Denoising Diffusion Models for Plug-and-Play Image Restoration

Yuanzhi Zhu

Supervisors: Dr. Kai Zhang, Jingyun Liang, Jiezhang Cao
Principal Investigator: Prof. Luc Van Gool

CVL Computer Vision Lab

ETH zürich

18/June/2023

Content

- Preliminaries
- Methods
- Results

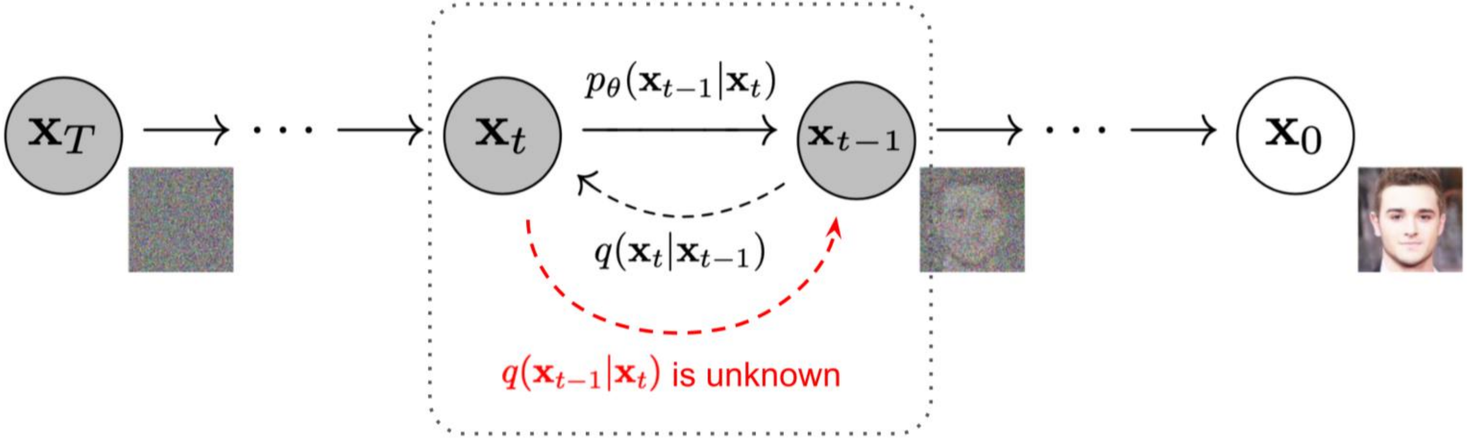
DDPM: Denoising Diffusion Probabilistic Models*

True data dist. : $x_0 \sim q(x_0)$

Forward process: $q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1})$

Reverse process: $p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$

Markov assumption



*[2006.11239] Denoising Diffusion Probabilistic Models (arxiv.org)

DDPM: Denoising Diffusion Probabilistic Models

Forward Diffusion Process

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1})$$

Each Step

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \underbrace{\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}}_{\text{norm invariant}}) \quad \text{or} \quad x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1}$$

noise schedule β_t controls the diffusion process

For arbitrary *t*

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}) \qquad \alpha_t = 1 - \beta_t \text{ and } \bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$$

$$\boxed{x_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t} \cdot z_t}$$

DDPM: Denoising Diffusion Probabilistic Models

Reverse Diffusion Process

if β_t is small enough, $q(x_{t-1}|x_t)$ will also be Gaussian

$$p_\theta(x_{0:T}) = p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \qquad \underline{p_\theta(x_{t-1}|x_t)} = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_\theta(x_t, t), \boldsymbol{\Sigma}_\theta(x_t, t))$$

match

Reverse when condition on x_0

all three are forward processes

$$\underline{q(x_{t-1}|x_t, x_0)} = \boxed{q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}}$$

$$\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$\longrightarrow q(x_{t-1}|x_t, x_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I})$$

$$\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_t\right)$$

model $\mathbf{z}_t$ with NN ☺

DDPM: Denoising Diffusion Probabilistic Models

Negative Log Likelihood to Variational Lower Bound

Known

$$-\log p_\theta(\mathbf{x}_0) \Longrightarrow \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right] \Longrightarrow \sum_{t=2}^{T} \underbrace{D_{\mathrm{KL}}(\boxed{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \,\|\, p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_t}$$

Explicit parameterization

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_\theta(x_t, t), \boldsymbol{\Sigma}_\theta(x_t, t)) \quad \begin{cases} \boldsymbol{\mu}_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} z_\theta(x_t, t)\right) \\ \boldsymbol{\Sigma}_\theta(x_t, t) = \sigma_t^2 \mathbf{I} \end{cases}$$

two options† $\sigma^2 = \begin{cases} \beta_t \\ \dfrac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \end{cases}$

Model The Noise (Residual)

Denoiser

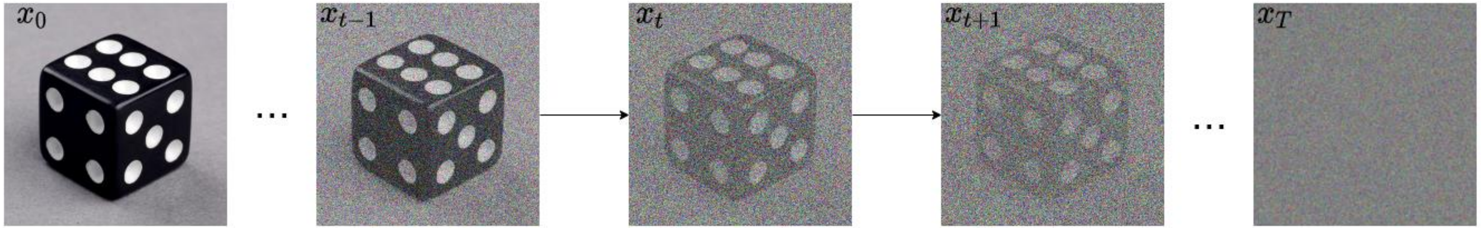
$$L_t^{\text{simple}} = \mathbb{E}_{x_0, z_t}\left[\left\|z_t - z_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z_t, t)\right\|^2\right]$$

$$\hat{\mathbf{x}}_0^{(t)}(\mathbf{x}_t) = \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\mathbf{z}_\theta^{(t)}(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}}$$

† Covariance has analytical optimal form (Estimating the Optimal Covariance with Imperfect Mean in Diffusion Probabilistic Models)

DDPM: Denoising Diffusion Probabilistic Models

$$x_0 \sim q(x_0) \longrightarrow q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \longrightarrow q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$



$$p_\theta(x_0) = \int p(x_T) \prod_{i=1}^{T} p_\theta(x_{t-1}|x_t) dx_{1:T} \longleftarrow p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \longleftarrow x_T \sim \mathcal{N}(0, I)$$

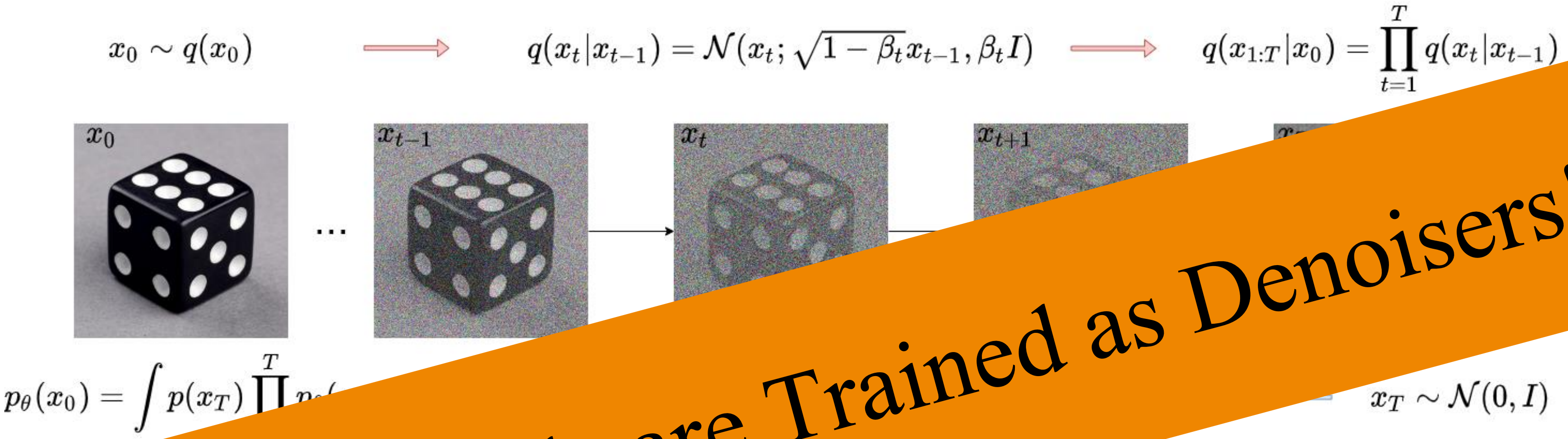
Algorithm 1 Training

repeat
$\mathbf{x}_0 \sim q(\mathbf{x}_0)$
$t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \mathbf{z}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
until converged

Algorithm 2 Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
$\boxed{\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \mathbf{z}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}}$
end for
return $\mathbf{x}_0$

DDPM: Denoising Diffusion Probabilistic Models



Diffusion Models are Trained as Denoisers!!

$\ldots, T\})$

$\mathbf{I})$

Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \mathbf{z}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\boxed{\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}}$
5: **end for**
6: **return** $\mathbf{x}_0$

DDIM: Denoising Diffusion Implicit Models*

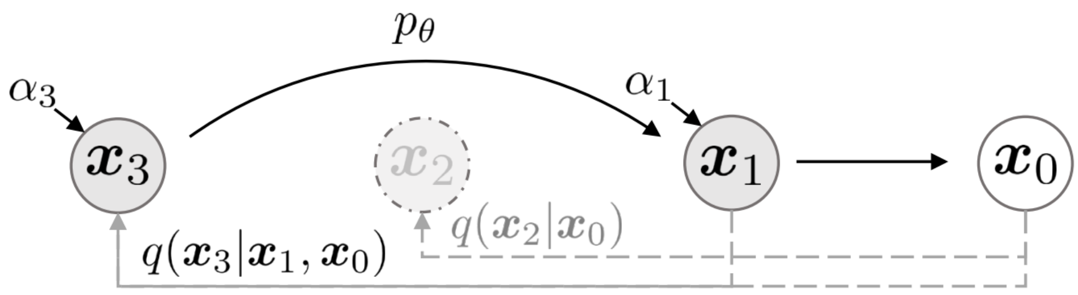
Reverse Process: deterministic given x_t, x_0, with $\sigma_t = 0$

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} x_0 + \frac{\sqrt{1-\bar{\alpha}_{t-1}}}{\sqrt{1-\bar{\alpha}_t}} (x_t - \sqrt{\bar{\alpha}_t} x_0)$$

Sampling:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t} z_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{"predicted } x_0\text{"}} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}} \cdot z_\theta^{(t)}(x_t)}_{\text{"direction pointing to } x_t\text{"}}$$

Accelerated Generation Processes



[2010.02502] Denoising Diffusion Implicit Models (arxiv.org)

Content

Plug-and-Play Image Restoration

Measurement (degradation model):

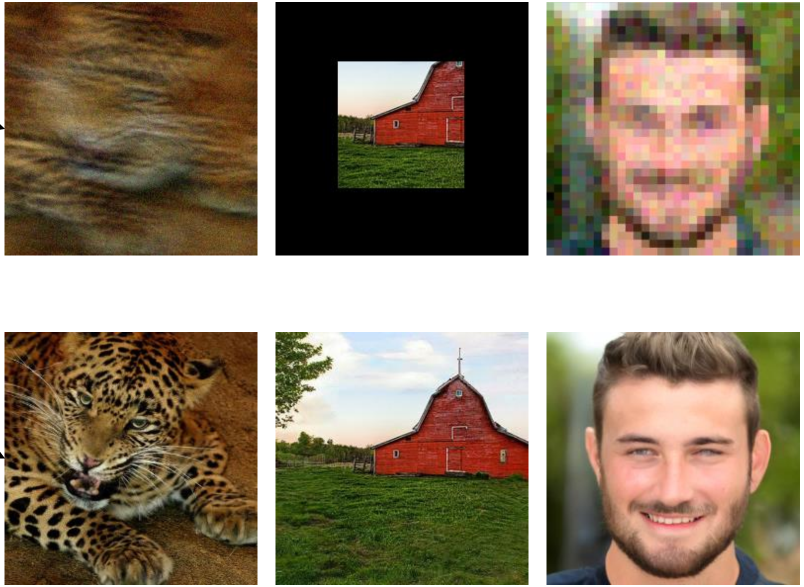
$$\mathbf{y} = \mathcal{H}\mathbf{x} + \mathbf{n}$$

