or may converge to a suboptimal solution.

Finally, Langevin dynamics may fail to mix, even when computed using the exact score. Consider a mixture distribution

$$p(\boldsymbol{x}) = c_1 p_1(\boldsymbol{x}) + c_2 p_2(\boldsymbol{x}) \tag{2.36}$$

When the score is taken, the mixture coefficients vanish: the logarithm separates the coefficients from the densities, and the gradient removes them entirely. Consequently, the score field encodes only the geometry of the component densities, not their relative weights. In the illustrative example above (Fig. 2.6), Langevin sampling from the given initialization point reaches each mode with roughly equal probability, despite one mode having higher mass in the true mixture distribution.

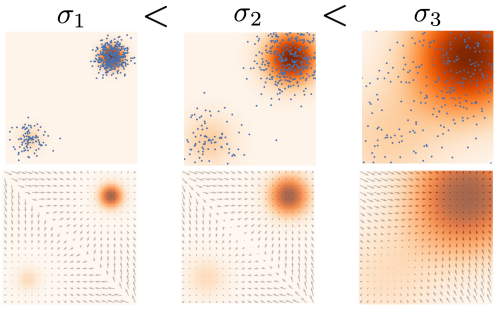


Figure 2.7. Application of multiple scales of Gaussian noise to perturb the data distribution (above), and jointly estimate the score functions for all of them (below)

These three limitations can be addressed simultaneously by perturbing the data with multiple levels of Gaussian noise (Fig. 2.7). First, because Gaussian noise has full support over the ambient space, a perturbed sample is no longer restricted to a low-dimensional manifold. Second, injecting sufficiently large noise expands the effective support of each mode, providing a stronger training signal in regions that would otherwise have very low density. Third, employing a hierarchy of noise scales induces intermediate distributions whose relative masses preserve the true mixing coefficients.

Formally, let $\{\sigma_t\}_{t=1}^T$ be a positive sequence of noise levels, and define a family of progressively corrupted data distributions

$$p_{\sigma_t}(\boldsymbol{x}_t) = \int p(\boldsymbol{x}) \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{x}, \sigma_t^2 \mathbf{I}) d\boldsymbol{x} \tag{2.37}$$

A neural network is then trained via score matching to estimate the score function at all noise levels jointly:

$$\arg\min_{\boldsymbol{\theta}} \sum_{t=1}^{T} \lambda(t) \mathbb{E}_{p_{\sigma_t}(\boldsymbol{x}_t)} \left[\| \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \nabla \log p_{\sigma_t}(\boldsymbol{x}_t) \|_2^2 \right], \tag{2.38}$$

where $\lambda(t)$ is a positive weighting function that depends on the noise level. This objective closely matches the form derived in Equation 2.32 for training a Variational Diffusion Model.

The associated generative procedure is *annealed Langevin dynamics*: samples are initialized from a fixed prior (e.g., uniform) and refined by running Langevin dynamics sequentially for $t = T, T-1, \ldots, 1$. Each stage begins from the output of the previous one, while both the noise scale and step size are gradually reduced. As the sequence progresses, trajectories concentrate around high-density regions and eventually converge to a mode. This procedure is directly analogous to the Markovian HVAE interpretation of a Variational Diffusion Model, in which a randomly initialized vector is iteratively denoised across decreasing noise levels.

We have therefore established an explicit correspondence between Variational Diffusion Models and Score-based Generative Models, both in their training objectives and in their sampling dynamics.

Continuous-time generalization. A natural question is how to extend diffusion models to an infinite number of timesteps. Under the Markovian HVAE view, this corresponds to taking the limit $T \to \infty$. The connection becomes clearer in the score-based perspective: with infinitely many noise scales, the corruption process evolves continuously over time and can be modeled as a stochastic process, which is described by a stochastic differential equation (SDE). Sampling is then performed by reversing this SDE, which requires estimating the score function at every continuous noise level [41]. Different parameterizations of the SDE correspond to different noising schedules over time, enabling flexible modeling of the diffusion process [18]. Next subsection deepens this continuous-time generalization.

2.2.1 Reverse-Time SDE and Probability Flow ODE

Hence, the discrete-time diffusion process admits a natural continuous-time limit, providing a unifying perspective in which sampling is described as the evolution of a time-dependent vector field in data space. Following *Score-Based Generative Modeling through Stochastic Differential Equations* (2021), many diffusion processes are solutions of stochastic differential equations (SDEs). In general, an SDE can be written as:

$$d\boldsymbol{x}_t = f(t)\,\boldsymbol{x}_t\,dt + g(t)\,d\mathbf{w}_t, \tag{2.39}$$

where $\mathbf{w}_t$ is a standard Wiener process, and the functions $f(t)$ and $g(t)$ define the drift and diffusion coefficients (noise schedule), respectively. In this continuous-time setting, the reverse-time dynamics of a diffusion model can be interpreted as integrating a time-dependent vector field in data space.

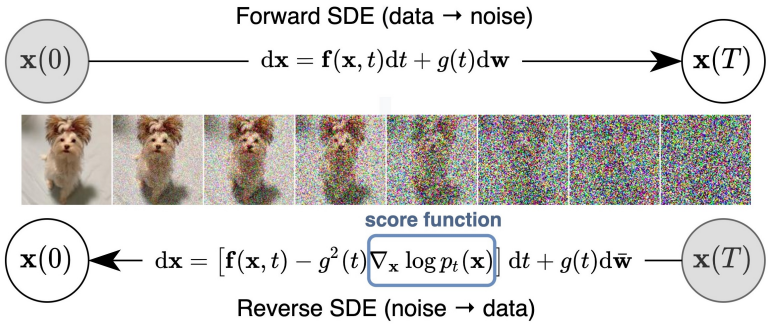


Figure 2.8. SGMs arise from solving the reverse-time SDE, which enables transforming noise into data given knowledge of the score at each intermediate time step.

With a finite number of noise levels, sample generation proceeds via annealed Langevin dynamics, i.e., sequentially sampling from each noise-perturbed distribution. In the continuous limit, this corresponds to reversing the perturbation process using a *reverse-time SDE*. For any SDE, the reverse process is also an SDE [1], with the closed form:

$$d\boldsymbol{x}_t = \Big[f(t)\,\boldsymbol{x}_t - g(t)^2 \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t) \Big]\, dt + g(t)\, d\bar{\mathbf{w}}_t, \tag{2.40}$$

where dt is a negative infinitesimal, and $\bar{\mathbf{w}}_t$ is a Wiener process under reverse-time parametrization. Replacing the unknown score function $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t)$ with a trained estimator $s_\theta(\boldsymbol{x}_t, t)$ gives the practical sampling SDE used in score-based generative models.

While SDE-based samplers can produce high-quality samples, they do not permit exact log-likelihood evaluation. By taking the expectation of the stochastic term, one obtains the associated *probability flow ODE* [41]:

$$\frac{d\boldsymbol{x}_t}{dt} = f(t)\,\boldsymbol{x}_t - \frac{1}{2} g(t)^2 \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t), \tag{2.41}$$

which deterministically transports samples from the prior to the data distribution while preserving marginal densities. Solving this ODE yields the same marginal distributions $\{p_t(\boldsymbol{x})\}_{t\in[0,T]}$ as the stochastic reverse SDE, although the trajectories are smoother (see Figure 2.9). Probability flow ODE hence admits exact likelihood computation because it defines a deterministic, invertible density flow, whose log-density evolution follows the instantaneous change-of-variables formula $\frac{d}{dt} \log p_t(x) = -\nabla \cdot f(x,t)$ (negative divergence of the vector field at the point $\boldsymbol{x}$) whereas the stochasticity of SDEs breaks pathwise invertibility and precludes exact change-of-variables. In practice a scheduler like DPM-Solver [46], a high-order numerical solver for the probability flow ODE to solve the induced reverse-time dynamics, is applied in models like Stable Diffusion for an improved sampling efficiency.

Both the stochastic reverse SDE (2.40) and the deterministic probability flow ODE (2.41) can be expressed as integrating a time-dependent vector field

$$v_t(\boldsymbol{x}) \;=\; f(t)\,\boldsymbol{x} - \gamma(t)\,\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}), \qquad \gamma(t) \in \{g(t)^2,\, \tfrac{1}{2} g(t)^2\},$$

where the factor $\gamma(t)$ depends on whether the dynamics are stochastic or deterministic. This formulation highlights that diffusion sampling—stochastic or deterministic—corresponds to integrating a learned, time-dependent vector field parameterized by the score

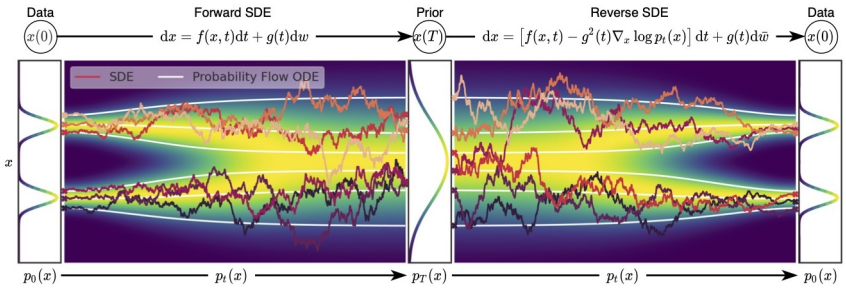


Figure 2.9. Data can be transformed into a noise distribution via an SDE, and generative modeling is performed by reversing this SDE. Alternatively, reversing the associated probability flow ODE produces a deterministic trajectory that samples from the same distribution. Both reverse-time processes are driven by the estimated score function.

function. This viewpoint is central in later spectral analyses, where we study how the frequency content and effective dimensionality of samples evolve along the reverse-time trajectory.

To conclude, in continuous time, score-based models are trained to approximate the score of the perturbed data distribution at all noise levels. Under appropriate weighting, this objective is closely related to maximum likelihood training. Once trained, the score model defines a reverse-time SDE whose numerical solution generates samples from the data distribution. In summary, sampling from diffusion models can be understood as integrating a learned reverse-time vector field in data space. For further discussions on alternative SDE parameterizations and numerical solvers, refer to [41,15].

2.3 Guidance

Thus far, we have focused on modeling the unconditional data distribution $p(\boldsymbol{x})$. In practice, we are often interested in conditional generation, i.e., modeling $p(\boldsymbol{x} \mid y)$, which allows explicit control over the generated data. Conditional diffusion models underpin applications such as image super-resolution [13] and image-text generation in DALL-E [25], Imagen [29], and Stable Diffusion [27].

A straightforward way to incorporate conditioning information y is by including it alongside the timestep t in the model inputs. The joint diffusion process of Equation 2.14 can be generalized to a conditional process:

$$p(\boldsymbol{x}_{0:T} \mid y) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y) \tag{2.42}$$

where y could represent, for example, a text embedding or a low-resolution image. The core neural network can then be trained as before, predicting $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \boldsymbol{x}_0$, $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \boldsymbol{\epsilon}_0$, or $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \nabla \log p(\boldsymbol{x}_t \mid y)$, depending on the interpretation used.

A potential limitation of this vanilla approach is that the model may ignore or underutilize the conditioning information. *Guidance* strategies are designed to explicitly control the influence of y on the generated samples, typically trading off sample diversity for adherence to the conditioning. The two most widely used methods are *Classifier Guidance* [41,7] and *Classifier-Free Guidance* [12].

Classifier Guidance. Classifier Guidance formulates conditional generation by combining an unconditional score $\nabla \log p(\boldsymbol{x}_t)$ with the gradient of a classifier predicting y from noisy inputs, i.e., $\nabla \log p(y \mid \boldsymbol{x}_t)$. During sampling, the overall conditional score is:

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \gamma \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{adversarial gradient}}, \tag{2.43}$$

where the hyperparameter γ controls the strength of the guidance: $\gamma = 0$ ignores conditioning, $\gamma > 1$ enforces stronger adherence at the cost of diversity. A drawback is the need to train a separate classifier capable of handling noisy inputs; most pretrained classifiers are not optimized for this setting, requiring the classifier to be trained specifically alongside the diffusion model.. For a full derivation see Appendix A.5.

2.3.1 Classifier-Free Guidance

Classifier-Free Guidance [12] eliminates the need for a separately trained classifier by relying solely on a conditional diffusion model and its unconditional counterpart. Notably, Stable Diffusion is trained using this approach.
Starting from Equation A.5.1, we can rewrite the classifier gradient as

$$\nabla \log p(y \mid \boldsymbol{x}_t) = \nabla \log p(\boldsymbol{x}_t \mid y) - \nabla \log p(\boldsymbol{x}_t) \tag{2.44}$$

Substituting this into Equation 2.43 yields

$$\begin{aligned} \nabla \log p(\boldsymbol{x}_t \mid y) &= \nabla \log p(\boldsymbol{x}_t) + \gamma \left(\nabla \log p(\boldsymbol{x}_t \mid y) - \nabla \log p(\boldsymbol{x}_t)\right) \\ &= \underbrace{\gamma \nabla \log p(\boldsymbol{x}_t \mid y)}_{\text{conditional score}} + \underbrace{(1-\gamma) \nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} . \end{aligned} \tag{2.45}$$

Again, γ is a term that controls how much our learned conditional model cares about the conditioning information. When $\gamma = 0$, the model ignores the conditioning and reduces to an unconditional diffusion model. When $\gamma = 1$, it reproduces the standard conditional diffusion. For $\gamma > 1$, the model emphasizes the conditional score while suppressing the unconditional score, prioritizing samples that strongly reflect the conditioning information.

Training two separate models can be computationally expensive. Classifier-Free Guidance addresses this by jointly learning a single conditional model, which can be queried as unconditional by replacing the conditioning information with a fixed value (e.g., zeros) [12,27]. This is equivalent to randomly dropping out the conditioning during training. The approach is elegant, as it allows flexible control over conditional generation without requiring a separate classifier.

2.4 Stable Diffusion: Latent-Space Diffusion Models

Denoising diffusion probabilistic models (DDPMs) in pixel space are effective but computationally costly for high-resolution image synthesis because of the large dimensionality and many timesteps. Stable Diffusion reduces this cost by performing the diffusion process in a learned, lower-dimensional latent space rather than directly on pixels [27].

2.4.1 Latent-space formulation

Let $x \in \mathbb{R}^{H \times W \times C}$ be an image and let E and D be a trained encoder and decoder satisfying

$$z = E(x), \qquad x \approx D(z), \tag{2.46}$$

where $z \in \mathbb{R}^{h \times w \times c}$ is a compressed latent representation with $h \ll H$ and $w \ll W$. The diffusion process is defined on z instead of x. A timestep-conditional denoiser $\epsilon_\theta(z_t, t, c)$ predicts the noise at step t; conditioning c typically denotes a text embedding (e.g., from a CLIP encoder). Operating in latent space preserves perceptually relevant structure while drastically lowering the dimensionality the diffusion model must process.

2.4.2 Architecture and sampling

Stable Diffusion uses a U-Net denoiser in latent space augmented with cross-attention to fuse conditioning information. The forward noising adds Gaussian noise to z_0 according

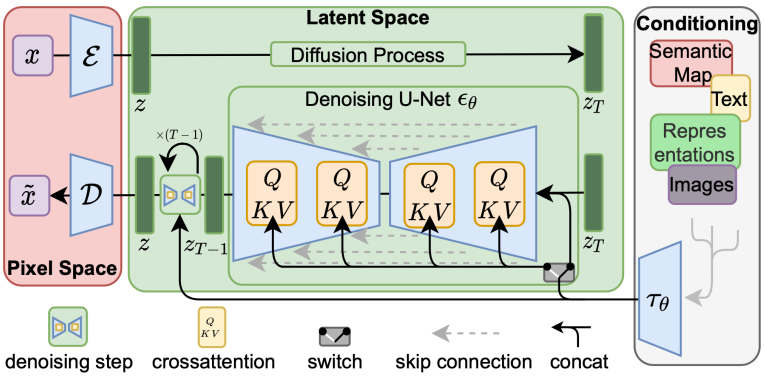


Figure 2.10. The architecture of the latent diffusion model (LDM) of *High-Resolution Image Synthesis with Latent Diffusion Models* (2022). See App. A.6 for details on U-Net and Transformer Architecture

to a schedule $\{\beta_t\}_{t=1}^{T}$. A common reverse update is

$$z_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(z_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(z_t, t, c)\right) + \sigma_t\,\eta_t, \tag{2.47}$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$, σ_t controls added stochasticity, and $\eta_t \sim \mathcal{N}(0, I)$. During sampling, classifier-free guidance is frequently used to trade off fidelity versus adherence to c [12]. See 2.10 and App. A.6

Intuition. Working in a learned latent compresses image information onto a lower-dimensional manifold that preserves semantic and mid/low-frequency content while attenuating some high-frequency details. The autoencoder thus acts as a perceptual filter: the diffusion model denoises and refines compressed structure in latent space, and the decoder reconstructs pixels from that refined latent. This combination yields high-quality, high-resolution images with far lower compute than pixel-space diffusion.

