Bayesian Inverse Problem with Denoising Diffusion model priors

Yazid Janati El Idrissi, Eric Moulines
CMAP, Ecole polytechnique

joint work with Gabriel Cardoso (CMAP), Sylvain Le Corff (LPSM)

Introduction

Generative modeling

We have a dataset $\mathcal{D}_N := \{X^1, \ldots, X^N\}$, where $X^i \in \mathbb{R}^{d_x}$.

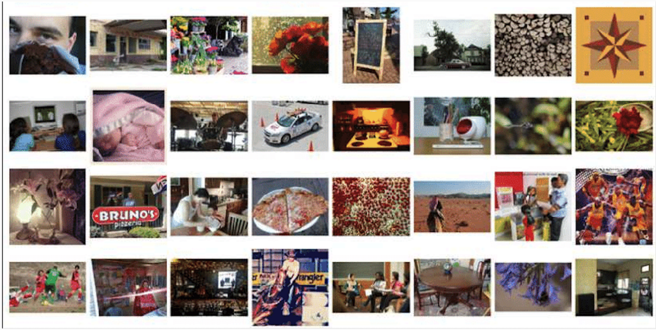


Figure 1: Samples from the `ImageNet` dataset.

Generative modeling

We have a dataset $\mathcal{D}_N := \{X^1, \dots, X^N\}$, where $X^i \in \mathbb{R}^{d_x}$.

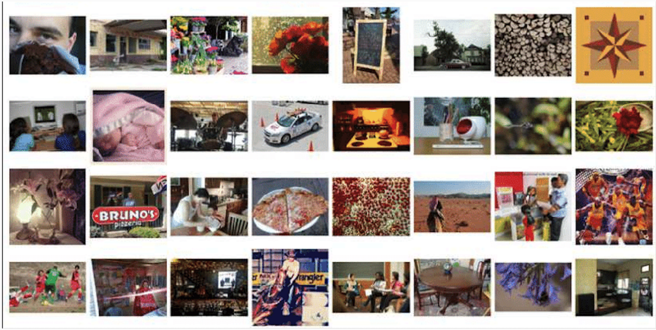


Figure 1: Samples from the `ImageNet` dataset.

Modeling assumption

$(X^1, \dots, X^N)$ are samples from some **unknown** distribution π_{data}

Generative modeling

① Approximate π_{data} with a parametric model.

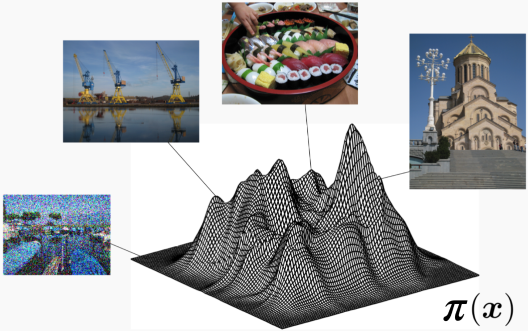


Figure 2: data distribution.

Bayesian inverse problems

② Sample reconstructions from the posterior distribution.

Figure 3: Reconstruction problems. Figure adapted from Lugmayr et al. (2022).

Generative modeling

① Approximate π_{data} with a parametric model p^{θ}.

Ackley et al. (1985); Kingma and Welling (2013); Goodfellow et al. (2014); Rezende and Mohamed (2015); Sohl-Dickstein et al. (2015); Ho et al. (2020); Song et al. (2021b)

Generative modeling

① Approximate π_{data} with a parametric model p^{θ}.

1. Choose a suitable parametric form for p^{θ}.

Ackley et al. (1985); Kingma and Welling (2013); Goodfellow et al. (2014); Rezende and Mohamed (2015); Sohl-Dickstein et al. (2015); Ho et al. (2020); Song et al. (2021b)

Generative modeling

① Approximate π_{data} with a parametric model p^θ.

1. Choose a suitable parametric form for p^θ.
2. Train p^θ to approximate π using the samples $(X^1, \ldots, X^N) \sim \pi$.

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} -\log \mathsf{p}^\theta(X^i)\,.$$

$\rightsquigarrow$ Minimize $\mathcal{L}(\theta) \rightarrow$ find optimal parameter θ_*.

Ackley et al. (1985); Kingma and Welling (2013); Goodfellow et al. (2014); Rezende and Mohamed (2015); Sohl-Dickstein et al. (2015); Ho et al. (2020); Song et al. (2021b)

Posterior sampling

(2) Perform controlled generation using p^{θ_*}.

$\rightsquigarrow$ Target distribution: weight p^{θ_*} with a function $x \mapsto g(x)$

Posterior sampling

(2) Perform controlled generation using p^{θ_*}.

⇝ Target distribution: weight p^{θ_*} with a function $x \mapsto g(x)$

$$\phi(\mathrm{d}x) = \frac{g(x)\mathsf{p}^{\theta_*}(\mathrm{d}x)}{\int g(z)\mathsf{p}^{\theta_*}(\mathrm{d}z)},$$

⇝ Posterior sampling: $g(x) = p(y|x)$.

⇝ Reinforcement learning: g is a reward function.

Denoising diffusion models

Introduction

- A denoising diffusion probabilistic model (DDPM) makes use of two Markov chains:
  1. a forward chain (process) that perturbs data to noise,
  2. a reverse chain (process) that converts noise back to data.
- The forward chain is typically hand-designed with the goal to transform the data distribution π_{data} into a (simple) reference distribution π_{ref} (e.g., standard Gaussian)
- The backward chain reverses the forward chain by learning transition kernels.
- New data points are generated by first sampling a random vector from the reference distribution, followed by ancestral sampling through the backward Markov chain.

Forward process

- Given a data distribution $x_0 \sim \pi_{\text{data}}(\mathrm{d}x_0) = q_0(\mathrm{d}x_0)$, the forward Markov chain generates a sequence of random variables $x_1, x_2 \ldots x_T$ with transition kernel $q_{t|t-1}(\mathrm{d}x_t \mid x_{t-1})$.
- The joint distribution of $x_1, x_2 \ldots x_T$ conditioned on x_0, denoted as $q_{0:T}(\mathrm{d}(x_1, \ldots, x_T) \mid x_0)$, may be written as

$$q_{0:T}(\mathrm{d}(x_1, \ldots, x_T) \mid x_0) = \prod_{t=1}^{T} q_{t|t-1}(\mathrm{d}x_t \mid x_{t-1}).$$

- In DDPMs, we handcraft the transition kernel $q_{t|t-1}(\mathrm{d}x_t \mid x_{t-1})$ to incrementally transform the data distribution $q_0(\mathrm{d}x_0)$ into a tractable reference distribution.
  - Typical design: Gaussian perturbation

$$q_{t|t-1}(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t \mathbf{I}\right),$$

    where $\beta_t \in (0, 1)$ is a hyperparameter chosen ahead of model training.

Forward process

- Gaussian transition kernel allows us to obtain the analytical form of $q_{t|0}(x_t \mid x_0)$ for all $t \in \{0, 1, \cdots, T\}$. Setting $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=0}^{t} \alpha_s$, we have

$$q_{t|0}(x_t \mid x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t)\mathbf{I}\right).$$

- Given x_0, we can easily obtain a sample of x_t by sampling a Gaussian vector $\epsilon_t \sim \mathcal{N}(0, \mathrm{I})$ and applying the transformation

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}_t.$$

- When $\bar{\alpha}_T \approx 0, x_T$ is almost Gaussian in distribution,

$$q_T(x_T) := \int q_{T|0}(x_T \mid x_0)\, q_0(x_0)\, \mathrm{d}x_0 \approx \mathcal{N}(x_T; \mathbf{0}, \mathbf{I}).$$

Backward process

- For generating new data samples, DDPMs start by sampling the reference distribution and then gradually remove noise by running a learnable Markov chain backward in time.
- The reverse Markov chain is parameterized by a reference distribution $\pi_{\text{ref}}(x_T) = \mathcal{N}(x_T; \mathbf{0}, \mathrm{I})$ and a learnable transition kernel

$$p^{\theta}_{t-1|t}(x_{t-1} \mid x_t) = \mathcal{N}\left(x_{t-1}; \mu^{\theta}_t(x_t), \Sigma^{\theta}_t(x_t)\right)$$

  where θ denotes model parameters, and the mean $\mu^{\theta}_t(x_t)$ and variance $\Sigma^{\theta}_t(x_t)$ are parameterized by deep neural networks.
- Data generation
  - Sample $x_T \sim \pi_{\text{ref}}(\cdot)$,
  - iteratively sample $x_{t-1} \sim p^{\theta}_{t-1|t}(\cdot \mid x_t)$ until $t = 1$.

Diffusion model principles

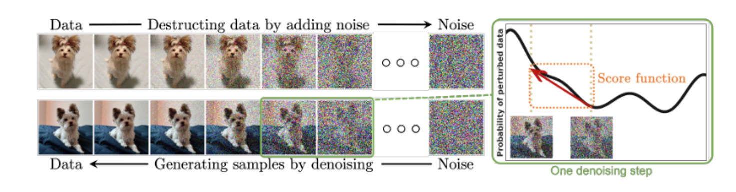


Figure 4: Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise.

Variational Inference

- Objective: Adjust the parameter θ so that the joint distribution of the reverse Markov chain

$$p_{0:T}^{\theta}\left(x_0, x_1, \cdots, x_T\right) = p_{\mathrm{ref}}(x_T) \prod_{t=1}^{T} p_{t-1|t}^{\theta}(x_{t-1} \mid x_t)$$

matches

$$q_{0:T}\left(x_0, x_1, \cdots, x_T\right) := q_0\left(x_0\right) \prod_{t=1}^{T} q_{t|t-1}\left(x_t \mid x_{t-1}\right).$$

- Training is performed by maximizing a variational bound:

$$\mathbb{E}_{q_0}\left[-\log p^{\theta}\left(x_0\right)\right] \leq \mathbb{E}_{q_{0:T}}\left[-\log \frac{p_{0:T}^{\theta}\left(x_{0:T}\right)}{q_{1:T|0}\left(x_{1:T} \mid x_0\right)}\right]$$

$$= \mathbb{E}_{q_{0:T}}\left[-\log p_T\left(x_T\right) - \sum_{t\geq 1} \log \frac{p_{t-1|t}^{\theta}\left(x_{t-1} \mid x_t\right)}{q_{t|t-1}\left(x_t | x_{t-1}\right)}\right] =: L^{\theta}$$

Variational inference with variance reduction

- L^θ might be rewritten using the backward representation of the forward noising process

$$q_{1:T|0}(x_{1:T}|x_0) = \prod_{t=1}^{T} q_{t|t-1}(x_t|x_{t-1})$$
$$= q_{T|0}(x_T|x_0) \prod_{t=2}^{T} q_{t-1|t}(x_{t-1}|x_t, x_0)$$

- With this backward decomposition, L^θ writes

$$L^\theta = \mathbb{E}_{q_{0:T}}\left[- \log \frac{p_T(x_T)}{q_{T|0}(x_T \mid x_0)} - \sum_{t=2}^{T} \log \frac{p^\theta_{t-1|t}(x_{t-1} \mid x_t)}{q_{t-1|t,0}(x_{t-1} \mid x_t, x_0)} - \log p^\theta_{0|1}(x_0 \mid x_1)\right]$$

$$= \mathbb{E}_{q_{0:T}}\left[D_{\mathrm{KL}}\left(q_{T|0}(\cdot \mid x_0) \| p_T(\cdot)\right) + \sum_{t=2}^{T} D_{\mathrm{KL}}\left(q_{t-1|t,0}(\cdot \mid x_t, x_0) \| p^\theta_{t-1|t}(\cdot \mid x_t)\right) - \log p^\theta_{0|1}(x_0 \mid x_1)\right]$$

Variational inference with variance reduction

- forward posteriors are tractable when conditioned on x_0 :

$$q_{t-1|t,0}\left(x_{t-1} \mid x_t, x_0\right) = \mathcal{N}\left(x_{t-1}; \tilde{\boldsymbol{\mu}}_t\left(x_t, x_0\right), \tilde{\beta}_t \mathbf{I}\right)$$

$$\text{where} \quad \tilde{\boldsymbol{\mu}}_t\left(x_t, x_0\right) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t} x_t$$

$$\text{and} \quad \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

- KL divergences are comparisons between Gaussian distributions with closed form expressions: taking $\Sigma_t^\theta(x_t) = \tilde{\beta}_t \mathrm{I}$,

$$D_{\mathrm{KL}}\left(q_{t-1|t,0}\left(\cdot \mid x_t, x_0\right) \| p^\theta_{t-1|t}\left(\cdot \mid x_t\right)\right) = \frac{1}{2\tilde{\beta}_t}\|\tilde{\boldsymbol{\mu}}_t(x_t, x_0) - \mu_t^\theta(x_t)\|^2 .$$

Variational inference with variance reduction

- Setting

$$\mu_t^\theta(x_t) = \tilde{\mu}_t(x_t, \hat{x}_{0|t}^\theta(x_t)),$$

we get

$$D_{\mathrm{KL}}\left(q_{t-1|t,0}\left(\cdot \mid x_t, x_0\right) \| p_{t-1|t}^\theta\left(\cdot \mid x_t\right)\right) = w_t \|x_0 - \hat{x}_{0|t}^\theta(x_t)\|^2 .$$

with $w_t = \bar{\alpha}_{t-1}\beta_t/(1-\bar{\alpha}_{t-1})(1-\bar{\alpha}_t)$.