Reconstruction (Bayes' theorem):

$$\begin{aligned}\hat{\mathbf{x}} &\sim p(\mathbf{x}|\mathbf{y}) \\ &\propto p(\mathbf{x})p(\mathbf{y}|\mathbf{x})\end{aligned}$$

Maximum A Posteriori (MAP) estimation:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x})$$



Plug-and-Play Image Restoration

Substitute degradation model $\mathbf{y} = \mathcal{H}\mathbf{x} + \mathbf{n}$:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \underbrace{\frac{1}{2\sigma_n^2}\|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2}_{\text{data term}} + \underbrace{\lambda\mathcal{P}(\mathbf{x})}_{\text{prior term}}$$

Introduce auxiliary variable **z**:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2 + \lambda\mathcal{P}(\mathbf{z}) \quad s.t. \quad \mathbf{z} = \mathbf{x}$$

Lagrange multiplier:

$$\mathcal{L}_\mu(\mathbf{x}, \mathbf{z}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2 + \lambda\mathcal{P}(\mathbf{z}) + \frac{\mu}{2}\|\mathbf{z} - \mathbf{x}\|^2$$

Half Quadratic Splitting (**HQS**) algorithm:

$$\begin{cases} \mathbf{z}_k = \arg\min_{\mathbf{z}} \underbrace{\frac{1}{2(\sqrt{\lambda/\mu})^2}\|\mathbf{z} - \mathbf{x}_k\|^2}_{\text{consistence}} + \underbrace{\mathcal{P}(\mathbf{z})}_{\text{prior}} & \textbf{Prior} \\ \mathbf{x}_{k-1} = \arg\min_{\mathbf{x}} \underbrace{\|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2}_{\text{condition}} + \underbrace{\mu\sigma_n^2\|\mathbf{x} - \mathbf{z}_k\|^2}_{\text{consistence}} & \textbf{Data} \end{cases}$$

Plug-and-Play Image Restoration

$$\arg\min_{\mathbf{z}} \underbrace{\frac{1}{2(\sqrt{\lambda/\mu})^2}\|\mathbf{z}-\mathbf{x}_k\|^2}_{\text{consistence}} + \underbrace{\mathcal{P}(\mathbf{z})}_{\text{prior}} \xrightarrow[\mathbf{x}_k = \mathbf{z}_k + \sqrt{\lambda/\mu}\epsilon]{\text{by definition}} \mathbf{z}_k = Denoiser(\mathbf{x}_k, \sqrt{\lambda/\mu})$$

degradation models

$$\arg\min_{\mathbf{x}} \underbrace{\|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2}_{\text{condition}} + \underbrace{\mu\sigma_n^2\|\mathbf{x}-\mathbf{z}_k\|^2}_{\text{consistence}}$$

Inpainting $\mathbf{x}_{k-1} = \dfrac{\mathbf{M}\odot\mathbf{y}+\rho_k\mathbf{z}_k}{\mathbf{M}+\rho_k}$ $\qquad \mathbf{y} = \mathbf{M}\odot\mathbf{x}$

Deblurring $\mathbf{x}_{k-1} = \mathcal{F}^{-1}\left(\dfrac{\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{y})+\rho_k\mathcal{F}(\mathbf{z}_k)}{\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{k})+\rho_k}\right)$ $\qquad \mathbf{y} = \mathbf{x}\otimes\mathbf{k}+\mathbf{n}$

SR $\mathbf{x}_{k-1} = \mathcal{F}^{-1}\left(\dfrac{1}{\rho_k}\left(\mathbf{d}-\overline{\mathcal{F}(\mathbf{k})}\odot_s\dfrac{(\mathcal{F}(\mathbf{k})\mathbf{d})\Downarrow_s}{(\overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{k}))\Downarrow_s+\rho_k}\right)\right)$ $\qquad \mathbf{y} = \mathbf{x}\downarrow_{sf}^{bicubic} + \mathbf{n}$

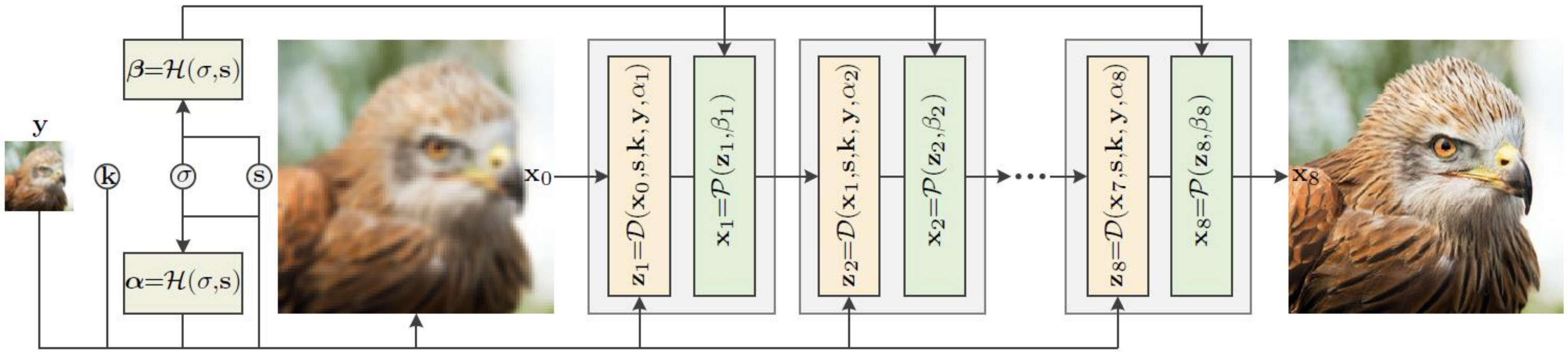
$$\mathbf{d} = \overline{\mathcal{F}(\mathbf{k})}\mathcal{F}(\mathbf{y}\uparrow_{sf}) + \rho_t\mathcal{F}(\mathbf{z}_k)$$

Approximately $\mathbf{x}_{k-1} \approx \mathbf{z}_k - \dfrac{1}{2\mu\sigma_n^2}\nabla_{\mathbf{z}_k}\|\mathbf{y}-\mathcal{H}(\mathbf{z}_k)\|^2$

Plug-and-Play Image Restoration

Previous Iterative Approaches:

- Empirically chosen *schedules* ☹
- *Discriminative* denoisers ☹



Introduce Diffusion Models:

- Well-defined *sampling schedules/trajectories* ☺
- *Generative* prior ☺

Sampling as Optimization

$$dx_t = -\nabla V(x_t)dt + \sqrt{2}dB_t$$

But where does the generative power come from?

Image from: cszn/USRNet: Deep Unfolding Network for Image Super-Resolution (CVPR, 2020) (PyTorch) (github.com)

Denoising Diffusion Models for Plug-and-Play Image Restoration

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{"predicted } x_0\text{"}} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}} \cdot \epsilon_\theta^{(t)}(x_t)}_{\text{"direction pointing to } x_t\text{"}}$$

One iteration **HQS → estimate** $\hat{\mathbf{x}}_0^{(t)}(\mathbf{x}_t, \mathbf{y})$

$$\begin{cases} \mathbf{x}_0^{(t)} = \arg\min_{\mathbf{z}} \frac{1}{2\bar{\sigma}_t^2} \|\mathbf{z} - \mathbf{x}_t\|^2 + \mathcal{P}(\mathbf{z}) \\ \hat{\mathbf{x}}_0^{(t)} = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2 + \rho_t \|\mathbf{x} - \mathbf{x}_0^{(t)}\|^2 \end{cases}$$

Calculate the predicted conditional noise

$$\hat{\epsilon}(\mathbf{x}_t, \mathbf{y}) = \frac{1}{\sqrt{1-\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \hat{\mathbf{x}}_0^{(t)}(\mathbf{x}_t, \mathbf{y}))$$

Finish one sampling step by adding noise back

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \hat{\mathbf{x}}_0^{(t)}(\mathbf{x}_t, \mathbf{y}) + \sqrt{1-\bar{\alpha}_{t-1}} (\sqrt{1-\zeta} \hat{\epsilon}(\mathbf{x}_t, \mathbf{y}) + \sqrt{\zeta} \epsilon_t)$$

$$\hat{\mathbf{x}}_0(\mathbf{x}_t, \mathbf{y}) = \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, y)) = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t + (1-\bar{\alpha}_t)(\mathbf{s}_\theta(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \right) = \hat{\mathbf{x}}_0(\mathbf{x}_t) + \frac{1-\bar{\alpha}_t}{\sqrt{\bar{\alpha}_t}} \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t).$$

In this page we use ϵ_θ instead z_θ to avoid confusion

Denoising Diffusion Models for Plug-and-Play Image Restoration

HQS algorithm

$$\begin{cases} \mathbf{z}_k = \arg\min_{\mathbf{z}} \underbrace{\frac{1}{2(\sqrt{\lambda/\mu})^2}\|\mathbf{z}-\mathbf{x}_k\|^2}_{\text{consistence}} + \underbrace{\mathcal{P}(\mathbf{z})}_{\text{prior}} \\ \mathbf{x}_{k-1} = \arg\min_{\mathbf{x}} \underbrace{\|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2}_{\text{condition}} + \underbrace{\mu\sigma_n^2\|\mathbf{x}-\mathbf{z}_k\|^2}_{\text{consistence}} \end{cases}$$

$$\Downarrow$$

DiffPIR

$$\begin{cases} \mathbf{x}_0^{(t)} = \arg\min_{\mathbf{z}} \frac{1}{2\bar{\sigma}_t^2}\|\mathbf{z}-\mathbf{x}_t\|^2 + \mathcal{P}(\mathbf{z}) \\ \hat{\mathbf{x}}_0^{(t)} = \arg\min_{\mathbf{x}} \|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2 + \rho_t\|\mathbf{x}-\mathbf{x}_0^{(t)}\|^2 \\ \mathbf{x}_{t-1} \longleftarrow \hat{\mathbf{x}}_0^{(t)} \end{cases}$$

unconditional

Algorithm 1 DiffPIR

Require: $\mathbf{s}_\theta, T, \mathbf{y}, \sigma_n, \{\bar{\sigma}_t\}_{t=1}^T, \zeta, \lambda$

1: Initialize $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, pre-calculate $\rho_t \triangleq \lambda\sigma_n^2/\bar{\sigma}_t^2$.

2: **for** $t = T$ **to** 1 **do**

3: $\quad \mathbf{x}_0^{(t)} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t))$ // *Predict $\hat{\mathbf{z}}_0$ with score model as denoisor*

4: $\quad \hat{\mathbf{x}}_0^{(t)} = \arg\min_{\mathbf{x}} \|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2 + \rho_t\|\mathbf{x}-\mathbf{x}_0^{(t)}\|^2$ // *Solving data proximal subproblem*

5: $\quad \hat{\epsilon} = \frac{1}{\sqrt{1-\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0^{(t)})$ // *Calculate effective $\hat{\epsilon}(\mathbf{x}_t, \mathbf{y})$*

6: $\quad \epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

7: $\quad \mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0^{(t)} + \sqrt{1-\bar{\alpha}_{t-1}}(\sqrt{1-\zeta}\hat{\epsilon} + \sqrt{\zeta}\epsilon_t)$ // *Finish one step reverse diffusion sampling*

8: **end for**

9: **return** $\mathbf{x}_0$

Approximately $\hat{\mathbf{x}}_0^{(t)} \approx \mathbf{x}_0^{(t)} - \frac{\bar{\sigma}_t^2}{2\lambda\sigma_n^2}\nabla_{\mathbf{x}_0^{(t)}}\|\mathbf{y}-\mathcal{H}(\mathbf{x}_0^{(t)})\|^2$

Denoising Diffusion Models for Plug-and-Play Image Restoration

But where does the generative power come from?

