Defining the problem

$$Y = \overline{X} + \sigma_y \varepsilon\,, \quad \varepsilon \sim \mathcal{N}(0, \mathrm{I}_{\mathsf{d}_y})\,, \quad \sigma \geq 0$$

One possible solution

Bayesian Inference and SMC Sampling

Part I

General scheme of generating

1. We have a dataset $D_N = \{X^1, X^2, \ldots, X^{\mathrm{N}}\}$ which we suppose having the distribution Π_0
2. We will then try to approximate Π with a parametric probability distribution p_θ
3. Calculate the Posterior distribution $\phi(\mathrm{d}x)$ from p_{θ_*}
4. Use it to sample new data

Main idea of diffusion models

A diffusion model consists of three major components: the forward process, the backward process, and the sampling procedure. The goal of diffusion models is to learn a diffusion process that generates the probability distribution of a given dataset. They learn the latent structure of a dataset by modeling the way in which data points diffuse through their latent space.

Forward process

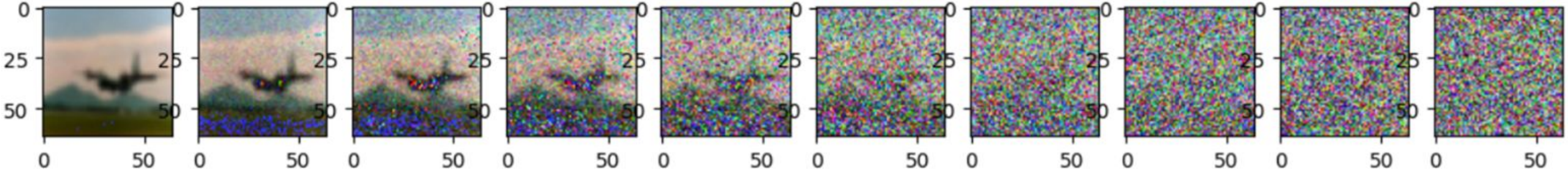
- The goal of this step will be to noise the image. The simplest way to do it is just the following linear transformation adding gaussian noise:

$$X_k = \sqrt{1-\beta_k} X_{k-1} + \sqrt{\beta_k} Z_k \,, \quad \beta_k \in [0,1], \quad X_0 \sim \pi_0 \,,$$

where $Z_k \sim \mathcal{N}(\mathbf{0}_{d_x}, \mathrm{I}_{d_x})$.

$X_k \sim \pi_k$ where [1]

$$\pi_k(\mathrm{d}x_k) := \int \pi_0(\mathrm{d}x_0) \mathcal{N}(\mathrm{d}x_k; \sqrt{\bar{\alpha}_k} x_0, (1-\bar{\alpha}_k) \mathrm{I}_{d_x}) \,.$$



Backward process

$$\pi_{1:n|0}(x_{1:n}|x_0) = \pi_{n|0}(x_n|x_0) \prod_{k=2}^{n} \pi_{k-1|0,k}(x_{k-1}|x_0, x_k)\,,$$

where $\pi_{n|0}(x_n|x_0) = \mathcal{N}(x_n; \bar{\alpha}_n^{1/2} x_0, (1-\bar{\alpha}_n)\mathrm{I})$ and $\pi_{k-1|0,k}$ is the bridge distribution $\pi_{k-1|0,k}(x_{k-1}|x_0, x_k) \propto \pi_{k-1|0}(x_{k-1}|x_0)\pi_{k|k-1}(x_k|x_{k-1})$, i.e.

$$\pi_{k-1|0,k}(x_{k-1}|x_0, x_k) = \mathcal{N}\left(x_{k-1}; \boldsymbol{\mu}_k(x_0, x_k), \sigma_k^2\,\mathrm{I}_d\right)\,,$$

with

$\boldsymbol{\mu}_k(x_0, x_k) = \bar{\alpha}_{k-1}^{1/2} x_0 + (1 - \bar{\alpha}_{k-1} - \sigma_k^2)^{1/2}(x_k - \bar{\alpha}_k^{1/2} x_0)/(1-\bar{\alpha}_k)^{1/2}$.