Summary. Stable Diffusion extends the DDPM framework into a learned latent representation, couples conditioning via cross-attention, and leverages classifier-free guidance to control fidelity and conditioning strength—enabling efficient, scalable text-to-image generation.

2.5 Related work

Research that links the generative dynamics of diffusion models to the multiscale structure of images and to phenomena of generalization and memorization is directly relevant to the analyses presented in this thesis. Two complementary strands of work are particularly pertinent.

First, a line of theoretical and empirical research has examined how information emerges throughout the reverse diffusion process. Several studies highlight that reverse-time sampling tends to resolve large-scale, high-variance features before finer details, suggesting an implicit coarse-to-fine generative hierarchy. For example, *Diffusion Models Generate Images Like Painters: an Analytical Theory of Outline First, Details Later* (2024) derive a closed-form description of the probability-flow ODE for Gaussian targets and demonstrate that

reverse diffusion trajectories often exhibit rotation-like dynamics that commit to coarse outlines prior to finer detail refinement; this perspective aligns closely with PCA-based measurements of intermediate model predictions and reveals low-dimensional structure in the generative trajectory (its sec 6.2 and fig. 5 are precursors of this thesis work). Parallel work from statistical physics identifies distinct dynamical regimes during backward sampling—including a phase akin to speciation, where broad structure emerges, followed by refinement and eventual collapse—formalizing the notion of dynamical phase transitions and predicting sharp changes in spectral properties at characteristic times [4,34]. Complementing these perspectives, recent analyses explore representation dynamics from a low-dimensional modeling viewpoint, showing how unimodal features and semantic structure evolve across noise scales and influence representation quality [21].

Second, a separate but related literature addresses why modern diffusion and flow-based methods tend to generalize rather than trivially memorize training samples. Work on implicit dynamical regularization and inductive biases suggests that training dynamics and model structure play a significant role in preventing memorization, beyond noise stochasticity alone. In the context of flow-matching and related objectives, replacing stochastic targets with closed-form alternatives does not qualitatively change generalization behaviour, indicating that implicit regularization dominates; complementary studies emphasize training-time windows and timescales that separate early generalization from later memorization regimes in diffusion models [2,5,32].

In addition to these theoretical strands, a growing body of work has focused on diagnostic and interpretative analyses of diffusion sampling. Several studies have examined intermediate predictions, denoising trajectories, and frequency dynamics to reveal hierarchical organization and coarse-to-fine emergence of semantic content across timesteps [44,15]. Related efforts analyse the controllability and robustness of sampling under perturbations of conditioning signals, showing that semantic decisions are often made early and subsequently refined [10]. In parallel to this prior work, which is often qualitative or task-driven, the present thesis adopts a qualitative analysis of the perturbation of reverse dynamics and a spectral and rank-based perspective to quantitatively characterize regime transitions in the reverse diffusion process, directly connecting empirical observations to recent theoretical accounts of speciation, collapse, and generalization in diffusion models. The experiments in Chapter 3 are designed to test and extend these perspectives. Concretely, PCA of the model's predicted clean image $\hat{x}_0(x_t, t)$ and its temporal evolution provide an empirical proxy for the coarse-to-fine emergence of structure. Likewise, measuring abrupt changes in spectral statistics across timesteps provides an empirical test of theoretical phase-transition and speciation predictions and situates our results within the wider discussion of memorization and dynamical regimes.

Chapter 3

Spectral Dynamics of the Reverse Regime

While diffusion models are typically evaluated through the quality and diversity of their final samples, such endpoint-based assessments obscure much of the internal structure of the generative process. The reverse diffusion trajectory—from pure noise to a structured image—encodes a rich sequence of intermediate representations that reflect how semantic information is progressively organized. Understanding these dynamics is essential for explaining both the empirical success of diffusion models and their ability to generalize in high-dimensional settings.

In this chapter, we investigate the reverse sampling process through a spectral and rank-based lens. Rather than analyzing the noisy latent variables x_t directly, we focus on the model's predicted clean-image estimates $\hat{x}_0(x_t, t)$, which provide a more interpretable view of the evolving generative state. These predictions reveal how large-scale structure, semantic commitment, and fine-grained details emerge over time, and they allow us to probe the effective dimensionality of the model's representations across denoising timesteps.

Recent theoretical work suggests that diffusion sampling is not a homogeneous process, but instead proceeds through distinct dynamical regimes. In particular, the reverse trajectory exhibits an early *speciation* phase in which global ambiguities are resolved and a coarse semantic configuration is selected, followed by refinement and eventual collapse toward individual modes (samples). Spectral diagnostics—such as the evolution of principal components and variance concentration—provide a natural empirical tool for detecting these regime transitions.

Figure 3.1. Is it an F1 car or a cruise ship? Example of attacking the generation of a diffusion model injecting two contrasting concepts: 'race car' and 'ship'.

Motivated by these insights, this chapter presents a systematic analysis of spectral dynamics during diffusion sampling. We first define all the components needed to carry out our experiments, then we characterize the evolution of frequency content and principal components across timesteps, and finally

introduce controlled perturbations to probe the stability and plasticity of the reverse dynamics. By combining PCA-based diagnostics with targeted interventions in both pixel and latent space, we aim to clarify when and how semantic structure becomes fixed during sampling, and how this timing constrains controllability, robustness, and potential acceleration of diffusion-based generation.

3.1 Spectral Dynamics of Reverse Trajectories

A naive visualization that displays samples across timesteps—from pure noise to the final output—often gives the impression that complete images simply ‘appear‘ from noise. Although visually suggestive, such animations do not fully expose the internal structure of the generative process. A more informative diagnostic is to inspect the model's *endpoint estimates* at intermediate timesteps, i.e. the predicted clean image $\hat{\boldsymbol{x}}_0$ computed from latent states along the reverse trajectory. As observed by *Diffusion Models Generate Images Like Painters: an Analytical Theory of Outline First, Details Later* (2024), these endpoint estimates reveal a consistent coarse-to-fine progression: large-scale, low-frequency structure is resolved early in the denoising schedule, while high-frequency, fine-grained details are incorporated at later timesteps. This behaviour is robust across architectures and sampler variants (cf. Fig. 6 in *Denoising Diffusion Probabilistic Models* (2020) and Fig. 1 in *Elucidating the Design Space of Diffusion-Based Generative Models* (2022)) and aligns qualitatively with human perceptual organization.

Complementary evidence comes from analyses of conditional diffusion models. For example, *Prompt-to-Prompt Image Editing with Cross Attention Control* (2022) study cross-attention dynamics and show that different semantic regions of an image exert influence at different times during sampling: global layout and object placement are typically resolved earlier, whereas local texture and fine appearance cues are refined subsequently. These findings reinforce the interpretation of diffusion sampling as a hierarchical generative procedure.

Beyond qualitative descriptions, recent work has proposed a quantitative decomposition of the reverse dynamics into distinct temporal regimes. *Dynamical regimes of diffusion models* (2024) identify three principal phases beginning from pure noise. An initial *speciation* phase is marked by the spontaneous emergence of coarse data structure, analogous to symmetry breaking in physical phase transitions. This is followed by a *consolidation* interval during which modal structure becomes more pronounced, and ultimately by a *collapse* phase in which trajectories become increasingly confined and attracted to individual data modes. These transitions admit quantitative detection: the speciation time can be located via spectral analysis of the data correlation matrix, while the collapse time can be estimated using measures related to excess entropy. Importantly, the observed dependence of collapse timing on data dimensionality and dataset size provides concrete insight into how diffusion models contend with high-dimensional data and the curse of dimensionality.

In the experiments that follow, we exploit spectral diagnostics of latent trajectories and controlled perturbations in latent space to characterize these regimes more precisely and to study their impact on generalization and mode coverage.

Motivated by these observations, in the remainder of this thesis we concentrate on the analysis of endpoint estimates of the reverse diffusion process, rather than on the raw

noisy states $\boldsymbol{x}_t$ themselves. In particular, for models parameterized to predict noise $\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t)$, we consider the corresponding clean-image estimate (see Eq. 2.26):

$$\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\, \hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t) \right), \tag{3.1}$$

and study how its spectral properties evolve as a function of the timestep. This choice enables both qualitative and quantitative investigation of the geometry of diffusion sampling trajectories, allowing us to characterize how frequency content and principal components develop throughout the denoising process.

To see why, figure 3.2 illustrates an example trajectory for the predicted noisy state $\boldsymbol{x}_t$ and the endpoint estimate $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t)$ at each timestep for a Stable Diffusion model. The noisy trajectory reveals a progressive uncovering of structure from pure white noise, whereas the endpoint estimate evolves from a coarse, blurred blob into a progressively refined and well-defined image. The impression conveyed by the $\boldsymbol{x}_t$ trajectory is that the image 'emerges' fully formed from noise, as if one were opening their eyes to reveal an image that was already present. In contrast, the endpoint-estimate trajectory displays a markedly different behaviour: already at early timesteps, recognizable object-level structure (such as the car body, tyres, and track curvature) is visible, plausibly representing a latent mean associated with the conditioning signal. Subsequent timesteps then refine this structure through the gradual addition of details in a smooth and coherent manner. This hence allows for a characterization of semantical representations along the reverse trajectory.

Relation between x_t and the endpoint estimate $\hat{x}_0$
Figure 3.4 compares the trajectories of the two quantities and visualizes their difference. A sample from the forward process in Eq. 2.17 may be written as

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\, \boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\, \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

while the corresponding endpoint estimate (see Eq. 3.1,) can be rearranged to yield

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\, \hat{\boldsymbol{x}}_0 + \sqrt{1-\bar{\alpha}_t}\, \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t).$$

We therefore visualize the residual

$$\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_t = (1 - \sqrt{\bar{\alpha}_t})\, \hat{\boldsymbol{x}}_0 \; - \; \sqrt{1-\bar{\alpha}_t}\, \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$$

which explicitly separates the contribution of the predicted endpoint from the model's predicted noise. Visualizing this residual across timesteps highlights the correction needed to transform the current sample into the

Figure 3.2. $\boldsymbol{x}_t$ and $\hat{\boldsymbol{x}}_0$ comparison across timesteps. Bottom: $t = 1000$; Top: t=0. Plot every 5 scheduler's steps (out of 50).

model's endpoint estimate. In other words, a coherent residual at the start of sampling provides direct evidence that the model has committed to a semantic hypothesis early and that subsequent steps refine that hypothesis rather than radically changing it.

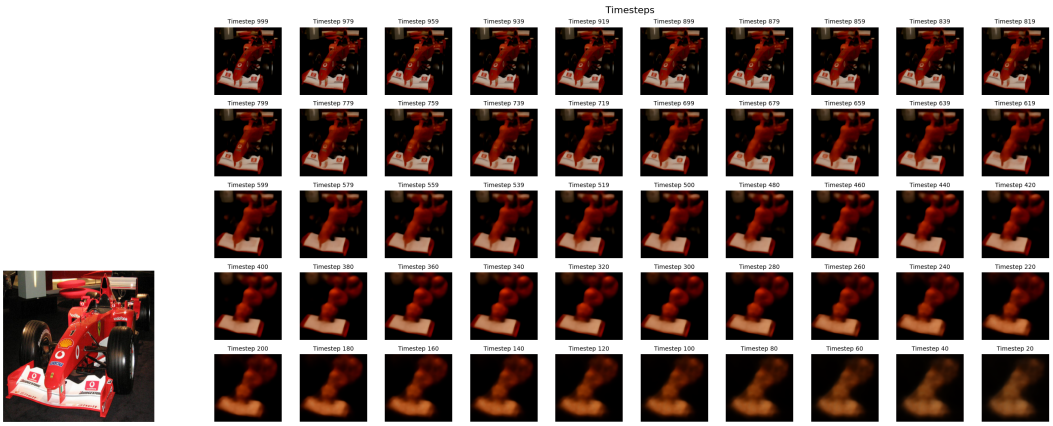
As shown in Figure 3.4, at early timesteps this residual already exhibits a coherent car-shaped structure. This indicates that the model's endpoint estimate stabilizes early in the trajectory (as expected), and that subsequent sampling steps act primarily to remove residual noise (and to converge to the estimate) rather than altering the semantic content of the generated sample: the model has greatly closed the gap with the estimate already at one fifth of the generation. The residual thus highlights the onset of *speciation*: the model commits to a semantic hypothesis at an early stage, and later timesteps serve mainly to refine and consolidate that hypothesis.

Interpretationally, the residual quantifies the part of the current sample that must change for $\boldsymbol{x}_t$ to reach the endpoint estimate $\hat{\boldsymbol{x}}_0$. It simultaneously captures (i) the noise that will be removed, and (ii) the structured signal toward which the model is steering as it converges to the final sample, thereby revealing the direction of change toward the emerging image.

3.1.1 Controlled Diffusion

To analyze the denoising behavior of diffusion models at specific noise levels, it is useful to consider first controlled inputs obtained by conditioning on a fixed clean data sample $\boldsymbol{x}_0$.
To probe the denoising behaviour while holding the underlying semantic content fixed, we adopt a *controlled diffusion* protocol: instead of constructing intermediate states through the model's usual sequential reverse sampler, we condition on a fixed clean image $\boldsymbol{x}_0$ and draw noisy observations directly from the forward marginal. Concretely,

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\,\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{3.2}$$



(a) Example ImageNet target.

(b) Controlled diffusion on the same ImageNet target. The model's endpoint estimates progressively recover the image as the noise level decreases; minor shifts in tone reflect the effect of latent compression and reconstruction in the latent diffusion pipeline.

Figure 3.3. Comparison between the original target image and controlled diffusion endpoint estimates sampled at decreasing noise levels.

(See Eq. 2.18), which corresponds to the **closed-form** expression (see Eq. 2.19)

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t \,;\, \sqrt{\bar{\alpha}_t}\,\boldsymbol{x}_0,\, (1-\bar{\alpha}_t)\mathbf{I}). \tag{3.3}$$

Because the noise schedule $\bar{\alpha}_t$ is known, samples from $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ can be drawn independently for any chosen timestep t without performing the sequential reverse-time recursion.

At each selected timestep t we then evaluate the model's *endpoint estimate* (Eq. 3.1). Importantly, the controlled protocol does *not* propagate $\boldsymbol{x}_t$ through the model's reverse transitions to construct $\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \ldots, \boldsymbol{x}_0$. Instead, for each t we (re)sample $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ and compute $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t)$. This choice yields inputs that are (i) precisely targeted to a given noise level and (ii) statistically independent across timesteps, which is desirable for diagnostics such as spectral analysis, principal component estimation, and controlled perturbations of latent structure.

For completeness, one could instead construct $\boldsymbol{x}_t$ iteratively using the reverse-time conditionals (Eq. 2.1.4)

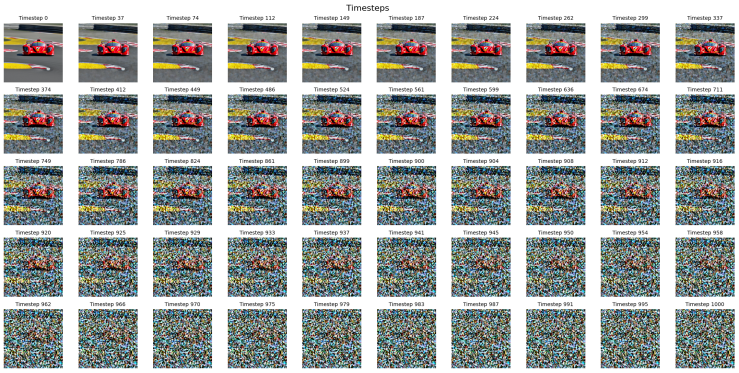
$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0)\, q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}, \tag{3.4}$$

thereby producing a trajectory of correlated noisy states. However, that sequential construction introduces extra stochasticity and temporal dependence between neighbouring timesteps while furnishing no additional information beyond the forward marginal. For targeted diagnostics we therefore prefer direct sampling from $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$.