HQS algorithm

$$\begin{cases} \mathbf{z}_k = \arg\min_{\mathbf{z}} \underbrace{\frac{1}{2(\sqrt{\lambda/\mu})^2}\|\mathbf{z}-\mathbf{x}_k\|^2}_{\text{consistence}} + \underbrace{\mathcal{P}(\mathbf{z})}_{\text{prior}} \\ \mathbf{x}_{k-1} = \arg\min_{\mathbf{x}} \underbrace{\|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2}_{\text{condition}} + \underbrace{\mu\sigma_n^2\|\mathbf{x}-\mathbf{z}_k\|^2}_{\text{consistence}} \end{cases}$$

$\mathbf{x}_k$ for denoising problem is not a "noisy image"

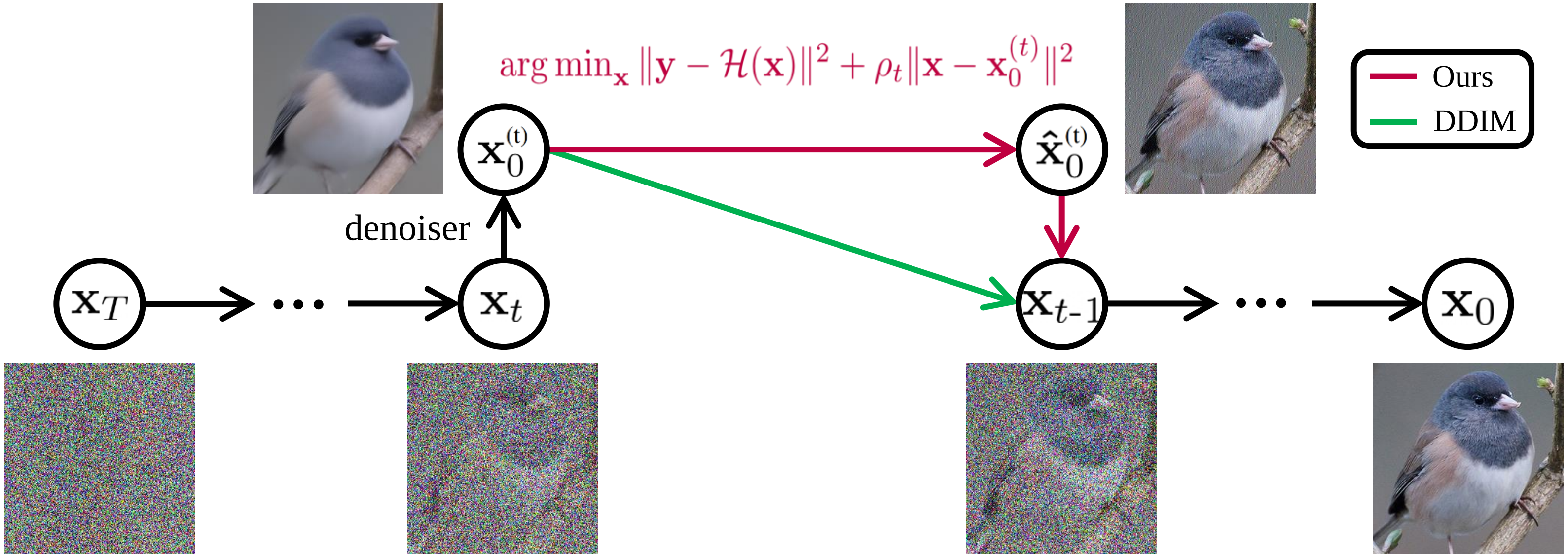
DiffPIR

$$\begin{cases} \mathbf{x}_0^{(t)} = \arg\min_{\mathbf{z}} \frac{1}{2\bar{\sigma}_t^2}\|\mathbf{z}-\mathbf{x}_t\|^2 + \mathcal{P}(\mathbf{z}) \\ \hat{\mathbf{x}}_0^{(t)} = \arg\min_{\mathbf{x}} \|\mathbf{y}-\mathcal{H}(\mathbf{x})\|^2 + \rho_t\|\mathbf{x}-\mathbf{x}_0^{(t)}\|^2 \\ \boxed{\mathbf{x}_{t-1} \longleftarrow \hat{\mathbf{x}}_0^{(t)}} \end{cases}$$

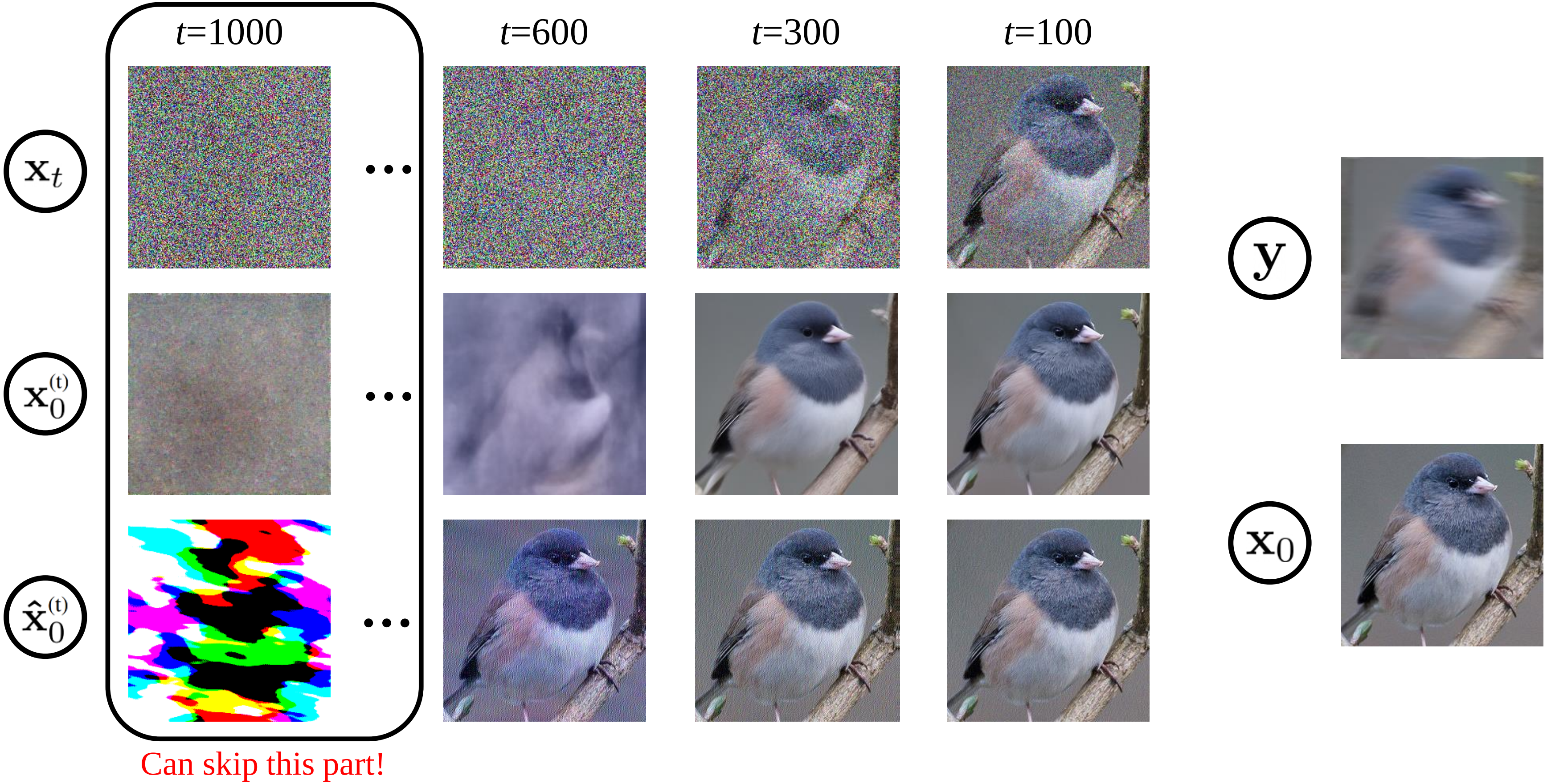
fix

Denoising Diffusion Models for Plug-and-Play Image Restoration

$$p_\theta(x_{t-1}|x_t)\begin{cases}\mathbf{x}_0^{(t)} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{(1-\bar{\alpha}_t)}\mathbf{z}_\theta(\mathbf{x}_t, t)) \\ \hat{\mathbf{x}}_0^{(t)} = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathcal{H}(\mathbf{x})\|^2 + \rho_t\|\mathbf{x} - \mathbf{x}_0^{(t)}\|^2 \\ \mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0^{(t)}(\mathbf{x}_t, \mathbf{y}) + \sqrt{1-\bar{\alpha}_{t-1}}(\sqrt{1-\zeta}\hat{\mathbf{z}}(\mathbf{x}_t, \mathbf{y}) + \sqrt{\zeta}\epsilon_t)\end{cases}$$

