A Unified and Systematic Lens on Diffusion Models

The Principles of Diffusion Models - Chapter 6

Sungwoo Park

Uncertainty Quantification Lab
Seoul National University

August 6, 2026

Overview

Contents

Three origin stories, one recipe

Chapters 2 $\sim$ 5 arrived from different directions.

- **Variational view**: learn reverse kernels by denoising one step at a time
- **Score view**: learn $\nabla_x \log p_t(x)$ and run a reverse-time SDE
- **Flow view**: learn a velocity field and solve a deterministic ODE

The common recipe

Choose a forward path $\{p_t\}_{t=0}^{T}$, learn a time-dependent field on that path, then transport p_{prior} back to p_{data}.

Marginal objectives are what we cannot compute

For a fixed t, each view asks for a different oracle quantity.

$$\textbf{Variational:}\quad \mathbb{E}_{p_t(x_t)}\left[D_{\mathrm{KL}}\left(\underline{p(x_{t-\Delta t} \mid x_t)} \,\|\, p_\phi(x_{t-\Delta t} \mid x_t)\right)\right],$$

$$\textbf{Score:}\quad \mathbb{E}_{p_t(x_t)}\left[\left\| s_\phi(x_t, t) - \underline{\nabla_{x_t} \log p_t(x_t)} \right\|_2^2\right],$$

$$\textbf{Flow:}\quad \mathbb{E}_{p_t(x_t)}\left[\left\| v_\phi(x_t, t) - \underline{v_t(x_t)} \right\|_2^2\right].$$

The reverse kernel, marginal score, and marginal velocity all depend on the unknown data distribution.

Condition on x_0, and every target becomes explicit

Use the Gaussian forward kernel

$$p_t(x_t \mid x_0) = \mathcal{N}(x_t; \alpha_t x_0, \sigma_t^2 I), \qquad x_t = \alpha_t x_0 + \sigma_t \epsilon.$$

Variational: $p(x_{t-\Delta t} \mid x_t, x_0),$

Score: $\nabla_{x_t} \log p_t(x_t \mid x_0) = -\dfrac{\epsilon}{\sigma_t},$

Flow: $v_t(x_t \mid x_0, \epsilon) = \alpha_t' x_0 + \sigma_t' \epsilon.$

What changed

The marginal target is unknown; the conditional target can be generated with each training example.

On finite data, posterior averaging is a finite sum

Replace the data distribution by the empirical measure

$$\widehat{p}_0 = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_0^{(i)}}.$$

Bayes' rule then puts all posterior mass on the training examples:

$$p(x_0 = x_0^{(i)} \mid x_t) = \frac{p_t(x_t \mid x_0^{(i)})}{\sum_{j=1}^{N} p_t(x_t \mid x_0^{(j)})} =: w_i(x_t, t), \qquad \sum_{i=1}^{N} w_i(x_t, t) = 1.$$

Therefore, for any quantity $h(x_0)$,

$$\mathbb{E}_{x_0 \sim p(\cdot \mid x_t)}[h(x_0)] = \sum_{i=1}^{N} w_i(x_t, t) h(x_0^{(i)}).$$

Finite-data form of Equation (6.1.1)

The abstract posterior expectation becomes a concrete weighted average over the training set.

The three finite-data oracles are explicit

Variational: a mixture of Gaussian bridges

$$p^*(x_{t-\Delta t} \mid x_t) = \sum_{i=1}^{N} w_i(x_t, t)\underline{p(x_{t-\Delta t} \mid x_t, x_0^{(i)})}$$

Score: points toward the posterior clean mean

$$s^*(x_t, t) = -\frac{1}{\sigma_t^2}\left(x_t - \alpha_t \sum_{i=1}^{N} w_i(x_t, t)\underline{x_0^{(i)}}\right)$$

Flow: averages explicit conditional velocities

$$v^*(x_t, t) = \sum_{i=1}^{N} w_i(x_t, t)\underline{v_t(x_t|x_0^{(i)})} = \sum_{i=1}^{N} w_i(x_t, t)\underline{\left[\alpha_t' x_0^{(i)} + \frac{\sigma_t'}{\sigma_t}\left(x_t - \alpha_t x_0^{(i)}\right)\right]}$$

One mechanism, three views

Every oracle is a posterior-weighted average of closed-form conditional quantities.

Contents

Four prediction heads, one forward kernel

Chapter 6 organizes the conditional targets as follows.

View	**Head**	**Conditional target**	**Oracle**
Variational	clean x_ϕ	x_0	$\mathbb{E}[x_0 \mid x_t]$
Variational	noise ϵ_ϕ	ϵ	$\mathbb{E}[\epsilon \mid x_t]$
Score-based	score s_ϕ	$-\epsilon/\sigma_t$	$\nabla \log p_t(x_t)$
Flow-based	velocity v_ϕ	$\alpha_t' x_0 + \sigma_t' \epsilon$	$\mathbb{E}[\dot{x}_t \mid x_t]$

All four targets have the same form:

$$A_t x_0 + B_t \epsilon.$$

The general diffusion loss has one template

$$\mathcal{L}(\phi) = \mathbb{E}_{x_0,\epsilon}\mathbb{E}_{t\sim p_{\text{time}}}\left[\underbrace{\omega(t)}_{\text{time weight}}\left\|\underbrace{\text{NN}_\phi(x_t,t)}_{\text{prediction}} - \underbrace{(A_t x_0 + B_t\epsilon)}_{\text{regression target}}\right\|_2^2\right].$$

The apparent variety of diffusion objectives comes from how we fill in this template, not from a different learning principle.