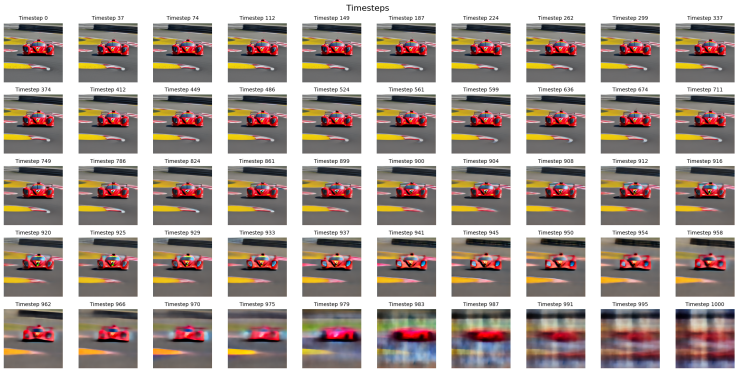
Operationally, the controlled diffusion protocol used in controlled diffusion is the following:

1. Fix a clean image $\boldsymbol{x}_0$ (for example, a training image or a held-out test example).
2. For each desired timestep t (which may be chosen sparsely or densely), draw $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ using Eq. (3.2).
3. Compute the model's endpoint estimate $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t)$ via Eq. (3.1).
4. Optionally, apply spectral diagnostics (e.g. eigendecomposition of the empirical covariance of $\hat{\boldsymbol{x}}_0$ over multiple noise realizations) or perform controlled perturbations in principal subspaces of $\boldsymbol{x}_t$ to study robustness and mode sensitivity. This is the setup for our spectral PCA analysis and latent-space steering.

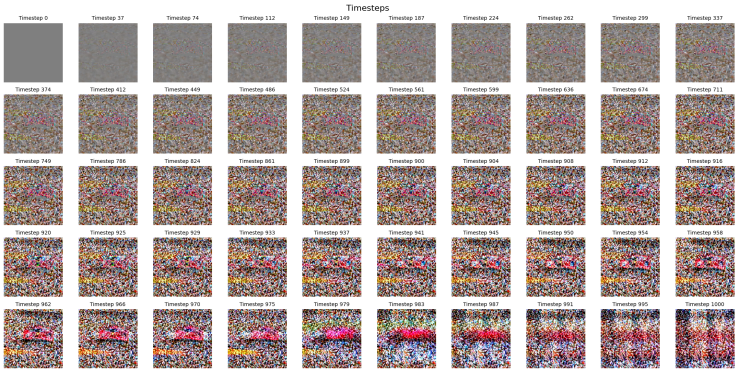
This protocol preserves the model's prediction function while removing correlations induced by iterative sampling, making it a clean and reproducible probe of how the model's internal estimate of the clean image changes with noise level. In practice, the endpoint estimates produced by this controlled procedure display the same coarse-to-fine progression described in Section 3.1: at large t (high noise) $\hat{\boldsymbol{x}}_0$ appears as a blurred, class-consistent blob that already captures coarse layout, and as t decreases the estimate becomes steadily sharper and more detailed; however, conditioning on Stable Diffusion latents induces a markedly stronger attraction of the endpoint estimate toward the geometric structure and semantic features of the training image. An example of these controlled trajectories is shown in Fig. 3.3.



(a) Trajectory of the noisy state $\boldsymbol{x}_t$. The visual impression is that the image progressively 'emerges' from noise, as if revealing a structure that was already present beneath the stochastic perturbations.



(b) Trajectory of the endpoint estimate $\hat{\boldsymbol{x}}_0$. The process begins from a coarse, class-consistent blob and rapidly converges toward a well-shaped image, with subsequent timesteps adding progressively finer details. This indicates that the model forms a meaningful estimate of the final sample already in the early stages.



(c) Difference between $\hat{\boldsymbol{x}}_0$ and $\boldsymbol{x}_t$ across timesteps. The residual highlights the structured signal toward which the model is steering the trajectory, together with the noise component that will be progressively removed.

Figure 3.4. Comparison between the trajectories of the noisy state $\boldsymbol{x}_t$, the endpoint estimate $\hat{\boldsymbol{x}}_0$, and their residual during sampling.

3.2 Experimental setup

Our experiments compare sampling dynamics across two representative samplers and model families. For the DDPM family we employ the ancestral (stochastic) reverse-diffusion sampler as introduced in *Denoising Diffusion Probabilistic Models* (2020). For the Stable Diffusion models we use the `DPMSolverMultistepScheduler` (a multistep integrator from the DPM-Solver family) [46]. These choices reflect a trade-off between (i) faithful, fully stochastic reverse-time sampling (ancestral sampler) and (ii) computationally efficient, high-quality multistep integration (DPM-Solver). For both a stochastic approach is utilized, using hence the SDE formulation (Eq. 2.40) for the reverse sampling trajectory.
Sampling schedulers implement the discrete integration scheme that converts noise into images; multistep solvers such as the DPMSolver family reuse past model evaluations to approximate higher-order integration, producing high-fidelity samples with far fewer function evaluations than naive single-step methods.
We choose the `DPMSolverMultistepScheduler` for Stable Diffusion because it delivers strong fidelity at low step counts while remaining computationally efficient, and we retain the ancestral sampler for DDPM to preserve a fully stochastic baseline.

Classifier guidance is applied to both DDPM and Stable Diffusion experiments. Because classifier guidance modifies the reverse-time vector field, it is expected to alter the spectral and principal-component dynamics of intermediate endpoint estimates: in particular, increasing guidance strength tends to concentrate probability mass and can accelerate dimensional collapse, thereby reducing sample diversity. Accordingly, in the following Stable Diffusion experiments we adopt classifier-free guidance (see Sec. 2.3.1) and vary guidance scale in the interval $[1.0, 7.5]$; in contrast, the Conditional DDPM baseline uses standard conditional sampling with vanilla guidance (see Sec. 2.3), and therefore does not expose a tunable guidance strength at inference time

For the DDPM experiments we used **CIFAR-10** [19], a labeled dataset of 60,000 32×32 color images across ten object classes, whose low resolution and constrained domain make it suitable for controlled diffusion experiments. Stable Diffusion is trained on **LAION** [33], a large-scale, web-scraped corpus of image–text pairs that provides diverse, higher-resolution imagery and enables text-conditioned generation. However, we performed experiments with **ImageNet**[6], a large-scale benchmark of over one million natural images spanning one thousand categories, commonly used to evaluate generative modeling performance at higher visual complexity.

3.3 PCA Dynamics

Analyzing the evolution of principal components of the model's endpoint estimates $\hat{\boldsymbol{x}}_0$ across denoising timesteps provides a compact, quantitative view of how diffusion models progressively organise semantic content during sampling. In our experiments we use the controlled-diffusion protocol (Section 3.1.1) with classifier guidance applied to a DDPM model.

All PCA results reported in this section are obtained from class-conditioned ensembles drawn from the CIFAR-10 training set. Concretely, we use the full set of $N = 5000$ training images belonging to the `bird` class ($y =$ `bird`; this ensures a stable estimate of the class-conditioned manifold at each time level) and apply the controlled-diffusion protocol (Section 3.1.1) with vanilla classifier guidance (sec. 2.3). The procedure used to construct the PCA at each timestep t is as follows.

Data collection For each training image $\boldsymbol{x}_0^{(i)}$, $i = 1, \ldots, N$, and for each chosen reverse timestep t we sample a noisy latent

$$\boldsymbol{x}_t^{(i)} \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0^{(i)}) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\, \boldsymbol{x}_0^{(i)},\ (1 - \bar{\alpha}_t)\mathbf{I}),$$

and compute the model's endpoint estimate

$$\hat{\boldsymbol{x}}_0^{(i)}(t) \ = \ \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t^{(i)}, t) \ = \ \frac{1}{\sqrt{\bar{\alpha}_t}}\Big(\boldsymbol{x}_t^{(i)} - \sqrt{1 - \bar{\alpha}_t}\, \hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t^{(i)}, t, y)\Big),$$

using the classifier-guided noise predictor $\hat{\boldsymbol{\epsilon}}_\theta(\cdot, t, y{=}\texttt{bird})$ at the vanilla guidance strength.

By default we draw one independent noisy sample per image per timestep (one $\boldsymbol{x}_t^{(i)}$ per $\boldsymbol{x}_0^{(i)}$); the protocol can be extended by drawing multiple noise realizations per image if required for variance estimation.

Forming the data matrix Each estimate $\hat{\boldsymbol{x}}_0^{(i)}(t)$ is vectorised into $\mathbb{R}^D$ (for CIFAR-10, $D = 3 \times 32 \times 32 = 3072$). At timestep t we assemble the data matrix

$$X_t \in \mathbb{R}^{N \times D}, \qquad X_t[i, :] = \operatorname{vec}\left(\hat{\boldsymbol{x}}_0^{(i)}(t)\right)^\top.$$

We mean-center the rows of X_t:

$$\bar{\boldsymbol{x}}_t = \frac{1}{N}\sum_{i=1}^{N} X_t[i, :]^\top, \qquad \widetilde{X}_t = X_t - \mathbf{1}\bar{\boldsymbol{x}}_t^\top,$$

where $\mathbf{1} \in \mathbb{R}^N$ is the all-ones vector.

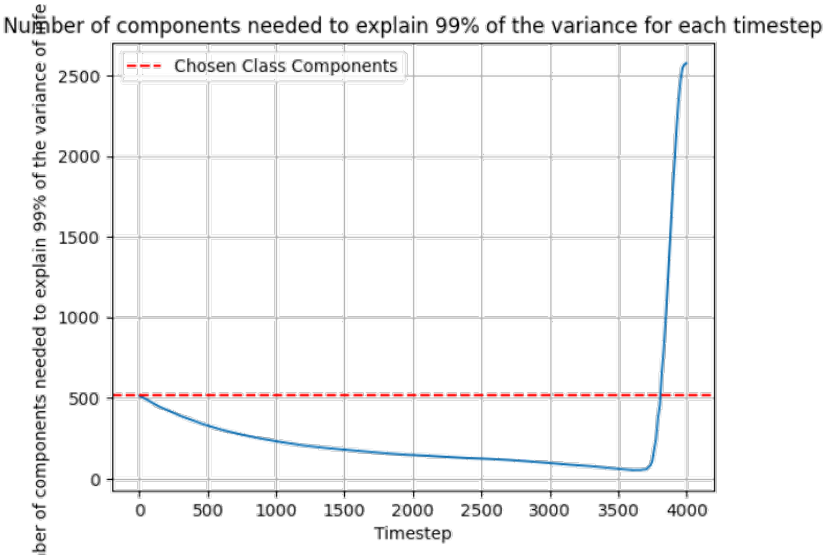


Figure 3.5. Principal Components explaining 99% of the variance for the endpoint estimate $\hat{x}_0$ across reverse diffusion. DDPM, Cifar10, Conditioned on class 'bird'.

PCA computation We compute the empirical covariance

$$\Sigma_t = \frac{1}{N-1} \widetilde{X}_t^\top \widetilde{X}_t \in \mathbb{R}^{D\times D},$$

and obtain its eigendecomposition

$$\Sigma_t = V_t \Lambda_t V_t^\top, \qquad \Lambda_t = \mathrm{diag}(\lambda_1(t), \lambda_2(t), \ldots),$$

with eigenvalues ordered $\lambda_1(t) \geq \lambda_2(t) \geq \cdots \geq 0$. Equivalently, for numerical stability we perform the SVD of $\widetilde{X}_t$ and recover the principal directions and eigenvalues from the singular vectors and singular values.

Explained-variance threshold Define the explained-variance ratio for the top k components at time t

$$R_k(t) = \frac{\sum_{j=1}^{k} \lambda_j(t)}{\sum_{j=1}^{D} \lambda_j(t)}.$$

We report the minimal integer

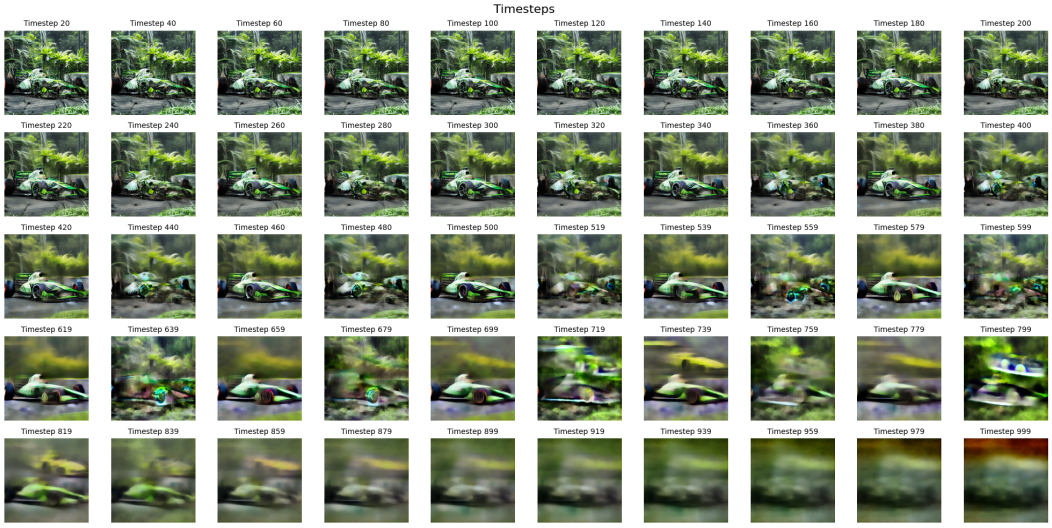
$$k_{0.99}(t) = \min\{k : R_k(t) \geq 0.99\},$$

i.e. the number of principal components required to explain 99% of the variance at timestep t. This represents a sort of compression of data while retaining most of the info and data structure. Plotting $k_{0.99}(t)$ as a function of t yields the curves shown in Fig. 3.5. This protocol yields a time-resolved characterization of the effective dimensionality of the class-conditioned endpoint estimates.

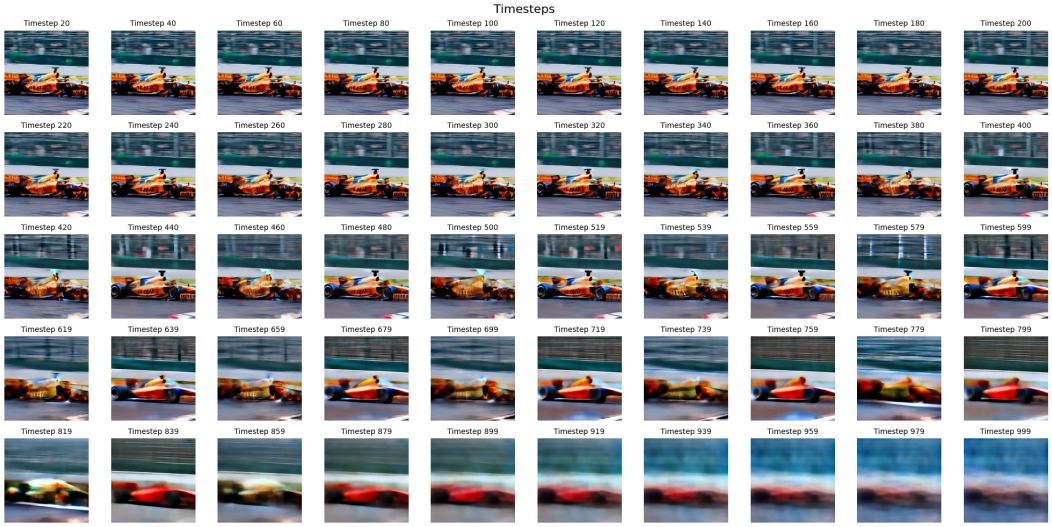
Commentary and implications The abrupt contraction of the PCA spectrum indicates that variance in the model's endpoint estimates becomes progressively confined to a low-dimensional subspace: representations transition from diffuse, high-variance patterns to a compact principal subspace that encodes the dominant semantic degrees of freedom. This behaviour is naturally interpreted as an early-stage *commitment* by the model to a particular semantic configuration of the conditioned class — consistent with the *speciation* phenomenon described in *Dynamical regimes of diffusion models* (2024) — where global ambiguities are resolved and sampling trajectories concentrate toward a class mode.

It is important, however, to distinguish this intermediate spectral contraction from the later *collapse* regime. A small number of principal components at very late timesteps mostly reflects that samples lie close to the data manifold and therefore share manifold-constrained structure; it does not by itself prove ongoing dynamical contraction or memorization. Consequently, PCA is most informative when evaluated as a time series across the full denoising trajectory: intermediate timesteps reveal structural transitions, while late timesteps reflect manifold geometry and final reconstruction fidelity.