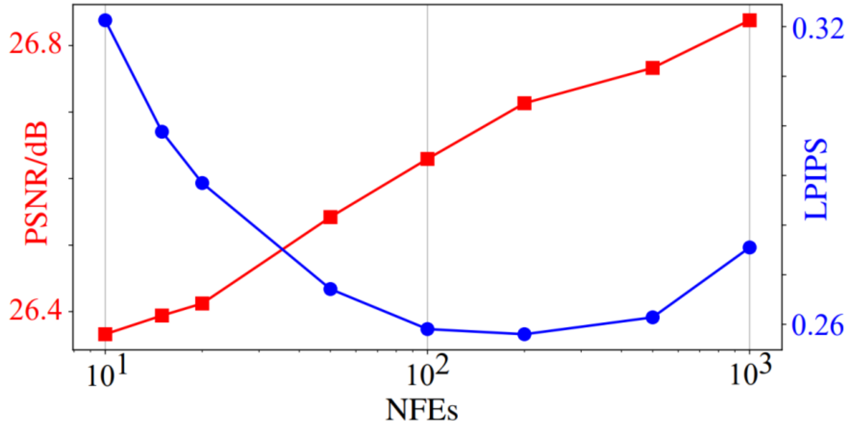


Denoising Diffusion Models for Plug-and-Play Image Restoration

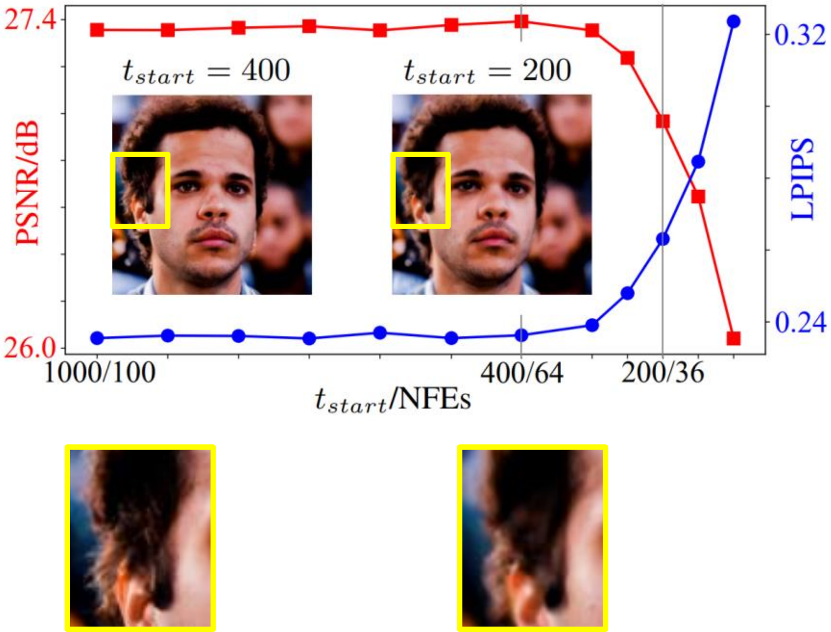


Ablation Study: Sampling Steps & Start Timestep

Effect of sampling steps

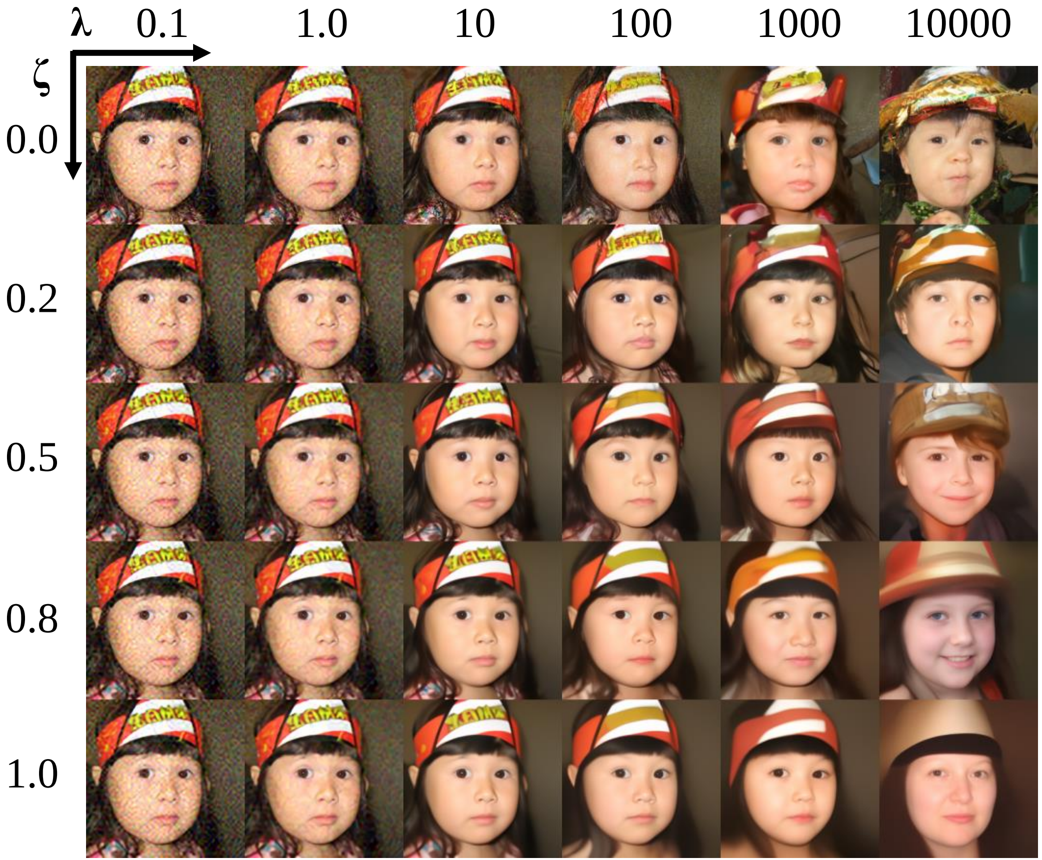


Effect of start sampling timestep



Ablation Study: Effect of Hyperparameters

- λ<1 → the noise is amplified
- λ>1000 → more *unconditional*
- ζ ~ 1 → more blurry



Content

- Preliminaries
- Methods
- Results

Quantitative Results

FFHQ		**Deblur (Gaussian)**			**Deblur (motion)**			**SR** (×4)		
Method	NFEs ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓
DiffPIR	100	27.36	**59.65**	**0.236**	**26.57**	**65.78**	**0.255**	26.64	**65.77**	0.260
DPS [8]	1000	25.46	65.57	0.247	23.31	73.31	0.289	25.77	67.01	**0.256**
DDRM [29]	20	25.93	101.89	0.298	-	-	-	27.92	89.43	0.265
DPIR [52]	>20	**27.79**	123.99	0.450	26.41	146.44	0.467	**28.03**	133.39	0.456

ImageNet		**Deblur (Gaussian)**			**Deblur (motion)**			**SR** (×4)		
Method	NFEs ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓
DiffPIR	100	22.80	**93.36**	**0.355**	**24.01**	**124.63**	**0.366**	23.18	**106.32**	0.371
DPS [8]	1000	19.58	138.80	0.434	17.75	184.45	0.491	22.16	114.93	0.383
DDRM [29]	20	22.33	160.73	0.427	-	-	-	23.89	118.55	**0.358**
DPIR [52]	>20	**23.86**	189.92	0.476	23.60	210.31	0.489	**23.99**	204.83	0.475

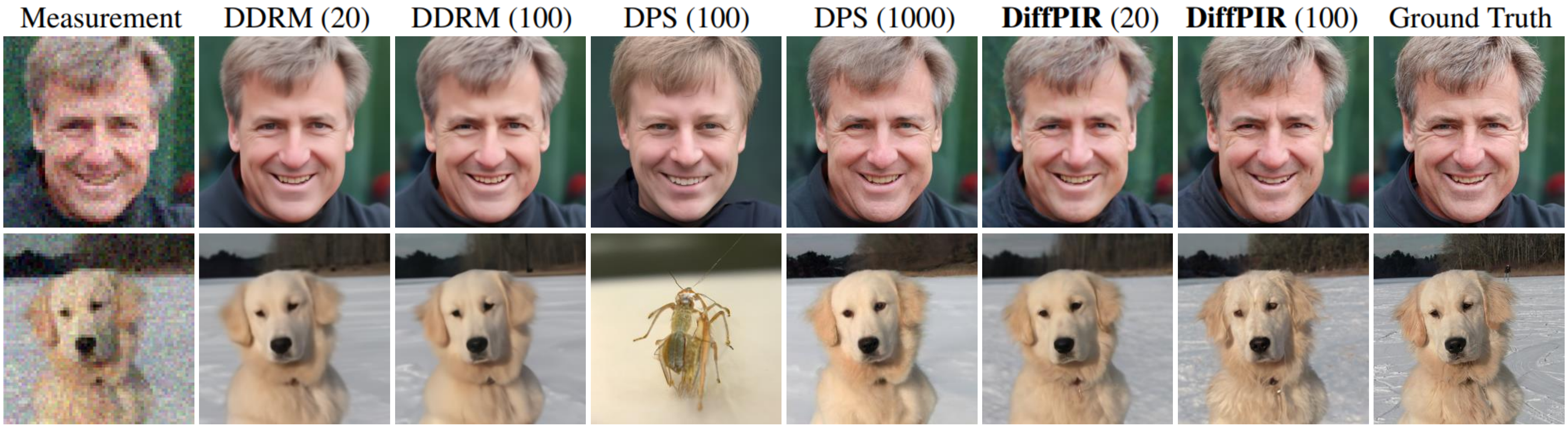
Table 1. **Noisy quantitative results on FFHQ (top) and ImageNet (bottom).** We compute the average PSNR (dB), FID and LPIPS of different methods on Gaussian deblurring, motion deblurring and 4× SR.

Quantitative Results

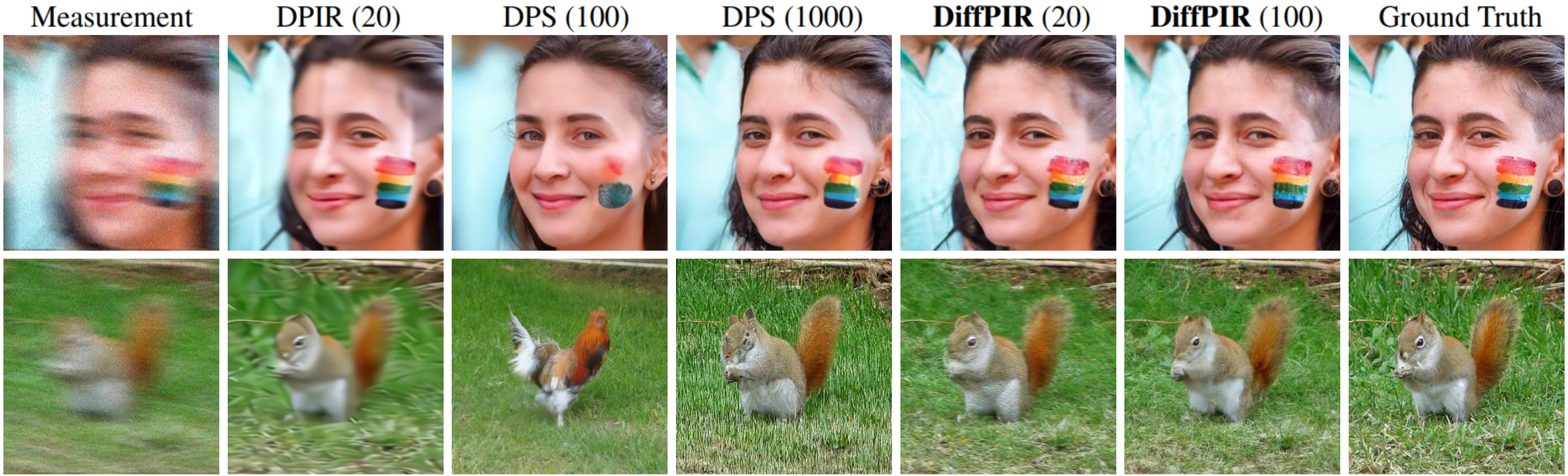
FFHQ		Inpaint (box)		Inpaint (random)			Deblur (Gaussian)			Deblur (motion)			SR (×4)		
Method	NFEs ↓	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓	PSNR ↑	FID ↓	LPIPS ↓
DiffPIR	20	35.72	0.117	34.03	30.81	0.116	30.74	46.64	0.170	37.03	20.11	0.084	29.17	58.02	0.187
DiffPIR	100	**25.64**	**0.107**	**36.17**	**13.68**	**0.066**	**31.00**	**39.27**	**0.152**	37.53	**11.54**	**0.064**	29.52	**47.80**	**0.174**
DPS [8]	1000	43.49	0.145	34.65	33.14	0.105	27.31	51.23	0.192	26.73	58.63	0.222	27.64	59.06	0.209
DDRM [29]	20	37.05	0.119	31.83	56.60	0.164	28.40	67.99	0.238	-	-	-	30.09	68.59	0.188
DPIR [52]	>20	-	-	-	-	-	30.52	96.16	0.350	**38.39**	27.55	0.233	**30.41**	96.16	0.362

Table 2. **Noiseless quantitative results on FFHQ.** We compute the average PSNR (dB), FID and LPIPS of different methods on inpainting, deblurring, and SR.

