Faster Sampling: Distillation

Lecturer: Qiang Sun | Lecture 09

Contents

1 Faster Sampling: Distillation

1.1 Prologue: why distillation?

Sampling from flow and diffusion models is computationally expensive because it typically requires *iteratively* transporting noise into data. Even with modern training-free ODE solvers from previous lectures, high-quality sampling often requires 10–20 *number of function evaluations (*NFE*)* for ODE-based samplers, and significantly more for naive discretizations. For instance, the original DDPM requires 1000 steps.

Distillation is a training-based alternative: we train a new *student* model that can generate samples with far fewer steps. Unlike training-free solvers, distillation does *not* keep the pretrained teacher model fixed at inference time; instead, the teacher provides a training signal that teaches the student a faster generation procedure.

Notation convention. Throughout, we write the forward process in general form

$$\mathrm{d}x_t = f_t(x_t)\mathrm{d}t + g_t \mathrm{d}W_t.$$

Whenever we use closed-form (α_t, β_t) Gaussian noising identities, we are specializing to the standard linear-Gaussian (VP/DDPM-style) case.

The reconstruction/x_0 perspective: Denoising and the score. A diffusion model can be viewed as a family of denoisers. Under the linear-Gaussian specialization of the forward process, the perturbation kernel is

$$x_t \mid x_0 \sim \mathcal{N}\big(\alpha_t x_0,\, \beta_t^2 I_d\big), \qquad 0 \le t \le 1, \tag{1.1}$$

the conditional score is

$$\nabla_x \log p_t(x \mid x_0) = -(x - \alpha_t x_0)/\beta_t^2.$$

Taking conditional expectations with respect to $p_t(x_0 \mid x)$ on both sides yields *Tweedie's formula*, which relates the *posterior mean denoiser* to the marginal score $\nabla_x \log p_t(x)$:

$$x_0^*(x,t) \triangleq \mathbb{E}[X_0 \mid X_t = x] = \frac{1}{\alpha_t}\Big(x + \beta_t^2\, \nabla_x \log p_t(x)\Big). \tag{1.2}$$

Thus, a score model $s_\theta(x,t) \approx \nabla_x \log p_t(x)$ can be converted into an x_0-prediction denoiser $x_{0,\theta}(x,t)$, and vice versa. In practice, we often parameterize the model to predict the noise ε or the data x_0. For example, the ε-prediction model $\varepsilon_\theta(x,t)$ is related to the score by $s_\theta(x,t) = -\varepsilon_\theta(x,t)/\beta_t$. Substituting this into (1.2) gives the reconstruction formula:

$$x_{0,\theta}(x,t) = \frac{x - \beta_t \varepsilon_\theta(x,t)}{\alpha_t}. \tag{1.3}$$

This perspective is crucial for distillation: if we can learn to predict the clean data x_0 in fewer steps, we can generate samples much faster.

Two distillation paradigms. We group existing approaches into two broad categories:

1. **Distribution-level distillation.** Train a generator G_θ with output distribution p_0^θ so that

   $$p_0^\theta \approx p_{\text{data}}. \tag{1.4}$$

   In practice, we often do not have an explicit objective that compares p_0^θ to p_{data} directly. Instead, we use a pretrained diffusion teacher distribution $p_0^{\phi^\star}$ and solve

   $$\min_\theta \; D\big(p_0^\theta \,\|\, p_0^{\phi^\star}\big), \tag{1.5}$$

   where D is some divergence (kullback-Leibler, an f-divergence, etc.). Examples include diffusion model distillation (DMD) (Yin et al., 2024), variational score distillation (VSD) (Wang et al., 2023), and score identity distillation (SiD) (Zhou et al., 2024).

2. **Flow-map-level distillation.** Here, we view the sampling process as a composition of implicit transport maps between probability distributions at different times. The goal is to train a student model to learn and approximate these maps on a coarser, more efficient time grid. Key examples include knowledge distillation (KD) (Luhman and Luhman, 2021) and progressive distillation (PD) (Salimans and Ho, 2022).

Chronology. Flow-map-level distillation (KD/PD) emerged earlier (around 2021), whereas distribution-level approaches gained prominence more recently (around 2023). For pedagogical clarity, we present distribution-level distillation first in Section 1.3, but first require a brief discussion of flow maps and solvers to establish the necessary notation.