The point of the decomposition

Separate what changes the forward path, what the network predicts, and where training effort is spent.

One loss, four knobs

1. **Noise schedule** (α_t, σ_t) — which marginal path do we use?
2. **Prediction type** (A_t, B_t) — clean, noise, score, or velocity?
3. **Time distribution** $p_{\text{time}}(t)$ — which noise levels are sampled more often?
4. **Time weighting** $\omega(t)$ — which noise levels matter more?

Roadmap

Schedule and head turn out to be largely change-of-variable choices. Weighting and sampling determine the effective emphasis of the objective.

A. The schedule chooses the marginal path

The forward kernel is fixed by two scalar functions: [Lai et al., 2026]

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad p_t(x_t \mid x_0) = \mathcal{N}(x_t; \alpha_t x_0, \sigma_t^2 I).$$

Boundary behavior

$$\begin{aligned} t = 0: &\quad \alpha_0 \simeq 1,\ \sigma_0 \simeq 0, \\ t = T: &\quad \alpha_T \simeq 0,\ x_T \simeq \sigma_T \epsilon. \end{aligned}$$

The schedule determines every intermediate marginal p_t.

Two canonical coordinates

$$\begin{aligned} \text{Linear:} &\quad (\alpha_t, \sigma_t) = (1 - t, t), \\ \text{Trigonometric:} &\quad (\alpha_t, \sigma_t) = (\cos t, \sin t). \end{aligned}$$

Any affine path can be converted to these by time reparameterization and spatial rescaling.

What the schedule changes

The path is theoretically convertible, but its induced loss scaling and finite-step geometry can differ.

B. Prediction heads choose output coordinates

The four targets share one affine form: [Lai et al., 2026]

$$\text{Target}_t = A_t x_0 + B_t \epsilon.$$

Head	A_t	B_t	**Target**
Clean	1	0	x_0
Noise	0	1	ϵ
Conditional score	0	$-1/\sigma_t$	$-\epsilon/\sigma_t$
Conditional velocity	α'_t	σ'_t	$\alpha'_t x_0 + \sigma'_t \epsilon$

Why the choice is not cosmetic

(A_t, B_t) depends on the head and schedule. The outputs convert algebraically (6.3.1), but their MSEs inherit different time weights (6.3.6).

C. The time distribution allocates training visits

Draw $t \sim p_{\text{time}}(t)$; this determines where training computation is spent. [Lai et al., 2026]

Choice	**What it does**
Uniform on $[0, T]$	equal visit rate across time
Log-normal noise level	focus on a selected noise range
Adaptive importance sampling	target high-variance or costly regions

Importance correction can preserve the target integral:

$$\int_0^T r(t)\ell_t \, dt = \mathbb{E}_{t\sim p_{\text{time}}}\left[\frac{r(t)}{p_{\text{time}}(t)}\ell_t\right].$$

Finite-minibatch effect

Even with correction, p_{time} changes visit frequency, gradient variance, and noise-level coverage.

D. The time weight defines the training objective

$\omega(t)$ directly controls how much each noise level contributes. [Lai et al., 2026]

Weighting choice	**Interpretation**
$\omega(t) \equiv 1$	standard equal per-sample MSE weight
$\omega(t) = g^2(t)$	likelihood-weighted score matching / KL upper bound
SNR-based	redistributes emphasis by signal-to-noise level
Monotone in log-SNR	admits an average-ELBO view over noisy data

Adaptive schemes can additionally use observed training behavior to change the emphasis.

Direct effect on optimization

Unless another factor compensates for it, changing ω changes the population objective and which corruption levels the model fits most accurately.

Sampling time and weighting are mathematically coupled

At the population level,

$$\mathcal{L} = \int_0^T \omega(t)\,\mathrm{mse}(t)\,dt = \mathbb{E}_{t\sim p_{\mathrm{time}}}\left[\frac{\omega(t)}{p_{\mathrm{time}}(t)}\,\mathrm{mse}(t)\right].$$

- Changing p_{time} can be undone by importance weighting.
- In minibatch SGD, it still changes gradient variance and noise-level coverage.
- So sampling and weighting are equivalent *as integrals*, not necessarily as optimizers.

Practical consequence

A mathematically unchanged objective can train differently under finite batches and finite compute.

Q&A: Time Sampling vs. Weighting

Question

p_{time}은 gradient variance와 effective weight를 함께 바꾸고, ω는 effective weight만 바꾼다고 보면 될까요?

First distinguish whether **importance correction is applied**. If the target is $\mathcal{L} = \int r(t)\ell_t \, dt$, then

$$t \sim p_{\text{time}}, \qquad \widehat{g}(t) = \frac{r(t)}{p_{\text{time}}(t)} \nabla_\phi \ell_t$$

is unbiased for the same population gradient.