Qualitative Results: Noisy 4x SR

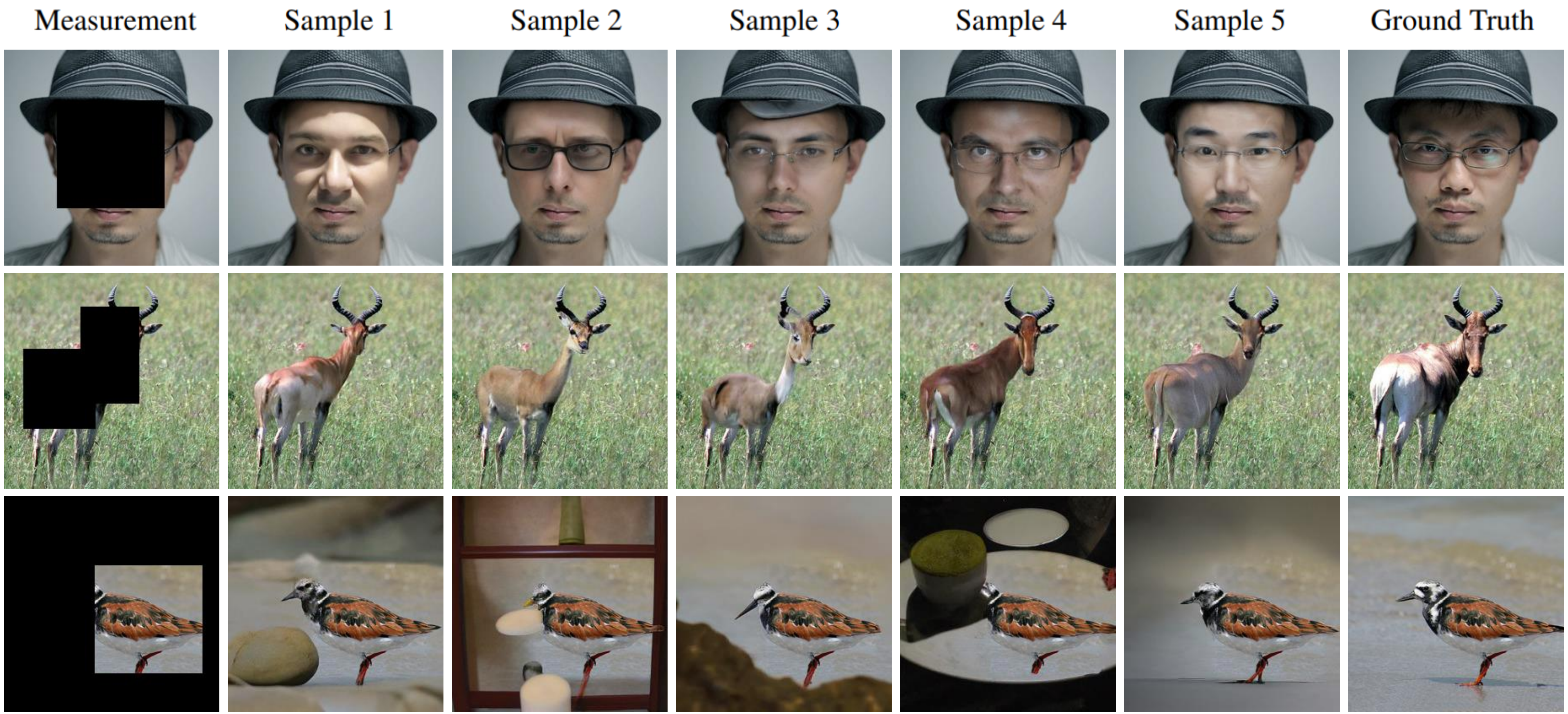


Qualitative Results: Noisy Motion Deblurring

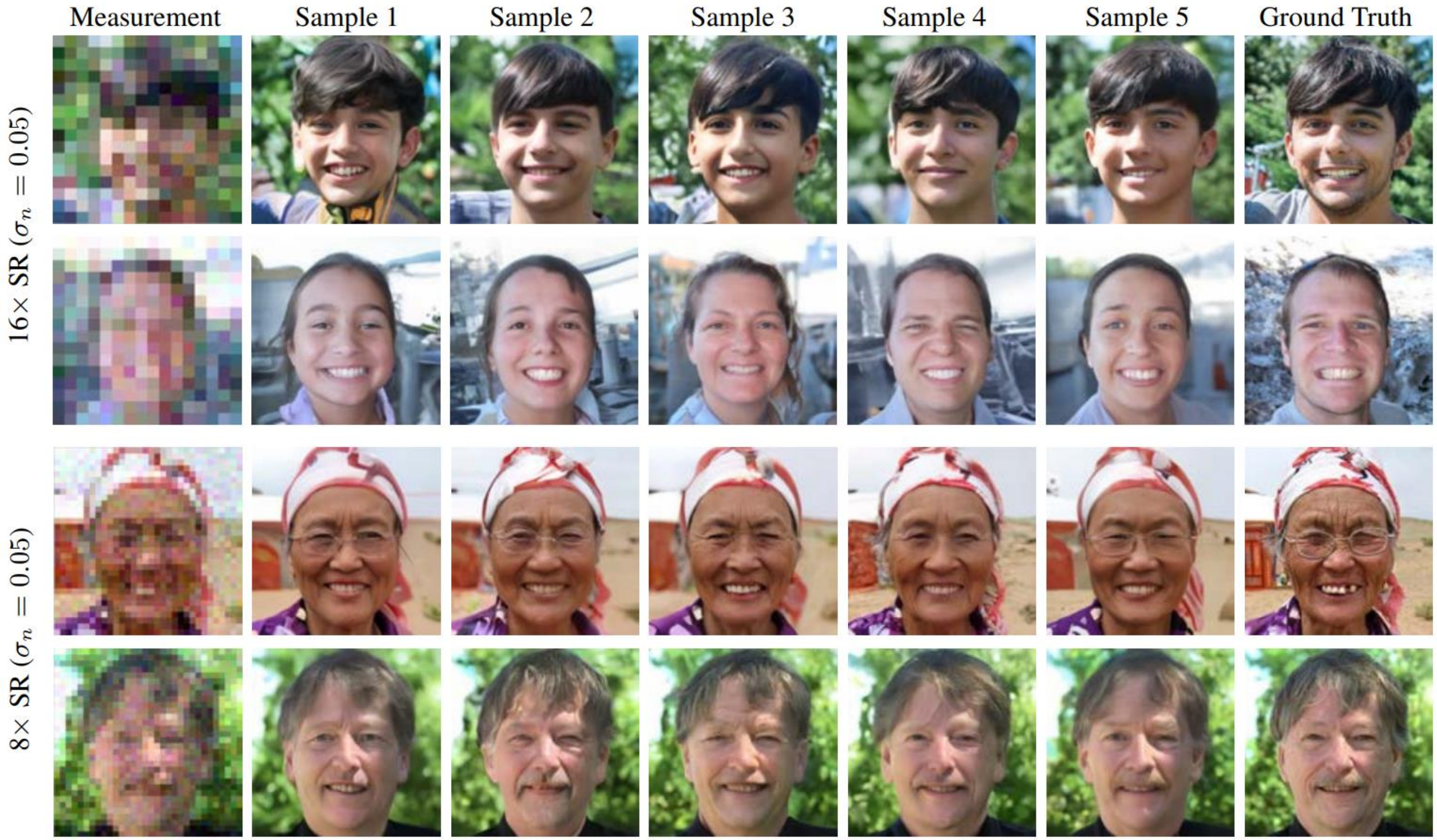


*All methods used the same diffusion model as denoiser

Diverse Reconstruction: Inpainting



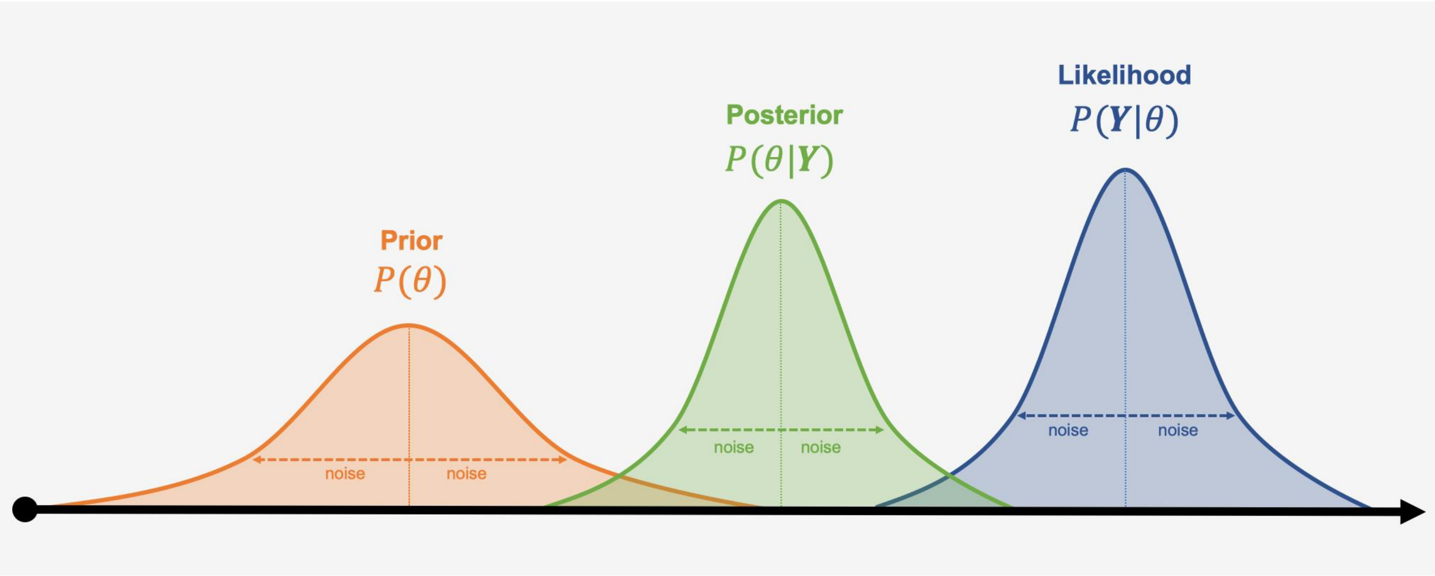
Diverse Reconstruction: Super Resolution



Thank You!

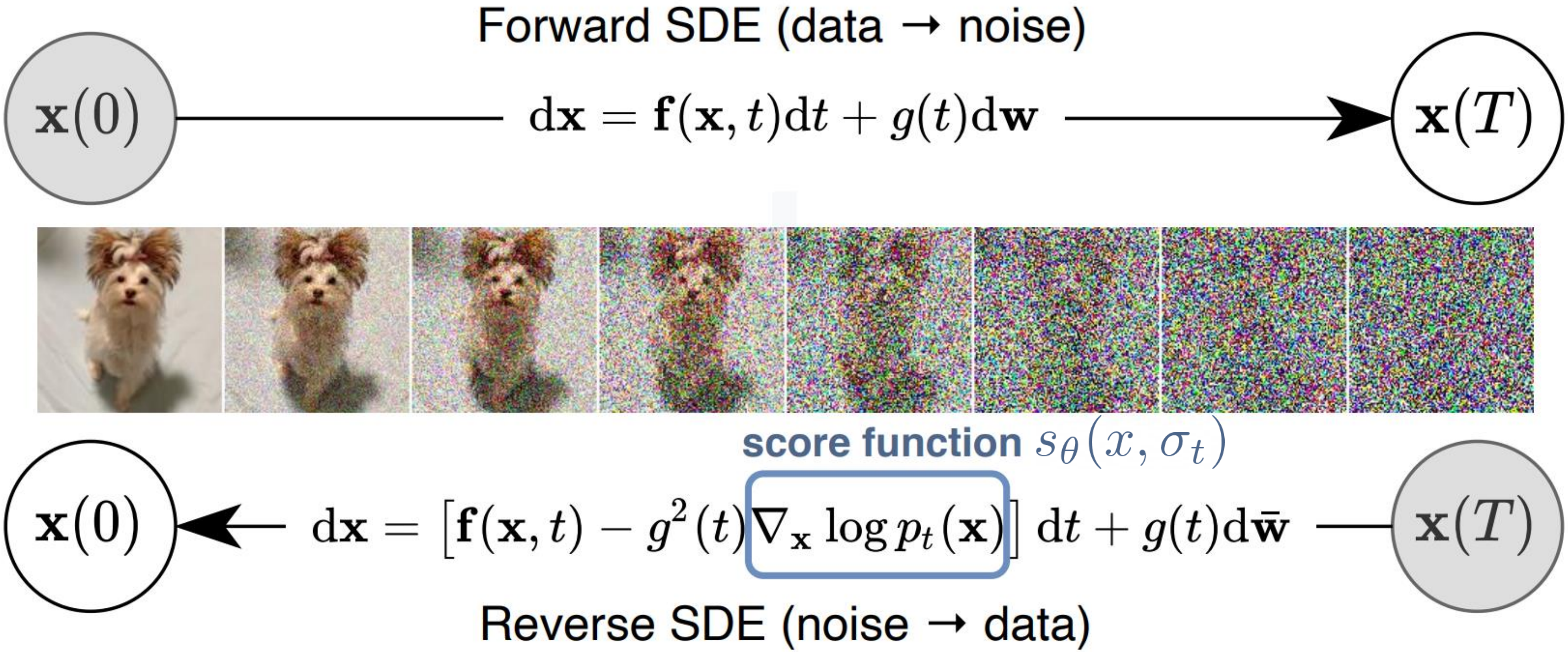
Additional Slides on Diffusion Models for IR

Sampling from the Posterior



REV. T. BAYES

SDE-based Generative Models: A Unified Framework*



*[2011.13456] Score-Based Generative Modeling through Stochastic Differential Equations (arxiv.org)

SDE-based Generative Models: A Unified Framework

Training Objective (DSM)

known Gaussian when $f(x,t)$ if affine

$$\theta^* = arg\min \mathbb{E}_{t\sim U(0,T)} \{\lambda(t)\mathbb{E}_{x(0)}\mathbb{E}_{x(t)|x(0)}[||s_\theta(x(t),t) - \nabla_{x(t)} \log \underline{p_{0t}(x(t)|x(0))}||_2^2]\}$$

Discretizations

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

SDE Form	Discrete Markov Chain	SDE Expression
Variance Exploding (VE) SDE (NCSN)	$x_i = x_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} z_{i-1}$	$\mathrm{d}x = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}w$
Variance Preserving (VP) SDE (DDPM)	$x_i = \sqrt{1-\beta_i}x_{i-1} + \sqrt{\beta_i}z_{i-1}$	$\mathrm{d}x = \frac{1}{2}\beta(t)x\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}w$

SDE-based Generative Models: A Unified Framework

Model: DDPM and SDE point of views

Score in score-based model is affine transformation of predicted noise in DDPM

$$\boldsymbol{x_t} = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\cdot\boldsymbol{\varepsilon}$$ Equivalent one step forward

$$\begin{aligned} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) &\approx \nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0) \\ &= -\frac{\boldsymbol{x}_t - \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0}{1-\bar{\alpha}_t} \\ &= -\frac{\boldsymbol{\varepsilon}}{\sqrt{1-\bar{\alpha}_t}} \\ &\approx -\frac{\boldsymbol{\varepsilon}_\theta(\boldsymbol{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}} \end{aligned}$$