Backward process

$\rightsquigarrow$ Use this decomposition to turn noise into samples from π_0.

$$\mathsf{p}^{\theta}_{0:n}(\mathrm{d}x_{0:n}) = \mathsf{p}_n(\mathrm{d}x_n)\prod_{k=0}^{n-1} p^{\theta}_k(\mathrm{d}x_k|x_{k+1})\,,$$

where p_n is a std Gaussian and

$$p^{\theta}_k(\mathrm{d}x_k|x_{k+1}) = \mathcal{N}(\mathrm{d}x_k; \mu^{\theta}_{k+1}(x_{k+1}), \beta_{k+1}\mathrm{I}_{d_x})$$

with $\mu^{\theta}_{k+1}(x_{k+1})$ obtained by replacing x_0 in $\boldsymbol{\mu}_{k+1}(x_0, x_{k+1})$ with a prediction

$$\hat{x}_{0|k,\theta}(x_{k+1}) := \bar{\alpha}_{k+1}^{-1/2}\left(x_{k+1} - (1-\bar{\alpha}_{k+1})^{1/2}\mathbf{e}^{\theta}(x_{k+1}, k+1)\right)\,,$$

where $\mathbf{e}^{\theta}(x, k+1)$ is typically a neural network parameterized by θ.

Backward process

The parameter θ is obtained by solving the following optimization problem:

$$\theta_* \in \operatorname{argmin}_\theta \sum_{k=1}^{n} (2 d_x \sigma_k^2 \alpha_k)^{-1} \mathbb{E}\left[\|\epsilon - \mathbf{e}^\theta(\sqrt{\alpha_k} x_0 + \sqrt{1-\alpha_k}\epsilon, k)\|_2^2\right] .$$

$\mathbf{e}^{\theta_*}(X_t, t)$ might be seen as the predictor of the noise added to X_0 to obtain X_t (in the forward pass) and justifies the *prediction* terminology.

Part II

Back to the problem

$$Y = \overline{X} + \sigma_y \varepsilon\,, \quad \varepsilon \sim \mathcal{N}(0, \mathrm{I}_{\mathsf{d}_y})\,, \quad \sigma \geq 0$$

Let’s suppose $\boldsymbol{\sigma}_y$=0

Main ideas

- $\phi_0^y(x_0) = p_0^\theta(x_0|y) \propto p_0^\theta(x_0) g_0^y(x_0)\,,$

Main ideas

- $$\begin{aligned}\phi_0^y(x_0) = p_0^\theta(x_0|y) &\propto p_0^\theta(x_0) g_0^y(x_0)\,, \\ &\propto \int p_n(x_n) \left\{ \prod_{t=1}^{n-1} p_t^\theta(x_t|x_{t+1}) \right\} p_0^\theta(x_0|x_1) g_0^y(x_0) \mathrm{d}x_{1:n}\end{aligned}$$

Main ideas

- $$\phi_0^y(x_0) = p_0^\theta(x_0|y) \propto p_0^\theta(x_0) g_0^y(x_0)\,,$$
  $$\propto \int p_n(x_n) \left\{ \prod_{t=1}^{n-1} p_t^\theta(x_t|x_{t+1}) \right\} p_0^\theta(x_0|x_1) g_0^y(x_0) \mathrm{d}x_{1:n}$$

Not a tractable integral

- Introduce a sequence of potential functions $(g_k^y)_{1<k<n}$ such that :

$$\phi_n^y(x_n) \propto \mathsf{p}_n(x_n) g_n^y(x_n)$$
$$\phi_t^y(x_t) \propto \int g_{t+1}^y(x_{t+1})^{-1} g_t^y(x_t) p_t(x_t|x_{t+1}) \phi_{t+1}^y(\mathrm{d}x_{t+1})$$
$$\phi_t^y(x_t) \propto \mathsf{p}_t(x_t) g_t^y(x_t)$$