Speciation time and acceleration hypothesis The spectral contraction naturally identifies an effective *speciation time* t_s, which is apparent as the abrupt reduction shown in Fig. 3.5. If t_s corresponds to the moment when class-specific semantic structure is established, a practical hypothesis follows: conditioned sampling may be accelerated by initializing the reverse dynamics near t_s — effectively sampling from points on the class-conditioned manifold at that noise level — instead of beginning at the maximal-noise time $t_{\max}$. This hypothesis will be investigated further in future work.



(a) Prompt: 'a photo of a f1 car'; **Adversarial prompt**: 'photo of a jungle'. Both conditionings bias sampling toward a hybrid image that combines features of an F1 car and a jungle. Interestingly, the two semantic concepts are compatible (a F1 car can plausibly appear within a natural scene), the model produces a coherent blend of both classes.



(b) Prompt: 'a photo of a f1 car'; **Adversarial prompt**: 'photo of an ocean'. Both conditionings influence early denoising, but the concepts are semantically incompatible (an F1 car cannot occupy an ocean). Around timestep 800 a clear "speciation" occurs: the trajectory collapses toward the car class while retaining residual ocean-like color/texture; prior to speciation both class signals are simultaneously present and visually overlapping.

Figure 3.6. Injecting adversarial conditioning into the Stable Diffusion sampling trajectory. Shown: estimated $\hat{\boldsymbol{x}}_0$ at each timestep. Adversarial weight $= 0.5$. See final generations in Fig. B.1

3.4 Perturbing the Sampling Trajectory

A complementary approach to studying reverse diffusion dynamics, and to probing their robustness, is to introduce controlled perturbations during sampling. By modifying either the predicted noise or the latent representations at intermediate timesteps, one can deliberately steer the reverse trajectory away from the path it would otherwise follow, thereby assessing how semantic structure is preserved, altered, or destabilized under intervention. In this setting, the diffusion model is treated as a white-box system: we assume full access to the sampling pipeline and direct control over the intermediate variables manipulated during generation.

3.4.1 Adversarial Noise Injection

A first class of interventions operates directly in the noise-prediction space. For diffusion models parameterized to predict noise, this corresponds to perturbing the model output $\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t)$ during the reverse update. Conceptually, such perturbations inject alternative denoising directions into the reverse dynamics. In extreme cases, the injected signal may resemble the noise prediction associated with a different conditioning signal or semantic class, thereby encouraging the trajectory to interpolate between—or compete across—multiple semantic hypotheses.
Although this procedure does not constitute an adversarial attack in the classical security sense, it provides a powerful diagnostic tool for studying the stability, sensitivity, and semantic commitments of the reverse diffusion process. In particular, it allows us to probe how strongly the model adheres to a given conditioning once speciation has occurred, and how early semantic conflicts are resolved.

For Stable Diffusion operating in latent space (ImageNet-scale semantics), an example of such an intervention is shown in Fig. 3.6. Two qualitatively distinct behaviours emerge when adversarial conditioning is blended into the sampling trajectory. When the injected concept is semantically compatible with the target (e.g. *F1 car + jungle*), the denoising dynamics converge to a coherent hybrid: low- and mid-level features from both concepts are integrated, and the final image (see Fig. 3.6a) remains semantically plausible. By contrast, when the concepts are incompatible (e.g. *F1 car + ocean*), the dynamics exhibit competitive mode selection (see Fig. 3.6b). During early timesteps both semantic signals coexist, but around a critical timestep ($t \approx 800$) the trajectory undergoes *speciation*: sampling collapses toward the car mode while retaining residual colour or texture cues from the ocean.

These observations suggest three conclusions. First, semantic compatibility determines whether conditioning signals combine additively or compete. Second, consistently with prior analyses of diffusion dynamics [4,34], timing is critical: early perturbations influence global layout and object identity, whereas later perturbations primarily affect texture and appearance. Third, some low-frequency or colour-level features may survive speciation even after semantic collapse, representing the shared semantic representations that are able to persist. Together, these findings reinforce the interpretation of diffusion sampling as an outline-first, detail-later process [44,34,15] and suggest practical control strategies based on time-dependent perturbation schedules.

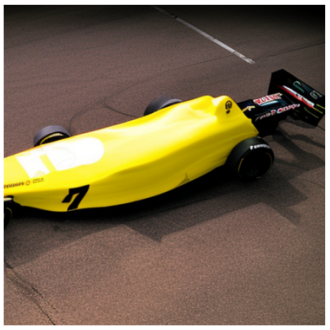
3.4.2 Attacks across Dynamical Regimes

To analyse the speciation phenomenon more precisely, we perform adversarial noise injection over restricted temporal windows, perturbing the predicted noise only for timesteps

$$t \in [T_{\text{lower}}, T_{\text{upper}}],$$

(a) No noise injection (baseline). Standard conditioned generation without intervention.

(b) Noise injection for $t \in [800, 1000]$. The model fully recovers the F1 car class. Despite the late perturbation, no banana-specific semantic features are retained, indicating that post-speciation interventions are largely corrected by denoising.



(c) Noise injection for $t \in [600, 1000]$. Partial degradation of car structure is observed. The model fails to reconstruct a coherent F1 car, suggesting that speciation occurs within this interval and that class commitment is no longer fully reversible.

(d) Noise injection for $t \in [400, 1000]$. Car-specific semantics are almost entirely lost. Only elongated metallic structures remain, visually resembling the banana stem rather than automotive components.

(e) Noise injection for $t \in [200, 1000]$. The adversarial class dominates the generation. The output collapses to a clear banana, with no residual car features.

(f) Noise injection for $t \in [0, 1000]$. The entire trajectory is steered, effectively reproducing unconditional generation under the adversarial conditioning alone.

Figure 3.7. Final decoded images obtained by adversarially perturbing the reverse diffusion trajectory with fixed adversarial weight $= 1.0$ over different temporal intervals. As the injection window extends into earlier timesteps, the trajectory undergoes a speciation transition and ultimately collapses to the adversarial class. Corresponding reverse trajectories are shown in Fig. B.1.

with both bounds treated as hyperparameters. This allows us to isolate how noise injections affect the reverse dynamics at different stages of sampling.

Following *Dynamical regimes of diffusion models* (2024), we distinguish two critical timescales. The *speciation time* t_S marks the transition from a high-entropy noise bundle to commitment toward a semantic class. The *collapse time* t_C denotes the later stage at which trajectories become attracted to individual training samples.

Identification of regime phases

Regime I: pre-speciation At the beginning of the reverse process, trajectories have not yet committed to a class and form a single high-dimensional bundle. As discussed in *Generative diffusion in very large dimensions* (2023), accurately learning the score is crucial to reproduce correct class proportions. In this regime, regularization that degrades score accuracy may delay or distort speciation, yielding realistic-looking samples with incorrect class weights. Conversely, regularization is beneficial at later stages to prevent collapse. Using adversarial conditioning with competing semantic signals provides an empirical means to estimate t_S: in our Stable Diffusion experiments on ImageNet, class commitment occurs at approximately $t_S \approx 800$ (see Fig. 3.6). We therefore define *Regime I* as timesteps $t \in [t_S, 1000]$.

Regime II: post-speciation, pre-collapse In *Regime II*, the trajectory has committed to a semantic class, but fine-grained features are still being constructed. Attacks in this regime are expected to preserve class identity while allowing adversarial attributes to influence texture, colour, or local structure. Recall that guidance tends to concentrate probability mass and can accelerate dimensional collapse, affecting the dynamics and distorting speciation time. Whether a full class switch is still feasible in this interval is a central empirical question addressed below.

Regime III: collapse *Regime III* is characterized by collapse: trajectories become attracted to the basin of individual training samples. Unless the dataset size is exponentially large in dimension, collapse is unavoidable at sufficiently late times [4,5]. The collapse time t_C is governed by the entropy of the noised data distribution. In this work, we do not analyse collapse in detail, as our focus is on early-time generalization and semantic commitment.

Interval experiments We perform a series of attacks in which adversarial noise is injected only within a chosen interval $t \in [T_{\text{lower}}, T_{\text{upper}}]$. Figure 3.7 shows final decoded images for different choices of T_{lower}, with $T_{\text{upper}} = 1000$. Images correspond to: no injection, $[800, 1000]$, $[600, 1000]$, $[400, 1000]$, $[200, 1000]$, and $[0, 1000]$.

When the injection is confined to late timesteps ($T_{\text{lower}} = 800$), the model reliably recovers the original class: adversarial noise introduced before speciation is largely corrected. As the perturbation window extends earlier, a qualitative transition occurs. For $T_{\text{lower}} = 600$, images exhibit partial degradation and hybrid features, indicating interference near speciation (the interval may be overlapping the speciation time t_S). For $T_{\text{lower}} = 400$, class structure is largely lost. For $T_{\text{lower}} = 200$ and below, the generated image is dominated by the adversarial class. Finally, perturbing across the whole trajectory $T_{\text{lower}} = 0$ produces an image that is essentially a sample of the adversarial conditioning alone.

These observations place the speciation time t_S near the boundary between the behaviours observed for $T_{\text{lower}} = 800$ and $T_{\text{lower}} = 600$. Perturbations applied strictly before t_S are ineffective at changing class identity, whereas perturbations that overlap the speciation regime can redirect the trajectory toward a different semantic attractor. These findings motivate a finer spectral/time-resolved analysis of the score and covariance evolution around t_S to understand which latent directions are most responsible for class commitment.

3.4.3 Latent-space perturbations

For latent diffusion models such as Stable Diffusion, an additional and often more interpretable intervention mechanism is available. Since sampling occurs in a learned latent space encoding high-level semantics, perturbations can be applied directly to latent variables rather than to pixel-space noise predictions. Ideally, latent perturbations allow steering along directions more closely aligned with semantic attributes while remaining compatible with the model's learned geometry. In the following section, we explore this intervention strategy in detail, leveraging principal directions in latent space to guide generation toward alternative semantic configurations.

3.4.4 Latent-Space Steering via PCA Directions

Latent representations provide a compact and semantically structured encoding of data, which diffusion models such as Stable Diffusion [27] exploit to enable efficient generation. In Stable Diffusion, images are first compressed into a latent space using a variational autoencoder (VAE). The diffusion process is then applied to these latent codes, and the final image is obtained by decoding the latent trajectory back into pixel space through the VAE decoder. Operating in latent space substantially reduces dimensionality while preserving high-level semantic information, leading to faster sampling and improved scalability.

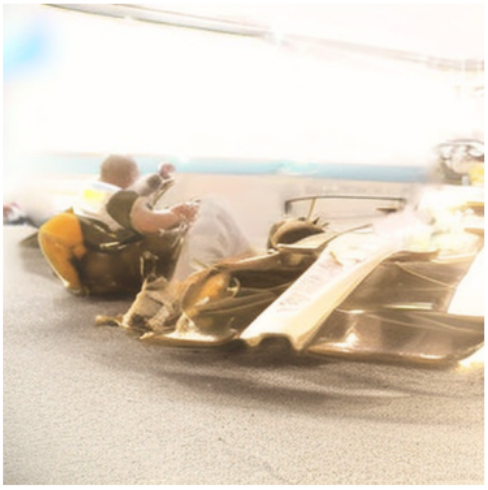
Directly injecting clean latent codes (e.g. principal components of dataset latents) corresponding to a different class into the denoising process is not straightforward. Such latents are defined at $t = 0$, and inserting them at intermediate timesteps would be inconsistent with the distribution of noised latents expected by the model, typically resulting in degenerate or unstable generations. A more principled direction is to construct datasets of noised latents for every timestep.
By performing principal component analysis (PCA) on VAE-encoded latents from a specific class, one can identify directions of maximal variance, which often correspond to semantically meaningful factors of variation. Similar analyses have been used to probe the local geometry of the estimated clean-image manifold $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t)$ [44], however, not in an adversarial manner. Steering the latent trajectory along such directions enables controlled manipulation of the generative process while remaining close to the model's learned distribution.

Concretely, a small perturbation vector $\boldsymbol{\Delta}$ is constructed by projecting the principal subspace of a source class onto that of a target class. Adding $\boldsymbol{\Delta}$ to the latent variable at intermediate timesteps introduces a controlled bias toward the target semantic manifold, while minimally disrupting the overall latent statistics. The magnitude of $\boldsymbol{\Delta}$ must be carefully tuned: it must be large enough to induce a semantic shift, yet sufficiently small to avoid corrupting the final decoded image.

PCA and orthogonal projection. Given a dataset of n mean-centered latent vectors $\{x^{(i)}\}_{i=1}^{n}$, stacked row-wise into $X \in \mathbb{R}^{n \times D}$, PCA computes the eigendecomposition of

(a) Standard Stable Diffusion (baseline). Clean generation with no latent-space intervention. No banana-related features are present.

(b) Weighted-sum PCA projection. Steering strength $s^* = 2.5$. Distinct banana-like attributes emerge, including elongated yellow front suspensions and a banana-shaped tyre warmer.



(c) SVD principal-direction projection. Steering strength $s^* = 200.0$. The image exhibits a global yellowish tint but no clearly identifiable banana geometry, consistent with perturbing a single dominant latent direction. An emergent human-like figure appears in the late stages of generation (see Fig. B.10).

(d) Shared–residual disentangling. Steering strength $s^* = 7.0$. Banana-related features are injected more coherently, appearing on the car nose and near the tyre, while overall image realism and structural integrity are preserved.

Figure 3.8. Final images obtained under different latent-space steering strategies. The prompt is *'photo of a white F1 race car'*. Steering vectors are computed from PCA of ImageNet classes *racer* (source) and *banana* (target), and applied over the full interval $\mathcal{I} = [0, 1000]$. Complete latent trajectories are shown in Fig. B.2.

the empirical covariance

$$\Sigma = \frac{1}{n-1} X^\top X = V \Lambda V^\top,$$

where $V = [v_1, \ldots, v_D] \in \mathbb{R}^{D \times D}$ is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_D)$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$. The vectors v_i are the principal directions and λ_i the corresponding explained variances, with standard deviations $\sigma_i = \sqrt{\lambda_i}$.

Collecting the first k principal directions into $V_k \in \mathbb{R}^{D \times k}$, the orthogonal projection of a vector $x \in \mathbb{R}^D$ onto this subspace is

$$P_{V_k}(x) = V_k V_k^\top x.$$

Latent subspace attack. Let $A \in \mathbb{R}^{k_A \times D}$ contain the top k_A principal directions of a *source* class (e.g. *race car/racer*), and let $B \in \mathbb{R}^{k_B \times D}$ denote those of a *target* class (e.g. *banana*). Let $V_B \in \mathbb{R}^{D \times k_B}$ be an orthonormal basis spanning the row space of B. The projection of a source component a_i onto the target subspace is

$$\mathrm{proj}_i = P_{V_B}(a_i) = V_B V_B^\top a_i.$$

The goal is to construct a perturbation vector $\Delta \in \mathbb{R}^D$, reshaped to the latent tensor shape $(e.g. [4, 64, 64])$, which is added to the latent variable z_t during sampling. We explore three construction strategies.

(1) Weighted-sum projection. Define projection norms $w_i = \|\mathrm{proj}_i\|_2$ and source variances $\sigma_i = \sqrt{\lambda_i}$. The aggregate direction is

$$\Delta_{\mathrm{raw}} = \sum_{i=1}^{k_A} w_i \, \mathrm{proj}_i,$$

with expected scale

$$\mathrm{std}_{\mathrm{exp}} = \sqrt{\sum_{i=1}^{k_A} (w_i \sigma_i)^2}.$$

The final perturbation is

$$\Delta = k \, \mathrm{std}_{\mathrm{exp}} \, \frac{\Delta_{\mathrm{raw}}}{\|\Delta_{\mathrm{raw}}\|_2 + \varepsilon}.$$

This construction emphasizes directions that are both strongly aligned with the target subspace and highly variant within the source class.

Results are shown in Fig. 3.8b and Fig. B.9. Clear banana-like features emerge, such as yellow bar-like (front suspension) structures on the car nose and banana-shaped elements (tyre warmer) near the tyres. Notably, these features appear only after $t \approx 800$, consistent with the model entering *Regime II*, where semantic commitment is fixed and details are refined. While effective, this method introduces the strongest latent corruption among the tested approaches.