Denoising score matching

Gaussian assumption

SDE-based Generative Models: A Unified Framework

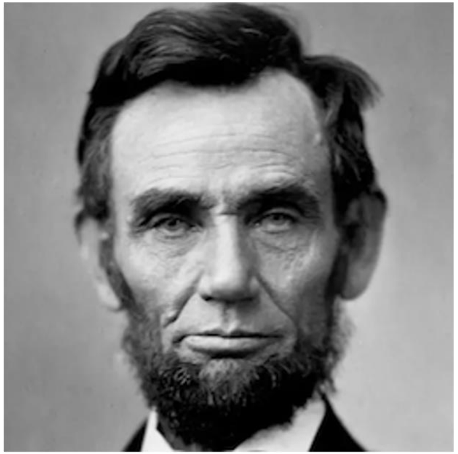
Controllable Generation

$$dx = [f(x,t) - g(t)^2 \nabla_x \log p_t(x|y)]dt + g(t)d\bar{w}$$

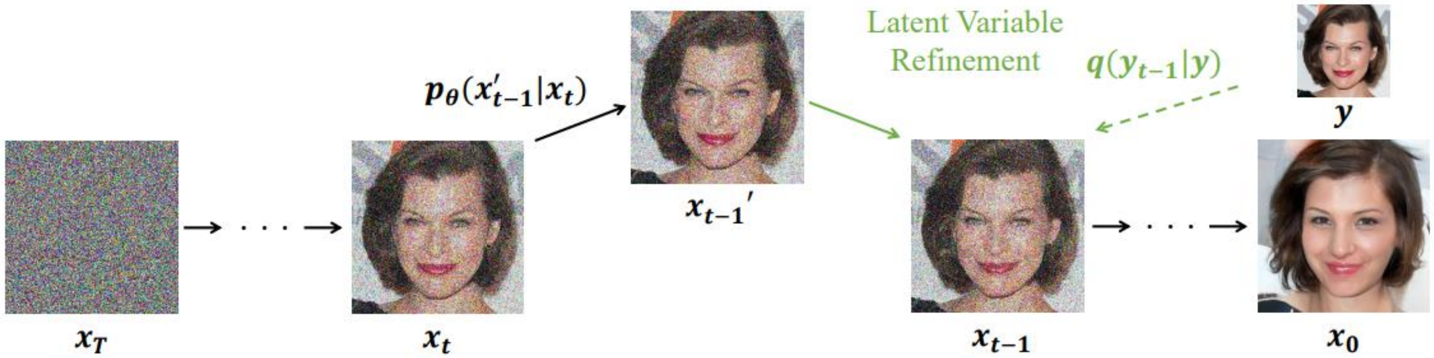
Bayesian

time-dependent classifier
(guidance term)

$$dx = \{f(x,t) - g(t)^2 [\underbrace{\nabla_x \log p_t(x)}_{\text{unconditional model}} + \nabla_x \log p_t(y|x)]\}dt + g(t)d\bar{w}$$



ILVR: Conditioning Method for DDPM*



Algorithm 1 Iterative Latent Variable Refinement

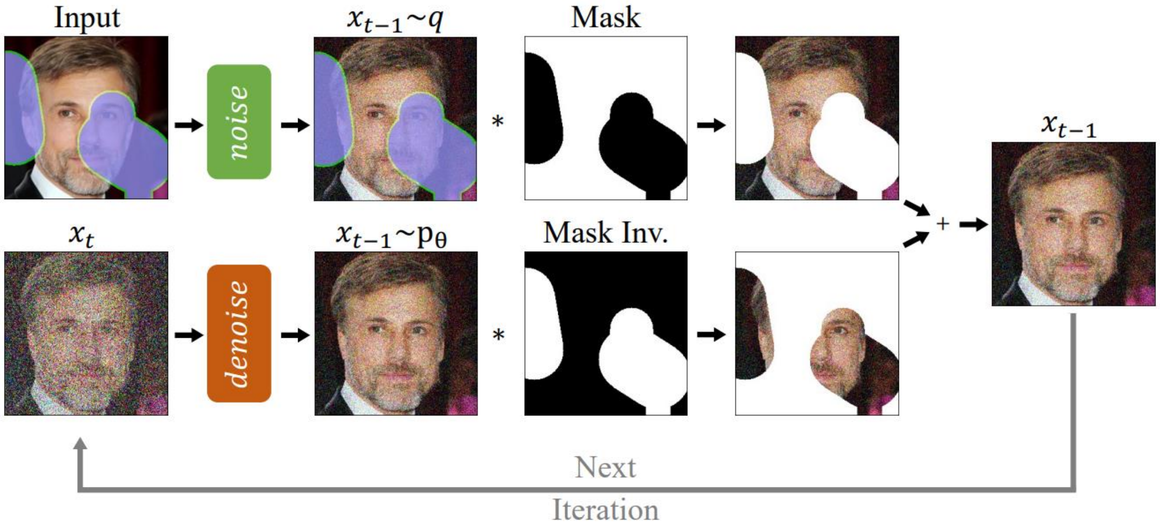
1: **Input**: Reference image y
2: **Output**: Generated image x
3: $\phi_N(\cdot)$: low-pass filter with scale N
4: Sample $x_T \sim N(\mathbf{0}, \mathbf{I})$
5: **for** $t = T, ..., 1$ **do**
6: $\quad \mathbf{z} \sim N(\mathbf{0}, \mathbf{I})$
7: $\quad x'_{t-1} \sim p_\theta(x'_{t-1}|x_t)$ ▷ unconditional proposal
8: $\quad y_{t-1} \sim q(y_{t-1}|y)$ ▷ condition encoding
9: $\quad \boxed{x_{t-1} \leftarrow \phi_N(y_{t-1}) + x'_{t-1} - \phi_N(x'_{t-1})}$
10: **end for**
11: **return** x_0

$$x_{t-1} = x'_{t-1} + a(\phi_N(y_{t-1}) - \phi_N(x'_{t-1}))$$

*[2108.02938] ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models (arxiv.org)

RePaint: Inpainting using Denoising Diffusion Probabilistic Models*

Same idea but different downstream tasks from ILVR



Algorithm 1 Inpainting using our RePaint approach.

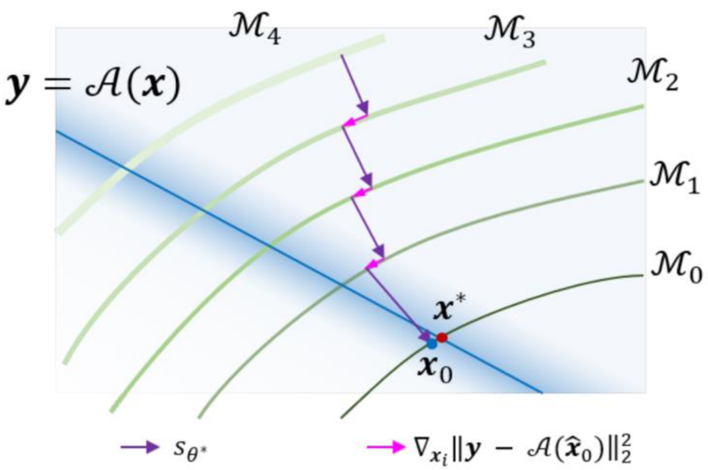
$x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
  for $u = 1, \dots, U$ **do**
    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\epsilon = \mathbf{0}$
    $x_{t-1}^{\text{known}} = \sqrt{\bar{\alpha}_t} x_0 + (1 - \bar{\alpha}_t)\epsilon$
    $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
    $x_{t-1}^{\text{unknown}} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(x_t, t)\right) + \sigma_t z$ unconditional
    $x_{t-1} = m \odot x_{t-1}^{\text{known}} + (1 - m) \odot x_{t-1}^{\text{unknown}}$
    if $u < U$ and $t > 1$ **then**
      $x_t \sim \mathcal{N}(\sqrt{1 - \beta_{t-1}} x_{t-1}, \beta_{t-1}\mathbf{I})$
    end if
  end for
end for
return x_0

*[2201.09865] RePaint: Inpainting using Denoising Diffusion Probabilistic Models (arxiv.org)

Diffusion Posterior Sampling for General Noisy Inverse Problems*

General forward model $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}_0) + \boldsymbol{n}, \quad \boldsymbol{y}, \boldsymbol{n} \in \mathbb{R}^n, \ \boldsymbol{x} \in \mathbb{R}^d$

$$p(\boldsymbol{y}|\boldsymbol{x}_0) = \frac{1}{\sqrt{(2\pi)^n \sigma^{2n}}} \exp\left[-\frac{\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x}_0)\|_2^2}{2\sigma^2}\right]$$



Algorithm 2 DPS - Gaussian [8]

Require: $N, y, \{\zeta_i\}_{i=1}^N, \{\tilde{\sigma}_i\}_{i=1^N}$

1: $x_N \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

2: **for** $i = N-1$ **to** 0 **do**

3: $\quad \hat{s} \leftarrow s_\theta(x_i, i)$

4: $\quad \hat{x}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}(x_i + \sqrt{1-\bar{\alpha}_i}\,\hat{s})$

5: $\quad z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

6: $\quad x'_{i-1} \leftarrow \frac{\sqrt{\alpha_i}(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i} x_i + \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_i}{1-\bar{\alpha}_i}\hat{x}_0 + \tilde{\sigma}_i z$

7: $\quad x_{i-1} \leftarrow x'_{i-1} - \zeta_i \nabla_{x_i}\|y - \mathcal{A}(\hat{x}_0)\|_2^2$

8: **return** x_0

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) = \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{y}|\boldsymbol{x}_t)$$

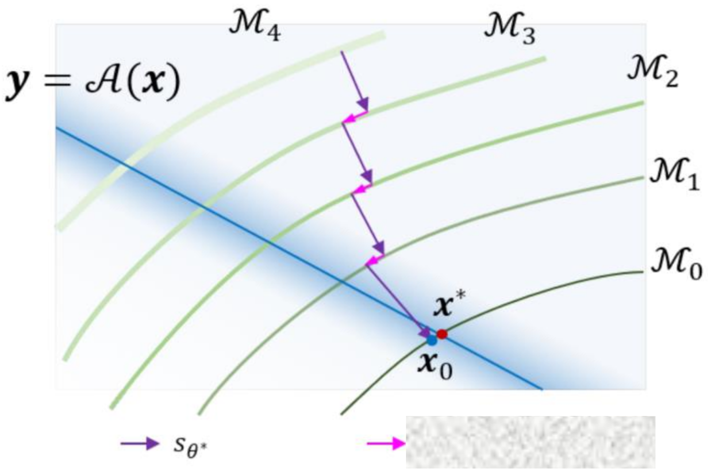
$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) \simeq \boldsymbol{s}_{\theta^*}(\boldsymbol{x}_t, t) - \rho\nabla_{\boldsymbol{x}_t}\|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0)\|_2^2$$

*[2209.14687] Diffusion Posterior Sampling for General Noisy Inverse Problems (arxiv.org)

Denoising Diffusion Models for Plug-and-Play Image Restoration

HQS as one diffusion step

$$\begin{cases} \hat{\mathbf{x}}_t = \arg\min\limits_{\mathbf{x}_t} \|\mathbf{y} - \mathcal{H}(\mathbf{x}_t)\|^2 + \mu\sigma_n^2 \|\mathbf{x}_t - \hat{\mathbf{z}}_t\|^2 \\ \hat{\mathbf{z}}_t = \arg\min\limits_{\mathbf{z}_t} \frac{1}{2(\sqrt{\lambda/\mu})^2} \|\mathbf{z}_t - \hat{\mathbf{x}}_t\|^2 + \mathcal{P}(\mathbf{z}_t) \end{cases}$$



Algorithm 2 Extended Sampling I: DPSy_t

Require: $\mathbf{s}_\theta, T, \mathbf{y}, \sigma_n, \{\sigma_t\}_{t=1}^T, \lambda$

Initialize $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

for $t = T$ **to** 1 **do**

  $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

  $\mathbf{z}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sqrt{\beta_t}\epsilon_t$ *// one step reverse diffusion sampling*