1.2 A unifying view: learning the probability-flow map

We use the same continuous-time convention as in Lecture 08: $t = 0$ is clean data and $t = 1$ is pure noise. Let p_t denote the marginal density of x_t. The probability-flow ODE (PF-ODE) associated with a forward SDE $\mathrm{d}x_t = f_t(x_t)\mathrm{d}t + g_t \mathrm{d}W_t$ is

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = f_t\big(x_t\big) \; - \; \frac{1}{2} g_t^2 \, \nabla_x \log p_t\big(x_t\big) \; \triangleq \; v^\star\big(x_t, t\big). \tag{1.6}$$

We call $v^\star$ the *oracle velocity field.* For any $s \geq t$, let $\Psi_{s\to t}$ denote the oracle flow map of (1.6):

$$\Psi_{s\to t}(x_s) \triangleq x_s + \int_s^t v^\star\big(x_\tau, \tau\big)\,\mathrm{d}\tau, \qquad x_\tau|_{\tau=s} = x_s. \tag{1.7}$$

In ideal sampling, we draw $x_1 \sim p_{\text{init}}$ and map it to data via $x_0 = \Psi_{1\to 0}(x_1)$.

1.2.1 Teacher solvers as surrogates for the oracle flow map

In practice $\Psi_{s\to t}$ is not available in closed form. Instead, we approximate it using a numerical solver driven by a pretrained teacher model. We denote such a teacher-induced transition map by

$$\mathrm{Solver}_{s\to t}(x_s;\, \phi^\star), \tag{1.8}$$

where $\phi^\star$ denotes teacher parameters (e.g., a pretrained diffusion model). The solver (Euler, DDIM, DPM-Solver, ...) determines the map.

Question 1 (Distilling the flow map)

Can we leverage intermediate steps produced by a slow teacher solver, $\mathrm{Solver}_{s\to t}(\cdot;\phi^\star)$, to directly learn the oracle map $\Psi_{s\to t}$? In particular, if we can learn a *single* map $G_\theta(\cdot, 1, 0) \approx \Psi_{1\to 0}$, we obtain a one-step generator.

Knowledge distillation (KD) as a first attempt. A standard idea (Luhman and Luhman, 2021) is to train a one-step generator G_θ to mimic a *long* teacher rollout. Let $\mathrm{Solver}^{\circ}_{1\to 0}(x_1; \phi^\star)$ denote the composition of many small teacher steps from $t = 1$ to $t = 0$. KD then solves

$$\min_\theta \ \mathcal{L}_{\mathrm{KD}}(\theta) \triangleq \mathbb{E}_{x_1 \sim p_{\mathrm{init}}}\left[\|G_\theta(x_1) - \mathrm{Solver}^{\circ}_{1\to 0}(x_1; \phi^\star)\|_2^2\right]. \tag{1.9}$$

This yields a fast sampler at inference time, but training requires expensive teacher rollouts and can be unstable.

A unified oracle objective. Many flow-map-level distillation methods can be phrased as trying to match the oracle flow map Ψ. For a family of student maps $G_\theta(\cdot, s, t)$, consider the ideal objective

$$\mathcal{L}_{\mathrm{oracle}}(\theta) \triangleq \mathbb{E}_{(s,t)\sim\pi}\mathbb{E}_{X_s \sim p_s}\Big[w(s,t)\, d\big(G_\theta(X_s, s, t), \Psi_{s\to t}(X_s)\big)\Big], \tag{1.10}$$

where π is a distribution over time pairs (s, t), $w(s, t) \geq 0$ is a weight, and d is a distance (often squared Euclidean).

Observation 2 (KD as a special case)

If π concentrates on $(s, t) = (1, 0)$ and $p_s = p_1 = p_{\mathrm{init}}$, then (1.10) reduces to regressing G_θ to $\Psi_{1\to 0}$. Replacing $\Psi_{1\to 0}$ by a teacher rollout $\mathrm{Solver}^{\circ}_{1\to 0}$ yields the KD loss (1.9).

Progressive distillation (PD) (Salimans and Ho, 2022) also fits (1.10) but uses many *local* time pairs (s, t) and enforces consistency between short transitions. We return to this view in (1.37).

1.3 Distribution-level distillation via score differences

1.3.1 Setup

Let $G_\theta(z)$ be a generator that maps $z \sim p_{\mathrm{init}}$ such as $\mathcal{N}(0, I_d)$ to a sample $\widehat{x}_0 \triangleq G_\theta(z)$. Denote the induced distribution by $\widehat{x}_0 \sim p_0^\theta$.