(2) SVD / principal-direction projection. Stack the projected vectors $\{\mathrm{proj}_i\}$ into a matrix $\mathrm{Proj} \in \mathbb{R}^{k_A \times D}$ and compute its singular value decomposition,

$$\mathrm{Proj} = USV^\top.$$

The leading right-singular vector v_1 defines the dominant shared direction, with singular value s_1 measuring its strength. We consider

$$\Delta = k \frac{v_1}{\|v_1\|_2},$$

which applies a focused, directed and normalized low-rank perturbation. (Note we could also have considered $\Delta = k \cdot s_1 \cdot v_1$, a magnitude-informed variant.)

Results are shown in Fig. 3.8c and Fig. B.10. No clear banana structure emerges, although a faint yellow halo suggests that the first principal direction captures limited colour-related semantics. A human-like figure appears late in the trajectory, possibly reflecting interpolation artifacts between class manifolds. This method requires a much larger steering magnitude (e.g. 200) and produces the least image corruption, consistent with its low-rank nature.

(3) Shared–residual disentangling. Each source component is decomposed as

$$a_i = P_{V_B}(a_i) + r_i,$$

where $P_{V_B}(a_i)$ captures the shared component and r_i the residual orthogonal to the target subspace. Collect either the shared parts or the residuals into a matrix and perform an SVD to obtain orthonormal directions $\{v_j\}$ and strengths $\{s_j\}$. Using the top m directions (hyperparameter) with normalized weights α_j (for example $\alpha_j \propto s_j$ and rescaled) yields

$$\Delta = \sum_{j=1}^{m} \alpha_j v_j.$$

Selecting the shared set yields attraction toward the common manifold between classes; selecting residuals yields pushes along source-unique directions (orthogonal to the target). We operate on the shared set to generate towards plausible semantic representations.

Results are shown in Fig. 3.8d and Fig. B.11. Banana features emerge similarly to the weighted-sum method, but with significantly improved image quality and realism. This confirms the benefit of isolating shared semantic components: steering along these directions injects plausible attributes without destabilizing the latent manifold. As before, feature emergence occurs around $t \approx 800$, reinforcing the connection to speciation.

Steering normalization implemented. To ensure comparability across steering intervals, the per-step steering magnitude is normalized such that the total injected strength remains constant.

Let $T_{\text{cont}} = 1000$ denote the continuous diffusion horizon, let $N = |\text{timesteps}|$ be the number of discrete scheduler timesteps (e.g. $N = 50$ in Stable Diffusion), and define the scheduler step size $\Delta_{\text{sched}} = T_{\text{cont}}/N$. Let the active steering interval be $\mathcal{I} = [b_0, b_1]$

and set the interval length in continuous indices $L = b_1 - b_0$. The number of discrete scheduler steps that fall inside $\mathcal{I}$ is computed as

$$N_{\text{interval}} = \left\lfloor \frac{L}{\Delta_{\text{sched}}} \right\rfloor,$$

The per-step update applied when $t \in \mathcal{I}$ is

$$z_t \leftarrow z_t + s\,\Delta\,\sqrt{\alpha_{\text{cum}}(t)},$$

where Δ is the reshaped 'steer_delta' and $s =$ 'steer_factor' is:

$$s = \frac{s^*}{N_{\text{interval}}},$$

with s^* an appropriate hyperparameter representing fixed steering we would like to apply at each timestep, selected to not produce out of distribution latents, (e.g. $s^* = 2.0$). Consequently the discrete sum of per-step scalars over the active interval equals the invariant value

$$\sum_{t\in\mathcal{I}} s \;=\; s \cdot\; N_{interval} = s^*,$$

This formulation makes the effective steering strength independent of the chosen interval bounds: the per–step scalar is rescaled so that the total steering weight over $\mathcal{I}$ remains constant. At the same time, each update is still multiplied by $\sqrt{\alpha_{\text{cum}}(t)}$, so the perturbation at step t is aligned with the scheduler's variance schedule; consequently, the injected offsets respect the latent noise scale expected by the model at each timestep, preventing the perturbation from pushing the trajectory outside the distribution encountered during standard sampling.

Final remarks on the results While latent-space steering yields informative qualitative results, it is less stable and less effective than direct adversarial noise injection. Noise-space interventions better exploit the inductive biases of diffusion models and their non-linear score dynamics, enabling clearer regime separation and speciation analysis. Nonetheless, latent-space perturbations provide valuable insight into how semantic structure is encoded in learned representations and reveal interpretable directions for semantic interpolation. These findings motivate future work on principled latent steering mechanisms that preserve distributional consistency while enabling controlled semantic manipulation.

3.5 Limitations

The analyses presented in this thesis are primarily empirical and diagnostic in nature, and several limitations should be acknowledged.

First, spectral and PCA-based analyses capture only linear structure in the model's intermediate representations. While abrupt changes in effective rank and variance concentration provide strong empirical signals of regime transitions, they do not fully characterize nonlinear manifold geometry. Consequently, conclusions about semantic commitment and speciation should be interpreted as descriptive rather than exhaustive.

Second, experiments are conducted on pretrained models and fixed architectures (DDPM

and Stable Diffusion). While this allows controlled comparisons across sampling strategies and perturbations, it limits the ability to attribute observed phenomena to specific architectural or training choices. Extending the analysis to models trained under different objectives or with varying inductive biases remains an open direction.

Third, the proposed notion of a speciation time is inferred indirectly from spectral contraction and perturbation sensitivity. Although consistent with recent theoretical work, it is not established as a formally defined quantity with universal invariance across datasets or model classes. Furthermore, guidance strength tends to concentrate probability mass and can accelerate dimensional collapse, this directly affects the reverse trajectory and distorts the speciation time, allowing for stronger semantic reconstructions when signal has been adversarially tampered. Further work is required to assess the speciation time robustness and practical exploitability for accelerated sampling.

Finally, the perturbation and steering experiments are designed as analytical probes rather than as optimized control mechanisms. While they demonstrate that semantic features can be injected or suppressed at specific stages of the reverse trajectory, they do not guarantee optimality or stability for downstream applications.

Chapter 4

Conclusions

Via a spectral and rank-based perspective, this thesis investigated the reverse sampling dynamics of diffusion models, and by the analysis of intermediate clean-image predictions and their evolution across denoising timesteps, we provided empirical evidence that diffusion sampling proceeds through distinct dynamical regimes characterized by changes in effective dimensionality and semantic commitment.

Principal component analyses revealed an abrupt contraction in the variance spectrum at early noise levels, indicating a transition from diffuse, high-dimensional representations toward a compact subspace encoding class-specific semantics. This phenomenon aligns with recent theoretical descriptions of a *speciation* regime, during which the generative trajectory resolves global ambiguities and commits to a particular mode of the data distribution. Importantly, we distinguished this intermediate contraction from the later collapse regime, where low-dimensionality primarily reflects manifold constraints rather than continued dynamical convergence to train data.

Building on these observations, we explored controlled perturbations of the reverse trajectory, including adversarial noise injection and latent-space steering. These experiments demonstrated that the effect of perturbations depends strongly on their temporal placement: early interventions are often corrected by subsequent denoising, while perturbations applied after speciation can irreversibly redirect the generation toward alternative semantic outcomes. Latent-space steering via PCA directions further showed that shared semantic components between classes can be selectively injected without fully destroying image realism. The results allow for a qualitative characterization of Diffusion Models' speciation behaviours.

Together, these findings support a view of diffusion sampling as a structured, temporally ordered process in which semantic decisions are made early and refined later. Beyond descriptive insights, the identification of an effective speciation time suggests potential avenues for accelerating conditioned sampling by initializing the reverse process closer to the (noised) class-conditioned manifold.

Overall, this work contributes diagnostic tools and empirical evidence toward a more mechanistic understanding of diffusion model sampling, generalization and inductive bias, bridging recent theoretical developments with observable behavior in practical, pretrained systems.

Acknowledgments

I extend my sincere gratitude to Professor Iacopo Masi and Dr. Maria Rosaria Briglia, whose support and guidance greatly enhanced this research. Working alongside them has been an invaluable learning experience, enjoying the exploration of the wonderful topics researched.
Heartfelt thanks are due to my family, whose constant support and attention have been a source of strength and encouragement. I appreciate their enduring emotional support and the assistance provided whenever needed.
I am also grateful to my friends, each of whom holds a special place in my heart. Their diverse perspectives and approaches to life have enriched my own journey, and I value the moments we've shared.
A debt of gratitude extends to all the educators who, from an early age, have contributed to shaping me into a better person. Special thanks go to the Scout community, which has illuminated my path with thoughtful guidance and provided continuous growth in both thought and spirit.
I also want to express my appreciation to my basketball teammates, whose camaraderie provided a welcome distraction through the joy of this wonderful sport.

Bibliography

[1] Brian D.O. Anderson. "Reverse-time diffusion equation models". In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326. ISSN: 0304-4149. DOI: https://doi.org/10.1016/0304-4149(82)90051-5. URL: https://www.sciencedirect.com/science/article/pii/0304414982900515.

[2] Quentin Bertrand et al. *On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity*. 2025. arXiv: 2506.03719 [cs.LG]. URL: https://arxiv.org/abs/2506.03719.

[3] Giulio Biroli and Marc Mézard. "Generative diffusion in very large dimensions". In: *Journal of Statistical Mechanics: Theory and Experiment* 2023.9 (Sept. 2023), p. 093402. ISSN: 1742-5468. DOI: 10.1088/1742-5468/acf8ba. URL: http://dx.doi.org/10.1088/1742-5468/acf8ba.

[4] Giulio Biroli et al. "Dynamical regimes of diffusion models". In: *Nature Communications* 15.1 (Nov. 2024). ISSN: 2041-1723. DOI: 10.1038/s41467-024-54281-3. URL: http://dx.doi.org/10.1038/s41467-024-54281-3.

[5] Tony Bonnaire et al. *Why Diffusion Models Don't Memorize: The Role of Implicit Dynamical Regularization in Training*. 2025. arXiv: 2505.17638 [cs.LG]. URL: https://arxiv.org/abs/2505.17638.

[6] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[7] Prafulla Dhariwal and Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. 2021. arXiv: 2105.05233 [cs.LG]. URL: https://arxiv.org/abs/2105.05233.

[8] Bradley Efron. "Tweedie's Formula and Selection Bias". In: *Journal of the American Statistical Association* 106.496 (2011). PMID: 22505788, pp. 1602–1614. DOI: 10.1198/jasa.2011.tm11181. eprint: https://doi.org/10.1198/jasa.2011.tm11181. URL: https://doi.org/10.1198/jasa.2011.tm11181.

[9] "Gaussian Channel". In: *Elements of Information Theory*. John Wiley & Sons, Ltd, 2005. Chap. 9, pp. 261–299. ISBN: 9780471748823. DOI: https://doi.org/10.1002/047174882X.ch9. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/047174882X.ch9. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/047174882X.ch9.

[10] Amir Hertz et al. *Prompt-to-Prompt Image Editing with Cross Attention Control*. 2022. arXiv: 2208.01626 [cs.CV]. URL: https://arxiv.org/abs/2208.01626.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. arXiv: 2006.11239 [cs.LG]. URL: https://arxiv.org/abs/2006.11239.

[12] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. arXiv: 2207.12598 [cs.LG]. URL: https://arxiv.org/abs/2207.12598.

[13] Jonathan Ho et al. *Cascaded Diffusion Models for High Fidelity Image Generation*. 2021. arXiv: 2106.15282 [cs.CV]. URL: https://arxiv.org/abs/2106.15282.

[14] Aapo Hyvärinen. "Estimation of Non-Normalized Statistical Models by Score Matching". In: *Journal of Machine Learning Research* 6.24 (2005), pp. 695–709. URL: http://jmlr.org/papers/v6/hyvarinen05a.html.

[15] Tero Karras et al. *Elucidating the Design Space of Diffusion-Based Generative Models*. 2022. arXiv: 2206.00364 [cs.CV]. URL: https://arxiv.org/abs/2206.00364.

[16] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. arXiv: 1312.6114 [stat.ML]. URL: https://arxiv.org/abs/1312.6114.

[17] Diederik P. Kingma et al. *Improving Variational Inference with Inverse Autoregressive Flow*. 2017. arXiv: 1606.04934 [cs.LG]. URL: https://arxiv.org/abs/1606.04934.

[18] Diederik P. Kingma et al. *Variational Diffusion Models*. 2023. arXiv: 2107.00630 [cs.LG]. URL: https://arxiv.org/abs/2107.00630.

[19] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. *CIFAR-10 (Canadian Institute for Advanced Research)*. URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[20] Yann LeCun et al. "A Tutorial on Energy-Based Learning". In: 2006. URL: https://api.semanticscholar.org/CorpusID:8531544.

[21] Xiao Li et al. *Understanding Representation Dynamics of Diffusion Models via Low-Dimensional Modeling*. 2025. arXiv: 2502.05743 [cs.LG]. URL: https://arxiv.org/abs/2502.05743.

[22] Calvin Luo. *Understanding Diffusion Models: A Unified Perspective*. 2022. arXiv: 2208.11970 [cs.LG]. URL: https://arxiv.org/abs/2208.11970.

[23] William Peebles and Saining Xie. *Scalable Diffusion Models with Transformers*. 2023. arXiv: 2212.09748 [cs.CV]. URL: https://arxiv.org/abs/2212.09748.

[24] Xiangyu Peng et al. *Open-Sora 2.0: Training a Commercial-Level Video Generation Model in $200k*. 2025. arXiv: 2503.09642 [cs.GR]. URL: https://arxiv.org/abs/2503.09642.

[25] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. arXiv: 2204.06125 [cs.CV]. URL: https://arxiv.org/abs/2204.06125.

[26] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. *Stochastic Backpropagation and Approximate Inference in Deep Generative Models*. 2014. arXiv: 1401.4082 [stat.ML]. URL: https://arxiv.org/abs/1401.4082.

[27] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV]. URL: https://arxiv.org/abs/2112.10752.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV]. URL: https://arxiv.org/abs/1505.04597.

[29] Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. arXiv: 2205.11487 [cs.CV]. URL: https://arxiv.org/abs/2205.11487.

[30] Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. arXiv: 2205.11487 [cs.CV]. URL: https://arxiv.org/abs/2205.11487.

[31] Saeed Saremi et al. *Deep Energy Estimator Networks*. 2018. arXiv: 1805.08306 [stat.ML]. URL: https://arxiv.org/abs/1805.08306.

[32] Christopher Scarvelis, Haitz Sáez de Ocáriz Borde, and Justin Solomon. *Closed-Form Diffusion Models*. 2025. arXiv: 2310.12395 [cs.LG]. URL: https://arxiv.org/abs/2310.12395.

[33] Christoph Schuhmann et al. *LAION-5B: An open large-scale dataset for training next generation image-text models*. 2022. arXiv: 2210.08402 [cs.CV]. URL: https://arxiv.org/abs/2210.08402.

[34] Antonio Sclocchi, Alessandro Favero, and Matthieu Wyart. *A Phase Transition in Diffusion Models Reveals the Hierarchical Nature of Data*. 2024. arXiv: 2402.16991 [stat.ML]. URL: https://arxiv.org/abs/2402.16991.

[35] Jascha Sohl-Dickstein et al. *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. 2015. arXiv: 1503.03585 [cs.LG]. URL: https://arxiv.org/abs/1503.03585.

[36] Casper Kaae Sønderby et al. *Ladder Variational Autoencoders*. 2016. arXiv: 1602.02282 [stat.ML]. URL: https://arxiv.org/abs/1602.02282.

[37] Jiaming Song, Chenlin Meng, and Stefano Ermon. *Denoising Diffusion Implicit Models*. 2022. arXiv: 2010.02502 [cs.LG]. URL: https://arxiv.org/abs/2010.02502.

[38] Yang Song and Stefano Ermon. *Generative Modeling by Estimating Gradients of the Data Distribution*. 2020. arXiv: 1907.05600 [cs.LG]. URL: https://arxiv.org/abs/1907.05600.

[39] Yang Song and Stefano Ermon. *Improved Techniques for Training Score-Based Generative Models*. 2020. arXiv: 2006.09011 [cs.LG]. URL: https://arxiv.org/abs/2006.09011.

[40] Yang Song and Diederik P. Kingma. *How to Train Your Energy-Based Models*. 2021. arXiv: 2101.03288 [cs.LG]. URL: https://arxiv.org/abs/2101.03288.

[41] Yang Song et al. *Score-Based Generative Modeling through Stochastic Differential Equations*. 2021. arXiv: 2011.13456 [cs.LG]. URL: https://arxiv.org/abs/2011.13456.