- In this case, p_{time} changes how often each noise level is **visited**.
- The expected gradient stays fixed, but minibatch gradient **variance** changes.

Role of corrected p_{time}

It allocates computation across noise levels while preserving the population objective.

Q&A: Time Sampling vs. Weighting

Question

p_{time}은 gradient variance와 effective weight를 함께 바꾸고, ω는 effective weight만 바꾼다고 보면 될까요?

Choice	Population objective	What changes in SGD
Corrected p_{time}	unchanged	visit rate, gradient variance
Uncorrected p_{time}	effective weight changes	visit rate, variance
Change $\omega(t)$	changes directly	gradient scale, variance

Bottom line

With correction, p_{time} allocates computation; ω directly defines the optimization target. Both affect gradient variance.

Q&A: Why Likelihood Weighting Uses $g^2(t)$

Question

Song et al.의 $\omega(t) = g^2(t)$는 어떤 맥락에서 등장하며, 어떤 직관을 갖나요?

For the forward SDE

$$dx = f(x,t)dt + g(t)dw$$

the reverse-drift error induced by score error $\Delta s = s_\phi - s^*$ is $g^2(t)\Delta s$.
Girsanov path-space energy divides by the diffusion variance $g^2(t)$, so

$$\frac{\left\| g^2(t)\Delta s \right\|_2^2}{g^2(t)} = g^2(t)\,\|\Delta s\|_2^2 .$$

One-line intuition

$g^2(t)$ is not a heuristic for equalizing score magnitude; it is **the coefficient left when reverse-path error is measured in likelihood units**. [Song et al., 2021a]

Q&A: Why Likelihood Weighting Uses $g^2(t)$

Question

Song et al.의 $\omega(t) = g^2(t)$는 어떤 맥락에서 등장하며, 어떤 직관을 갖나요?

This yields the forward-KL upper bound. [Song et al., 2021a]

$$D_{\mathrm{KL}}(p_{\mathrm{data}} \| p_\phi^{\mathrm{SDE}}) \leq \frac{1}{2} \int_0^T g^2(t) \mathbb{E}_{p_t} \left\| s_\phi - s^* \right\|_2^2 dt + D_{\mathrm{KL}}(p_T \| \pi).$$

In noise-prediction coordinates,

$$s_\phi - s^* = -\frac{\epsilon_\phi - \epsilon^*}{\sigma_t}, \qquad \omega_\epsilon(t) = \frac{g^2(t)}{\sigma_t^2}.$$

Changing coordinates changes the visible weight

"g^2 weighting" is the score-coordinate form; in noise coordinates, the same objective appears as g^2/σ_t^2.

Q&A: Score Scale vs. Likelihood Weighting

Question

$g^2(t)$ weighting을 "noise step별 score 크기를 정규화한다"고 이해해도 될까요?

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_t\Big\{\lambda(t)\mathbb{E}_{\mathbf{x}(0)}\mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)}\big[\,\big\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p_{0t}(\mathbf{x}(t)\mid\mathbf{x}(0))\big\|_2^2\,\big]\Big\}. \quad (7)$$

Here $\lambda : [0,T] \to \mathbb{R}_{>0}$ is a positive weighting function, t is uniformly sampled over $[0,T]$, $\mathbf{x}(0) \sim p_0(\mathbf{x})$ and $\mathbf{x}(t) \sim p_{0t}(\mathbf{x}(t)\mid\mathbf{x}(0))$. With sufficient data and model capacity, score matching ensures that the optimal solution to Eq. (7), denoted by $\mathbf{s}_{\boldsymbol{\theta}*}(\mathbf{x},t)$, equals $\nabla_{\mathbf{x}}\log p_t(\mathbf{x})$ for almost all $\mathbf{x}$ and t. As in SMLD and DDPM, we can typically choose $\lambda \propto 1/\mathbb{E}\big[\,\big\|\nabla_{\mathbf{x}(t)}\log p_{0t}(\mathbf{x}(t)\mid\mathbf{x}(0))\big\|_2^2\,\big]$. Note that Eq. (7) uses denoising score matching, but other score matching objectives, such as sliced

Under the Gaussian conditional, [Song et al., 2021b]

$$\nabla_{x_t}\log p(x_t \mid x_0) = -\frac{\epsilon}{\sigma_t}, \qquad \frac{1}{\mathbb{E}\,\|\nabla\log p(x_t \mid x_0)\|_2^2} = \frac{\sigma_t^2}{D}.$$

Thus inverse score-norm weighting compensates for the large target scale at small σ_t.

Q&A: Score Scale vs. Likelihood Weighting

Question

$g^2(t)$ weighting을 "noise step별 score 크기를 정규화한다"고 이해해도 될까요?

The two weights start from different principles.

Weight	Motivation	Gaussian form
Scale normalization	inverse score norm	σ_t^2/D
Likelihood weighting	path-space KL bound	$g^2(t)$

They can be proportional for some VE schedules, but generally differ for VP because $g^2(t) = \beta(t)$. [Song et al., 2021a, Song et al., 2021b]

Keep the two intuitions separate

σ_t^2/D equalizes target scale; $g^2(t)$ measures path-space likelihood error.