- Hence, criterion L^θ rewrites

$$L^\theta = \sum_{t=2}^{T} w_t \mathbb{E}_{q_0 \otimes \mathcal{N}(0,\mathrm{I})}[\|x_0 - \hat{x}_{0|t}^\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon})\|^2]$$

which amount to compute $\hat{x}_{0|t}^\theta(x_t)$ as a predictor of the initial state x_0 from the current state x_t.

- This criterion is the denoising score matching.

Noise prediction

- Using that $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t$, we have

$$x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t)$$

- Choosing $\hat{x}^{\theta}_{0|t}(x_t) = (1/\sqrt{\bar{\alpha}_t})(x_t - \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\epsilon}}^{\theta}_{0|t}(x_t))$, the criterion L^{θ} may be equivalently expressed as

$$L^{\theta} = \sum_{t=2}^{T} \tilde{w}_t \mathbb{E}_{q_0 \otimes \mathcal{N}(0,\mathrm{I})}[\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}^{\theta}_{0|t}(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon})\|^2]$$

where

$$\tilde{w}_t = \frac{\beta_t}{\alpha_t(1-\bar{\alpha}_{t-1})}$$

A continuous-time perspective

Ornstein-Uhlenbeck Noising process

- Consider a diffusion process $\{X_t\}_{t=0}^{T}$ that starts from the data distribution $q_0(\mathrm{d}x) \equiv \pi_{\mathsf{data}}(\mathrm{d}x)$ at time $t = 0$. The notation $q_t(\mathrm{d}x)$ refers to the marginal distribution of the diffusion at time $0 \leq t \leq T$.
- Assume furthermore that at time $t = T$, the marginal distribution is (very close to) a reference distribution $q_T(\mathrm{d}x) = \pi_{\mathsf{ref}}(\mathrm{d}x)$ that is straightforward to sample from, e.g. $\mathcal{N}(0, \mathrm{I})$.
- This diffusion process is the noising process. It is often chosen as an Ornstein-Uhlenbeck (OU) diffusion,

$$\mathrm{d}X_t = -\frac{1}{2} X_t \mathrm{d}t + \mathrm{d}W_t$$

OU noising process

- OU diffusion is reversible w.r.t. $\pi_{\text{ref}} = \mathcal{N}(0, \mathrm{I})$: the conditional distribution of $X_{t+s} \mid X_t = x_t$ is $\mathcal{N}(\alpha_s x_t, \sigma_s^2 \mathrm{I})$, with

$$\alpha_s = \sqrt{1 - \sigma_s^2} \quad \sigma_s^2 = 1 - \mathrm{e}^{-s}$$

- Denote

$$F(s, x, y) \propto \exp\left\{-\frac{(y - \alpha_s x)^2}{2\sigma_s^2}\right\}.$$

the forward transition from x to y in " s " amount of time.

Reverse diffusion I (informal)

- the DDPM strategy consists in sampling from the Gaussian reference measure π_{ref} at time $t = T$ and simulate the OU process backward in time.
- In other words, one would like to simulate from the reverse process $\overleftarrow{X}_t$ defined as

$$\overleftarrow{X}_s = X_{T-s}$$

- The reverse process is distributed as $\overleftarrow{X}_0 \sim \pi_{\text{ref}}$ at time $t = 0$ and, crucially, we have that $\overleftarrow{X}_T \sim \pi_{\text{data}}$.
- The reverse diffusion follows the dynamics (Hausmann, Pardoux, 1986; Millet, Nualart, Sanz, 1989)

$$d\overleftarrow{X}_t = +\frac{1}{2}\overleftarrow{X}_t dt + \nabla \log q_{T-t}\left(\overleftarrow{X}_t\right) \mathrm{d}t + \mathrm{d}B_t$$

where B is another Wiener process [the notation B emphasizes that there is no link between this Wiener process and the one used to simulate the forward process].

Reverse diffusion II (informal)

- To simulate the reverse diffusion, one needs to be able to estimate the score $\nabla \log q_{T-t}(x)$.
- In practice, the score is unknown and need to be approximated

$$s_t^\theta(x) \approx \nabla_x \log q_t(x)$$

which is often parameterized by a neural network.

- Since

$$\log q_t(x) = \log \int F(t, x_0, x) \, \pi_{\mathsf{data}} \, (\mathrm{d}x_0)$$

the analytical expression of $F(t, x_0, x)$ gives that (Tweedie formula)

$$\nabla_x \log q_t(x) = -\frac{x - \alpha_t \widehat{x}_0(x,t)}{\sigma_t^2}$$

where $\widehat{x}_0(x,t) = \mathbb{E}\left[X_0 | X_t = x\right]$ is a denoising estimate of x_0 given a noisy estimate $X_t = x$ at time t

Estimation of the score

- To estimate the score, one only needs to train a denoising function $\widehat{x}^{\theta}_{0|t}(x)$.
- It is a simple regression problem: take pairs (X_0, X_t) that can be generated as

$$X_0 \sim \pi_{\text{data}} \quad \text{and} \quad X_t = \alpha_t X_0 + \sigma_t Z_t$$

  with $Z_t \sim \mathcal{N}(0, \mathrm{I})$ and minimize the Mean Squared Error (MSE) loss, i.e.

$$\mathbb{E}_{q_{0,t}}\left[\left\|X_0 - \hat{x}^{\theta}_{0|t}(X_t)\right\|^2\right]$$

  with stochastic gradient descent or any other stochastic optimization procedure.
- The score is then defined as

$$s^{\theta}_t(x) = -\frac{x - \alpha_t \widehat{x}^{\theta}_t(x)}{\sigma_t^2}$$

Time reversal formula for a diffusion process

General time reversal formulas for diffusion processes are well known since the 80 's. Consider a diffusion process Y in $\mathbb{R}^n$ satisfying

$$dY_t = b_t\,(Y_t)\,dt + \sigma_t\,(Y_t)\,dB_t, \quad 0 \le t \le T,$$

with B a Brownian motion, b a drift vector field and σ a matrix field associated to the diffusion field a $:= \sigma\sigma^\top$

Assuming that the law of Y_t is absolutely continuous at each time t, under appropriate assumptions, the time-reversed process Y^* is again a diffusion process with diffusion matrix field $\mathrm{a}_t^* = \mathrm{a}_{T-t}$ and drift field

$$b_t^*(y) = -b_{T-t}(y) + \nabla \cdot (\mu_{T-t}\mathrm{a}_{T-t})\,(y)/\mu_{T-t}(y),$$

where μ_t is the density of the law of Y_t with respect to Lebesgue measure.

This is not a straightforward result because a reversed semimartingale might not be a semimartingale !.

Time reversal formula for a diffusion process

For the identity

$$b_t^*(y) = -b_{T-t}(y) + \nabla \cdot \left(\mu_{T-t} \mathrm{a}_{T-t}\right)(y)/\mu_{T-t}(y),$$

to hold, it is assumed in that b is locally Lipschitz and that either a is bounded away from zero or that the derivative ∇a in the sense of distribution is controlled locally.

Haussmann and Pardoux take a PDE approach; Millet, Nualart and Sanz rely on stochastic calculus of variations.

The existence of an absolutely continuous density follows from a Hörmander type condition (PDE formulation in Haussman et al. and consequence of Malliavin calculus in Millet et al.).

Time reversal formula for a diffusion process

Föllmer's approach significantly departs from these strategies. Under the simplifying hypothesis that a is the identity matrix, the law P of Y has a finite entropy

$$H(P \mid R) < \infty$$

with respect to the law R of a Brownian motion with some given initial probability distribution.

In particular, the drift field b of P satisfies $\int_{[0,T]\times\mathbb{R}^n} |b_t(y)|^2 \mu_t(y) dt dy < \infty$ and might be singular, rather than locally Lipschitz.

As a consequence of this finite entropy assumption, Föllmer proves the time reversal formula

$$b_t^*(y) = -b_{T-t}(y) + \nabla \log \mu_{T-t}(y)$$

(recall a = Id) where the derivative is in the sense of distributions, without invoking any already known result about the regularity of μ.

Summary

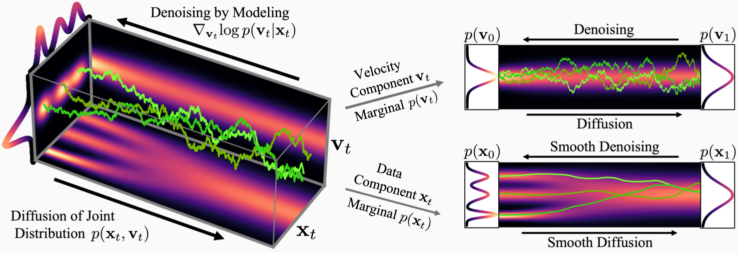


Figure 5: From Dockhorn et al. (2022)

Feyman-Kac representation

Context

Bayesian linear inverse problem:

$$Y = AX + \sigma_y Z, \quad \text{where} \quad Z \sim \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x}), \quad X \sim p_0, \quad \sigma_y \geq 0\,.$$

Context

Bayesian linear inverse problem:

$$Y = AX + \sigma_y Z, \quad \text{where} \quad Z \sim \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x}), \quad X \sim p_0, \quad \sigma_y \geq 0\,.$$

Objective: Sample the distribution of X given a realisation y of Y.

Context

Bayesian linear inverse problem:

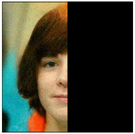
$$Y = AX + \sigma_y Z, \quad \text{where} \quad Z \sim \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x}), \quad X \sim p_0, \quad \sigma_y \geq 0\,.$$

Objective: Sample the distribution of X given a realisation y of Y.