Let p_t be the *teacher* diffusion model marginal at time t, and let p_t^θ be the marginal obtained by forward noising the student samples (under the same linear-Gaussian specialization):

$$\widehat{x}_t \mid \widehat{x}_0 \sim \mathcal{N}(\alpha_t \widehat{x}_0,\, \beta_t^2 I_d), \qquad \widehat{x}_0 \sim p_0^\theta, \qquad \widehat{x}_t \sim p_t^\theta. \tag{1.11}$$

Equivalently,

$$z \sim p_{\mathrm{init}},\ \varepsilon \sim \mathcal{N}(0, I_d), \qquad \widehat{x}_t = \alpha_t G_\theta(z) + \beta_t \varepsilon. \tag{1.12}$$

1.3.2 Variational score distillation

Variational score distillation (VSD) matches teacher and student *noised* distributions across time by minimizing a weighted Kullback-Leibler (KL) divergence:

$$\mathcal{L}_{\mathrm{VSD}}(\theta) \triangleq \mathbb{E}_{t\sim p(t)}\Big[\omega(t)\, D_{\mathrm{KL}}\big(p_t^\theta \,\|\, p_t\big)\Big], \tag{1.13}$$

where $p(t)$ is a time-sampling distribution, often uniform on $[0, 1]$, and $\omega(t) \geq 0$ is a weighting function. Expanding the KL gives

$$\mathcal{L}_{\mathrm{VSD}}(\theta) = \mathbb{E}_{t\sim p(t)}\mathbb{E}_{\widehat{x}_t\sim p_t^\theta}\big[\omega(t)\left(\log p_t^\theta(\widehat{x}_t) - \log p_t(\widehat{x}_t)\right)\big]. \tag{1.14}$$

At the population optimum, $p_t^\theta = p_t$ for all t, and in particular p_0^θ matches the teacher's data distribution.

1.3.3 Gradient as a score-difference signal

A key feature of KL is that its gradient can be written using *score differences*.

Proposition 1 (VSD gradient)

Let $\widehat{x}_t = \alpha_t G_\theta(z) + \beta_t\varepsilon$ with $z \sim p_{\mathrm{init}}$, $\varepsilon \sim \mathcal{N}(0, I_d)$. Then

$$\nabla_\theta\mathcal{L}_{\mathrm{VSD}}(\theta) = \mathbb{E}_{t,z,\varepsilon}\Big[\omega(t)\,\alpha_t\left(\nabla_x \log p_t^\theta(\widehat{x}_t) - \nabla_x \log p_t(\widehat{x}_t)\right)^\top \nabla_\theta G_\theta(z)\Big]. \tag{1.15}$$

Proof of Proposition 1. Substitute (1.12) into (1.14) and differentiate with respect to θ, exchanging the order of differentiation and expectation:

$$\nabla_\theta(\log p_t^\theta(\widehat{x}_t) - \log p_t(\widehat{x}_t)) = \nabla_\theta \log p_t^\theta(\widehat{x}_t) + \left(\nabla_x \log p_t^\theta(\widehat{x}_t) - \nabla_x \log p_t(\widehat{x}_t)\right)^\top \nabla_\theta\widehat{x}_t.$$

Since $\widehat{x}_t = \alpha_t G_\theta(z) + \beta_t\varepsilon$, we have $\nabla_\theta\widehat{x}_t = \alpha_t\nabla_\theta G_\theta(z)$. Taking expectation over $\widehat{x}_t \sim p_t^\theta$, the first term vanishes by the score identity

$$\mathbb{E}_{\widehat{x}_t\sim p_t^\theta}[\nabla_\theta \log p_t^\theta(\widehat{x}_t)] = \nabla_\theta \int p_t^\theta(x)\mathrm{d}x = 0.$$

Substituting $\nabla_\theta\widehat{x}_t = \alpha_t\nabla_\theta G_\theta(z)$ into the remaining term gives (1.15).

□

What needs to be estimated? Equation (1.15) requires two scores:

- the **teacher score** $\nabla_x \log p_t(x)$, assumed to be provided by a pretrained diffusion model;
- the **student score** $\nabla_x \log p_t^\theta(x)$, which depends on the current generator G_θ.

The teacher score is obtained by querying the pretrained model. The student score must be estimated from samples.

Algorithm 1: VSD training algorithm.