[42] Yang Song et al. *Sliced Score Matching: A Scalable Approach to Density and Score Estimation*. 2019. arXiv: 1905.07088 [cs.LG]. URL: https://arxiv.org/abs/1905.07088.

[43] Pascal Vincent. "A Connection Between Score Matching and Denoising Autoencoders". In: *Neural Computation* 23 (July 2011), pp. 1661–1674. DOI: 10.1162/NECO_a_00142.

[44] Binxu Wang and John J. Vastola. *Diffusion Models Generate Images Like Painters: an Analytical Theory of Outline First, Details Later*. 2024. arXiv: 2303.02490 [cs.CV]. URL: https://arxiv.org/abs/2303.02490.

[45] Lilian Weng. "What are diffusion models?" In: *lilianweng.github.io* (July 2021). URL: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[46] Kaiwen Zheng et al. *DPM-Solver-v3: Improved Diffusion ODE Solver with Empirical Model Statistics*. 2023. arXiv: 2310.13268 [cs.CV]. URL: https://arxiv.org/abs/2310.13268.

Appendix A

Derivations

Below, we present the main derivations of the equations introduced in Chapter 2. These derivations encompass proofs for the ELBO of VAEs, HVAEs and VDMs, as well as VDMs recursive parameterization, KL Divergence between gaussians, and the equivalance of the three Diffusion Models' formulations. In addition, Classifier Guidance and the architectures of Diffusion Models are examined in detail.

A.1 Evidence Lower Bound

Derivations to see why ELBO is an objective we would like to maximize.

Deriving the ELBO using equation Equation 2.1:

$$\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} \\
&= \log \int \frac{p(\boldsymbol{x}, \boldsymbol{z}) q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})} d\boldsymbol{z} \\
&= \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})} \right] \\
&\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})} \right]
\end{aligned} \tag{A.1.1}$$

Where Jensen's Inequality was applied to arrive at the lower bound. This derivation however does not supply much useful intuition on exactly why the ELBO is actually a lower bound of the evidence, as Jensen's Inequality handwaves it away. Furthermore, knowing that ELBO is truly a lower bound of the data does not really tell us why we want to maximize it as an objective. To better understand the relationship between the evidence and the ELBO, we can perform another derivation, using Equation 2.2:

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log p(\boldsymbol{x}) \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) dz \\
&= \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \log p(\boldsymbol{x}) dz \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log p(\boldsymbol{x})\right] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{z} \mid \boldsymbol{x})}\right] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}) q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}{p(\boldsymbol{z} \mid \boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}{p(\boldsymbol{z} \mid \boldsymbol{x})}\right] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] + \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z} \mid \boldsymbol{x})) \qquad \text{(A.1.2)} \\
&\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right] \qquad \text{(A.1.3)}
\end{aligned}
$$

From this derivation, we clearly observe from Equation A.1.2 that the evidence is equal to the ELBO plus the (non-symmetric) Kullback-Leibler Divergence $\left(D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}\right)$ between the approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ and the true posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$. This KL Divergence was removed by Jensen's Inequality in Equation A.1.1. Now we now why the ELBO is indeed a lower bound: the difference between the evidence and the ELBO is a strictly non-negative KL term, thus the value of the ELBO can never exceed the evidence.

Having introduced latent variables $\boldsymbol{z}$ that we would like to model, our goal is to learn this underlying latent structure that describes observed data. We want to optimize the parameters of our variational posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ to exactly match the true posterior distribution $p(\boldsymbol{z} \mid \boldsymbol{x})$, which is achieved by minimizing their KL Divergence, ideally to zero. Unfortunately, it is intractable to minimize this KL Diverge term directly, as we do not have access to the ground truth $p(\boldsymbol{z} \mid \boldsymbol{x})$ distribution. However on the left hand side of Equation A.1.2, the likelihood of our data (and therefore the evidence term $\log p(\boldsymbol{x})$) is always a constant with respect to $\boldsymbol{\phi}$, as it is computed by marginalizing out all latents $\boldsymbol{z}$ from the joint distribution $p(\boldsymbol{x}, \boldsymbol{z})$ and does not depend on $\boldsymbol{\phi}$ whatsoever. Since the ELBO and KL Divergence terms sum up to a constant, any maximization of the ELBO term with respect to $\boldsymbol{\phi}$ necessarily invokes an equal minimization of the KL Divergence term: $\frac{\partial}{\partial \phi}\mathcal{L}(\phi) = -\frac{\partial}{\partial \phi}\mathrm{KL}(q_\phi||p)$, where $\mathcal{L}(\phi)$ is the ELBO. Thus, the ELBO can be maximized as a proxy for learning how to perfectly model the true latent posterior distribution; the more optimized the ELBO is, the closer the approximate posterior gets to the true posterior (KL ≥ 0, so $\mathcal{L}(\phi) \leq \log p(x)$, with equality iff $q_\phi(z|x) = p(z|x)$). Additionally, once trained, the ELBO can be used to estimate the likelihood of observed or generated data as well, since it is learned to approximate the model evidence $\log p(\boldsymbol{x})$.

A.2 Hierarchical VAE ELBO

Derivation for the ELBO of HVAE:

$$\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) d\boldsymbol{z}_{1:T} \\
&= \log \int \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})} d\boldsymbol{z}_{1:T} \\
&= \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})} \right] \\
&\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x})} \right]
\end{aligned}$$

A.3 ELBO form with two-variable expectations

Like any HVAE, the VDM can be optimized by maximing the ELBO [35]. One derivation of the ELBO, depending only on $\boldsymbol{x}_0$ results in the following form:

$$\begin{aligned}
\log p(\boldsymbol{x}) &\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)} \right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1}) \,||\, p(\boldsymbol{x}_T)) \right]}_{\text{prior matching term}} \\
&\quad - \sum_{t=1}^{T-1} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) \,||\, p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})) \right]}_{\text{consistency term}}
\end{aligned} \tag{A.3.1}$$

The derived form of the ELBO can be interpreted in terms of its individual components: $\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right]$ can be interpreted as a reconstruction term, predicting the log probability of the original data sample given the first-step latent. This term also appears in a vanilla VAE, and can be trained similarly; $\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1})) \,||\, p(\boldsymbol{x}_T)) \right]$ is a prior matching term, it is minimized when the final latent distribution matches the Gaussian prior. This term does not require any optimization, as it has no trainable parameters. Furthermore as we have assumed a large enough T such that the final distribution is Gaussian, this term effectively becomes zero; $\mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})) \right]$ is a consistency term, it endeavors to make the distribution at $\boldsymbol{x}_t$ consistent, from both the forward and backward processes. That is, a denoising step from a noisier image should match the corresponding noising step from a cleaner image, for every intermediate timestep, this is reflected mathematically by the KL Divergence. This term is minimized when we train $p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})$ to match the Gaussian distribution $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, defined in Eq. 2.13.

Visually, this interpretation of the ELBO is depicted in Fig. A.1 below. The cost of optimizing a VDM is primarily dominated by the third term, since we must optimize over all timesteps t.

Under this derivation, all terms of the ELBO are computed as expectations, and can therefore be approximated using Monte Carlo estimates. However, optimizing ELBO using these terms might be suboptimal, as the consistency term is computed as an expectation over two random variables $\{\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}\}$ for every timestep, the variance of its Monte

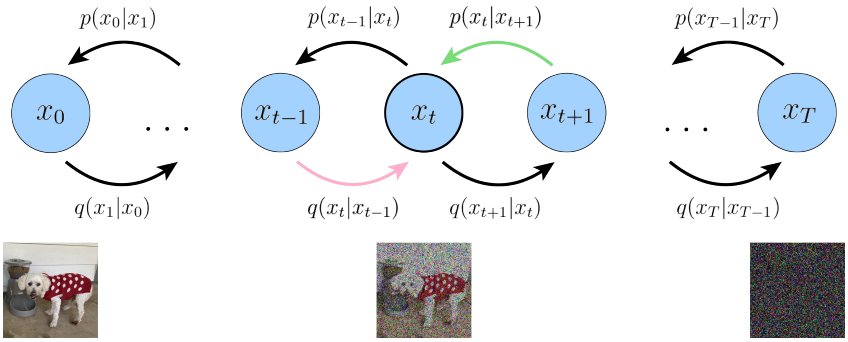


Figure A.1. A VDM can be optimized by ensuring that for every intermediate latent, the posterior from the latent above it matches the Gaussian corruption of the latent before it. In this figure, for each intermediate latent, we minimize the difference between the distributions represented by the pink and green arrows.

Carlo estimate could potentially be higher than a term that is estimated using only one random variable per timestep. As it is computed by summing up $T-1$ consistency terms, the final estimated value of the ELBO may then have high variance for large T values.

Proof of Eq. A.3.1

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}_{0:T})\, d\boldsymbol{x}_{1:T} \\
&= \log \int p(\boldsymbol{x}_{0:T}) \frac{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\, d\boldsymbol{x}_{1:T} \\
&= \log \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)} \right] \\
&\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{\prod_{t=1}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \prod_{t=2}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1}) \prod_{t=1}^{T-1} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \prod_{t=1}^{T-1} p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1}) \prod_{t=1}^{T-1} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1})} \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \prod_{t=1}^{T-1} \frac{p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1})} \right] \\
&+ \sum_{t=1}^{T-1} \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{T-1}, \boldsymbol{x}_T|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1})} \right] \\
&+ \sum_{t=1}^{T-1} \mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[\log \frac{p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_{T-1}) \,\|\, p(\boldsymbol{x}_T)) \right]}_{\text{prior matching term}} \\
&\underbrace{- \sum_{t=1}^{T-1} \mathbb{E}_{q(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t+1}|\boldsymbol{x}_0)} \left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) \,\|\, p_\theta(\boldsymbol{x}_t \mid \boldsymbol{x}_{t+1})) \right]}_{\text{consistency term}}
\end{aligned}
$$

A.4 Variational Diffusion Models

A.4.1 Proof of Equation 2.16.

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}_{0:T})\, d\boldsymbol{x}_{1:T} \\
&= \log \int p(\boldsymbol{x}_{0:T}) \frac{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\, d\boldsymbol{x}_{1:T} \\
&= \log \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)} \right] \\
&\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{\prod_{t=1}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \prod_{t=2}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0) \prod_{t=2}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \prod_{t=2}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0) \prod_{t=2}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)} + \log \prod_{t=2}^{T} \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)} + \log \prod_{t=2}^{T} \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{\frac{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \cancel{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}}{\cancel{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}}} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{\cancel{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}} + \log \frac{\cancel{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)} + \log \prod_{t=2}^{T} \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)} \right] + \\
&\sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_T|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)} \right] + \\
&\sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}|\boldsymbol{x}_0)} \left[\log \frac{p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1) \right]}_{\text{reconstruction term}} - \underbrace{\mathrm{KL}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_0) \,||\, p(\boldsymbol{x}_T))}_{\text{prior matching term}} \\
&- \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[\mathrm{KL}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \right]}_{\text{denoising matching term}}
\end{aligned}
$$

A.4.2 Recursive reparameterization

$$
\begin{aligned}
\boldsymbol{x}_t &= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}^*_{t-1} \\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}^*_{t-2}\right) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}^*_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t - \alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}^*_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}^*_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\sqrt{\alpha_t - \alpha_t\alpha_{t-1}}^2 + \sqrt{1-\alpha_t}^2}\boldsymbol{\epsilon}_{t-2} \qquad \text{(A.4.1)} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t - \alpha_t\alpha_{t-1} + 1 - \alpha_t}\boldsymbol{\epsilon}_{t-2} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1 - \alpha_t\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2} \\
&= \dots \\
&= \sqrt{\prod_{i=1}^{t}\alpha_i}\boldsymbol{x}_0 + \sqrt{1 - \prod_{i=1}^{t}\alpha_i}\boldsymbol{\epsilon}_0 \\
&= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0 \\
&\sim \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})
\end{aligned}
$$

where in Eq. A.4.1 we use the fact that the sum of independent Gaussian random variables remains a Gaussian, with mean being the sum of the two means, and variance being the sum of two variances, and the fact that $\boldsymbol{x}_t$ is expressed as a linear combination of independent Gaussian terms, each scaled by its standard deviation. That is, if $a\,\epsilon_1 + b\,\epsilon_2$ with $\epsilon_1, \epsilon_2 \sim \mathcal{N}(0,1), \epsilon_1 \perp \epsilon_2$ then

$$
a\,\epsilon_1 + b\,\epsilon_2 \sim \sqrt{a^2 + b^2}\,\epsilon \quad \text{with} \quad \epsilon \sim \mathcal{N}(0,1).
$$

A.4.3 Proof of Equation 2.20.

$$
\begin{aligned}
q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)} \\
&= \frac{\mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I}) \mathcal{N}(\boldsymbol{x}_{t-1}; \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0, (1-\bar{\alpha}_{t-1})\mathbf{I})}{\mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\mathbf{I})} \\
&\propto \exp\left\{ -\frac{1}{2}\Big[\frac{(\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_{t-1})^2}{1-\alpha_t} + \frac{(\boldsymbol{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(\boldsymbol{x}_t - \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0)^2}{1-\bar{\alpha}_t} \Big] \right\} \\
&= \exp\left\{ -\frac{1}{2}\Big[\frac{-2\sqrt{\alpha_t}\boldsymbol{x}_t\boldsymbol{x}_{t-1} + \alpha_t \boldsymbol{x}_{t-1}^2}{1-\alpha_t} + \frac{\boldsymbol{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_{t-1}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} + C(\boldsymbol{x}_t, \boldsymbol{x}_0) \Big] \right\} \\
&\propto \exp\left\{ -\frac{1}{2}\Big[\frac{-2\sqrt{\alpha_t}\boldsymbol{x}_t\boldsymbol{x}_{t-1}}{1-\alpha_t} + \frac{\alpha_t \boldsymbol{x}_{t-1}^2}{1-\alpha_t} + \frac{\boldsymbol{x}_{t-1}^2}{1-\bar{\alpha}_{t-1}} - \frac{2\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_{t-1}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \Big] \right\} \\
&= \exp\left\{ -\frac{1}{2}\Big[\Big(\frac{\alpha_t}{1-\alpha_t} + \frac{1}{1-\bar{\alpha}_{t-1}} \Big) \boldsymbol{x}_{t-1}^2 - 2\Big(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \Big) \boldsymbol{x}_{t-1} \Big] \right\} \\
&= \exp\left\{ -\frac{1}{2}\Big[\frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})} \boldsymbol{x}_{t-1}^2 - 2\Big(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \Big) \boldsymbol{x}_{t-1} \Big] \right\} \\
&= \exp\left\{ -\frac{1}{2}\left[\frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})} \boldsymbol{x}_{t-1}^2 - 2\left(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \right) \boldsymbol{x}_{t-1} \right] \right\} \\
&= \exp\left\{ -\frac{1}{2}\left[\frac{1-\bar{\alpha}_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})} \boldsymbol{x}_{t-1}^2 - 2\left(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \right) \boldsymbol{x}_{t-1} \right] \right\} \\
&= \exp\left\{ -\frac{1}{2}\frac{1-\bar{\alpha}_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})} \left[\boldsymbol{x}_{t-1}^2 - 2\frac{\left(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{1-\bar{\alpha}_{t-1}} \right)}{\frac{1-\bar{\alpha}_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}} \boldsymbol{x}_{t-1} \right] \right\} \\
&= \exp\left\{ -\frac{1}{2}\frac{1-\bar{\alpha}_t}{(1-\alpha_t)(1-\bar{\alpha}_{t-1})} \left[\boldsymbol{x}_{t-1}^2 - 2\frac{\left(\frac{\sqrt{\alpha_t}\boldsymbol{x}_t}{(1-\alpha_t)} + \frac{\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0}{(1-\bar{\alpha}_{t-1})} \right) + (1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \boldsymbol{x}_{t-1} \right] \right\} \\
&= \exp\left\{ -\frac{1}{2}\frac{1}{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}} \Big[\boldsymbol{x}_{t-1}^2 - 2\Big(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \Big) \boldsymbol{x}_{t-1} \Big] \right\} \\
&\propto \mathcal{N}(\boldsymbol{x}_{t-1}; \underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)})
\end{aligned}
$$

A.4.4 KL Divergence between two Gaussians

We utilize the fact that the KL Divergence between two Gaussian distributions is [9]:

$$
\begin{aligned}
&\mathrm{KL}(\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) \,||\, \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)) \\
&= \frac{1}{2}\left[\log \frac{|\,\boldsymbol{\Sigma}_y\,|}{|\,\boldsymbol{\Sigma}_x\,|} - d + \mathrm{tr}(\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_x) + (\boldsymbol{\mu}_y - \boldsymbol{\mu}_x)^T \boldsymbol{\Sigma}_y^{-1}(\boldsymbol{\mu}_y - \boldsymbol{\mu}_x) \right] \qquad \text{(A.4.2)}
\end{aligned}
$$

In our case, where we can set the variances of the two Gaussians to match exactly, optimizing the KL Divergence term reduces to minimizing the difference between the means

of the two distributions:

$$
\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&= \arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \mid\mid \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right)\right) \\
&= \arg\min_{\theta} \frac{1}{2}\Big[\log\frac{|\boldsymbol{\Sigma}_q(t)|}{|\boldsymbol{\Sigma}_q(t)|} - d + \mathrm{tr}(\boldsymbol{\Sigma}_q(t)^{-1}\boldsymbol{\Sigma}_q(t)) + (\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)^\top \boldsymbol{\Sigma}_q(t)^{-1}(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)\Big] \\
&= \arg\min_{\theta} \frac{1}{2}\Big[\log 1 - d + d + (\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)^\top \boldsymbol{\Sigma}_q(t)^{-1}(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)\Big] \\
&= \arg\min_{\theta} \frac{1}{2}\Big[(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)^\top \boldsymbol{\Sigma}_q(t)^{-1}(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)\Big] \\
&= \arg\min_{\theta} \frac{1}{2}\Big[(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)^\top (\sigma_q^2(t)\mathbf{I})^{-1}(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_q)\Big] \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)}\left[\left\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right\|_2^2\right]
\end{aligned}
$$

where $\boldsymbol{\mu}_q$ is shorthand for $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$, and $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ as shorthand for $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ for brevity.