  $\mathbf{x}_{t-1} = \mathbf{z}_{t-1} - \frac{\sigma_t^2}{2\lambda\sigma_n^2}\nabla_{\mathbf{z}_{t-1}}\|\mathbf{y}_{t-1} - \mathcal{H}(\mathbf{z}_{t-1})\|^2$ *// Solving data proximal subproblem*

end for

return $\mathbf{x}_0$

$$\arg_{\mathbf{x}_t}\min \|\mathbf{y} - \mathcal{H}(\mathbf{x}_t)\|^2 + \mu\sigma_n^2\|\mathbf{x}_t - \hat{\mathbf{z}}_t\|^2$$

$$\Downarrow$$

$$\hat{\mathbf{x}}_t \approx \hat{\mathbf{z}}_t - \frac{\sigma_t^2}{2\lambda\sigma_n^2}\nabla_{\mathbf{z}_t}\|\mathbf{y} - \mathcal{H}(\mathbf{z}_t)\|^2$$

Diffusion Model Based Posterior Sampling for Noisy Linear Inverse Problems*

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y} \mid \mathbf{x}_t) \simeq \nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)$$
$$= \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{A}^T \left(\sigma^2 \mathbf{I} + \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} \mathbf{A}\mathbf{A}^T \right)^{-1} \left(\mathbf{y} - \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{A}\mathbf{x}_t \right)$$

A itself is row-orthogonal

$$[\nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)]_m = \frac{\mathbf{a}_m^T \left(\mathbf{y} - \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{A}\mathbf{x}_t \right)}{\sigma^2 \sqrt{\bar{\alpha}_t} + \frac{1-\bar{\alpha}_t}{\sqrt{\bar{\alpha}_t}} \|\mathbf{a}_m\|_2^2}$$

efficient computation via SVD

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y} \mid \mathbf{x}_t) \simeq \nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)$$
$$= \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{V}\mathbf{\Sigma} \left(\sigma^2 \mathbf{I} + \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} \mathbf{\Sigma}^2 \right)^{-1} \left(\mathbf{U}^T \mathbf{y} - \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{\Sigma}\mathbf{V}^T \mathbf{x}_t \right), \tag{12}$$

Algorithm 1: DMPS: DM based posterior sampling

Input: $\mathbf{y}, \mathbf{A}, \sigma^2, \{\tilde{\sigma}_t\}_{t=1}^T, \lambda$

Initialization: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

for $t = T$ ***to*** 1 **do**

- Draw $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \right) + \tilde{\sigma}_t \mathbf{z}_t$
- Compute $\nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)$ as (12)
- $\mathbf{x}_{t-1} = \mathbf{x}_{t-1} + \lambda \frac{1-\alpha_t}{\sqrt{\alpha_t}} \nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)$

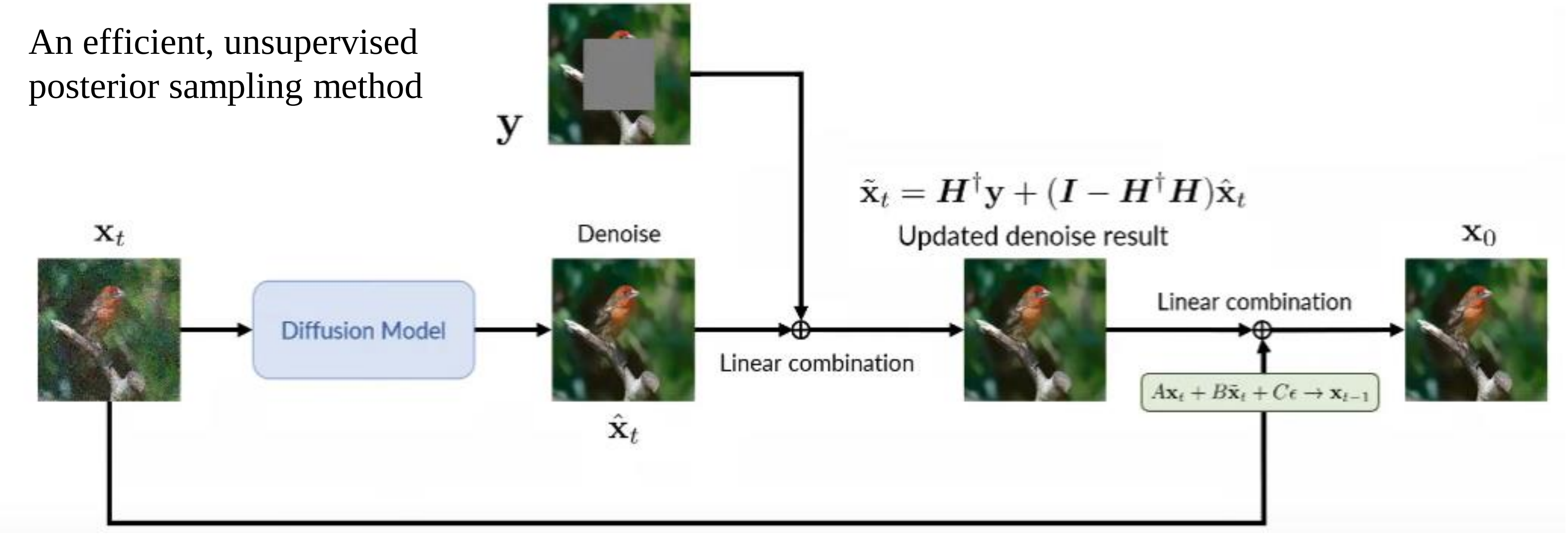
Output: $\mathbf{x}_0$

---

*[2211.12343] Diffusion Model Based Posterior Sampling for Noisy Linear Inverse Problems (arxiv.org)

Denoising Diffusion Restoration Models (DDRM)*

An efficient, unsupervised posterior sampling method



[H = Diagonal with 0 and 1's]

$$\mathbf{y} = H\mathbf{x}_0 + \mathbf{z} \quad \xRightarrow{\text{SVD}} \quad \boldsymbol{U}^\top \mathbf{y} = \boldsymbol{\Sigma}(\boldsymbol{V}^\top \mathbf{x}_0) + \boldsymbol{U}^\top \mathbf{z}$$

*[2201.11793] Denoising Diffusion Restoration Models (arxiv.org)

Denoising Diffusion Restoration Models (DDRM)*

Linear inverse problem $\mathbf{y} = \boldsymbol{H}\mathbf{x} + \mathbf{z} \Rightarrow \bar{\mathbf{y}} = \bar{\mathbf{x}}_0 + \bar{\mathbf{z}}$ $\quad q(\bar{\mathbf{y}}^{(i)}|\mathbf{x}_0) = \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_{\mathbf{y}}^2 / s_i^2)$

SVD $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \Rightarrow$

$$\bar{\mathbf{x}}_t = V^T \mathbf{x}_t$$
$$\bar{\mathbf{y}} = \Sigma^\dagger U^T \mathbf{y}$$

$$\underbrace{q^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{x}_0, \mathbf{y})}_{\text{forward}} = \begin{cases} \mathcal{N}(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}) & \text{if } s_i > 0 \\ \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_T^2) & \text{if } s_i = 0 \end{cases}$$

$$\underbrace{q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y})}_{\text{reverse}} = \begin{cases} \mathcal{N}(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{t+1}}, \eta^2\sigma_t^2) & \text{if } s_i = 0 \\ \mathcal{N}(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{\mathbf{y}}/s_i}, \eta^2\sigma_t^2) & \text{if } \sigma_t < \frac{\sigma_{\mathbf{y}}}{s_i} \\ \mathcal{N}((1-\eta_b)\bar{\mathbf{x}}_0^{(i)} + \eta_b \bar{\mathbf{y}}^{(i)}, \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2) & \text{if } \sigma_t \geq \frac{\sigma_{\mathbf{y}}}{s_i} \end{cases}$$

$$p_\theta^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}) & \text{if } s_i > 0 \\ \mathcal{N}(0, \sigma_T^2) & \text{if } s_i = 0 \end{cases}$$

singular values $s_1 \geq s_2 \geq \ldots \geq s_m$

DDRM
$$p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1}, \mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_{t+1}}, \eta^2\sigma_t^2) & \text{if } s_i = 0 \\ \mathcal{N}(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_{\mathbf{y}}/s_i}, \eta^2\sigma_t^2) & \text{if } \sigma_t < \frac{\sigma_{\mathbf{y}}}{s_i} \\ \mathcal{N}((1-\eta_b)\bar{\mathbf{x}}_{\theta,t}^{(i)} + \eta_b \bar{\mathbf{y}}^{(i)}, \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2) & \text{if } \sigma_t \geq \frac{\sigma_{\mathbf{y}}}{s_i} \end{cases}$$

y null-space

final steps

generative part

*[2201.11793] Denoising Diffusion Restoration Models (arxiv.org)

These equations in DDRM are in VESDE form

Zero-Shot Image Restoration Using Denoising Diffusion Null-Space Model*

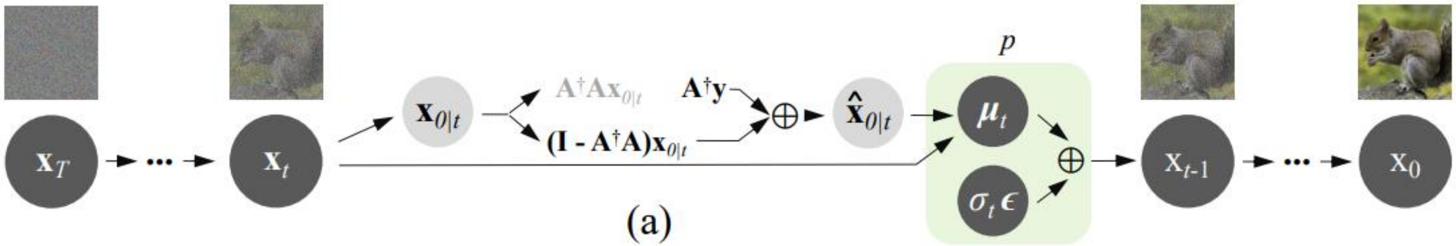
Decouple $$\mathbf{x} \equiv \underbrace{\mathbf{A}^{\dagger}\mathbf{A}\mathbf{x}}_{\text{range-space of } \mathbf{A}} + \underbrace{(\mathbf{I} - \mathbf{A}^{\dagger}\mathbf{A})\mathbf{x}}_{\text{null-space of } \mathbf{A}}$$

$$\textit{Consistency}: \quad \mathbf{A}\hat{\mathbf{x}} \equiv \mathbf{y}, \qquad \textit{Realness}: \quad \hat{\mathbf{x}} \sim q(\mathbf{x})$$

Reconstruction $$\hat{\mathbf{x}} = \mathbf{A}^{\dagger}\mathbf{y} + (\mathbf{I} - \mathbf{A}^{\dagger}\mathbf{A})\bar{\mathbf{x}}$$

find a proper $\bar{x}$ that makes the null-space term is in harmony with the range-space term

Diffusion Models $$\hat{\mathbf{x}}_{0|t} = \mathbf{A}^{\dagger}\mathbf{y} + (\mathbf{I} - \mathbf{A}^{\dagger}\mathbf{A})\mathbf{x}_{0|t}$$



(a)

*[2212.00490] Zero-Shot Image Restoration Using Denoising Diffusion Null-Space Model (arxiv.org)