Require: pretrained teacher score model s_{teacher}; generator G_θ; auxiliary score model s_ζ; weights $\omega(t)$

for *each iteration* **do**

  Sample $t \sim p(t)$, $z \sim p_{\text{init}}$, $\varepsilon \sim \mathcal{N}(0, I_d)$

  $\widehat{x}_0 \leftarrow G_\theta(z)$, $\widehat{x}_t \leftarrow \alpha_t \widehat{x}_0 + \beta_t \varepsilon$

  `// (A) score estimation step`

  $\zeta \leftarrow \zeta - \eta_\zeta \nabla_\zeta \left\| s_\zeta(\widehat{x}_t, t) + (\widehat{x}_t - \alpha_t \widehat{x}_0)/\beta_t^2 \right\|^2$

  `// (B) generator step`

  $\theta \leftarrow \theta - \eta_\theta \nabla_\theta\, \omega(t)\, \alpha_t \big(s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t)\big)^\top G_\theta(z)$

1.3.4 Bilevel training: estimating the student score

Fix θ, and sample $\widehat{x}_0 = G_\theta(z)$ and $\widehat{x}_t$ as in (1.12). Because the conditional distribution $\widehat{x}_t \mid \widehat{x}_0$ is Gaussian, its score is available in closed form:

$$\nabla_x \log p_t(\widehat{x}_t \mid \widehat{x}_0) = -\frac{\widehat{x}_t - \alpha_t \widehat{x}_0}{\beta_t^2}. \tag{1.16}$$

We can therefore train an auxiliary score network $s_\zeta(x, t)$ on *student-generated* data by denoising score matching (DSM):

$$\mathcal{L}_{\text{DSM}}(\zeta; \theta) \triangleq \mathbb{E}_{t,z,\varepsilon}\Big[\Big\| s_\zeta(\widehat{x}_t, t) + \frac{\widehat{x}_t - \alpha_t \widehat{x}_0}{\beta_t^2} \Big\|_2^2\Big], \qquad \widehat{x}_0 = G_\theta(z),\ \widehat{x}_t = \alpha_t \widehat{x}_0 + \beta_t \varepsilon. \tag{1.17}$$

For fixed θ, the optimum satisfies $s_\zeta(x, t) \approx \nabla_x \log p_t^\theta(x)$.

Generator update. With s_ζ trained (or partially trained), we update the generator by treating s_ζ as fixed and using the approximate gradient

$$\nabla_\theta \mathcal{L}_{\text{VSD}}(\theta) \approx \mathbb{E}_{t,z,\varepsilon}\Big[\omega(t)\, \alpha_t \left(s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t)\right)^\top \nabla_\theta G_\theta(z)\Big], \tag{1.18}$$

where $s_{\text{teacher}}(x, t) \approx \nabla_x \log p_t(x)$ is the pretrained teacher score. Training then alternates between two coupled steps:

(i) *score estimation*, which updates ζ via DSM on student samples, and

(ii) *generator update*, which updates θ using the score-difference signal in (1.18).

Why matching scores implies matching distributions. At convergence, (1.18) drives $s_\zeta(\widehat{x}_t, t) \approx s_{\text{teacher}}(\widehat{x}_t, t)$ on the support of p_t^θ, i.e., alignment of student and teacher scores at noisy times. When this alignment holds across time, it implies $p_t^\theta \approx p_t$. Because the Gaussian noising map $p_0 \mapsto p_t$ is injective, matching noisy marginals then forces p_0^θ to match the teacher distribution, and hence p_{data} when the teacher is accurate.

1.3.5 Beyond KL: other divergences

One may replace D_{KL} by an f-divergence

$$D_f(p_t^\theta \| p_t) \triangleq \int p_t(x)\, f\Big(\frac{p_t^\theta(x)}{p_t(x)}\Big) \mathrm{d}x. \tag{1.19}$$

For generic f, the gradient typically involves the (unknown) density ratio p_t^θ/p_t, requiring additional ratio estimation. A common way is to introduce an auxiliary critic or discriminator that approximates the density ratio via the variational formulation of f-divergences, as in f-GAN (Nowozin et al., 2016), although this introduces an extra network and a nested minimax optimization. The KL choice is special because it yields the clean score-difference form in Proposition 1, thus enabling tractable update rules without extra estimation steps.

For the order-α Rényi divergence,

$$D_\alpha(p_t^\theta \| p_t) \triangleq \frac{1}{\alpha - 1} \log \int \big(p_t^\theta(x)\big)^\alpha p_t(x)^{1-\alpha}\, \mathrm{d}x, \qquad \alpha > 0,\ \alpha \neq 1, \tag{1.20}$$

direct differentiation gives

$$\nabla_\theta D_\alpha(p_t^\theta \| p_t) = \frac{\alpha}{\alpha - 1}\, \mathbb{E}_{x \sim q_{\alpha,t}^\theta} \big[\nabla_\theta \log p_t^\theta(x)\big], \qquad q_{\alpha,t}^\theta(x) \propto \big(p_t^\theta(x)\big)^\alpha p_t(x)^{1-\alpha}. \tag{1.21}$$

Thus the gradient is taken under a tilted distribution $q_{\alpha,t}^\theta$, rather than reducing to a simple score-difference expectation under p_t^θ; in practice this again introduces ratio-based reweighting or density-ratio estimation.