A.4.5 Proof of Equation 2.24.

$$
\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&= \arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \mid\mid \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right)\right) \\
&= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)}\left\|\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t)}{1-\bar{\alpha}_t} - \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}\right\|_2^2 \\
&= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)}\left\|\frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\left(\hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right)\right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)}\frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2\right]
\end{aligned}
$$

A.4.6 Proof of Equation 2.27.

$$
\begin{aligned}
\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + (1-\alpha_t)\frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\alpha_t}}}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{(1-\alpha_t)\sqrt{1-\bar{\alpha}_t}}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0 \\
&= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\right)\boldsymbol{x}_t - \frac{(1-\alpha_t)\sqrt{1-\bar{\alpha}_t}}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0 \\
&= \left(\frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\right)\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0 \\
&= \frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0 \\
&= \frac{1-\bar{\alpha}_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0 \\
&= \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0
\end{aligned}
$$

A.4.7 Proof of Equation 2.31.

$$
\begin{aligned}
\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\frac{\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t)}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + (1-\alpha_t)\frac{\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t)}{\sqrt{\alpha_t}}}{1-\bar{\alpha}_t} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{(1-\alpha_t)(1-\bar{\alpha}_t)}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t) \\
&= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\right)\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t) \\
&= \left(\frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\right)\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t) \\
&= \frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t) \\
&= \frac{1-\bar{\alpha}_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t) \\
&= \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\nabla\log p(\boldsymbol{x}_t)
\end{aligned}
$$

A.4.8 Proof of Equation 2.32.

$$
\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&= \arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \mid\mid \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right)\right) \\
&= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)} \left\| \frac{1}{\sqrt{\alpha_t}} \boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \frac{1}{\sqrt{\alpha_t}} \boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t}} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) \right\|_2^2 \\
&= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)} \left\| \frac{1-\alpha_t}{\sqrt{\alpha_t}} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \frac{1-\alpha_t}{\sqrt{\alpha_t}} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) \right\|_2^2 \\
&= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)} \left\| \frac{1-\alpha_t}{\sqrt{\alpha_t}} \left(\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)\right) \right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{\alpha_t} \left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \nabla \log p(\boldsymbol{x}_t)\|_2^2 \right]
\end{aligned}
$$

From here, the constant prefactor drops, giving the standard score-matching objective

$$
\arg\min_{\boldsymbol{\theta}} \quad \mathbb{E}_{p(\boldsymbol{x})} \|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)\|_2^2
$$

which cleanly connects diffusion training to denoising score matching.

A.4.9 Equivalence of the Three Views

Combining Tweedie's formula (Eq. 2.30) with the reparameterization trick (Eq. 2.26) shows the similarity between Eq. 2.32 and Eq. 2.28:

$$
\begin{aligned}
\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla \log p(\boldsymbol{x}_t)}{\sqrt{\bar{\alpha}_t}} &= \frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{\alpha}_t}} \\
\therefore (1-\bar{\alpha}_t)\nabla \log p(\boldsymbol{x}_t) &= -\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0 \\
\nabla \log p(\boldsymbol{x}_t) &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_0
\end{aligned} \tag{A.4.3}
$$

showing that the score and the negative noise differ only by a time–dependent scaling factor. The score function measures how to move in data space to maximize the log probability; intuitively, since the source noise is added to a natural image to corrupt it, moving in its opposite direction 'denoises' the image and would be the best update to increase the subsequent log probability. Consequently, the three training formulations are mathematically equivalent: a VDM may be learned by predicting the clean image $\boldsymbol{x}_0$, the source noise $\boldsymbol{\epsilon}_0$, or the score$\nabla \log p(\boldsymbol{x}_t)$ [11,18]. In practice, training proceeds by sampling timesteps t and minimizing the corresponding prediction error across noise levels.

A.5 Classifier Guidance

Consider a score-based diffusion model, where the goal is to learn the conditional score $\nabla \log p(\boldsymbol{x}_t \mid y)$ at arbitrary noise levels t. For brevity, we write ∇ as shorthand for $\nabla_{\boldsymbol{x}_t}$.

Using Bayes' rule, the conditional score can be expressed as

$$\begin{aligned}\nabla \log p(\boldsymbol{x}_t \mid y) &= \nabla \log \left(\frac{p(\boldsymbol{x}_t) p(y \mid \boldsymbol{x}_t)}{p(y)} \right) \\ &= \nabla \log p(\boldsymbol{x}_t) + \nabla \log p(y \mid \boldsymbol{x}_t) - \nabla \log p(y) \\ &= \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{adversarial gradient}}\end{aligned} \tag{A.5.1}$$

where $\nabla \log p(y) = 0$ since it does not depend on $\boldsymbol{x}_t$.

Equation (A.5.1) shows that the conditional score can be decomposed into the unconditional score function and the gradient of a classifier predicting y from $\boldsymbol{x}_t$. Classifier Guidance leverages this decomposition by jointly learning:

1. An unconditional score function $\nabla \log p(\boldsymbol{x}_t)$, and
2. A classifier $p(y \mid \boldsymbol{x}_t)$ capable of handling noisy inputs $\boldsymbol{x}_t$.

During sampling, the conditional score used in annealed Langevin dynamics is computed as the sum of these two components.

To allow flexible control over the influence of the conditioning information, Classifier Guidance introduces a scaling factor $\gamma \in \mathbb{R}^+$ on the classifier gradient:

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \nabla \log p(\boldsymbol{x}_t) + \gamma \nabla \log p(y \mid \boldsymbol{x}_t) \tag{A.5.2}$$

When $\gamma = 0$, the model ignores the conditioning information entirely, whereas large γ values enforce strong adherence to y, potentially reducing sample diversity, as the model favors regions of the data space that are easily predictable from y even under noise.

A notable limitation of Classifier Guidance is the need to train a separate classifier that can handle arbitrarily noisy inputs. Most pretrained classifiers are not optimized for this setting, requiring the classifier to be trained specifically alongside the diffusion model.

A.6 Diffusion Models Architectures

There are two common backbone architecture choices for diffusion models: U-Net and Transformer.

U-Net [28] consists of a downsampling stack and an upsampling stack.

- Downsampling: Each step consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a ReLU and a 2x2 max pooling with stride 2. At each downsampling step, the number of feature channels is doubled.
- Upsampling: Each step consists of an upsampling of the feature map followed by a 2x2 convolution and each halves the number of feature channels.
- Shortcuts: Shortcut connections result in a concatenation with the corresponding layers of the downsampling stack and provide the essential high-resolution features to the upsampling process.

Diffusion Transformer [23] for diffusion modeling operates on latent patches, using the same design space of LDM (Latent Diffusion Model) [27]. DiT has the following setup:

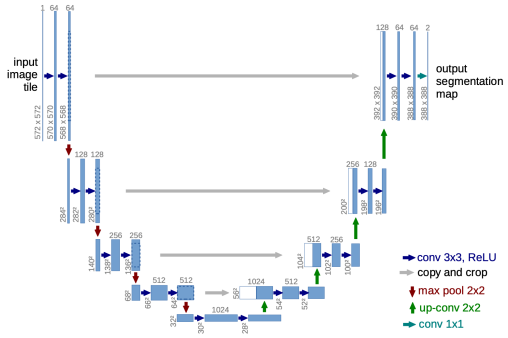


Figure A.2. The U-net architecture. Each blue square is a feature map with the number of channels labeled on top and the height x width dimension labeled on the left bottom side. The gray arrows mark the shortcut connections. [28]

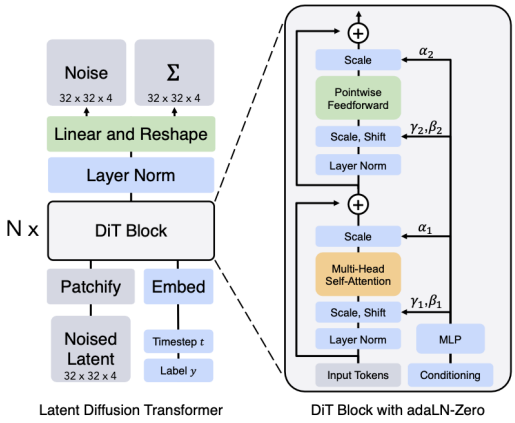


Figure A.3. The Diffusion Transformer (DiT) architecture. [23]

1. ake the latent representation of an input $\boldsymbol{z}$ as input to DiT.

2. "Patchify" the noise latent of size $I \times I \times C$ into patches of size p and convert it into a sequence of patches of size $(I/p)^2$.

3. Then this sequence of tokens go through Transformer blocks. They are exploring three different designs for how to do generation conditioned on contextual information like timestep t or class label c. Among three designs, adaLN (Adaptive layer norm)-Zero works out the best, better than in-context conditioning and cross-attention block. The scale and shift parameters, γ and β, are regressed from the sum of the embedding vectors of t and c. The dimension-wise scaling parameters α are also regressed and applied immediately prior to any residual connections within the DiT block.

4. The transformer decoder outputs noise predictions and an output diagonal covariance prediction.

Transformer architecture can be easily scaled up and it is well known for that. This is one of the biggest benefits of DiT as its performance scales up with more compute and larger DiT models are more compute efficient according to the experiments.

Appendix B

Additional Figures

Additional figures are provided in this appendix to offer deeper insights into the implemented experiments. The primary results are reported in Chapter 3.

B.1 Noise-injection Trajectories

Figures B.2–B.7 report final decoded images obtained when we adversarially perturb the reverse trajectory only during the interval $t \in [T_{\text{lower}}, T_{\text{upper}}]$. Each figure corresponds to a different choice of the perturbed interval: the baseline (no injection) and five experiments where perturbations begin progressively earlier (from $T_{\text{lower}} = 800$ down to $T_{\text{lower}} = 0$). These runs use an adversarial weight of 1.0 applied only inside the specified interval. The sequence illustrates the transition from a regime where late-time perturbations are corrected by the denoising dynamics (no class switch), through a partial-speciation zone where hybrid / intermediate features appear, to an earlier-time regime where the trajectory collapses to the adversarial class and the original class cannot be recovered.

Interpretation. Taken together these single-figure panels make explicit that (i) perturbations ending before the operational speciation time t_S are typically corrected and do not change class membership, (ii) perturbations ending near or after t_S can induce partial or full speciation depending on strength and duration, and (iii) there is a transitional window where trajectories are most susceptible to redirection.

(a) Prompt: 'a photo of a f1 car'; **Adversarial prompt**: 'photo of a jungle'. Both conditionings bias sampling toward a hybrid image that combines features of an F1 car and a jungle. Interestingly, the two semantic concepts are compatible (a F1 car can plausibly appear within a natural scene), the model produces a coherent blend of both classes.

(b) Prompt: 'a photo of a f1 car'; **Adversarial prompt**: 'photo of an ocean'. Both conditionings influence early denoising, but the concepts are semantically incompatible (an F1 car cannot occupy an ocean). The model generates only one mode.

Figure B.1. Injecting adversarial conditioning into the Stable Diffusion sampling trajectory. Shown: final generated image. Adversarial weight $= 0.5$.

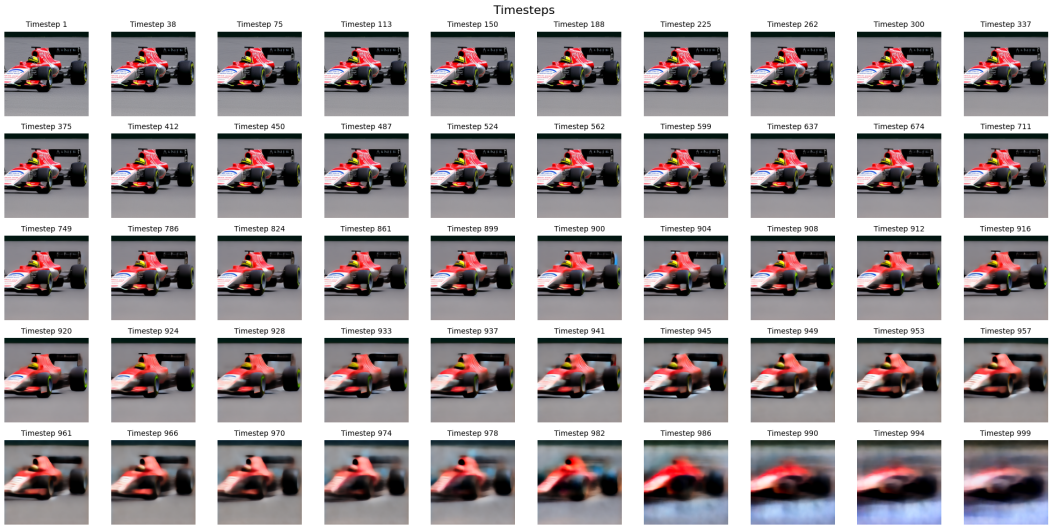


Figure B.2. No noise injection (baseline). Standard generation with the original conditioning (no adversarial perturbation). The model produces a realistic F1 car.

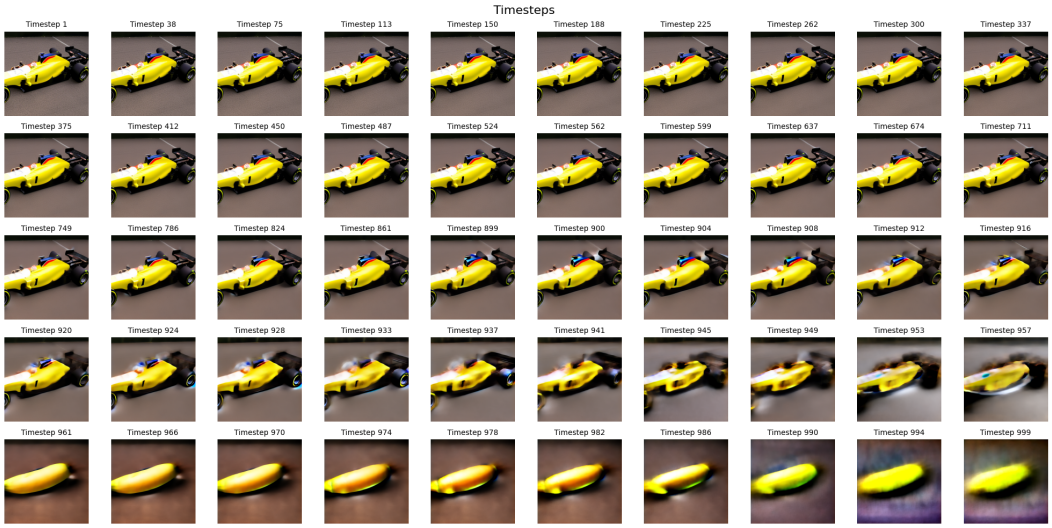


Figure B.3. Noise injection for $t \in [800, 1000]$. Perturbations confined to the latest timesteps are largely corrected by subsequent denoising: the model still produces a clear F1 car with no reliable evidence of the adversarial class.

