CS 6501: Constrained-Aware Generative AI

Lecture Notes 7
Diffusion Models and Guidance

DDPM objectives, score perspectives, and algorithmic insertion points for constraints

Prof. Ferdinando Fioretto
Department of Computer Science, University of Virginia

Tuesday, February 3, 2026

Abstract

Diffusion models instantiate a powerful design pattern for constrained-aware generation: a sample is produced by a long sequence of small, local updates that progressively refine a simple noise distribution into a complex data distribution. This iterative structure is algorithmically valuable because it creates many natural insertion points for control mechanisms, including guidance forces (soft constraints as energy shaping), projections (hard constraints as feasibility restoration), and repair operators.

These notes develop the denoising diffusion probabilistic model (DDPM) objective and its score interpretation. We derive the forward noising process, the noise-prediction training objective, and the reverse sampling recursion. We then introduce guidance mechanisms, emphasizing the score view in which guidance adds a gradient term to the reverse dynamics, thereby implementing the course's constrained target distribution from Lecture 1. Finally, we connect these updates to the optimization primitives of Lecture 6 by writing a generic reverse sampler with explicit slots for projection and proximal restoration, and we work through representative examples such as inpainting and linear inverse problems.

1 Why diffusion is a natural vehicle for constraint injection

In Lecture 1 we emphasized the "fundamental equation" of constrained-aware generation: define a target distribution by combining a plausibility model with soft constraint potentials and hard feasibility sets. In Lecture 6 we isolated the optimization primitives that reappear when we try to compute under such targets, including penalties, proximal maps, and projections.

Diffusion models are a particularly convenient instantiation of this view. They generate by iterating a local update

$$\mathbf{x}_{t-1} \leftarrow \mathrm{Update}_\theta(\mathbf{x}_t, t; \mathrm{conditioning}) + \mathrm{noise}, \tag{1}$$

where t is a discrete time index descending from a large T to 0. Algorithmically, (1) already resembles the "one model step, one constraint step" template that appears in proximal splitting. Because the reverse chain takes T steps, we can couple each step to a control or repair operator.

Throughout the course, we will repeatedly use three families of insertions. First, **guidance** adds a drift term that pushes samples toward satisfying a property or condition, often by adding a gradient of a log-likelihood or an energy penalty. Second, **projection and proximal repair** restore feasibility or reduce violation after a model update. Third, **discrete repair** replaces gradients by search or rule-based operations when the constraints are symbolic or combinatorial (Lectures 4 and 9).

This lecture focuses on continuous diffusion and the first two mechanisms, but we phrase results in a way that anticipates discrete diffusion and constrained decoding.

2 DDPM: forward process, reverse process, and objective

We follow Ho et al. (2020) and Sohl-Dickstein et al. (2015). The object is a continuous vector $\mathbf{x}_0 \in \mathbb{R}^d$ (an image, a trajectory representation, a relaxed embedding). The diffusion model defines a latent Markov chain $(\mathbf{x}_t)_{t=0}^T$ by specifying a *forward* noising process q and learning a *reverse* denoising process p_θ.

2.1 Forward diffusion

Choose a variance schedule $(\beta_t)_{t=1}^T$ with $\beta_t \in (0,1)$, and define $\alpha_t \triangleq 1-\beta_t$ and the cumulative product $\bar{\alpha}_t \triangleq \prod_{s=1}^t \alpha_s$. The forward process is

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

Because the forward process is linear Gaussian, we can marginalize in closed form:

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

Equivalently, we can sample $\mathbf{x}_t$ from $\mathbf{x}_0$ using a single noise draw $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$:

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

For reasonable schedules, the end distribution $q(\mathbf{x}_T)$ is close to $\mathcal{N}(0, \mathbf{I})$, so sampling can start from pure noise.

2.2 Reverse diffusion and the variational objective

We learn a reverse Markov chain

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

with a parameterized mean μ_θ and a chosen covariance schedule Σ_t (fixed or learned). Sampling begins from $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$ and iterates $t = T, \dots, 1$.