Sample from p_0

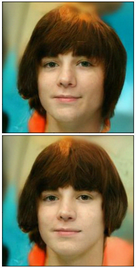
$\xrightarrow{AX+\sigma_y Z}$

Observation y



$\longrightarrow$

Posterior samples



Feynman-Kac representation

We focus on the specific case where the prior p_0 is the marginal w.r.t. x_0 of Denoising Diffusion Model. The posterior is

$$p_0^y(\mathrm{d}x_0) = \frac{1}{\mathcal{Z}^y} \int g_0^y(x_0) \prod_{k=0}^{n-1} p_{k|k+1}(\mathrm{d}x_k|x_{k+1})\, p_n(\mathrm{d}x_n)\,.$$

- The posterior can be interpreted as the marginal of a (time-reversed) Feynman–Kac (FK) model with non-trivial potential only at $k = 0$!
- In this work, we twist, without modifying the law of the FK model, the backward transitions $p_{k|k+1}$ by potentials depending on the observation y; see *e.g.* for a similar idea for rare event simulation (see, *e.g.*, Cérou et al., 2012).

"Forward" smoothing decomposition

- Define, for all $k \in [\![0, n]\!]$, the backward functions

$$\beta^y_{0|k}(x_k) := \int g^y_0(x_0)\, p_{0|k}(\mathrm{d}x_0|x_k)$$

- The backward functions satisfy the recursion:

$$\beta^y_{0|k+1}(x_{k+1}) = \int \beta^y_{0|k}(x_k)\, p_{k|k+1}(\mathrm{d}x_k|x_{k+1})\,.$$

- Define the forward smoothing kernels (FSK) for $k \in [\![0, n-1]\!]$

$$p^y_{k|k+1}(\mathrm{d}x_k|x_{k+1}) := \frac{\beta^y_{0|k}(x_k)}{\beta^y_{0|k+1}(x_{k+1})}\, p_{k|k+1}(\mathrm{d}x_k|x_{k+1})\,,$$

$$(= \mathrm{Law}(X_k \mid Y = y, X_{k+1} = x_{k+1}))\,.$$

"Forward" smoothing decomposition

The posterior distribution can be written in terms of forward smoothing kernels

$$p_0^y(\mathrm{d}x_0) = \int p_n^y(\mathrm{d}x_n) \prod_{k=0}^{n-1} p_{k|k+1}^y(\mathrm{d}x_k|x_{k+1}) \,.$$

where

$$p_n^y(\mathrm{d}x_n) = \frac{\beta_{0|n}^y(x_n) p_n(\mathrm{d}x_n)}{\mathcal{Z}^y}$$

- Most of the recent works to sample from p_0^y use the forward smoothing decomposition with different approximation of the intractable forward smoothing kernels. Chung et al. (2023); Song et al. (2023); Zhang et al. (2023); Boys et al. (2023); Trippe et al. (2023); Wu et al. (2023).

DDPM approximation

The DDPM is based on the assumption the forward smoothing decomposition is a good approximation the time reversal of the forward Markov chain initialized at p_0^y, i.e.

$$p_0^y(\mathrm{d}x_0) \prod_{k=1}^{n} q_{k|k-1}(\mathrm{d}x_k|x_{k-1}) \approx p_n^y(\mathrm{d}x_n) \prod_{k=0}^{n-1} p_{k|k+1}^y(\mathrm{d}x_k|x_{k+1}) \,,$$

which suggests the following approximation

$$p_{k|k+1}^y(\mathrm{d}x_k|x_{k+1}) \approx \int q_{k|0,k+1}(\mathrm{d}x_k|x_0, x_{k+1}) p_{0|k+1}^y(\mathrm{d}x_0|x_{k+1})$$

where

$$p_{0|k+1}^y(\mathrm{d}x_0|x_{k+1}) \propto p_0^y(\mathrm{d}x_0) q_{k+1|0}(x_{k+1}|x_0)$$

DDPM approximation

(Ho et al., 2020; Song et al., 2021a) suggested to use the DDPM approximation of the backward kernel is :

$$p^y_{k|k+1}(\mathrm{d}x_k|x_{k+1}) = q_{k|0,k+1}(\mathrm{d}x_k|\mathbb{E}[X_0|X_{k+1} = x_{k+1}, Y = y], x_{k+1})$$

where

$$\mathbb{E}[X_0|X_{k+1}, Y = y] := \int x_0 \, p^y_{0|k+1}(\mathrm{d}x_0|X_{k+1}) \, .$$

Conditional score

By Tweedie's formula,

$$\mathbb{E}\big[X_0|X_k, Y = y\big] = \frac{X_k + (1-\alpha_k)\nabla_{x_k}\log p_k^y(X_k)}{\sqrt{\alpha_k}}\,,$$

where

$$\begin{aligned} p_k^y(x_k) &:= \int p_0^y(\mathrm{d}x_0) q_{k|0}(x_k|x_0) \\ &\propto \int g_0^y(x_0) p_0(\mathrm{d}x_0) q_{k|0}(x_k|x_0) \\ &\propto \int g_0^y(x_0) p_{0|k}(\mathrm{d}x_0|x_k)\, p_k(x_k)\,. \end{aligned}$$

Hence,

$$\nabla_{x_k}\log p_k^y(x_k) = \nabla_{x_k}\log \beta_{0|k}^y(x_k) + \nabla_{x_k}\log p_k(x_k)\,.$$

Diffusion posterior sampling I

$$\nabla_{x_k} \log p_k^y(x_k) = \nabla_{x_k} \log \beta_{0|k}^y(x_k) + \nabla_{x_k} \log p_k(x_k)\,,$$

- A pre-trained score network (for $\nabla_{x_k} \log p_k(x_k)$) is available.
- But the gradient of the log backward function is intractable in practice.

Using the pre-trained approximation $\hat{x}_{0|k}(X_k)$ of $\mathbb{E}[X_0|X_k]$, Chung et al. (2023) proposed the following approximation,

$$\nabla_{x_k} \log \beta_{0|k}^y(x_k) \approx \nabla_{x_k} \log g_0^y(\hat{x}_{0|k}(x_k))\,.$$

They then sample approximately from the FSK in the following way; given X_k^y

- First sample $X_{k-1} \sim p_{k-1|k}(\cdot|X_k^y)$
- Then set $X_{k-1}^y = X_{k-1} + \gamma_k \nabla_{x_k} \log g_0^y(\hat{x}_{0|k}(X_k^y))$
- γ_k is in practice a highly sensitive parameter, crucial for good performance.

Diffusion posterior sampling II

- The DPS approximation by Chung et al. (2023) boils down to assuming that $p_{0|k}(\mathrm{d}x_0|x_k) \approx \delta_{\hat{x}_{0|k}(x_k)}(\mathrm{d}x_0)$.
- This is a very crude approximation that becomes accurate only as $k \to 0$.

Song et al. (2023) consider the sample sampling scheme but propose instead the following Gaussian approximation

$$p_{0|k}(\mathrm{d}x_0|x_k) \approx \mathcal{N}(\mathrm{d}x_0; \hat{x}_{0|k}(x_k), r_k^2 \,\mathrm{I}_{d_x}), \quad r_k^2 = \frac{\sigma_k^2}{1+\sigma_k^2}\,,$$

in which case, we obtain the following approximation

$$\beta^y_{0|k}(x_k) \approx \mathcal{N}(y; A\hat{x}_{0|k}(x_k), r_k^2 A A^\intercal + \sigma_y^2 \,\mathrm{I}_{d_y})\,.$$

- The Gaussian approximation above becomes exact in the case where $p_0 = \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x})$ and *variance exploding* is used.
- Still, this is not a realistic approximation in the more general case.

Tweedie Moment Projected diffusion

Boys et al. (2023) instead consider a Gaussian approximation $\hat{p}_{0|k}(\cdot|x_k)$ of $p_{0|k}(\cdot|x_k)$:

$$\hat{p}_{0|k}(\cdot|x_k) := \underset{\mu,\Sigma}{\operatorname{argmin}} \mathsf{KL}(p_{0|k}(\cdot|x_k) \parallel \mathcal{N}(\mu, \Sigma)) .$$

and

$$\hat{p}_{0|k}(\cdot|x_k) = \mathcal{N}\big(\mathbb{E}[X_0|X_k = x_k], \mathbb{C}\text{ov}(X_0|X_k = x_k)\big) ,$$

where the expectation and covariance are under $p_{0|k}(\cdot|x_k)$. Under the same assumption as previously (backward=forward), it can be shown that

$$\mathbb{C}\text{ov}(X_0|X_k) = \frac{1-\alpha_k}{\sqrt{\alpha_k}} \nabla_{x_k} \mathbb{E}[X_0|X_k]$$

which may be approximated by plugging in $\hat{x}_{0|k}(X_k)$ to approximate $\nabla_{x_k} \mathbb{E}[X_0|X_k]$.

- The resulting covariance approximation is not symmetric nor positive definite.
- Extremely expensive to compute. In practice further crude approximations are introduced.

Monte Carlo guided diffusion

General Feynman–Kac model

Introduce intermediate positive potentials $(g_k^y)_{k=0}^n$, each being a function on $\mathbb{R}^{d_x}$, and write

$$p_0^y(\mathrm{d}x_0) = \frac{1}{\mathcal{Z}^y} \int g_n^y(x_n)\, p_n(\mathrm{d}x_n) \times \prod_{k=0}^{n-1} \frac{g_k^y(x_k)}{g_{k+1}^y(x_{k+1})}\, p_{k|k+1}(\mathrm{d}x_k|x_{k+1})\,.$$

- Because the $g_n^y(x_n) \prod_{k=0}^{n-1} \frac{g_k^y(x_k)}{g_{k+1}^y(x_{k+1})} = g_0^y(x_0)$, the FK is not modified - the potentials are used to render the sampling easier.
- This allows the posterior of interest to be expressed as the time-zero marginal of a Feynman-Kac model with
  - initial law p_n,
  - Markov transition kernels $(p_{k|k+1})_{k=0}^{n-1}$
  - Potentials g_n^y and $(x_k, x_{k+1}) \mapsto g_k^y(x_k)/g_{k+1}^y(x_{k+1})$.

Posterior sampling proposal

Alternatively, the previous decomposition defines a sequence of distributions

$$p_k^y(\mathrm{d}x_k) \propto g_k^y(x_k) p_k(\mathrm{d}x_k)\,, \quad k \in [\![0, n]\!]\,,$$

where the posterior of interest is the terminal distribution at $k = 0$.

- If we have a particle approximation of p_{k+1}^y then we can evolve it into a particle approximation of p_k^y $\rightsquigarrow$ we recursively build an empirical approximation of p_0^y.
- The choice of potentials $\{g_k^y\}_{k \in [\![0,n]\!]}$ is crucial; we need to ensure that p_k^y is close enough to p_{k+1}^y so that we can bridge the intermediate distributions efficiently.