Let derive some alternative forms of the gradient. Writing the density ratio as

$$r_t^\theta(x) \triangleq \frac{p_t^\theta(x)}{p_t(x)},$$

we can rewrite the tilted law as

$$q_{\alpha,t}^\theta(x) = \frac{r_t^\theta(x)^{\alpha-1} p_t^\theta(x)}{\int r_t^\theta(u)^{\alpha-1} p_t^\theta(u)\, \mathrm{d}u}. \tag{1.22}$$

Hence

$$\nabla_\theta D_\alpha(p_t^\theta \| p_t) = \frac{\alpha}{\alpha - 1} \frac{\mathbb{E}_{x \sim p_t^\theta}\big[r_t^\theta(x)^{\alpha-1} \nabla_\theta \log p_t^\theta(x)\big]}{\mathbb{E}_{x \sim p_t^\theta}\big[r_t^\theta(x)^{\alpha-1}\big]}. \tag{1.23}$$

So one need not sample from $q_{\alpha,t}^\theta$ directly; the expectation can be written as a self-normalized importance-weighted average under p_t^θ. Nevertheless, the difficult object is still the unknown ratio r_t^θ, so unlike KL this does not lead to a clean score-difference estimator.

1.3.6 An important application: VSD for 3D generation

The VSD idea extends beyond distilling diffusion models into one-step generators. A striking application is 3D generation from a pretrained *2D* image diffusion model. Let $\theta \in \mathbb{R}^d$ denote parameters of a 3D scene (geometry, texture, etc.), and let $\mathcal{R}(\theta)$ be a differentiable renderer that outputs an image $\widehat{x}_0 \triangleq \mathcal{R}(\theta)$. Forward-noise the rendered image:

$$\widehat{x}_t = \alpha_t\,\mathcal{R}(\theta) + \beta_t\varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I_d). \tag{1.24}$$

Let $s_{\text{teacher}}(x, t \mid c)$ be a pretrained *text-conditioned* teacher score (prompt c). VSD introduces an auxiliary score $s_\zeta(x, t)$ modeling the marginal score of the distribution of noisy renderings. A simple score-alignment objective is

$$\mathcal{L}^{3\text{D}}_{\text{VSD}}(\theta) \triangleq \frac{1}{2} \cdot \mathbb{E}_{t,\varepsilon}\Big[\omega(t)\,\big\|s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t \mid c)\big\|_2^2\Big]. \tag{1.25}$$

Define $\mathrm{sg}[\cdot]$ as the stop-gradient operator: identity in the forward pass, zero derivative in the backward pass. During the θ-update, we use the surrogate objective

$$\widetilde{\mathcal{L}}^{3\text{D}}_{\text{VSD}}(\theta) \triangleq \mathbb{E}_{t,\varepsilon}\Big[\omega(t)\;\mathrm{sg}\big[s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t \mid c)\big]^\top \widehat{x}_t\Big], \qquad \widehat{x}_t = \alpha_t\mathcal{R}(\theta) + \beta_t\varepsilon. \tag{1.26}$$

Intuitively, this objective treats the score difference as a fixed guidance vector field during the θ-step, and only differentiates how the renderer moves samples in image space. This is a local first-order approximation to (1.25): it preserves the desired descent direction while avoiding Jacobian-through-score terms that are typically expensive and noisy. Since $\nabla_\theta\,\mathrm{sg}[u] = 0$, differentiation gives

$$\nabla_\theta\widetilde{\mathcal{L}}^{3\text{D}}_{\text{VSD}}(\theta) = \mathbb{E}_{t,\varepsilon}\Big[\omega(t)\;\mathrm{sg}\big[s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t \mid c)\big]^\top \nabla_\theta\widehat{x}_t\Big] \tag{1.27}$$

$$= \mathbb{E}_{t,\varepsilon}\Big[\omega(t)\,\alpha_t\;\mathrm{sg}\big[s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t \mid c)\big]^\top \nabla_\theta\mathcal{R}(\theta)\Big]. \tag{1.28}$$