Training can be derived via a variational bound on $-\log p_\theta(\mathbf{x}_0)$, using the forward process as an approximate posterior over the latent chain:

$$\mathcal{L}_{\mathrm{VLB}}(\theta) \;=\; \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[-\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1) + \sum_{t=2}^{T} \mathrm{KL}\big(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\big)\right] + \text{const.} \tag{6}$$

Here $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$ is tractable because the forward chain is Gaussian:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0),\ \tilde{\beta}_t \mathbf{I}), \tag{7}$$

where

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\,\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\,\mathbf{x}_t, \qquad \tilde{\beta}_t \triangleq \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \tag{8}$$

2.3 Noise prediction and the "simple" objective

The practical DDPM objective is usually written in the *noise prediction* form. Define a network $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ that predicts the noise $\boldsymbol{\epsilon}$ in (4). Then we can estimate $\mathbf{x}_0$ by

$$\hat{\mathbf{x}}_0(\mathbf{x}_t, t) \;=\; \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right). \tag{9}$$

Plugging (9) into (8) yields a reverse mean parameterization that matches the true posterior when $\hat{\mathbf{x}}_0 = \mathbf{x}_0$:

$$\mu_\theta(\mathbf{x}_t, t) \;=\; \tilde{\mu}_t\big(\mathbf{x}_t, \hat{\mathbf{x}}_0(\mathbf{x}_t, t)\big) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right). \tag{10}$$

A key simplification in Ho et al. (2020) is that, under standard covariance choices, the variational bound reduces (up to weights) to a denoising regression:

$$\mathcal{L}_{\text{simple}}(\theta) \;=\; \mathbb{E}_{t\sim\text{Unif}\{1,\dots,T\},\,\mathbf{x}_0\sim p_{\text{data}},\,\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|_2^2\right], \qquad \mathbf{x}_t \text{ from (4)}. \tag{11}$$

Training repeatedly samples a data point $\mathbf{x}_0$, samples a time t, creates $\mathbf{x}_t$, and regresses to the known noise.

3 The score view: diffusion as score estimation

Guidance is most naturally expressed by adding gradients to a score-based update. We therefore connect the noise predictor $\boldsymbol{\epsilon}_\theta$ to a score estimator $s_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$.

3.1 Score of the forward marginal

The forward conditional (3) is Gaussian, so its score is explicit:

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t \mid \mathbf{x}_0) = -\frac{1}{1-\bar{\alpha}_t}\left(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0\right). \tag{12}$$

Using (4), $\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0 = \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}$, so

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t \mid \mathbf{x}_0) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\,\boldsymbol{\epsilon}. \tag{13}$$

If $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ predicts $\boldsymbol{\epsilon}$, then a natural score estimator is

$$s_\theta(\mathbf{x}_t, t) \;\triangleq\; -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\,\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t). \tag{14}$$

Conceptually, training (11) learns the score fields of the noised data distributions across noise levels, connecting DDPMs to classical score matching (Hyvärinen, 2005) and to the general score-based SDE framework (Song et al., 2021).

3.2 Reverse updates as "score plus noise"

The DDPM sampler can be written as

$$\mathbf{x}_{t-1} = \mu_\theta(\mathbf{x}_t, t) + \Sigma_t^{1/2}\,\mathbf{z}, \qquad \mathbf{z} \sim \mathcal{N}(0, \mathbf{I}), \tag{15}$$

with μ_θ given by (10). Since μ_θ depends linearly on $\boldsymbol{\epsilon}_\theta$ (and hence on the score via (14)), the reverse update is "a score-driven step plus Gaussian noise". This is the main entry point for guidance: if we want the reverse chain to sample from a *tilted* target distribution, we add an extra score term coming from the tilting factor.

4 Guidance as energy shaping

We now connect diffusion guidance to the constrained target distribution perspective from Lecture 1. Suppose we want samples from

$$\pi(\mathbf{x}_0 \mid \mathbf{c}) \;\propto\; p_{\text{data}}(\mathbf{x}_0)\,\exp\big(-\lambda\phi(\mathbf{x}_0, \mathbf{c})\big)\,\mathbf{1}\{\mathbf{x}_0 \in \mathcal{C}(\mathbf{c})\}. \tag{16}$$

If we knew the score of the intermediate marginals π_t at each noise level t, we could run the reverse chain under that score. Guidance approximates the score of a tilted distribution by adding a term derived from a conditioner.

4.1 Classifier guidance

Assume a classifier provides $p_\varphi(\mathbf{y} \mid \mathbf{x}_t)$ for a label or property $\mathbf{y}$. Under standard approximations, the guided score is

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid \mathbf{y}) \;=\; \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_\varphi(\mathbf{y} \mid \mathbf{x}_t). \tag{17}$$

Operationally, replace $s_\theta(\mathbf{x}_t, t)$ by

$$s_{\text{guided}}(\mathbf{x}_t, t) \;=\; s_\theta(\mathbf{x}_t, t) + w\, \nabla_{\mathbf{x}_t} \log p_\varphi(\mathbf{y} \mid \mathbf{x}_t), \tag{18}$$

where $w \geq 0$ is a guidance scale. This increases adherence to $\mathbf{y}$, at the cost of potential distribution shift and reduced diversity.