Posterior sampling proposal: recursion

Consider the following particle approximation of p^y_{k+1}

$$p^{N,y}_{k+1} = \frac{1}{N} \sum_{i=1}^{N} \delta_{\xi^i_{k+1}} \,,$$

Recall that $p_k(\mathrm{d}x_k) = \int p_{k|k+1}(\mathrm{d}x_k|x_{k+1}) p_{k+1}(\mathrm{d}x_{k+1})$,

Posterior sampling proposal: recursion

Consider the following particle approximation of p^y_{k+1}

$$p^{N,y}_{k+1} = \frac{1}{N}\sum_{i=1}^{N} \delta_{\xi^i_{k+1}} \,,$$

Recall that $p_k(\mathrm{d}x_k) = \int p_{k|k+1}(\mathrm{d}x_k|x_{k+1})p_{k+1}(\mathrm{d}x_{k+1})$,

$$p^y_k(\mathrm{d}x_k) = \frac{\int \frac{g^y_k(x_k)}{g^y_{k+1}(x_{k+1})} p_{k|k+1}(\mathrm{d}x_k|x_{k+1})p^y_{k+1}(\mathrm{d}x_{k+1})}{\int \frac{g^y_k(z_k)}{g^y_{k+1}(z_{k+1})} p_{k|k+1}(\mathrm{d}z_k|z_{k+1})p^y_{k+1}(\mathrm{d}z_{k+1})} \;,$$

Posterior sampling proposal: recursion

Consider the following particle approximation of p^y_{k+1}

$$p^{N,y}_{k+1} = \frac{1}{N}\sum_{i=1}^{N} \delta_{\xi^i_{k+1}} \,,$$

Recall that $p_k(\mathrm{d}x_k) = \int p_{k|k+1}(\mathrm{d}x_k|x_{k+1})p_{k+1}(\mathrm{d}x_{k+1})$,

$$p^y_k(\mathrm{d}x_k) = \frac{\int \frac{g^y_k(x_k)}{g^y_{k+1}(x_{k+1})} p_{k|k+1}(\mathrm{d}x_k|x_{k+1})p^y_{k+1}(\mathrm{d}x_{k+1})}{\int \frac{g^y_k(z_k)}{g^y_{k+1}(z_{k+1})} p_{k|k+1}(\mathrm{d}z_k|z_{k+1})p^y_{k+1}(\mathrm{d}z_{k+1})} \,,$$

and hence

$$p^y_k(\mathrm{d}x_k) \propto \int \underbrace{\frac{\int g^y_k(z_k)p_k(\mathrm{d}z_k|x_{k+1})}{g^y_{k+1}(x_{k+1})}}_{:=\widetilde{\omega}_k(x_{k+1})} p^y_k(\mathrm{d}x_k|x_{k+1})p^y_{k+1}(\mathrm{d}x_{k+1}) \,,$$

where $p^y_k(\mathrm{d}x_k|x_{k+1}) \propto g^y_k(x_k)p_{k|k+1}(\mathrm{d}x_k|x_{k+1}) \rightarrow$ available in closed form if we use a Gaussian potential with mean linear in x_k.

Posterior sampling proposal: SMC approximation

$$p_k^y(\mathrm{d}x_k) = \int p_k^y(\mathrm{d}x_k|x_{k+1}) \frac{\widetilde{\omega}_k(x_{k+1}) p_{k+1}^y(\mathrm{d}x_{k+1})}{\int \widetilde{\omega}_k(z_{t+1}) p_{k+1}^y(\mathrm{d}z_{k+1})} ,$$

Assume $p_k^{N,y} = \frac{1}{N}\sum_{i=1}^N \delta_{\xi_{k+1}^i}$ is a particle approximation of $p_{k+1}^{N,y}$.

⇝ **Weight**:

$$p_k^{N,y}(\cdot) \approx \sum_{i=1}^N \frac{\widetilde{\omega}_k(\xi_{k+1}^i)}{\sum_{j=1}^N \widetilde{\omega}_k(\xi_{k+1}^j)} p_k^y(\cdot|\xi_{k+1}^i) .$$

⇝ **Resample**: Draw $A_{k+1}^{1:N} \overset{\text{iid}}{\sim} \text{Categorical}(\{\omega_k^j\}_{j=1}^N)$ where $\omega_k^j \propto \widetilde{\omega}_t(\xi_{k+1}^j)$.

⇝ **Mutate**: Sample $\xi_k^i \sim p_k^y(\cdot|\xi_{k+1}^{A_{k+1}^i})$ for $i \in [1:N]$,

$$p_k^{N,y} = \frac{1}{N}\sum_{i=1}^N \delta_{\xi_k^i} .$$

Gordon et al. (1993); Del Moral (2004); Cappe et al. (2005); Chopin et al. (2020)

Potentials: heuristic

For simplicity (and only in this slide) let $p_0(y)$ be the posterior of the inverse problem

$$Y = \overline{X}_0, \quad X_0 \sim p_0\,,$$

The marginals of the *forward process* initialized at p_0^y are

$$X_k \stackrel{\mathcal{L}}{=} \sqrt{\bar{\alpha}_k} X_0 + \sqrt{1 - \bar{\alpha}_k} Z, \quad X_0 \sim p_0^y, \quad Z \sim \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x})\,,$$

and so

$$\overline{X}_k \stackrel{\mathcal{L}}{=} \sqrt{\bar{\alpha}_k} y + \sqrt{1 - \bar{\alpha}_k}\, \overline{Z}\,, \quad \overline{Z} \sim \mathcal{N}(\mathbf{0}_{d_y}, \mathrm{I}_{d_y})\,.$$

- This suggests that one relevant choice of potentials is

$$g_k^y(x_k) = \mathcal{N}(\sqrt{\alpha_k} y; x_k, (1 - \alpha_k)\mathrm{I}_{d_y})\,.$$

Choice of potentials

- More generally, we let the variance be a free parameter $\sigma^2_{y,k}$.

Our proposal in the general case is

$$p^y_k(\mathrm{d}x_k) \propto g^y_k(x_k) p_k(\mathrm{d}x_k) \,, \quad g^y_k(x_k) := \mathcal{N}(\sqrt{\bar{\alpha}_k} y; A x_k, \sigma^2_{y,k} \mathrm{I}_{d_y})$$

- This particular choice of potential allows us to compute in closed form the auxiliary transition kernel $\propto g^y_k(x_k) p_{k|k+1}(\mathrm{d}x_k | x_{k+1})$ we use for our particle approximations.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

Illustration

⇝ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

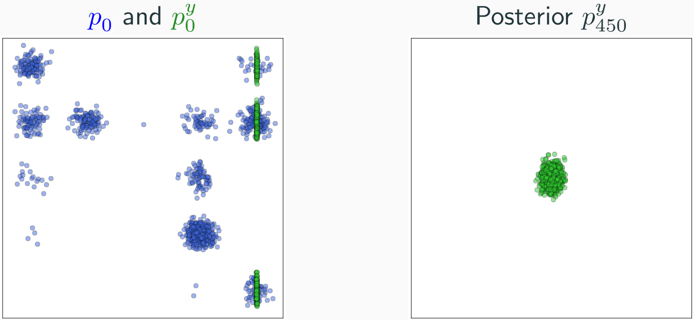


Figure 6: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

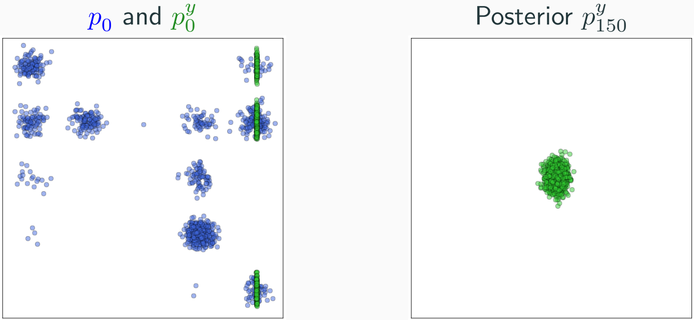


Figure 7: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

⇝ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

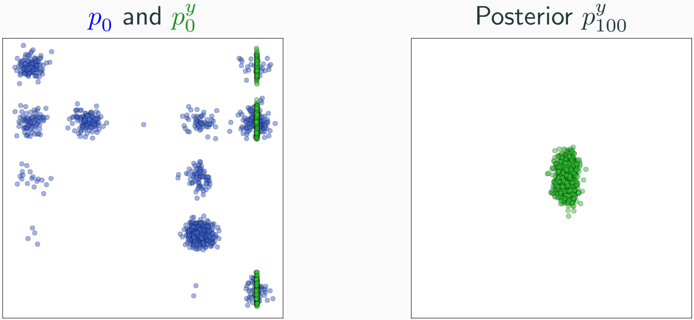


Figure 8: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

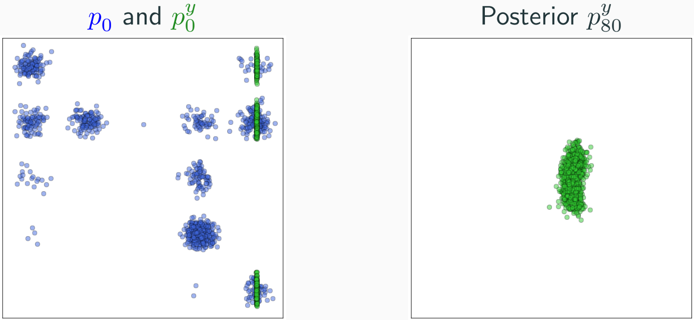


Figure 9: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

⇝ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

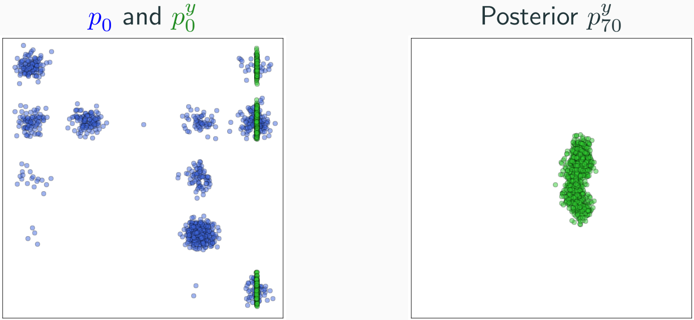


Figure 10: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

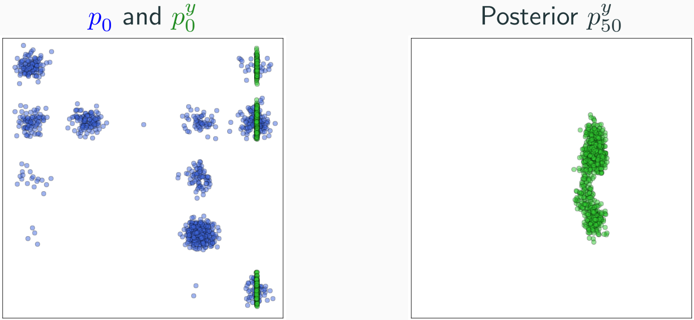


Figure 11: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

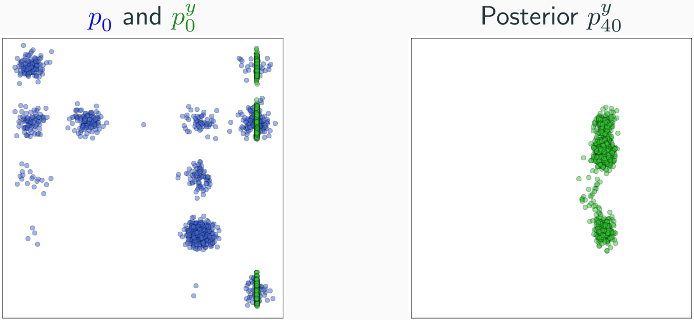


Figure 12: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow \{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

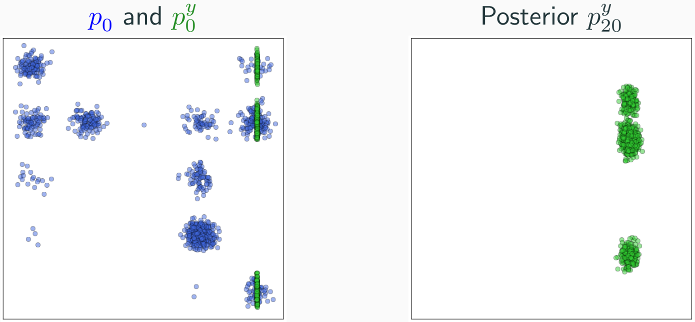


Figure 13: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

⇝ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

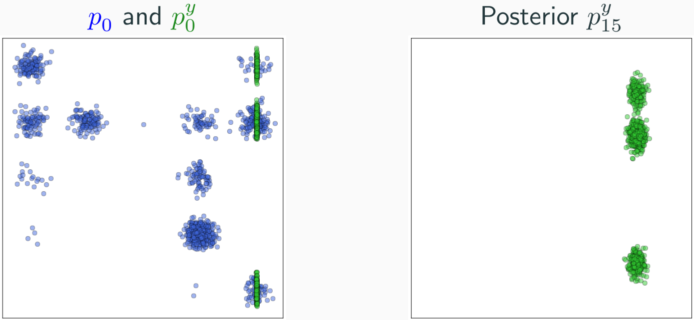


Figure 14: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

$\rightsquigarrow$ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

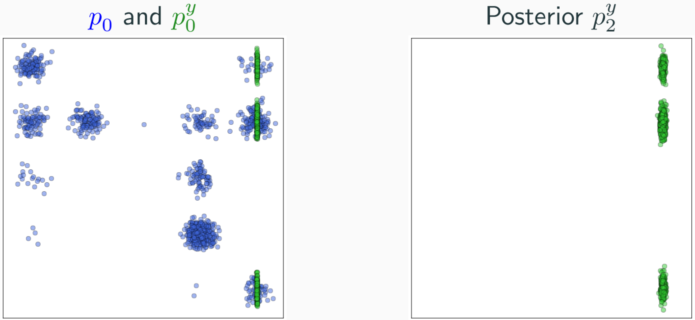


Figure 15: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Illustration

⇝ $\{p_k^y\}_{k=1}^n$ is available in closed form for the Gaussian mixture example.