Zero-Shot Image Restoration Using Denoising Diffusion Null-Space Model

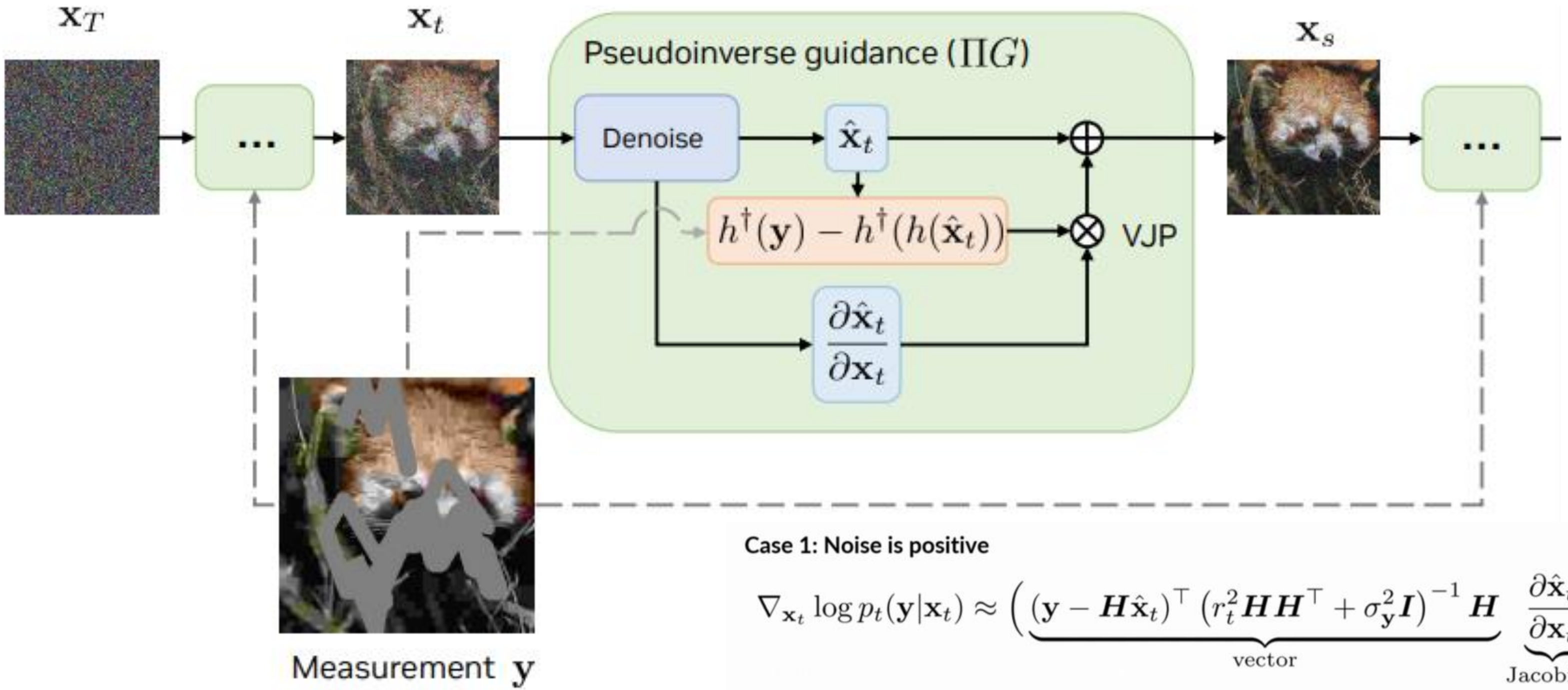
Algorithm 1 Sampling of DDNM

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, ..., 1$ **do**
    $\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \mathcal{Z}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\sqrt{1-\bar{\alpha}_t}\right)$
    $\hat{\mathbf{x}}_{0|t} = \mathbf{A}^{\dagger}\mathbf{y} + (\mathbf{I} - \mathbf{A}^{\dagger}\mathbf{A})\mathbf{x}_{0|t}$
    $\mathbf{x}_{t-1} \sim p(\mathbf{x}_{t-1}|\mathbf{x}_t, \hat{\mathbf{x}}_{0|t})$
return $\mathbf{x}_0$

Algorithm 2 Sampling of DDNM^{+}

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, ..., 1$ **do**
    $L = \min\{T - t, l\}$
    $\mathbf{x}_{t+L} \sim q(\mathbf{x}_{t+L}|\mathbf{x}_t)$
    for $j = L, ..., 0$ **do**
        $\mathbf{x}_{0|t+j} = \frac{1}{\sqrt{\bar{\alpha}_{t+j}}} \left(\mathbf{x}_{t+j} - \mathcal{Z}_{\boldsymbol{\theta}}(\mathbf{x}_{t+j}, t+j)\sqrt{1-\bar{\alpha}_{t+j}}\right)$
        $\hat{\mathbf{x}}_{0|t+j} = \mathbf{x}_{0|t+j} - \boldsymbol{\Sigma}_{t+j}\mathbf{A}^{\dagger}(\mathbf{A}\mathbf{x}_{0|t+j} - \mathbf{y})$
        $\mathbf{x}_{t+j-1} \sim \hat{p}(\mathbf{x}_{t+j-1}|\mathbf{x}_{t+j}, \hat{\mathbf{x}}_{0|t+j})$
return $\mathbf{x}_0$

ΠGDM: Pseudoinverse-Guided Diffusion Models for Inverse Problems*



Case 1: Noise is positive

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx \Big(\underbrace{(\mathbf{y} - \boldsymbol{H}\hat{\mathbf{x}}_t)^\top \left(r_t^2 \boldsymbol{H}\boldsymbol{H}^\top + \sigma_{\mathbf{y}}^2 \boldsymbol{I}\right)^{-1} \boldsymbol{H}}_{\text{vector}} \underbrace{\frac{\partial \hat{\mathbf{x}}_t}{\partial \mathbf{x}_t}}_{\text{Jacobian}}\Big)^\top.$$

Backprop through diffusion model

Case 2: Noise is zero

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx r_t^{-2}\Big((\boldsymbol{H}^\dagger \mathbf{y} - \boldsymbol{H}^\dagger \boldsymbol{H}\hat{\mathbf{x}}_t)^\top \frac{\partial \hat{\mathbf{x}}_t}{\partial \mathbf{x}_t}\Big)^\top$$

$\boldsymbol{H}^\dagger = \boldsymbol{H}^\top(\boldsymbol{H}\boldsymbol{H}^\top)^{-1}$ is matrix pseudoinverse!

[*Pseudoinverse-Guided Diffusion Models for Inverse Problems | OpenReview]

On Equivalence of Diffusion Posterior Sampling Strategies

DPS $\hat{\mathbf{x}}_t \approx \mathbf{x}_t - \zeta_t \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y} \mid \mathbf{x}_t) \simeq \nabla_{\mathbf{x}_t} \log \tilde{p}(\mathbf{y} \mid \mathbf{x}_t)$$

$$= \frac{1}{\sqrt{\bar{\alpha}_t}} \boxed{\mathbf{A}^T} \underbrace{\left(\sigma^2 \mathbf{I} + \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} \mathbf{A}\mathbf{A}^T\right)^{-1}}_{\text{some coefficient}} \boxed{\left(\mathbf{y} - \frac{1}{\sqrt{\bar{\alpha}_t}} \mathbf{A}\mathbf{x}_t\right)}$$

DDNM

$$\begin{aligned} \hat{\mathbf{x}}_{0|t} &= \mathbf{A}^\dagger \mathbf{y} + (\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\mathbf{x}_{0|t} \\ &= \mathbf{x}_{0|t} - (\mathbf{A}^\dagger \mathbf{A}\mathbf{x}_{0|t} - \mathbf{A}^\dagger \mathbf{y}) \\ &= \mathbf{x}_{0|t} - \boxed{\mathbf{A}^\dagger (\mathbf{A}\mathbf{x}_{0|t} - \mathbf{y})} \end{aligned}$$

Solving Image Restoration Tasks Iteratively (Traditional PnP Methods)

[Image Restoration by Iterative Denoising and Backward Projections]

Algorithm 2 Iterative Denoising and Backward Projections (IDBP)

Input: $\boldsymbol{H}, \boldsymbol{y}, \sigma_e$, denoising operator $\mathcal{D}(\cdot;\sigma)$, stopping criterion. $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{e}$, such that $\boldsymbol{e} \sim \mathcal{N}(\boldsymbol{0}, \sigma_e^2 \boldsymbol{I}_m)$ and $\boldsymbol{x}$ is an unknown signal whose prior model is specified by $\mathcal{D}(\cdot;\sigma)$.

Output: $\hat{\boldsymbol{x}}$ an estimate for $\boldsymbol{x}$.

Initialize: $\tilde{\boldsymbol{y}}_0 =$ some initialization, $k = 0$, δ approx. satisfying (12).

while *stopping criterion not met* **do**

- $k = k + 1$;
- $\tilde{\boldsymbol{x}}_k = \mathcal{D}(\tilde{\boldsymbol{y}}_{k-1}; \sigma_e + \delta)$;
- $\tilde{\boldsymbol{y}}_k = \boldsymbol{H}^\dagger \boldsymbol{y} + (\boldsymbol{I}_n - \boldsymbol{H}^\dagger \boldsymbol{H})\tilde{\boldsymbol{x}}_k$;

end

$\hat{\boldsymbol{x}} = \tilde{\boldsymbol{x}}_k$;

[Plug-and-Play Image Restoration with Deep Denoiser Prior]

Algorithm 1: Plug-and-play image restoration with deep denoiser prior (DPIR).

Input : Deep denoiser prior model, degraded image $\mathbf{y}$, degradation operation $\mathcal{T}$, image noise level σ, σ_k of denoiser prior model at k-th iteration for a total of K iterations, trade-off parameter λ.

Output: Restored image $\mathbf{z}_K$.

1 Initialize $\mathbf{z}_0$ from $\mathbf{y}$, pre-calculate $\alpha_k \triangleq \lambda\sigma^2/\sigma_k^2$.

2 **for** $k = 1, 2, \cdots, K$ **do**

3 $\quad \mathbf{x}_k = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathcal{T}(\mathbf{x})\|^2 + \alpha_k \|\mathbf{x} - \mathbf{z}_{k-1}\|^2$; // *Solving data subproblem*

4 $\quad \mathbf{z}_k = Denoiser(\mathbf{x}_k, \sigma_k)$; // *Denoising with deep DRUNet denoiser and periodical geometric self-ensemble*

5 **end**

What are the advantages of diffusion sampling framework?

→ well-defined path connecting two distributions

In our experiment all methods use the same diffusion model checkpoints

→ schedule is all you need!!

Sampling from Langevin Dynamics?

[2103.04715] Bayesian imaging using Plug & Play priors: when Langevin meets Tweedie (arxiv.org)

Algorithm 1 PnP-ULA

Require: $n \in \mathbb{N}$, $y \in \mathbb{R}^m$, $\varepsilon, \lambda, \alpha, \delta > 0$, $\mathsf{C} \subset \mathbb{R}^d$ convex and compact
Ensure: $2\lambda(2\mathrm{L}_y + \alpha\mathrm{L}/\varepsilon) \leqslant 1$ and $\delta < (1/3)(\mathrm{L}_y + 1/\lambda + \alpha\mathrm{L}/\varepsilon)^{-1}$
  Initialization: Set $X_0 \in \mathbb{R}^d$ and $k = 0$.
  for $k = 0 : N$ **do**
    $Z_{k+1} \sim \mathcal{N}(0, \mathrm{Id})$
    $X_{k+1} = X_k + \delta\nabla\log(p(y|X_k)) + (\alpha\delta/\varepsilon)(D_\varepsilon(X_k) - X_k) + (\delta/\lambda)(\Pi_{\mathsf{C}}(X_k) - X_k) + \sqrt{2\delta}Z_{k+1}$
  end for
  return $\{X_k : k \in \{0, \ldots, N+1\}\}$

[1611.02862] The Little Engine that Could: Regularization by Denoising (RED) (arxiv.org)
An Interpretation Of Regularization By Denoising And Its Application With The Back-Projected Fidelity Term

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \mu\left(\nabla\ell_{LS}(\boldsymbol{x}_k) + \lambda\boldsymbol{g}_{\text{RED}}(\boldsymbol{x}_k)\right) \\ &= \boldsymbol{x}_k - \mu\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \lambda(\boldsymbol{x}_k - \mathcal{D}(\boldsymbol{x}_k;\sigma))\right) \end{aligned}$$

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \mu\left(\nabla\ell_{BP}(\boldsymbol{x}_k) + \lambda\boldsymbol{g}_{\text{RED}}(\boldsymbol{x}_k)\right) \\ &= \boldsymbol{x}_k - \mu\left(\boldsymbol{A}^\dagger(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \lambda(\boldsymbol{x}_k - \mathcal{D}(\boldsymbol{x}_k;\sigma))\right) \end{aligned}$$

PnP Generative Networks: Conditional Iterative Generation of Images in Latent Space*

Metropolis-adjusted Langevin algorithm (MALA) sampler

$$x_{t+1} = x_t + \epsilon_1 \frac{\partial \log p(x_t)}{\partial x_t} + \epsilon_2 \frac{\partial \log p(y = y_c | x_t)}{\partial x_t} + N(0, \epsilon_3^2)$$

$$\frac{\partial \log p(x)}{\partial x} \approx \frac{R_x(x) - x}{\sigma^2}$$

Denoising AutoEncoder output as score

$$h_{t+1} = h_t + \epsilon_1 (R_h(h_t) - h_t) + \epsilon_2 \frac{\partial \log C_c(G(h_t))}{\partial G(h_t)} \frac{\partial G(h_t)}{\partial h_t} + N(0, \epsilon_3^2)$$

latent	prior	condition	noise
	realistic	*e.g. class-specific*	*diverse*

*[1612.00005] Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space (arxiv.org)