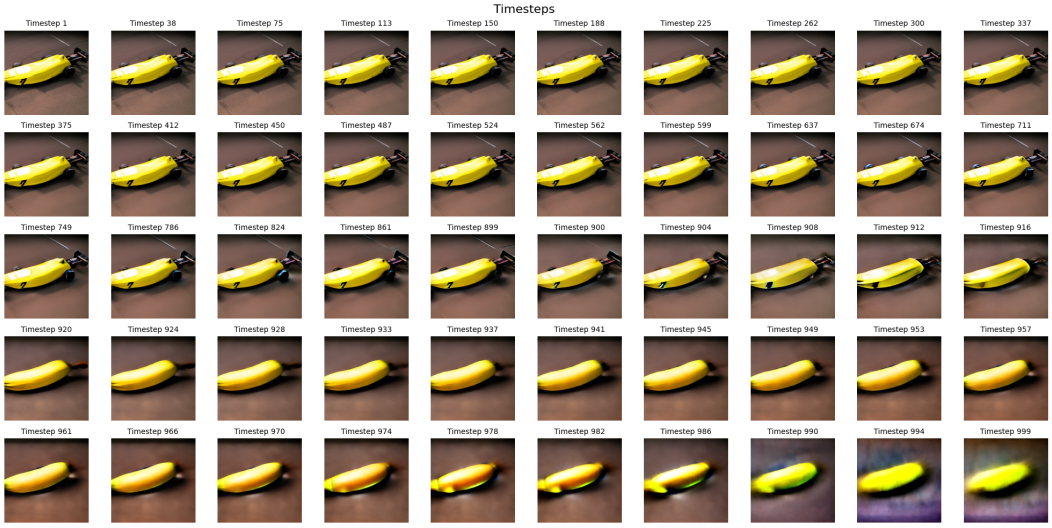


Figure B.4. Noise injection for $t \in [600, 1000]$. The image shows partial degradation and hybrid features: some car structure remains but adversarial (banana-like) cues persist, indicating partial speciation.

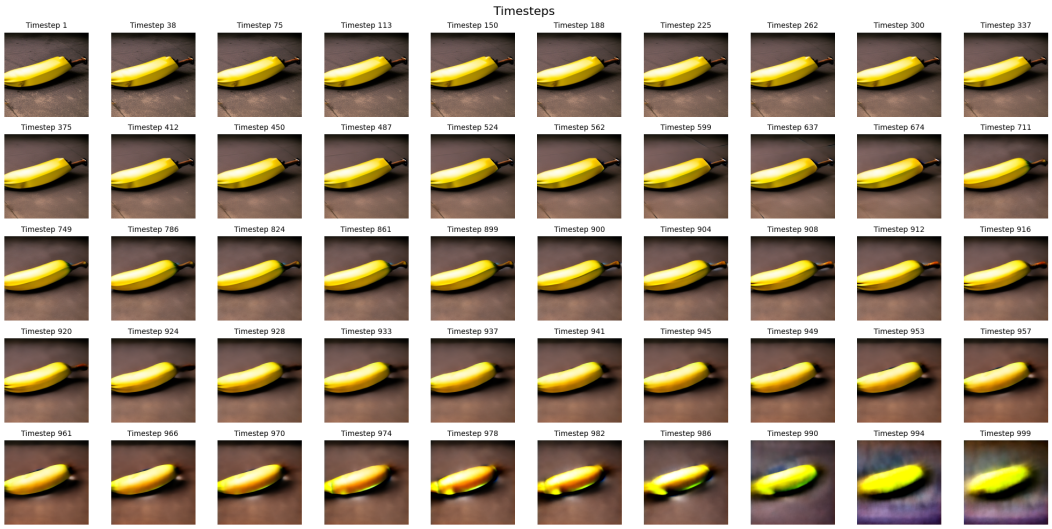


Figure B.5. Noise injection for $t \in [400, 1000]$. The model can no longer reliably recover explicit car features; the trajectory has been pushed into a different basin and only residual or hybrid shapes appear.

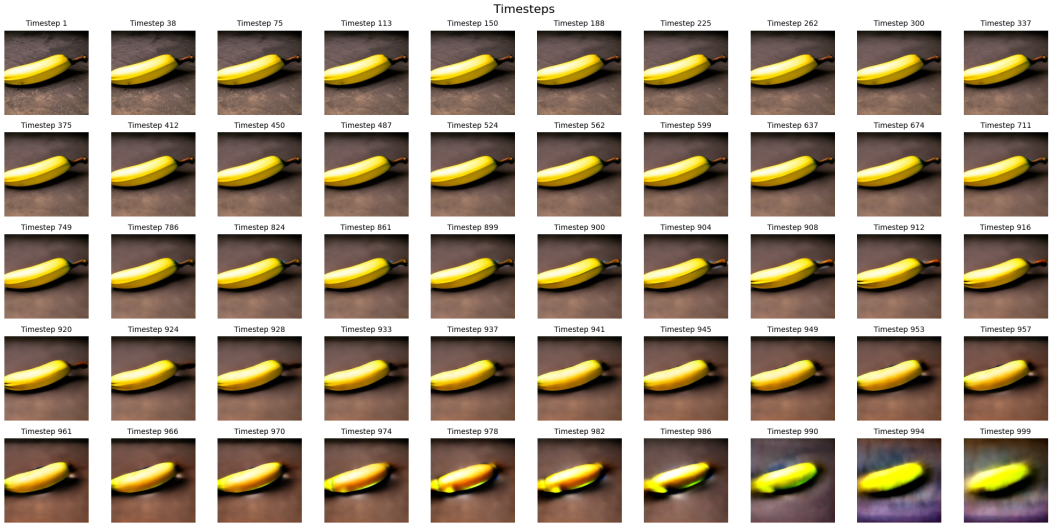


Figure B.6. Noise injection for $t \in [200, 1000]$. The generated image is dominated by the adversarial class and car features are essentially lost: collapse toward the adversarial attractor has occurred.

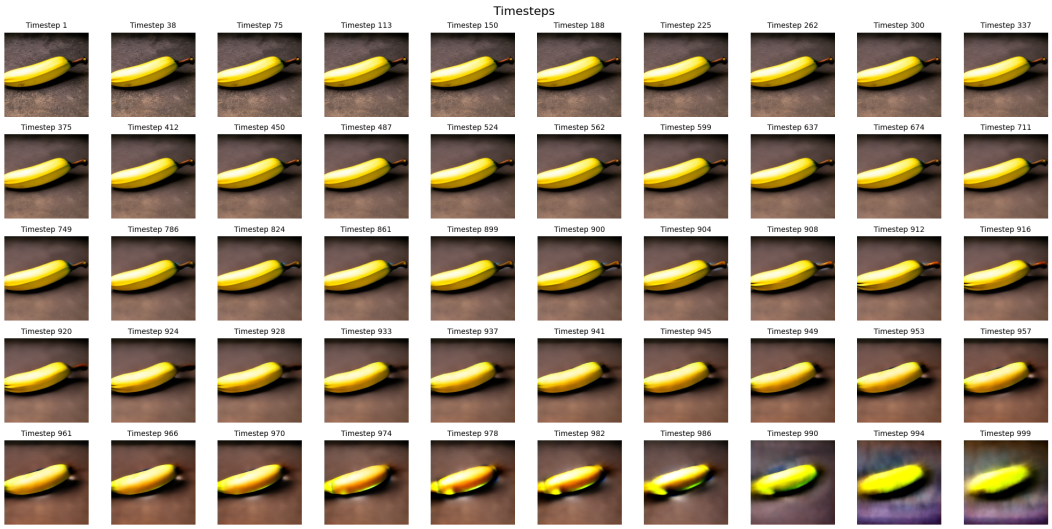


Figure B.7. Noise injection for $t \in [0, 1000]$. Perturbing across the entire denoising trajectory produces a sample that effectively matches the adversarial conditioning (the adversarial class), similar to sampling directly from the adversarial prompt.

B.2 Latent steering Trajectories

Figures B.8–B.11 show decoded images obtained when applying the latent-space steering procedures described in Sec. 3.4.4. Fig. B.8 is the baseline (no steering), Fig. B.9-B.11 show results for three different constructions of the perturbation Δ: weighted-sum projection, SVD/principal-direction projection, and shared–residual disentangling. All runs use the same prompt ('photo of a f1 race car') and the same attacked interval $\mathcal{I} = [0, 1000]$; only the construction of Δ and its magnitude differ.

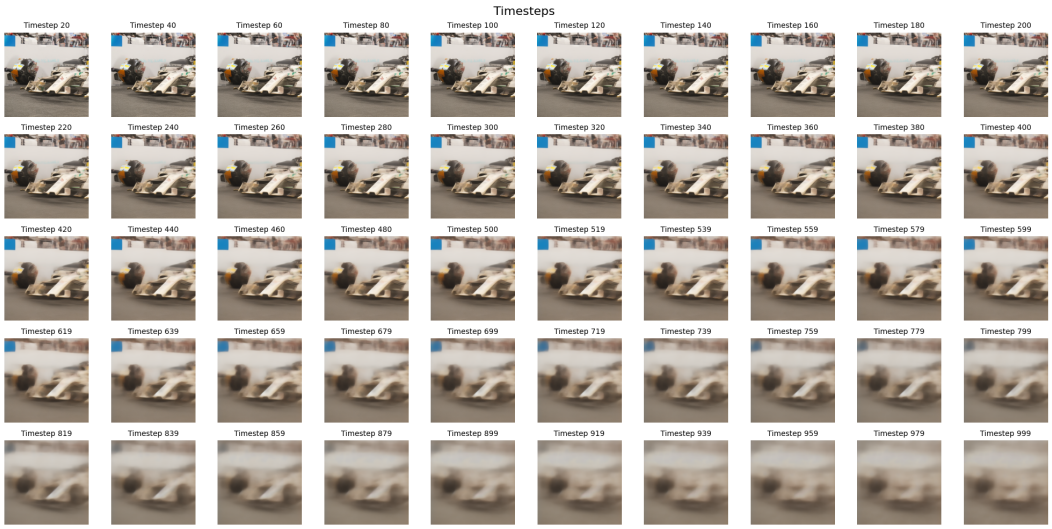


Figure B.8. Standard Stable Diffusion generation (baseline). No latent steering applied; the model produces a realistic F1 car without banana-like features.

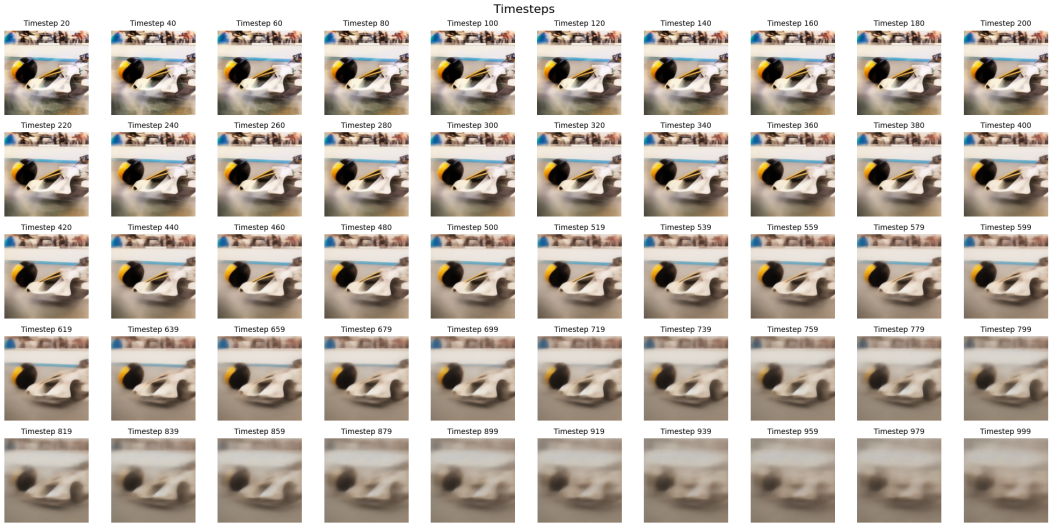


Figure B.9. Weighted-sum projection. Steering Δ constructed by variance-weighted aggregation of source-to-target projections (steer delta $= 2.5$). After speciation ($t \approx 800$) banana-like features appear strongly (yellow bars/suspensions, tyre warmers).

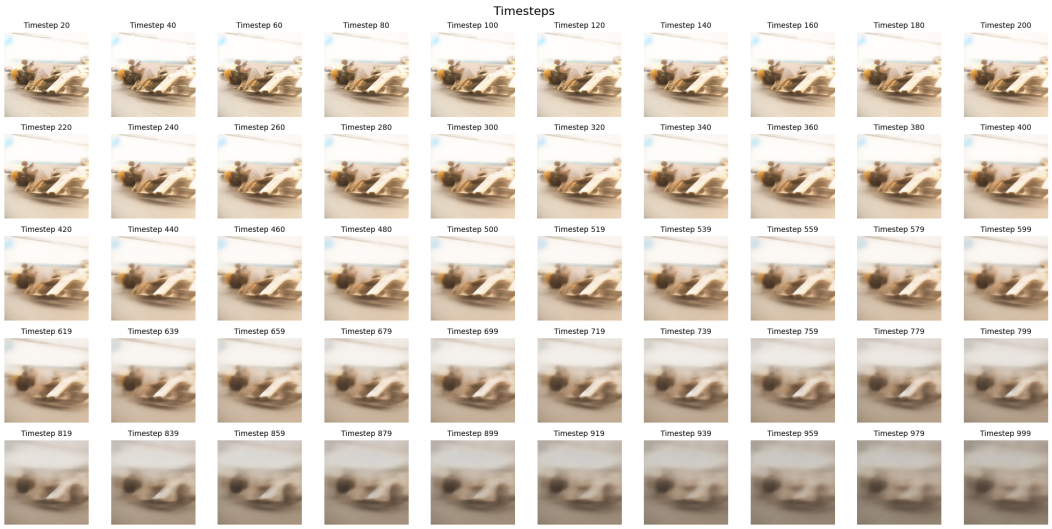


Figure B.10. SVD / principal-direction projection. Steering along the dominant right-singular vector (steer delta $= 200$). The image is strongly yellowed but shows fewer explicit banana shapes — attacking a single principal axis can produce color-dominant shifts rather than detailed object morphologies.

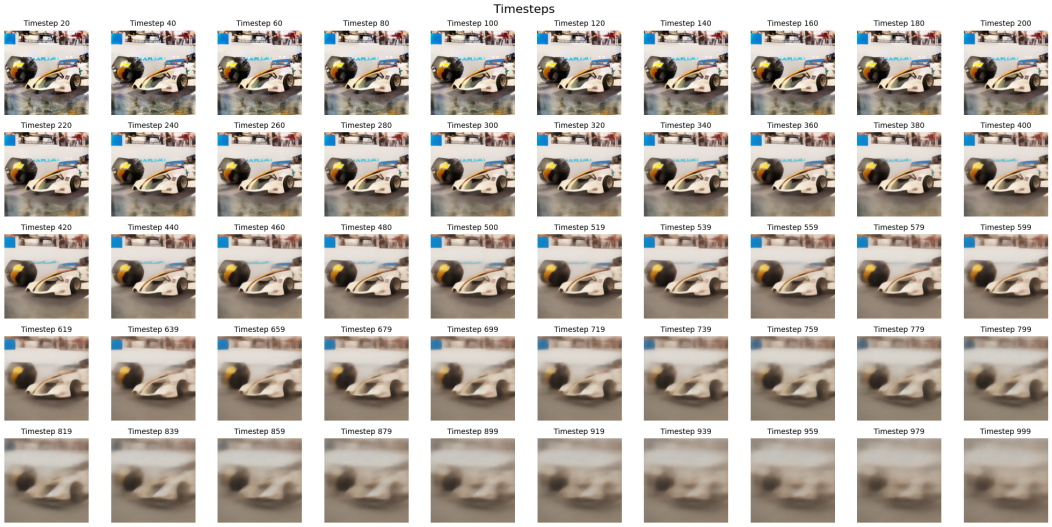


Figure B.11. Shared–residual disentangling. Steering created from the top shared residual directions (steer delta = 7.0). Banana-like cues appear in the car nose and tyre region, producing hybrid / localized attribute changes.