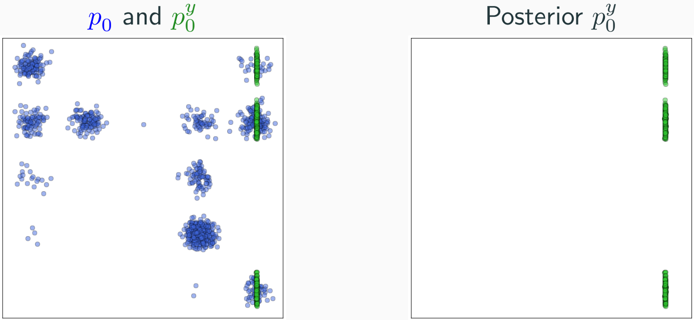


Figure 16: Left plot: samples from the prior p_0 and posterior p_0^y. **Right plot:** samples from the posterior proposals p_k^y for time steps ranging from $n := 500$ to 0.

Toy examples

$\rightsquigarrow$ 25 Gaussian mixture example with means

$$\mu_{i,j} = (8i, 8j, \ldots, 8i, 8j), \quad (i,j) \in \{-2, \ldots, 2\}$$

with unit convariance matrices. We randomly draw the weights of the mixture and the forward operator A and σ_y for the inverse problem $\rightsquigarrow$ $\nabla \log p_k$ is available in **closed form**.

$\rightsquigarrow$ 20 component mixture of translated and rotated Funnel distributions. We learn the score and consider the ground truth to be samples from parallel NUTS with very long chains.

Toy examples

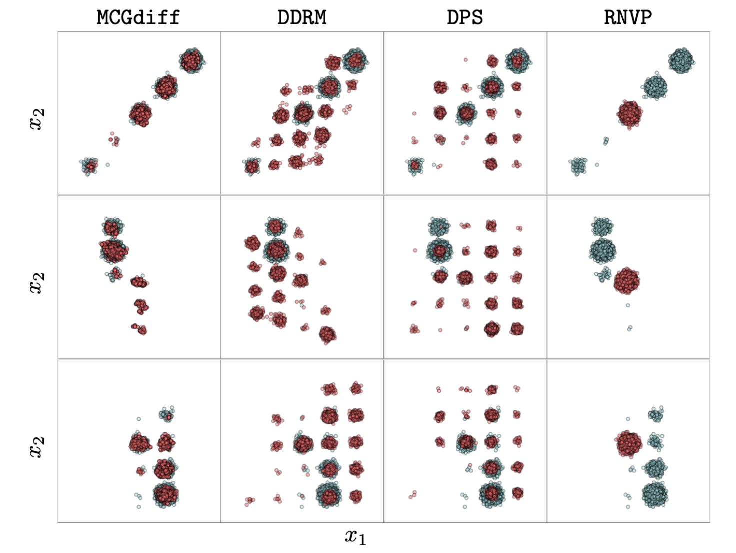
d	d_y	MCGdiff	DDRM	DPS	RNVP
80	1	**1.39 ± 0.45**	5.64 ± 1.10	4.98 ± 1.14	6.86 ± 0.88
80	2	**0.67 ± 0.24**	7.07 ± 1.35	5.10 ± 1.23	7.79 ± 1.50
80	4	**0.28 ± 0.14**	7.81 ± 1.48	4.28 ± 1.26	7.95 ± 1.61
800	1	**2.40 ± 1.00**	7.44 ± 1.15	6.49 ± 1.16	7.74 ± 1.34
800	2	**1.31 ± 0.60**	8.95 ± 1.12	6.88 ± 1.01	8.75 ± 1.02
800	4	**0.47 ± 0.19**	8.39 ± 1.48	5.51 ± 1.18	7.81 ± 1.63

d	d_y	MCGdiff	DDRM	DPS	RNVP
6	1	**1.95 ± 0.43**	4.20 ± 0.78	5.43 ± 1.05	6.16 ± 0.65
6	3	**0.73 ± 0.33**	2.20 ± 0.67	3.47 ± 0.78	4.70 ± 0.90
6	5	**0.41 ± 0.12**	0.91 ± 0.43	2.07 ± 0.63	3.52 ± 0.93
10	1	**2.45 ± 0.42**	3.82 ± 0.64	4.30 ± 0.91	6.04 ± 0.38
10	3	**1.07 ± 0.26**	4.94 ± 0.87	5.38 ± 0.84	5.91 ± 0.64
10	5	**0.71 ± 0.12**	2.32 ± 0.74	3.74 ± 0.77	5.11 ± 0.69

Figure 17: Sliced Wasserstein between samples of the target posterior and the empirical measure returned by each method. **Top**: Gaussian mixture. **Bottom**: Funnel mixture. We show the 95% CLT interval over 20 seeds.

DPS: Chung et al. (2023), DDRM: Kawar et al. (2022)

Toy examples

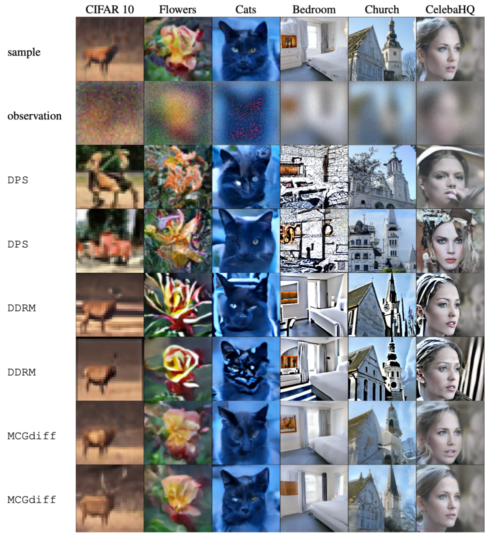


Imaging experiments

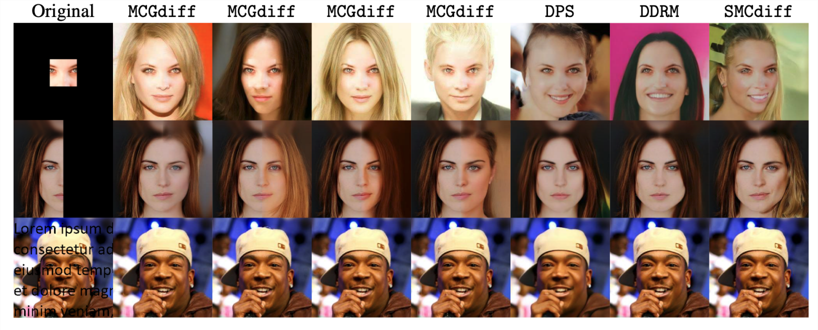
- $\rightsquigarrow$ Diffusion models learned on different datasets of image sizes varying from $(64, 64, 3)$ to $(256, 256, 3)$.
- $\rightsquigarrow$ We run parallel SMCs with $\mathbf{N = 64}$ particles.

Super-resolution example

Deblurring example



Inpainting example



Divide-and-conquer posterior sampling

Sequence of distributions

Let $(k_\ell)_{\ell=0}^L$ be an increasing sequence in $[\![0, n]\!]$ with $k_0 = 0$ and $k_L = n$.

Consider

$$p^y_{k_\ell}(\mathrm{d}x_{k_\ell}) \propto g^y_{k_\ell}(x_{k_\ell}) p_{k_\ell}(\mathrm{d}x_\ell)\,,$$

with

$$g^y_{k_\ell}(x_{k_\ell}) = \mathcal{N}(\sqrt{\alpha_{k_\ell}}\, y; Ax_{k_\ell}, \sigma^2_{y,k_\ell}\mathrm{I}_{d_y})\,.$$

- L is typically much smaller than n.
- This is the same sequence of distribution as in our SMC approach but now we only consider a small number L of intermediate distributions.
- Our goal is to recursively sample from each one of them without having to evolve N particles in parallel.
- We also want to solve the "image inconsistency" problem observed in our SMC method.

Recursion

Since

$$p_{k_\ell}(\mathrm{d}x_{k_\ell}) = \int \left\{ \prod_{j=k_\ell}^{k_{\ell+1}-1} p_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \right\} p_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}}) \,,$$

we can write $p^y_{k_\ell}$ in terms of forward smoothing kernels, i.e.

$$p^y_{k_\ell}(\mathrm{d}x_{k_\ell}) = \int \left\{ \prod_{j=k_\ell}^{k_{\ell+1}-1} p^{y,\ell}_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \right\} p^{y,\ell}_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}})$$

where

$$p^{y,\ell}_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}}) \propto \beta^{y,\ell}_{k_\ell|k_{\ell+1}}(x_{k_{\ell+1}})\, p_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}}) \,,$$
$$p^{y,\ell}_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \propto \beta^{y,\ell}_{k_\ell|j}(x_j)\, p_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \,,$$

and for all $j \in [\![k_\ell, k_{\ell+1}]\!]$

$$\beta^{y,\ell}_{k_\ell|j}(x_j) := \int g^y_{k_\ell}(x_{k_\ell}) p_{k_\ell|j}(\mathrm{d}x_{k_\ell}|x_j).$$

DCPS summary

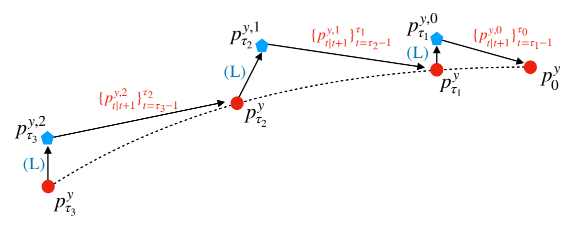


Figure 18: Illustration of idealized DCPS.

Starting at an approximate sample $X^y_{k_{\ell+1}}$ from $p^y_{k_{\ell+1}}$

- Use ULA initialized at $X^y_{k_{\ell+1}}$ to obtain an approximate sample from $X^{y,\ell}_{k_{\ell+1}}$.
- Starting from $X^{y,\ell}_{k_{\ell+1}}$, simulate a Markov chain with transition kernels $(p^{y,\ell}_{j|j+1})^{k_\ell}_{j=k_{\ell+1}-1}$
- Repeat until the posterior of interest is reached.

Backward function approximation

- The first source of intractability are the backward functions $\beta^{y,\ell}_{k_\ell|j}$.
- This is the same problem as before, however note that now they are expressed as an integral under $p_{k_\ell|j}(\cdot|x_j)$ with $j \in [\![k_\ell + 1, k_{\ell+1}]\!]$ instead of $p_{0|j}(\cdot|x_j)$ for $j \in [\![0, n]\!]$.
- This is more convenient since we expect Gaussian approximations of $p_{k_\ell|j}(\cdot|x_j)$ to be more accurate than those of $p_{0|j}(\cdot|x_j)$.