It is helpful to rewrite (18) as an energy shaping statement. Define $E(\mathbf{x}_t; \mathbf{y}) \triangleq -\log p_\varphi(\mathbf{y} \mid \mathbf{x}_t)$. Then $\nabla_{\mathbf{x}_t} \log p_\varphi(\mathbf{y} \mid \mathbf{x}_t) = -\nabla_{\mathbf{x}_t} E(\mathbf{x}_t; \mathbf{y})$, so guidance adds a force that decreases the energy, mirroring the role of the soft constraint potential ϕ in (16).

4.2 Classifier-free guidance

Classifier guidance requires a separate classifier trained on noisy inputs. Classifier-free guidance (CFG) avoids this by training a single denoiser that can operate both conditionally and unconditionally (Ho and Salimans, 2022). Let $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c})$ be the conditional noise predictor and $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \varnothing)$ the unconditional predictor (implemented by dropping conditioning during training). CFG forms a guided predictor

$$\boldsymbol{\epsilon}_{\text{cfg}}(\mathbf{x}_t, t, \mathbf{c}) = (1 + w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}) - w\, \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \varnothing), \tag{19}$$

and then uses (10) with $\boldsymbol{\epsilon}_\theta$ replaced by $\boldsymbol{\epsilon}_{\text{cfg}}$. The difference $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \mathbf{c}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, \varnothing)$ estimates the direction in which conditioning changes the score, and scaling this difference amplifies the conditional signal.

4.3 General energy guidance and property guidance

More generally, suppose we can evaluate a differentiable energy $E(\mathbf{x}_0, \mathbf{c})$ that penalizes violation of a desired property. A simple plug-in strategy is to map E to the noisy space using the estimate $\hat{\mathbf{x}}_0(\mathbf{x}_t, t)$ from (9), and add the gradient of $-E(\hat{\mathbf{x}}_0(\mathbf{x}_t, t), \mathbf{c})$ to the reverse drift. This gives a training-free way to impose differentiable soft constraints, and it mirrors the penalty-force insertion discussed in Lecture 6.

Remark 1 (When guidance can fail)**.** *Guidance is only as reliable as the energy or conditional model it uses. Large guidance weights can push samples away from the learned data manifold, producing unrealistic outputs. This is a concrete instance of the plausibility versus validity tension from Lecture 1. A robust remedy is to combine guidance with feasibility restoration via projection or a proximal map, which limits drift while still biasing toward constraints.*

5 Algorithmic view: reverse diffusion with explicit insertion points

We now write a generic reverse sampler that exposes explicit insertion points for (i) guidance forces and (ii) feasibility restoration via projection or proximal steps. This is the key algorithmic bridge to Lecture 6.

The two lines to focus on are the guidance drift G and the restoration operator $\mathcal{R}_t$. Common choices of $\mathcal{R}_t$ include

$$\mathcal{R}_t(\mathbf{x}) = \text{Proj}_{\mathcal{C}}(\mathbf{x}) \qquad \text{or} \qquad \mathcal{R}_t(\mathbf{x}) = \text{prox}_{\eta_t \phi}(\mathbf{x}), \tag{20}$$

with Proj and prox as defined in Lecture 6. In this view, the reverse chain becomes a splitting method: a model-driven stochastic step followed by a feasibility or penalty step.

Algorithm 1 Reverse diffusion with guidance and feasibility restoration (generic template)

Require: Noise schedule $(\beta_t)_{t=1}^T$; denoiser $\boldsymbol{\epsilon}_\theta(\cdot)$; guidance function $G(\mathbf{x}_t, t, \mathbf{c})$ returning a drift vector; feasibility operator $\mathcal{R}_t(\cdot)$ (projection, prox, or repair); guidance scale $w \geq 0$.

Ensure: Sample $\mathbf{x}_0$.

1: Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2: **for** $t = T, T-1, \ldots, 1$ **do**
3: Compute base mean $\mu_\theta(\mathbf{x}_t, t)$ via (10) (or using a guided predictor such as (19)).
4: Compute guided mean $\tilde{\mu} \leftarrow \mu_\theta(\mathbf{x}_t, t) + w\, G(\mathbf{x}_t, t, \mathbf{c})$.
5: Sample $\mathbf{x}_{t-1} \leftarrow \tilde{\mu} + \Sigma_t^{1/2}\mathbf{z}$, with $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ (or set noise to 0 in implicit samplers).
6: Feasibility restoration $\mathbf{x}_{t-1} \leftarrow \mathcal{R}_t(\mathbf{x}_{t-1})$.
7: **end for**
8: **return** $\mathbf{x}_0$.

6 Worked examples of constraint insertion

We now give concrete examples that connect diffusion guidance to standard constraint forms from Lectures 1 and 6.