Q&A: Score Scale vs. Likelihood Weighting

Question

$g^2(t)$ weighting을 "noise step별 score 크기를 정규화한다"고 이해해도 될까요?

> Option 2: Choose $\lambda(t) \to 0$ as $t \to 0$ so that $\lambda(t)/\sigma_t^2$ does not blow up. This makes the mean well-behaved, but the variance of the stochastic gradients may still blow up as $t \to 0$.

At $\sigma_0 = 0$, the effective noise-prediction weight $\lambda(t)/\sigma_t^2$ can diverge. Common remedies are: [Song et al., 2021b]

- Start integration at a small $\delta > 0$.
- Set $\lambda(t) = O(\sigma_t^2)$ or use importance sampling.

Caution

The numerator $\lambda(t)$ of the effective weight must vanish here. The statement $g(0) = 0$ does not hold for every SDE.

Monotone weighting recovers a variational picture

Let $\lambda_t = \log(\alpha_t^2/\sigma_t^2)$ be the log-SNR and define the remaining reverse-process cost [Kingma and Gao, 2023]

$$\mathcal{V}(t; x_0) = D_{\mathrm{KL}}(p(x_{t:1} \mid x_0) \,\|\, p_\phi(x_{t:1}))\,.$$

The per-level denoising error is the rate of change of this global cost:

$$\frac{d}{dt}\mathcal{V}(t; x_0) = \frac{1}{2}\frac{d\lambda_t}{dt}\mathbb{E}_\epsilon \left\|\epsilon_\phi(x_t, t) - \epsilon\right\|_2^2.$$

Integration by parts turns a monotone $\omega(\lambda_t)$ into

$$\mathcal{L}_\omega(x_0) = \mathbb{E}_{t\sim p_\omega}[\mathcal{V}(t; x_0)] + C.$$

Interpretation

The weight chooses a distribution over Gaussian augmentation strengths — an average ELBO view.

Q&A: Does Monotone Weighting Help?

Question

Monotone weighting은 실제로 다른 weighting보다 이점이 있나요?

In an ImageNet-64 comparison using the same ϵ-prediction head and EDM sampler, monotone weights yielded an empirical gain. [Kingma and Gao, 2023]

ϵ-prediction + EDM sampler	Monotone?	FID ↓
$\mathrm{sech}(\lambda/2)$ baseline	no	1.55
$\mathrm{sigmoid}(-\lambda + 2)$	yes	1.46
adaptive + EDM-monotone	yes	1.44

In this setup, monotone weights achieved lower FID (Frechet Inception Distance) than the non-monotone baseline.

Q&A: Does Monotone Weighting Help?

Question

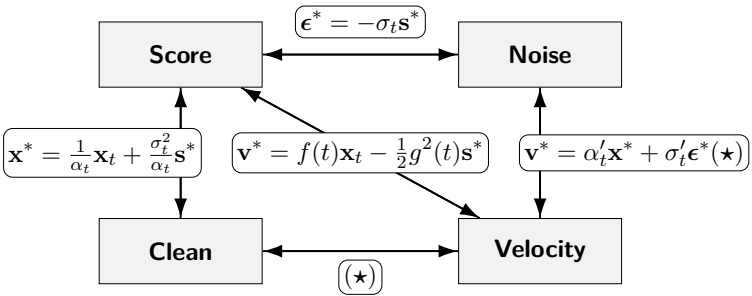
Monotone weighting은 실제로 다른 weighting보다 이점이 있나요?

Even within the same study, results depended on the exact weight shape.
[Kingma and Gao, 2023]

- For reproduced EDM, both non-monotone and EDM-monotone reached FID 1.43.
- Even the monotone sigmoid ranged from FID 1.68 to 1.94 as the shift changed.
- The theory provides **a noisy-data ELBO interpretation, not a performance-dominance theorem**.

Contents

The four parameterizations are one network head



At each non-degenerate t, clean, noise, score, and velocity predictions are related by deterministic algebraic conversions.

Proof sketch: why the four heads agree

For fixed t with $\omega(t) > 0$, each weighted objective is an ordinary MSE, whose unique minimizer is a conditional mean.

$$h^*(x_t, t) = \mathbb{E}[Y_t \mid x_t].$$

Applying this identity to the four targets gives

$$\begin{aligned} s^*(x_t, t) &= \mathbb{E}\left[-\frac{\epsilon}{\sigma_t}\middle| x_t\right] = -\frac{1}{\sigma_t}\epsilon^*(x_t, t), \\ x^*(x_t, t) &= \mathbb{E}[x_0 \mid x_t], \qquad \alpha_t x^* = x_t + \sigma_t^2 s^*, \\ v^*(x_t, t) &= \mathbb{E}[\alpha_t' x_0 + \sigma_t' \epsilon \mid x_t] = \alpha_t' x^* + \sigma_t' \epsilon^*. \end{aligned}$$

Substitution, together with Lemma 4.5.1, yields

$$v^* = \frac{\alpha_t'}{\alpha_t} x_t + \left(\frac{\alpha_t' \sigma_t^2}{\alpha_t} - \sigma_t \sigma_t'\right) s^* = f(t) x_t - \frac{1}{2} g^2(t) s^*.$$

Therefore, at every non-degenerate t, the four optimal heads are one-to-one coordinates of the same oracle.