Backward kernel approximation

Assume again that forward=backward. Then for $j \in [\![k_\ell + 1, k_{\ell+1}]\!]$,

$$p_{k_\ell|j}(\mathrm{d}x_{k_\ell}|x_j) = \int q_{k_\ell|0,j}(\mathrm{d}x_{k_\ell}|x_0, x_j) p_{0|j}(\mathrm{d}x_0|x_j)\,,$$

Let $\hat{p}_{0|j}(\cdot|x_j)$ be an approximation of $p_{0|j}(\cdot|x_j)$ and define

$$\hat{p}_{k_\ell|j}(\mathrm{d}x_{k_\ell}|x_j) = \int q_{k_\ell|0,j}(\mathrm{d}x_{k_\ell}|x_0, x_j)\hat{p}_{0|j}(\mathrm{d}x_0|x_j)$$

- For `DPS` (Chung et al., 2023), $\hat{p}_{0|j}(\mathrm{d}x_0|x_j) = \delta_{\hat{x}^\theta_{0|j}(x_j)}(\mathrm{d}x_0)$.
- For Song et al. (2023), $\hat{p}_{0|j}(\mathrm{d}x_0|x_j) = \mathcal{N}(\mathrm{d}x_0; \hat{x}^\theta_{0|j}(x_j), r_j^2\,\mathrm{I}_{d_y})$.
- In both cases, $\hat{p}_{k_\ell|j}(\cdot|x_j)$ is computable in closed form. We write

  $$\hat{p}_{k_\ell|j}(\mathrm{d}x_{k_\ell}|x_j) = \mathcal{N}(\mathrm{d}x_{k_\ell}; \mu_{k_\ell|j}(x_j), \sigma^2_{k_\ell|j}\,\mathrm{I}_{d_x})\,.$$

  where both the mean and variance depend on the approximation used.

Backward kernel approximation

Proposition
Assume forward=backward. For all $\ell \in [\![0, L]\!]$, $j \in [\![k_\ell + 1, k_{\ell+1}]\!]$,

$$W_2(\hat{p}_{k_\ell|j}(\cdot|x_j), p_{k_\ell|j}(\cdot|x_j)) \leq \frac{\sqrt{\alpha_{k_\ell}}(1 - \alpha_j/\alpha_{k_\ell})}{1 - \alpha_j} W_2(\hat{p}_{0|j}(\cdot|x_j), p_{0|j}(\cdot|x_j)) \,.$$

where $\frac{\sqrt{\alpha_{k_\ell}}(1-\alpha_j/\alpha_{k_\ell})}{1-\alpha_j} < 1$ *and goes to* 0 *as* $j \to k_\ell$.

- We improve upon the previous approximations by performing Gaussian approximations on intervals $[\![k_\ell, k_{\ell+1}]\!]$ of moderate size.
- Our approximation of the backward function is then

$$\begin{aligned}\beta^{y,\ell}_{k_\ell|j}(x_j) \approx \hat{\beta}^{y,\ell}_{k_\ell|j}(x_j) &:= \int g^y_{k_\ell}(x_{k_\ell}) \hat{p}_{k_\ell|j}(\mathrm{d}x_{k_\ell}|x_j) \\ &= \mathcal{N}(\sqrt{\alpha_{k_\ell}}\, y; A\mu_{k_\ell|j}(x_j), \sigma^2_{k_\ell|j} AA^\intercal + \sigma^2_{y,\ell} \mathrm{I}_{d_y}) \,.\end{aligned}$$

FSK approximation

Recall that the quantities of interest are

$$p^{y,\ell}_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \propto \beta^{y,\ell}_{k_\ell|j}(x_j)\, p_{j|j+1}(\mathrm{d}x_j|x_{j+1})\,,$$
$$p^{y,\ell}_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}}) \propto \beta^{y,\ell}_{k_\ell|k_{\ell+1}}(x_{k_{\ell+1}})\, p_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}})\,.$$

Given the previous approximation of the backward function, we replace them instead with

$$\hat{p}^{y,\ell}_{j|j+1}(\mathrm{d}x_j|x_{j+1}) \propto \hat{\beta}^{y,\ell}_{k_\ell|j}(x_j)\, p_{j|j+1}(\mathrm{d}x_j|x_{j+1})\,,$$
$$\hat{p}^{y,\ell}_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}}) \propto \hat{\beta}^{y,\ell}_{k_\ell|k_{\ell+1}}(x_{k_{\ell+1}})\, p_{k_{\ell+1}}(\mathrm{d}x_{k_{\ell+1}})\,,$$

- Still, while now we can evaluate the density $\hat{p}^{y,\ell}_{j|j+1}(\cdot|x_{j+1})$ we still cannot sample from it.
- We can approximately sample from $\hat{p}^{y,\ell}_{k_{\ell+1}}$ using `ULA`.

Variational approximation I

For a fixed x_{j+1} we seek a mean-field Gaussian variational approximation of $\hat{p}^{y,\ell}_{j|j+1}(\cdot|x_{j+1})$ by solving

$$\underset{r^{y,\ell}_{j|j+1}(\cdot|x_{j+1}) \in \mathcal{G}_{\mathrm{D}}}{\operatorname{argmin}} \; \mathsf{KL}(r^{y,\ell}_{j|j+1}(\cdot|x_{j+1}) \parallel \hat{p}^{y,\ell}_{j|j+1}(\cdot|x_{j+1}))\,,$$

where $\mathcal{G}_{\mathrm{D}} := \{\mathcal{N}(\mu, \mathrm{diag}(\sigma)) : \mu \in \mathbb{R}^{d_x}, \sigma \in \mathbb{R}^{d_x}_{>0}\}$.

- We only learn vectors (μ, σ) that depend on the value of $X^{y,\ell}_{j+1}$ and do not seek to generalize as this incurs problem dependent, heavy training.

Variational approximation II

Letting $r^{y,\ell}_{j|j+1}(\cdot|X^{y,\ell}_{j+1}) = \mathcal{N}(\mu^{y,\ell}_{j|j+1}, \mathrm{diag}(\mathrm{e}^{s^{y,\ell}_{j|j+1}}))$ where $s^{y,\ell}_{j|j+1} \in \mathbb{R}^{d_x}$,

$$
\begin{aligned}
&\mathsf{KL}(r^{y,\ell}_{j|j+1}(\cdot|X^{y,\ell}_{j+1}) \parallel \hat{p}^{y,\ell}_{j|j+1}(\cdot|X^{y,\ell}_{j+1})) \\
&= -\mathbb{E}\big[\log \hat{\beta}^{y,\ell}_{k_\ell|j}(\mu^{y,\ell}_{j|j+1} + \mathrm{diag}(\mathrm{e}^{s^{y,\ell}_{j|j+1}})Z)\big] + \frac{\|\mu^{y,\ell}_{j|j+1} - \mu_{j|j+1}(X^{y,\ell}_{j+1})\|^2}{2\sigma^2_{m|m+1}} \\
&\qquad - \frac{1}{2}\sum_{i=1}^{d_x}\left(s^{y,\ell}_{j|j+1,i} - \frac{\mathrm{e}^{s^{y,\ell}_{j|j+1,i}}}{\sigma^2_{m|m+1}}\right),
\end{aligned}
$$

- We perform the optimization using SGD.
- Crucially, we normalize the gradients to ensure the stability of the training procedure.
- In practice, we only perform 2 or 3 SGD steps.

Tamed ULA steps

We now turn to the Langevin steps on $\hat{p}^{y,\ell}_{k_{\ell+1}}$.

As the marginals $(p_k)_{k=0}^n$ approximate the true marginals of the forward process initialized at the data distribution π, we may use

$$s^\theta_k(x_k) = -(x_k - \sqrt{\alpha_k}\hat{x}^\theta_{0|k}(x_k))/(1-\alpha_k)\,,$$

as a substitute for $\nabla_{x_k} \log p_k(x_k)$, following Dhariwal and Nichol (2021).

We sample approximately from $\hat{p}^{y,\ell}_{k_{\ell+1}}$ by running M steps of the Tamed Unadjusted Langevin scheme (Brosse et al., 2019)

$$X_{j+1} = X_j + \gamma G^{y,\ell}_\gamma(X_j) + \sqrt{2\gamma} Z_j\,, \quad X_0 = X^y_{k_{\ell+1}}\,, \tag{1}$$

where

$$G^{y,\ell}_\gamma(x) := \frac{\nabla \log \hat{\beta}^{y,\ell}_{k_\ell|k_{\ell+1}}(x) + s^\theta_{k_{\ell+1}}(x)}{1 + \gamma \|\nabla \log \hat{\beta}^{y,\ell}_{k_\ell|k_{\ell+1}}(x) + s^\theta_{k_{\ell+1}}(x)\|}\,,$$

and set $X^{y,\ell}_{k_{\ell+1}} := X_M$.

Summary

Given an approximate sample $X^y_{k_{\ell+1}}$ from $\hat{p}^y_{k_{\ell+1}}$,

- Run `TULA` starting from $X^y_{k_{\ell+1}}$ to obtain $X^{y,\ell}_{k_{\ell+1}}$ approximately distributed according $\hat{p}^{y,\ell}_{k_{\ell+1}}$.
- Sample $(X^{y,\ell}_j)^{k_\ell}_{j=k_{\ell+1}}$: given $X^{y,\ell}_{j+1}$ with $j \in [\![k_\ell, k_{\ell+1} - 1]\!]$,
  - Find variational approximation $r^{y,\ell}_{j|j+1}(\cdot|X^{y,\ell}_{j+1})$.
  - Draw $X^{y,\ell}_j \sim r^{y,\ell}_{j|j+1}(\cdot|X^{y,\ell}_{j+1})$.
- Repeat these steps.

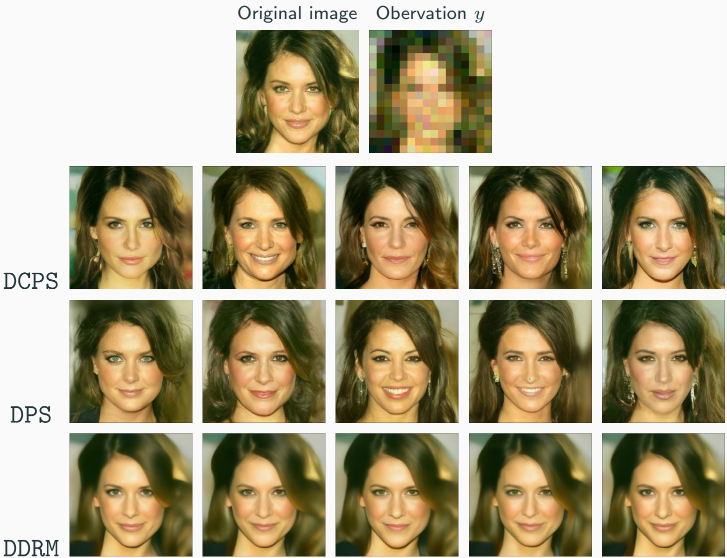
Toy experiments

- Same 25 Gaussian mixture example.
- $\texttt{DCPS}_M$ refers to our algorithm with M Langevin steps at the beginning of each block.
- We use $L = 4$.
- We also estimate the empirical weights of each Gaussian mixture mode and compare with the ground truth.