Dropping $\mathrm{sg}[\cdot]$ in notation for readability yields

$$\nabla_\theta\widetilde{\mathcal{L}}^{3\text{D}}_{\text{VSD}}(\theta) = \mathbb{E}_{t,\varepsilon}\Big[\omega(t)\,\alpha_t\,\big(s_\zeta(\widehat{x}_t, t) - s_{\text{teacher}}(\widehat{x}_t, t \mid c)\big)^\top\,\nabla_\theta\mathcal{R}(\theta)\Big]. \tag{1.29}$$

Compared with the exact chain-rule gradient of (1.25), this removes Jacobian-through-score terms, is is computationally cheaper, and usually lower-variance.

If we *suppress* the student score term (set $s_\zeta \equiv 0$), then (1.29) reduces to the *score distillation sampling* (SDS) gradient used in many text-to-3D pipelines. Keeping s_ζ yields a more principled *distribution matching* interpretation.

1.4 Flow-map-level distillation: progressive distillation

We now turn to distilling the sampling trajectory itself. We focus on deterministic samplers (PF-ODE / DDIM-style maps), which makes the distillation targets explicit.

1.4.1 DDIM transition map in x_0-prediction form

Fix times $s > t$. Given an x_0-prediction denoiser $\widehat{x}_0(\cdot, s)$, the DDIM transition from s to t is the affine map

$$\mathrm{DDIM}_{s\to t}(x_s;\, \widehat{x}_0) = \frac{\beta_t}{\beta_s}x_s + \alpha_s\Big(\frac{\alpha_t}{\alpha_s} - \frac{\beta_t}{\beta_s}\Big)\,\widehat{x}_0(x_s, s). \tag{1.30}$$

This is the concrete instance of (1.8) we use for progressive distillation.

1.4.2 The distillation operation: two teacher steps into one student step

Progressive distillation (Salimans and Ho, 2022) trains a student so that one student step reproduces two teacher steps. Fix three times $s > u > t$ (typically consecutive points on a discrete grid). Let $x_{0,\phi^\star}(x, \tau)$ be the teacher x_0-prediction denoiser. Given input $x_s \sim p_s$, compute the two-step teacher trajectory:

$$\widetilde{x}_u \leftarrow \mathrm{DDIM}_{s\to u}(x_s;\, x_{0,\phi^\star}) = \frac{\beta_u}{\beta_s} x_s + \alpha_s \Big(\frac{\alpha_u}{\alpha_s} - \frac{\beta_u}{\beta_s} \Big) x_{0,\phi^\star}(x_s, s), \tag{1.31}$$

$$\widetilde{x}_t \leftarrow \mathrm{DDIM}_{u\to t}(\widetilde{x}_u;\, x_{0,\phi^\star}) = \frac{\beta_t}{\beta_u} \widetilde{x}_u + \alpha_u \Big(\frac{\alpha_t}{\alpha_u} - \frac{\beta_t}{\beta_u} \Big) x_{0,\phi^\star}(\widetilde{x}_u, u). \tag{1.32}$$

We would like to train a student denoiser $f_\theta(x_s, s)$ such that a single DDIM step from s to t using f_θ matches $\widetilde{x}_t$.

Pseudo-target denoiser output. Define $\widetilde{x}$ (a pseudo-target at time s) as the value that makes a single DDIM step match the two-step teacher output:

$$\widetilde{x}_t = \mathrm{DDIM}_{s\to t}(x_s;\, \widetilde{x}) = \frac{\beta_t}{\beta_s} x_s + \alpha_s \Big(\frac{\alpha_t}{\alpha_s} - \frac{\beta_t}{\beta_s} \Big) \widetilde{x}. \tag{1.33}$$

Solving (1.33) gives a closed-form pseudo-target.

Lemma 2 (Two-steps-in-one DDIM target)

Let $s > u > t$ and let $\widetilde{x}_t$ be produced by two teacher DDIM steps (1.31)–(1.32). The pseudo-target $\widetilde{x}$ defined by (1.33) is

$$\widetilde{x} = \frac{\beta_s}{\alpha_t \beta_s - \alpha_s \beta_t}\, \widetilde{x}_t - \frac{\beta_t}{\alpha_t \beta_s - \alpha_s \beta_t}\, x_s. \tag{1.34}$$

Proof of Lemma 2. Equation (1.33) is linear in $\widetilde{x}$. Rearrange:

$$\alpha_s \Big(\frac{\alpha_t}{\alpha_s} - \frac{\beta_t}{\beta_s} \Big) \widetilde{x} = \widetilde{x}_t - \frac{\beta_t}{\beta_s} x_s.$$