In practice, train one head and derive the rest

Suppose the network predicts noise $\epsilon_\phi(x_t, t)$. Then

$$s_\phi(x_t, t) = -\frac{1}{\sigma_t}\epsilon_\phi(x_t, t),$$

$$x_\phi(x_t, t) = \frac{x_t - \sigma_t \epsilon_\phi(x_t, t)}{\alpha_t},$$

$$v_\phi(x_t, t) = \alpha_t' x_\phi(x_t, t) + \sigma_t' \epsilon_\phi(x_t, t).$$

At the oracle optimum,

$$v^*(x_t, t) = f(t)x_t - \frac{1}{2}g^2(t)s^*(x_t, t).$$

Not four separate models

The prediction type selects the network's output coordinates; the underlying oracle field is shared.

PF-ODE does not care which head we chose

The same probability-flow ODE can be written in four equivalent forms

$$\begin{aligned}
\dot{x} &= \frac{\alpha_t'}{\alpha_t} x - \sigma_t \left(\frac{\alpha_t'}{\alpha_t} - \frac{\sigma_t'}{\sigma_t} \right) \epsilon^*(x, t) \\
&= \frac{\sigma_t'}{\sigma_t} x + \alpha_t \left(\frac{\alpha_t'}{\alpha_t} - \frac{\sigma_t'}{\sigma_t} \right) x^*(x, t) \\
&= \frac{\alpha_t'}{\alpha_t} x + \sigma_t^2 \left(\frac{\alpha_t'}{\alpha_t} - \frac{\sigma_t'}{\sigma_t} \right) s^*(x, t) \\
&= \alpha_t' x^*(x, t) + \sigma_t' \epsilon^*(x, t) = v^*(x, t).
\end{aligned}$$

With $f(t) = \alpha_t'/\alpha_t$, this is the familiar

$$\dot{x} = f(t)x - \frac{1}{2} g^2(t) s^*(x, t).$$

Derivation sketch: the PF-ODE forms are substitutions

At a non-degenerate t, conditional expectation preserves the affine identity.

$$x_t = \alpha_t x^* + \sigma_t \epsilon^*, \qquad \dot{x} = v^* = \alpha_t' x^* + \sigma_t' \epsilon^*.$$

Eliminate one oracle at a time:

$$\begin{aligned} v^* &= \frac{\alpha_t'}{\alpha_t} x_t - \sigma_t \left(\frac{\alpha_t'}{\alpha_t} - \frac{\sigma_t'}{\sigma_t} \right) \epsilon^*, \\ &= \frac{\sigma_t'}{\sigma_t} x_t + \alpha_t \left(\frac{\alpha_t'}{\alpha_t} - \frac{\sigma_t'}{\sigma_t} \right) x^*. \end{aligned}$$

Finally, $\epsilon^* = -\sigma_t s^*$ converts the first line into the score form, while the defining equation above is already the velocity form.

What this proves

Equation (6.3.2) introduces no new dynamics; it only rewrites one vector field in four coordinates.

Every affine schedule is a warped canonical path

Canonical FM path

$$x_\tau^{\mathrm{FM}} = (1-\tau)x_0 + \tau\epsilon.$$

For $c(t) = \alpha_t + \sigma_t \neq 0$ and $\tau(t) = \sigma_t / c(t)$,

$$x_t = \alpha_t x_0 + \sigma_t \epsilon = c(t) x_{\tau(t)}^{\mathrm{FM}}.$$

Time is reparameterized; space is rescaled.

Trigonometric path

$$x_u^{\mathrm{Trig}} = \cos u\, x_0 + \sin u\, \epsilon.$$

With $R_t = \sqrt{\alpha_t^2 + \sigma_t^2}$,

$$\cos\tau_t = \frac{\alpha_t}{R_t}, \quad \sin\tau_t = \frac{\sigma_t}{R_t}, \quad x_t = R_t x_{\tau_t}^{\mathrm{Trig}}.$$

Radius gives scale; angle gives time.

Derivation sketch: every affine path is a warped FM path

Define $c(t) = \alpha_t + \sigma_t$ and $\tau(t) = \sigma_t / c(t)$, with $c(t) \neq 0$. Then

$$\begin{aligned} c(t) x^{\mathrm{FM}}_{\tau(t)} &= c(t)\big[(1-\tau(t))x_0 + \tau(t)\epsilon\big] \\ &= \alpha_t x_0 + \sigma_t \epsilon = x_t. \end{aligned}$$

Differentiating the same identity gives the velocity conversion:

$$\dot{x}_t = c'(t) x^{\mathrm{FM}}_{\tau(t)} + c(t)\tau'(t) v^{\mathrm{FM}}\Big(x^{\mathrm{FM}}_{\tau(t)}, \tau(t)\Big).$$

The trigonometric form follows from the polar decomposition of (α_t, σ_t) into a radius and an angle.

What this proves

Pointwise, the state and velocity follow from the map $t \mapsto \tau(t)$ and a scalar spatial rescaling.