6.1 Example 1: image inpainting as a hard projection

Consider an image vector $\mathbf{x} \in \mathbb{R}^d$ and a binary mask operator M that selects observed pixels. In inpainting, we require $M\mathbf{x} = \mathbf{b}$ for observed values $\mathbf{b}$. This is an affine constraint set

$$\mathcal{C} \;=\; \{\mathbf{x} \in \mathbb{R}^d : M\mathbf{x} = \mathbf{b}\}. \tag{21}$$

The Euclidean projection is simple: replace the observed pixels and keep the rest unchanged:

$$\mathrm{Proj}_{\mathcal{C}}(\mathbf{x}) = (I - M)\mathbf{x} + M^\top \mathbf{b}, \tag{22}$$

where $M^\top$ denotes the embedding back into full space. Thus, in algorithm 1, we can set $\mathcal{R}_t = \mathrm{Proj}_{\mathcal{C}}$ for all t. This guarantees exact data consistency at every step, which is often more stable than enforcing it only at the end.

6.2 Example 2: linear inverse problems and DDRM-style data consistency

Consider noisy linear measurements $\mathbf{y} = A\mathbf{x}_0 + \eta$ with $\eta \sim \mathcal{N}(0, \sigma^2 I)$. A natural soft constraint is the negative log-likelihood

$$\phi(\mathbf{x}_0) = \frac{1}{2\sigma^2}\|A\mathbf{x}_0 - \mathbf{y}\|_2^2. \tag{23}$$

If we use plug-in energy guidance based on $\hat{\mathbf{x}}_0(\mathbf{x}_t, t)$, then the added drift is proportional to

$$-\nabla_{\mathbf{x}_t}\phi(\hat{\mathbf{x}}_0(\mathbf{x}_t, t)) = -\left(\nabla_{\mathbf{x}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t, t)\right)^\top \frac{1}{\sigma^2}A^\top(A\hat{\mathbf{x}}_0(\mathbf{x}_t, t) - \mathbf{y}). \tag{24}$$

This implements a data-consistency force and is closely related in spirit to diffusion restoration methods such as DDRM (Kawar et al., 2022). From an optimization viewpoint, this is exactly the penalty-force insertion from Lecture 6, applied within the reverse chain.

6.3 Example 3: feasibility via proximal restoration

Suppose we have a nonsmooth penalty, such as an ℓ_1 regularizer or total variation:

$$\phi(\mathbf{x}_0) = \|\mathbf{x}_0\|_1 \qquad \text{or} \qquad \phi(\mathbf{x}_0) = \mathrm{TV}(\mathbf{x}_0). \tag{25}$$

Then choosing $\mathcal{R}_t(\mathbf{x}) = \mathrm{prox}_{\eta_t\phi}(\mathbf{x})$ yields a principled restoration step that remains well-defined even when ϕ is nonsmooth. This makes the reverse chain resemble proximal splitting, except that the model update is a learned denoiser rather than an explicit gradient step.

7 Implicit sampling: DDIM and an ODE-like interpretation

The ancestral DDPM sampler (15) injects Gaussian noise at every step. Denoising Diffusion Implicit Models (DDIM) (Song et al., 2020) show that one can define a family of reverse processes that share the same marginals but vary the amount of stochasticity. A common instantiation uses a deterministic update (noise set to zero) coupled to the same $\hat{\mathbf{x}}_0(\mathbf{x}_t, t)$ estimate. This enables fewer steps and a more ODE-like interpretation.

The controlled-iteration view remains the same: deterministic updates make projection or proximal steps feel closer to standard optimization algorithms, and they often reduce the variance of constraint satisfaction across runs.

8 Summary and looking ahead

Diffusion models provide an algorithmic interface for constraint-aware generation that is often more direct than one-shot generators: the reverse process is a sequence of local updates, and each update can be augmented with guidance forces and feasibility restoration. The DDPM objective (11) trains a noise predictor that induces a score estimator (14). Guidance then implements energy shaping by adding score terms, either from a classifier (18) or via classifier-free combinations (19). Hard feasibility can be implemented step-wise via projection or proximal restoration, aligning diffusion sampling with the optimization primitives from Lecture 6.

In Lecture 8 we will make the continuous-time connection more explicit via flow matching and rectified flows, where the reverse dynamics become an ODE and constraint injection resembles adding a control term or splitting a vector field. In Lecture 9 we will return to discrete spaces, where the same insertion-point logic applies but gradients are replaced by search and combinatorial repair.

References

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

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

A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning (ICML)*, 2021.

J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021.

B. Kawar, M. Elad, S. E. Y. Shimoni, and M. Irani. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.