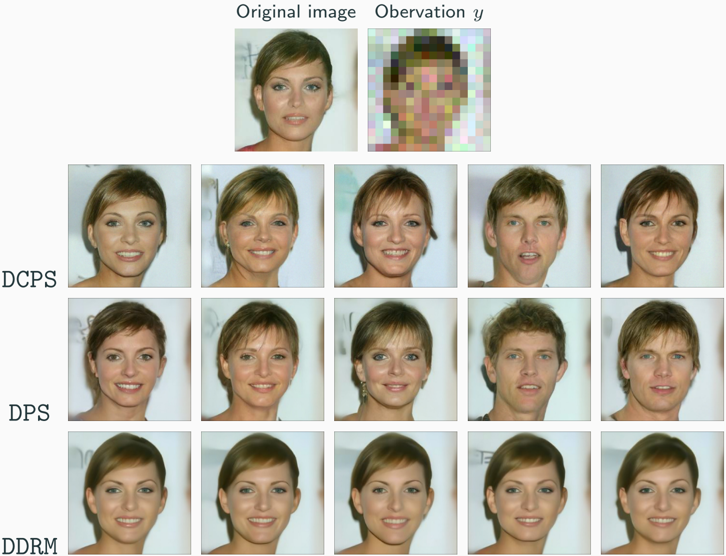
	$d_x = 10,\, d_y = 1$		$d_x = 100,\, d_y = 1$	
	SW	Δw	SW	Δw
MCGDiff	$2.25/2.69 \pm 2.07$	0.32 ± 0.20	$2.72/3.13 \pm 1.76$	0.42 ± 0.19
DPS	$3.12/5.64 \pm 8.45$	0.20 ± 0.12	$4.29/4.93 \pm 4.85$	0.35 ± 0.25
DDRM	$2.66/3.06 \pm 1.90$	0.36 ± 0.16	$5.97/6.26 \pm 2.33$	0.52 ± 0.19
$\texttt{DCPS}_{50}$	$1.95/2.70 \pm 2.28$	0.17 ± 0.25	$4.40/4.72 \pm 2.18$	0.44 ± 0.16
$\texttt{DCPS}_{500}$	$1.26/2.59 \pm 2.83$	0.13 ± 0.30	$2.81/3.22 \pm 2.21$	0.32 ± 0.18