The losses differ only by time-dependent weights

Under the corresponding head conversions,

$$\begin{aligned}\|s_\phi - s^*\|_2^2 &= \frac{1}{\sigma_t^2}\,\|\epsilon_\phi - \epsilon^*\|_2^2 \\ &= \frac{\alpha_t^2}{\sigma_t^4}\,\|x_\phi - x^*\|_2^2 \\ &= \left[\frac{\alpha_t}{\sigma_t(\alpha_t'\sigma_t - \alpha_t\sigma_t')}\right]^2 \|v_\phi - v^*\|_2^2 .\end{aligned}$$

The hidden qualification

The objectives are gradient-equivalent only after correcting for the induced time weights.

Conditions: $\sigma_t > 0$ and $\alpha_t'\sigma_t - \alpha_t\sigma_t' \neq 0$.

Why linear v-prediction is natural

For $\alpha_t = 1 - t$ and $\sigma_t = t$, [Liu, 2022, Lipman et al., 2023]

$$v_t(x_t \mid x_0, \epsilon) = \epsilon - x_0.$$

1. **Constant target scale**

$$\mathbb{E}\,\|\epsilon - x_0\|_2^2 = D + \operatorname{Tr}\operatorname{Cov}[x_0] + \|\mathbb{E}x_0\|_2^2,$$

   which does not depend on t.
2. **No schedule curvature**: $\alpha_t'' = \sigma_t'' = 0$.
3. **Direct ODE field**: $\dot{x} = v_\phi(x, t)$ has no explicit semilinear split.

Conditional and oracle energy prefer different schedules

Let $\alpha_t = 1 - \rho(t)$ and $\sigma_t = \rho(t)$. Then

$$v_t(x_t \mid x_0, \epsilon) = \rho'(t)(\epsilon - x_0).$$

Conditional kinetic energy

$$\mathcal{K}[\rho] = C \int_0^1 \rho'(t)^2 \, dt, \quad C = \mathbb{E} \, \|\epsilon - x_0\|_2^2 .$$

With $\rho(0) = 0$ and $\rho(1) = 1$,

$$\rho''(t) = 0 \quad \Longrightarrow \quad \rho(t) = t.$$

The canonical path is the least-energy affine interpolation.

Marginalized oracle energy

$$\mathcal{K}_{\text{oracle}}[\rho] = \int_0^1 \rho'(t)^2 \kappa(t) \, dt,$$

$$\kappa(t) = \mathbb{E} \, \|\mathbb{E}[\epsilon - x_0 \mid x_t]\|_2^2 .$$

$$(\kappa(t)\rho'(t))' = 0 \quad \Longrightarrow \quad \rho'(t) \propto \frac{1}{\kappa(t)}.$$

The oracle-optimal clock is generally nonlinear.

Section 6.3.4 caveat [Lai et al., 2026]

Linear time minimizes conditional energy; it minimizes oracle energy only if $\kappa(t)$ is constant.

Why it is not automatically the best choice

The previous advantages describe an idealized target and schedule, not end-to-end performance.

- The marginalized oracle velocity can still be nonlinear in time.
- Model error interacts with normalization, architecture, and guidance.
- Sampling quality depends on the solver, step schedule, and function-evaluation budget.
- Alternative heads may align better with a particular solver or data scaling.

Conclusion

Linear v-prediction is theoretically clean. Whether it wins is an empirical question.

Q&A: Prediction Heads and Solvers

Question

Fast ODE sampling에는 ϵ or x, few-NFE training에는 x or v가 유리하다고 보는 이유는 무엇인가요?

Advanced ODE solvers decompose the PF-ODE as

$$\dot{x} = L(t)x + N_\phi(x, t)$$

They treat $L(t)x$ analytically and approximate only the integral containing the model output.

- DPM-Solver exploits this semilinear structure in noise-prediction coordinates.
- DPM-Solver++ uses the same idea in data-prediction coordinates.

What matters for fast sampling

When the structure exposed by the head matches what the solver handles analytically, discretization error can decrease at the same NFE. [Lu et al., 2022]

Q&A: Prediction Heads and Solvers

Question

Fast ODE sampling에는 ϵ or x, few-NFE training에는 x or v가 유리하다고 보는 이유는 무엇인가요?

For few-step generation, conditioning of the endpoint conversion is key. [Salimans and Ho, 2022]

1. At low SNR, noise error in
$$\widehat{x} = \frac{x_t - \sigma_t \widehat{\epsilon}}{\alpha_t}$$
is amplified by the factor $1/\alpha_t$.
2. x-prediction directly predicts the endpoint.
3. Under VP, v-prediction uses $\widehat{x} = \alpha_t x_t - \sigma_t \widehat{v}$, whose coefficients are bounded.

Conclusion

With plain Euler/Heun, v is convenient because it directly outputs the full PF-ODE field. This does not make v more accurate with every solver.

Q&A: Parameterization and Performance

Question

Parameterization에 따른 실제 성능 차이를 분석한 연구가 있나요?

Salimans–Ho: CIFAR-10 comparison [Salimans and Ho, 2022]

- Performance was broadly similar across configurations that trained stably.
- v was the most stable, while x was slightly better on the best metric.
- The ϵ+truncated-SNR configuration diverged.