Divide by the scalar coefficient and simplify:

$$\widetilde{x} = \frac{\widetilde{x}_t - (\beta_t/\beta_s) x_s}{\alpha_s \big(\alpha_t/\alpha_s - \beta_t/\beta_s \big)} = \frac{\beta_s}{\alpha_t \beta_s - \alpha_s \beta_t}\, \widetilde{x}_t - \frac{\beta_t}{\alpha_t \beta_s - \alpha_s \beta_t}\, x_s.$$

□

Student training objective. The student denoiser f_θ is trained to regress to $\widetilde{x}$. In continuous time one may sample $s \sim \mathrm{Unif}[0, 1]$ and set $u = s - \Delta$, $t = s - 2\Delta$. In practice we use a discrete decreasing grid $t_0 = 1 > t_1 > \cdots > t_N = 0$. Writing $s = t_k$, $u = t_{k+1}$, $t = t_{k+2}$, the PD objective becomes

$$\min_\theta\ \mathbb{E}_{k\sim \mathrm{Unif}\{0,\ldots,N-2\}}\ \mathbb{E}_{x_{t_k} \sim p_{t_k}} \Big[w\big(\lambda_{t_k}\big)\, \| f_\theta(x_{t_k}, t_k) - \widetilde{x}^{(k)} \|_2^2 \Big], \tag{1.35}$$

where $\widetilde{x}^{(k)}$ is computed from Lemma 2 and $\lambda_t \triangleq \log(\alpha_t/\beta_t)$ is the half-log SNR.

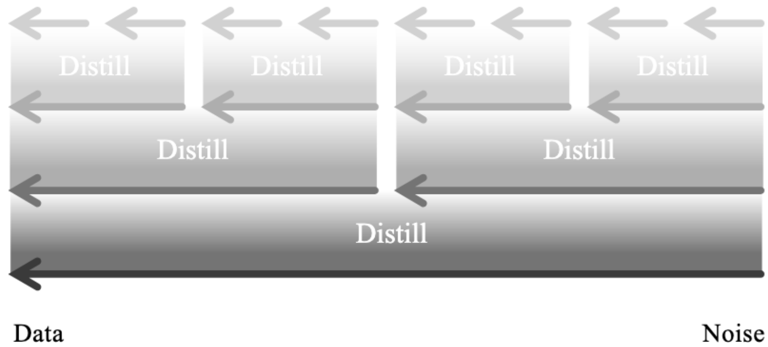

Figure 1: Progressive distillation (PD): at each round, learn a student so that one step matches two teacher steps, then repeat on a coarser grid.

Algorithm 2: Progressive distillation (DDIM, conceptual)

Require: initial teacher denoiser $x_{0,\phi^\star}$; base grid $t_0 > \cdots > t_N$; number of rounds R
for $r = 1, \ldots, R$ **do**
  `// train a student to match two teacher steps`
  for *each training step* **do**
    Sample $k \sim \mathrm{Unif}\{0, \ldots, N-2\}$ and $x_{t_k} \sim p_{t_k}$
    Compute $\widetilde{x}_{t_{k+1}}$ and $\widetilde{x}_{t_{k+2}}$ via (1.31)–(1.32)
    Form $\widetilde{x}^{(k)}$ by (1.34)
    Update f_θ to regress $f_\theta(x_{t_k}, t_k) \approx \widetilde{x}^{(k)}$
  `// replace teacher and coarsen the grid`
  $\phi^\star \leftarrow \theta$
  Keep only every other time point: $t_0, t_2, t_4, \ldots$ (so $N \leftarrow N/2$)

1.4.3 Training and sampling pipeline

After one PD round, the student covers two teacher steps at a time. At inference we therefore sample on the *skip-2* grid $t_0 \to t_2 \to t_4 \to \cdots \to t_N$, reducing the number of steps from N to $N/2$.

Then we repeat:

1. Replace the teacher denoiser $x_{0,\phi^\star}$ by the student f_θ.
2. Coarsen the time grid by keeping every other time point.
3. Train a new student on the coarser grid using the same two-steps-in-one construction.

Each round halves the number of steps: $N \to N/2 \to N/4 \to \cdots$.