Table 1: Results for the Gaussian mixture experiment. Results for the SW

Super-resolution experiments



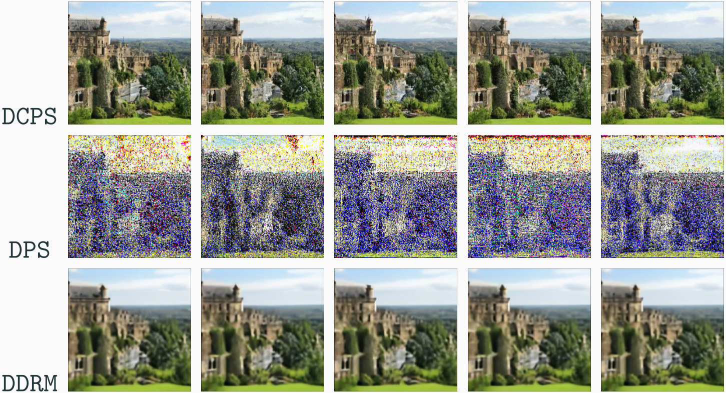
Super-resolution experiments



Super-resolution experiments

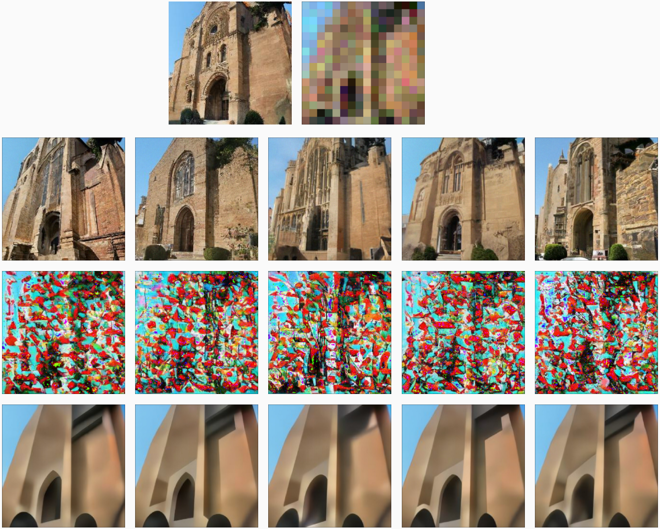
Original image

Obervation y



Super-resolution experiments

Original image Obervation y

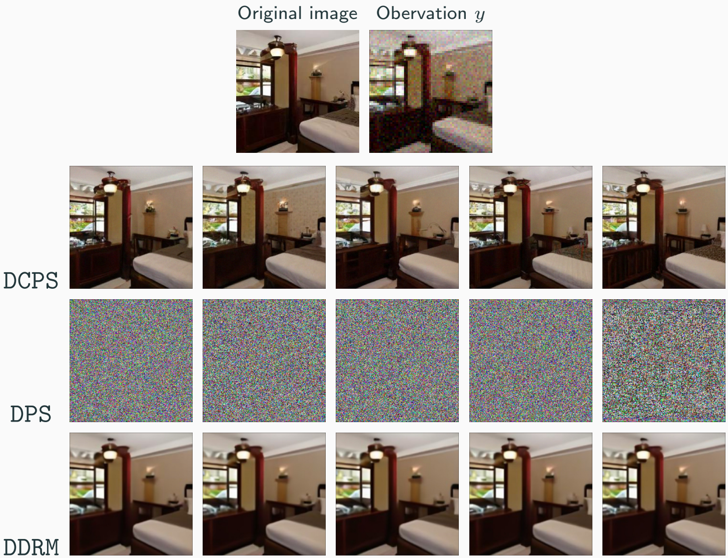


DCPS

DPS

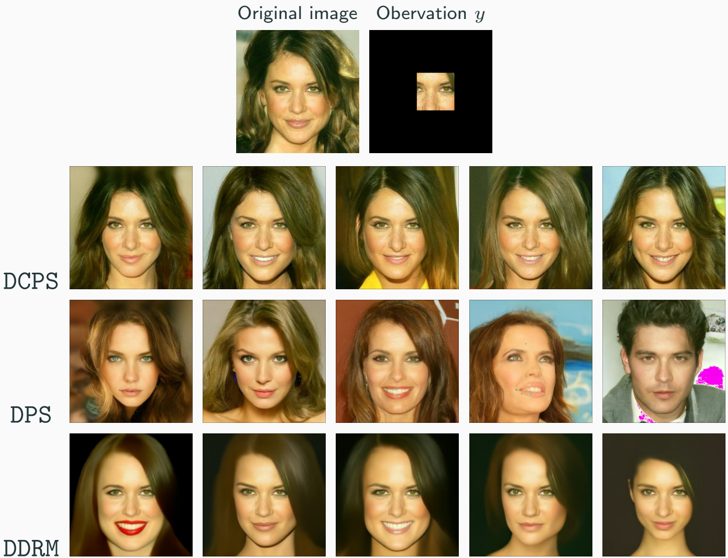
DDRM

Super-resolution experiments



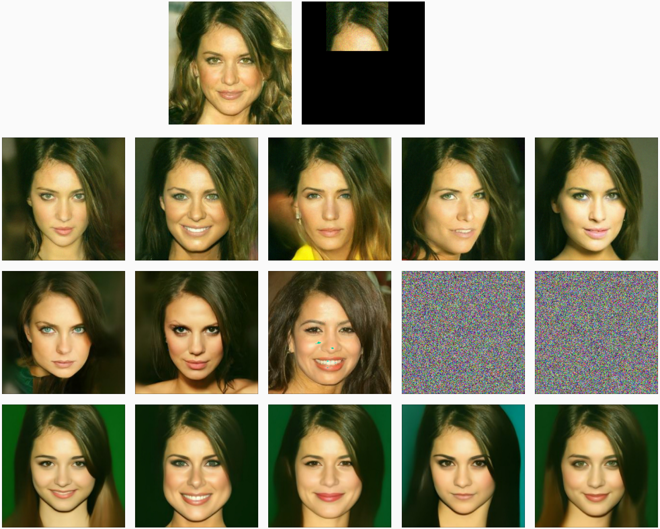
Super-resolution experiments

Inpainting and outpainting experiments



Inpainting and outpainting experiments

Original image Obervation y

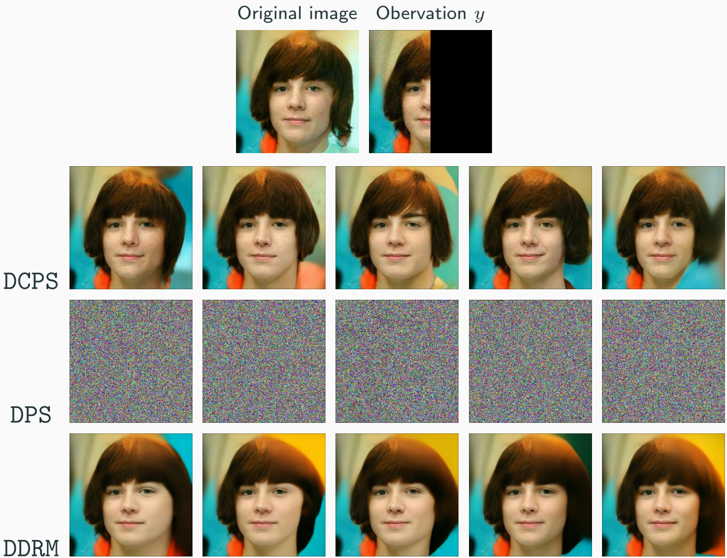


DCPS

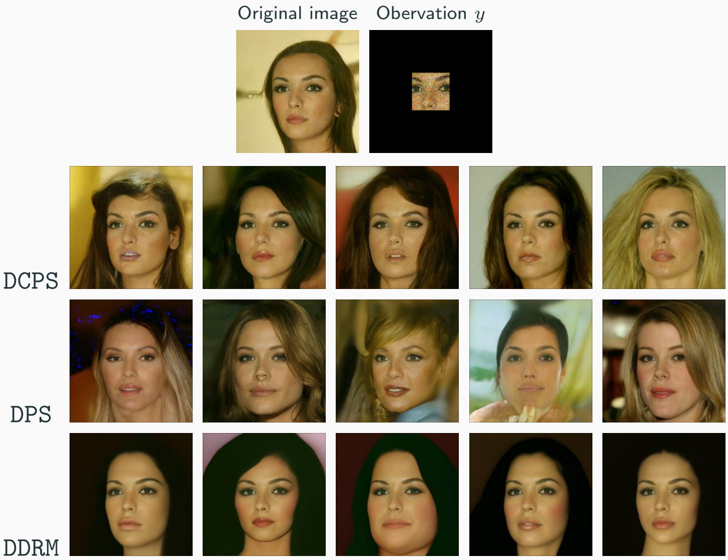
DPS

DDRM

Inpainting and outpainting experiments

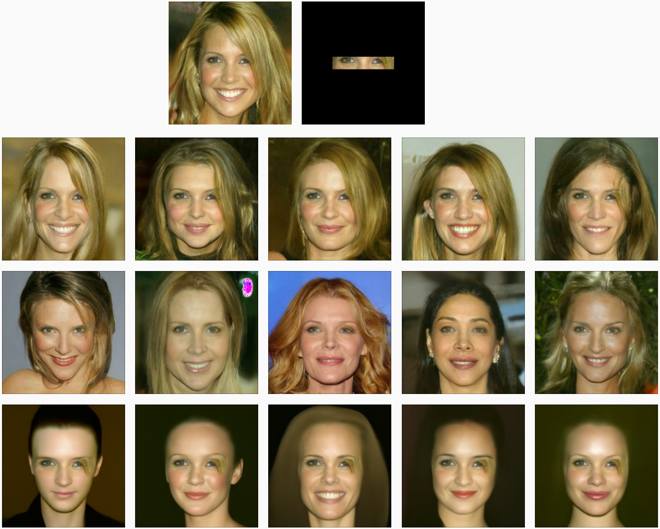


Inpainting and outpainting experiments



Inpainting and outpainting experiments

Original image Obervation y

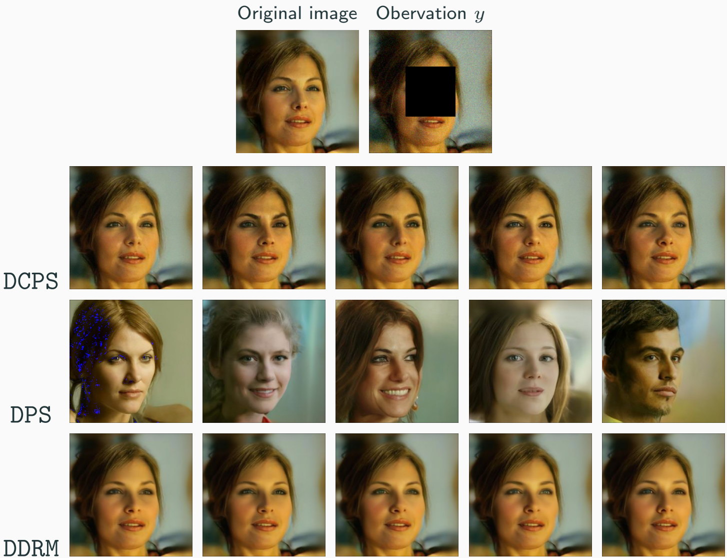


DCPS

DPS

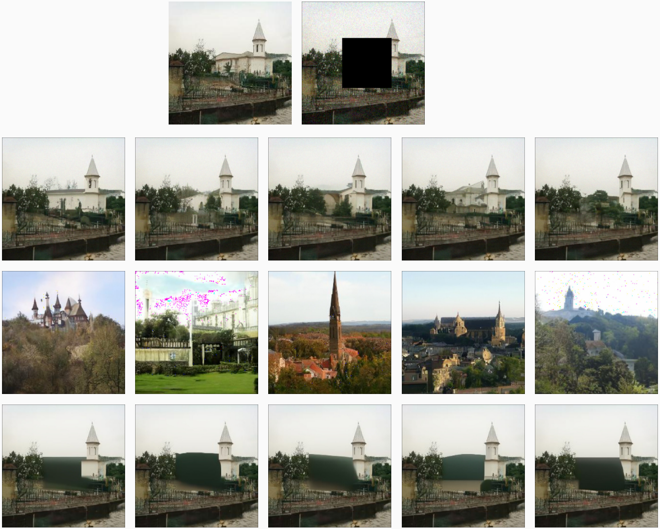
DDRM

Inpainting and outpainting experiments



Inpainting and outpainting experiments

Original image Obervation y



DCPS

DPS

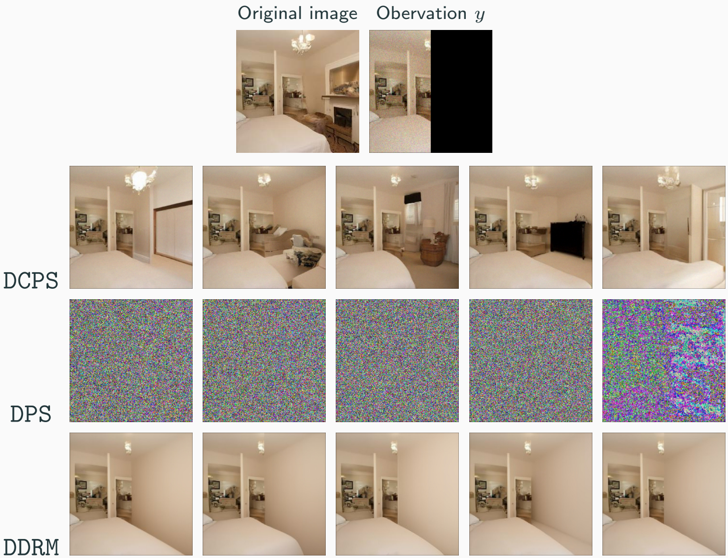
DDRM

Inpainting and outpainting experiments

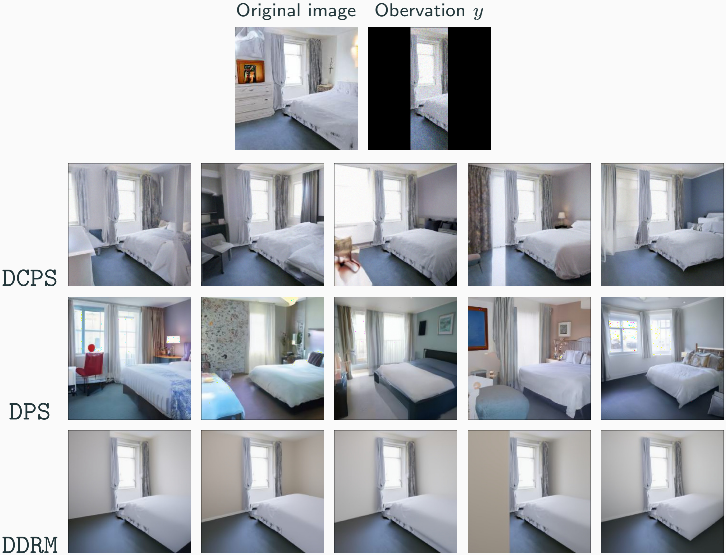
Inpainting and outpainting experiments

Inpainting and outpainting experiments

Inpainting and outpainting experiments



Inpainting and outpainting experiments



Colorization experiments

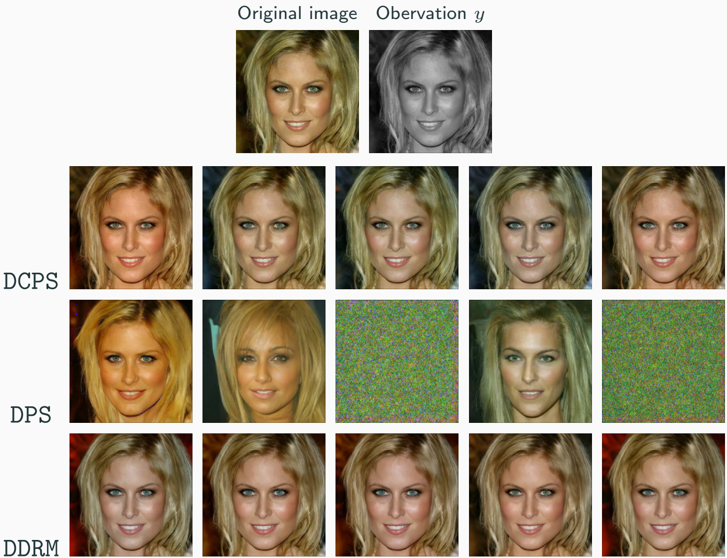
Original image Obervation y

DCPS

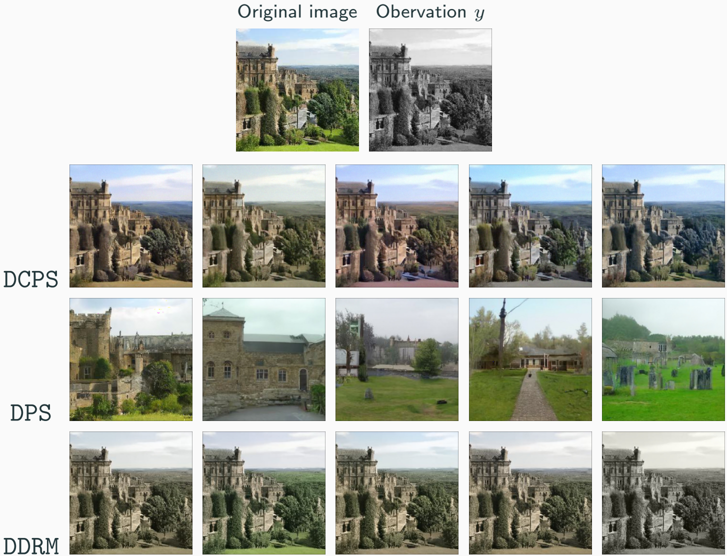
DPS

DDRM

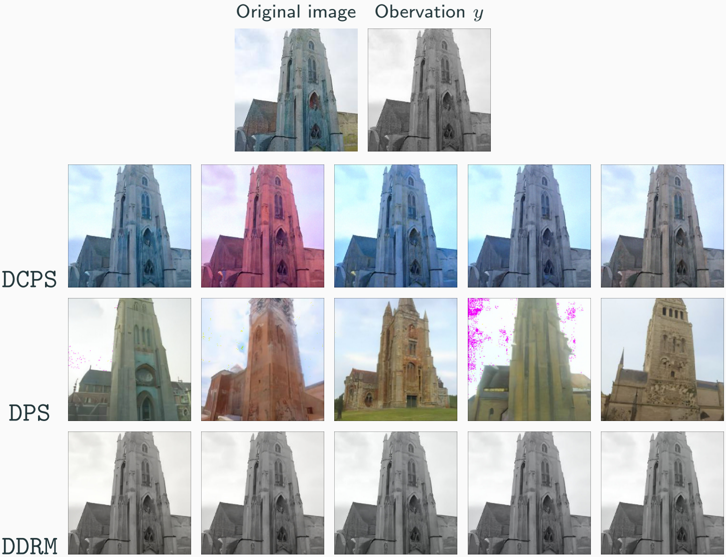
Colorization experiments



Colorization experiments



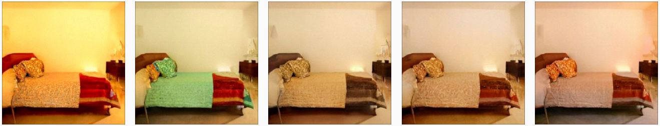
Colorization experiments



Colorization experiments

Original image   Obervation y

DCPS



DPS

DDRM

Thank you!

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

Boys, B., Girolami, M., Pidstrigach, J., Reich, S., Mosca, A., and Akyildiz, O. D. (2023). Tweedie moment projected diffusions for inverse problems. *arXiv preprint arXiv:2310.06721*.

Brosse, N., Durmus, A., Moulines, É., and Sabanis, S. (2019). The tamed unadjusted langevin algorithm. *Stochastic Processes and their Applications*, 129(10):3638–3663.

Cappe, O., Moulines, E., and Ryden, T. (2005). *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Cérou, F., Del Moral, P., Furon, T., and Guyader, A. (2012). Sequential Monte Carlo for rare event estimation. *Statistics and computing*, 22(3):795–808.

Chopin, N., Papaspiliopoulos, O., et al. (2020). *An introduction to sequential Monte Carlo*, volume 4. Springer.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. (2023). Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*.

Del Moral, P. (2004). Feynman-kac formulae. In *Feynman-Kac Formulae*, pages 47–93. Springer.

Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Dockhorn, T., Vahdat, A., and Kreis, K. (2022). Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations (ICLR)*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/ non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, 140(2):107–113.

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

Kawar, B., Elad, M., Ermon, S., and Song, J. (2022). Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Van Gool, L. (2022). Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR.

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

Song, J., Vahdat, A., Mardani, M., and Kautz, J. (2023). Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*.

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

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. S. (2023). Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. In *The Eleventh International Conference on Learning Representations*.

Wu, L., Trippe, B. L., Naesseth, C. A., Cunningham, J. P., and Blei, D. (2023). Practical and asymptotically exact conditional sampling in diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhang, G., Ji, J., Zhang, Y., Yu, M., Jaakkola, T., and Chang, S. (2023). Towards coherent image inpainting using denoising diffusion implicit models. *arXiv preprint arXiv:2304.03322*.