Kingma–Gao: ImageNet-64 comparison [Kingma and Gao, 2023]

$$\text{FID}: \quad \epsilon\ 1.44 \quad / \quad \text{EDM denoiser}\ 1.43 \quad / \quad v\ 1.45$$

Takeaway so far

Among stable configurations, results are effectively close to a tie; interactions between head and weighting are substantial.

Q&A: Parameterization and Performance

Question

Parameterization에 따른 실제 성능 차이를 분석한 연구가 있나요?

DPM-Solver-v3 uses a model-specific sampling parameterization instead of fixed noise/data coordinates. [Zheng et al., 2023]

ScoreSDE, 10 NFE	DPM-Solver++	DPM-Solver-v3
FID ↓	4.01	**3.40**

Even for the same model, rankings can change with the sampling coordinates and solver.

Ablation pitfall

Head conversion itself changes the time weights. Meaningful head ablations must control for schedule, weighting, preconditioning, and solver.

Q&A: Which Head Should We Use?

Question

대수적으로 교환 가능하다면, 실무에서는 어떤 prediction head를 주로 사용하나요?

Head	Common setting	Main reason
ϵ / **score**	DDPM, Score SDE	legacy checkpoint and sampler ecosystem [Ho et al., 2020, Song et al., 2021b]
x / **denoiser**	EDM, DPM-Solver++, consistency	low-SNR behavior and thresholding [Karras et al., 2022, Lu et al., 2022]

In practice, the head is often chosen to match the existing model and sampler ecosystem.

Q&A: Which Head Should We Use?

Question

대수적으로 교환 가능하다면, 실무에서는 어떤 prediction head를 주로 사용하나요?

v/velocity is common in VP distillation [Salimans and Ho, 2022] and flow matching [Lipman et al., 2023]. Its advantages are bounded conversion and a direct vector field. Practitioners do not choose the head in isolation.

1. **Existing checkpoint**: keep the trained head and original sampler when possible.
2. **Few-step or low-SNR endpoint**: x and v amplify conversion error less.
3. **Solver structure**: exponential integrators exploit an x/ϵ split; plain Euler/Heun readily uses a direct v field.

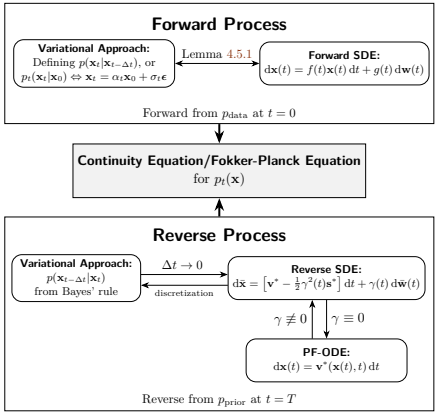
Short answer

For a new model, choose the head jointly with schedule–weight–solver. Which combination is well-conditioned matters more than which head is universally best.

Contents

Fokker–Planck is the common backbone

All three views share one forward and one reverse density equation.



One density path admits many dynamics

The flow view starts from the ODE

$$\frac{dx(t)}{dt} = v^*(x(t), t), \qquad x(T) \sim p_T, \quad t : T \to 0,$$

whose density obeys the continuity equation

$$\partial_t p_t = -\nabla \cdot (v^* p_t).$$

Question

Can we add stochasticity while keeping every one-time marginal p_t unchanged?

The answer is yes: there is an entire family of reverse-time SDEs.

A reverse-SDE family shares the ODE marginals

For any $\gamma(t) \geq 0$, integrate backward from T to 0. [Ma et al., 2024]

$$d\bar{x}(t) = \left[v^*(\bar{x}(t), t) - \frac{1}{2}\gamma^2(t) s^*(\bar{x}(t), t) \right] dt + \gamma(t) d\bar{w}(t).$$

Here $s^*(x, t) = \nabla_x \log p_t(x)$ and

$$s^*(x, t) = \frac{1}{\sigma_t} \frac{\alpha_t v^*(x, t) - \alpha_t' x}{\alpha_t' \sigma_t - \alpha_t \sigma_t'}.$$

Proposition 6.4.1

Every choice of γ above produces the same prescribed marginal path $\{p_t\}$ as the PF-ODE.

Proof sketch: why the SDE keeps the same marginals

The reverse-time Fokker–Planck equation is

$$\begin{aligned}\partial_t p_t &= -\nabla\cdot\left[\left(v^* - \frac{1}{2}\gamma^2 s^*\right) p_t\right] - \frac{1}{2}\gamma^2 \Delta p_t \\ &= -\nabla\cdot(v^* p_t) + \frac{1}{2}\gamma^2 \underbrace{\nabla\cdot(s^* p_t)}_{\Delta p_t} - \frac{1}{2}\gamma^2 \Delta p_t \\ &= -\nabla\cdot(v^* p_t).\end{aligned}$$

The key identity is $s^* p_t = \nabla p_t$.

What is shared

The ODE and SDE have different sample paths, but the same one-time marginal densities.

$\gamma(t)$ is a stochasticity dial, not a new path

Choice	Dynamics	Marginals
$\gamma(t) = 0$	deterministic PF-ODE	p_t
$\gamma(t) = g(t)$	Chapter 4 reverse Score SDE	p_t
other $\gamma(t) \geq 0$	intermediate stochastic process	p_t

- γ can be selected after training when the oracle fields are exact.
- With learned fields and numerical solvers, different γ can produce different errors and sample quality.