1.4.4 PD as local semigroup matching

The oracle flow maps satisfy the semigroup property

$$\Psi_{s\to t} = \Psi_{u\to t} \circ \Psi_{s\to u}. \tag{1.36}$$

Progressive distillation enforces (1.36) *locally* by matching a student "long" step to the composition of two teacher "short" steps. Abstractly, with grid step size Δs, PD minimizes objectives of the form

$$\mathbb{E}_s \mathbb{E}_{x_s\sim p_s} \Big\| G_\theta(x_s, s, s-2\Delta s) - \text{Solver}_{s-\Delta s\to s-2\Delta s}\big(\text{Solver}_{s\to s-\Delta s}(x_s)\big) \Big\|_2^2. \tag{1.37}$$

This uses only *short* teacher fragments. If we denote the full teacher rollout by

$$\text{Solver}^{\circ}_{s\to 0} \triangleq \text{Solver}_{\Delta s\to 0} \circ \cdots \circ \text{Solver}_{s-\Delta s\to s-2\Delta s} \circ \text{Solver}_{s\to s-\Delta s}, \tag{1.38}$$

then (1.37) can be seen as a local surrogate for regressing to $\text{Solver}^{\circ}_{s\to 0}$ (and ultimately to $\Psi_{s\to 0}$).

1.4.5 Other solvers and stochastic samplers

The closed-form pseudo-target in Lemma 2 relies on the *explicit affine* DDIM update (1.30). More general solvers (higher-order Runge–Kutta, multistep methods) may not admit an analytic inversion that yields $\widetilde{x}$. In such cases one can still distill by directly matching transition maps as in (1.37).

For stochastic samplers, a standard trick is to freeze the randomness (noise seed) per example so that the teacher transition becomes deterministic. Then the student is regressed to that fixed transition.

1.4.6 PD with classifier-free guidance

Classifier-free guidance (CFG) typically doubles per-step computation because it evaluates the model twice (conditional/unconditional). A two-stage approach (Meng et al., 2023) distills both guidance and steps:

1. **Distill guidance into a single network.** Let $x_{0,\phi^\star}(x_s, s, c)$ be a pretrained x_0-prediction model conditioned on prompt c, and let $\varnothing$ denote the null condition. For guidance weight $\omega \geq 0$, the CFG combination is

$$x^{\omega}_{0,\phi^\star}(x_s, s, c) \triangleq (1+\omega)\, x_{0,\phi^\star}(x_s, s, c) - \omega\, x_{0,\phi^\star}(x_s, s, \varnothing). \tag{1.39}$$

   Train a new model $x_{0,\theta_1}(x_s, s, c, \omega)$ that takes ω as input and regresses to (1.39):

$$\min_{\theta_1} \; \mathbb{E}_{\omega\sim p_\omega} \mathbb{E}_s \mathbb{E}_{x\sim p_{\text{data}},\, x_s\sim p(x_s|x)} \Big[\lambda(s) \, \| x_{0,\theta_1}(x_s, s, c, \omega) - x^{\omega}_{0,\phi^\star}(x_s, s, c) \|_2^2 \Big]. \tag{1.40}$$

2. **Apply PD to reduce steps.** Use x_{0,θ_1} as the teacher denoiser in the PD pipeline and distill it into x_{0,θ_2} that samples on a much coarser grid.

The final sampler uses one network call per step and very few steps.

1.5 Closing remarks

Distillation aims to avoid long iterative sampling by learning a faster student. We covered two complementary families:

- **Distribution-level** methods (e.g., VSD) match student and teacher distributions through divergence minimization and score-difference training signals.
- **Flow-map-level** methods (KD/PD) learn approximate transport maps between time marginals, often by matching short teacher fragments.

There are many closely related approaches. For instance, *consistency models* (Song et al., 2023) and *consistency distillation* can be interpreted as learning time-consistent maps, which is conceptually similar to the semigroup-matching view in (1.37).

In the next lecture we will revisit limitations of distillation: how errors accumulate when we coarsen the time grid, how imperfect teachers bias the student, and what diagnostics or corrections are available.

References

Luhman, E. and Luhman, T. (2021). Knowledge distillation in iterative generative models for improved sampling speed.

Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., and Salimans, T. (2023). On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Nowozin, S., Cseke, B., and Tomioka, R. (2016). f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems 29 (NeurIPS 2016).*

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

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. (2023). Consistency models. In *International Conference on Machine Learning (ICML).*

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. (2023). ProlificDreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *Advances in Neural Information Processing Systems (NeurIPS).*

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. (2024). One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Zhou, M., Zheng, H., Yi, Z., Gu, M., and Peng, Z. (2024). Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *International Conference on Machine Learning (ICML).*