The unification

Variational kernels, reverse SDEs, and PF-ODEs are different dynamics constrained by one density evolution.

Q&A: KL-Optimal $\gamma(t)$

Question

Data distribution과 model distribution의 KL gap을 줄이는 $\gamma(t)$는 어떻게 정해지나요?

Let $w_t := \gamma^2(t)$, and let $\mathcal{L}_t$ denote the learned-velocity error at time t. Then the following forward-KL upper bound holds. [Ma et al., 2024]

$$D_{\mathrm{KL}}(p_{\mathrm{data}} \| p_{\theta,0}) \leq \frac{1}{2} \int_0^1 \frac{\mathcal{L}_t}{w_t} \left(1 + \frac{w_t}{2\kappa_t \sigma_t}\right)^2 dt, \qquad \kappa_t := \dot{\sigma}_t - \frac{\dot{\alpha}_t}{\alpha_t}\sigma_t.$$

First distinction

This does not minimize the actual KL directly; it minimizes a **KL upper bound** expressed in terms of learned-velocity error.

Q&A: KL-Optimal $\gamma(t)$

Question

Data distribution과 model distribution의 KL gap을 줄이는 $\gamma(t)$는 어떻게 정해지나요?

At fixed t, the w_t-dependent part is

$$\frac{\mathcal{L}_t}{2}\left[\frac{1}{w_t}+\frac{w_t}{4(\kappa_t\sigma_t)^2}\right]$$

The two terms grow at opposite ends; balancing them gives [Ma et al., 2024]

$$\boxed{\gamma^2_{\mathrm{KL}}(t)=w_t^{\mathrm{KL}}=2\kappa_t\sigma_t=2\left(\dot\sigma_t\sigma_t-\frac{\dot\alpha_t\sigma_t^2}{\alpha_t}\right)}.$$

Intuition

The bound grows when diffusion is too small or large enough to amplify score-conversion error.

Q&A: KL-Optimal $\gamma(t)$

Question

Data distribution과 model distribution의 KL gap을 줄이는 $\gamma(t)$는 어떻게 정해지나요?

For representative schedules, [Ma et al., 2024]

Schedule	$w_t^{\mathrm{KL}} = \gamma_{\mathrm{KL}}^2(t)$	Practical note
VP	β_t	standard reverse-SDE scale
Linear $(\alpha_t, \sigma_t) = (1-t, t)$	$2t/(1-t)$	regularize near $t \to 1$

Precise statement

With oracle fields, every γ produces the same marginals. This choice reduces a **KL upper bound in the presence of model error**; it is not the unique minimizer of the actual KL.

Contents

Takeaways

1. **Conditioning is the tractability trick.** Regress against a computable conditional target; its conditional mean recovers the marginal oracle.
2. **Diffusion losses have one common template.** Schedule, prediction head, time weighting, and time sampling are the visible design choices.
3. **Heads and affine schedules are theoretically convertible.** Their losses match after schedule-dependent weighting, but optimization and numerics can still differ.
4. **Fokker–Planck is the unifying law.** Variational, score-based, and flow-based models all implement the same marginal density transport.

References I

Ho, J., Jain, A., and Abbeel, P. (2020).
Denoising diffusion probabilistic models.
In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851.

Karras, T., Aittala, M., Aila, T., and Laine, S. (2022).
Elucidating the design space of diffusion-based generative models.
In *Advances in Neural Information Processing Systems*, volume 35, pages 26565–26577.

Kingma, D., Salimans, T., Poole, B., and Ho, J. (2021).
Variational diffusion models.
In *Advances in Neural Information Processing Systems*, volume 34, pages 21696–21707.

Kingma, D. P. and Gao, R. (2023).
Understanding diffusion objectives as the ELBO with simple data augmentation.
In *Advances in Neural Information Processing Systems*.

References II

Lai, C.-H., Song, Y., Kim, D., Mitsufuji, Y., and Ermon, S. (2026).
The principles of diffusion models.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. (2023).
Flow matching for generative modeling.
In *International Conference on Learning Representations*.

Liu, Q. (2022).
Rectified flow: A marginal preserving approach to optimal transport.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. (2022).
DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models.

Ma, N., Goldstein, M., Albergo, M. S., Boffi, N. M., Vanden-Eijnden, E., and Xie, S. (2024).
SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers.
In *European Conference on Computer Vision*, pages 23–40.

References III

Salimans, T. and Ho, J. (2022).
Progressive distillation for fast sampling of diffusion models.
In *International Conference on Learning Representations*.

Song, Y., Durkan, C., Murray, I., and Ermon, S. (2021a).
Maximum likelihood training of score-based diffusion models.
In *Advances in Neural Information Processing Systems*, volume 34, pages 1415–1428.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021b).
Score-based generative modeling through stochastic differential equations.
In *International Conference on Learning Representations*.

Zheng, K., Lu, C., Chen, J., and Zhu, J. (2023).
DPM-Solver-v3: Improved diffusion ODE solver with empirical model statistics.
In *Advances in Neural Information Processing Systems*.

Thank You