MCGdiff

$$\begin{aligned}\phi_k^y(\mathrm{d}x_k) &\propto \int \frac{g_k^y(x_k)}{g_{k+1}^y(x_{k+1})} p_t(\mathrm{d}x_k|x_{k+1})\phi_{k+1}^y(\mathrm{d}x_{k+1}) \\ &\propto \int \underbrace{\frac{\int g_k^y(z_k)p_k(\mathrm{d}z_k|x_{k+1})}{g_{k+1}^y(x_{k+1})}}_{:=\widetilde{\omega}_k(x_{k+1})} p_k^y(\mathrm{d}x_k|x_{k+1})\phi_{k+1}^y(\mathrm{d}x_{k+1})\end{aligned}$$

ϕ_{t+1}^y is $\phi_{t+1}^N = N^{-1}\sum_{i=1}^N \delta_{\xi_{t+1}^i}$

$$\widehat{\phi}_t^N(x_t) = \sum_{i=1}^N \widetilde{\omega}_t(\xi_{t+1}^i)p_t^y(x_t|\xi_{t+1}^i) / \sum_{j=1}^N \widetilde{\omega}_t(\xi_{t+1}^j)$$

MCGdiff

$$\phi_k^y(\mathrm{d}x_k) \propto \int \frac{g_k^y(x_k)}{g_{k+1}^y(x_{k+1})} p_t(\mathrm{d}x_k|x_{k+1}) \phi_{k+1}^y(\mathrm{d}x_{k+1})$$

$$\propto \int \underbrace{\frac{\int g_k^y(z_k) p_k(\mathrm{d}z_k|x_{k+1})}{g_{k+1}^y(x_{k+1})}}_{:=\widetilde{\omega}_k(x_{k+1})} p_k^y(\mathrm{d}x_k|x_{k+1}) \phi_{k+1}^y(\mathrm{d}x_{k+1})$$

ϕ_{t+1}^y is $\phi_{t+1}^N = N^{-1} \sum_{i=1}^N \delta_{\xi_{t+1}^i}$

$$\widehat{\phi}_t^N(x_t) = \sum_{i=1}^N \widetilde{\omega}_t(\xi_{t+1}^i) p_t^y(x_t|\xi_{t+1}^i) / \sum_{j=1}^N \widetilde{\omega}_t(\xi_{t+1}^j)$$

Algorithm 1: `MCGdiff` ($\sigma = 0$)

Input: Number of particles N

Output: $\xi_0^{1:N}$

`// Operations involving index` i `are repeated for each` $i \in [1:N]$

$\overline{z}_n^i \sim \mathcal{N}(\mathbf{0}_{\mathsf{d}_y}, \mathrm{I}_{\mathsf{d}_y})$, $\underline{z}_n^i \sim \mathcal{N}(\mathbf{0}_{\mathsf{d}_x - \mathsf{d}_y}, \mathrm{I}_{\mathsf{d}_x - \mathsf{d}_y})$, $\overline{\xi}_n^i = \mathsf{K}_n \bar{\alpha}_n^{1/2} y + (1 - \bar{\alpha}_n) \mathsf{K}_n \overline{z}_n^i$, $\xi_n^i = \overline{\xi}_n^i \frown \underline{z}_n^i$;

for $s \leftarrow n-1 : 0$ **do**

  if $s = n-1$ **then**

    $\widetilde{\omega}_{n-1}(\xi_n^i) = \mathcal{N}\left(\bar{\alpha}_n^{1/2} y; \overline{\mathsf{m}}_n(\xi_n^i), 2 - \bar{\alpha}_n\right)$;

  else

    $\widetilde{\omega}_s(\xi_{s+1}^i) = \mathcal{N}\left(\bar{\alpha}_s^{1/2} y; \overline{\mathsf{m}}_{s+1}(\xi_{s+1}^i), \sigma_{s+1}^2 + 1 - \bar{\alpha}_s\right) / \mathcal{N}\left(\bar{\alpha}_{s+1}^{1/2} y; \overline{\xi}_{s+1}^i, 1 - \bar{\alpha}_{s+1}\right)$;

  $I_{s+1}^i \sim \mathrm{Cat}\left(\{\widetilde{\omega}_s(\xi_{s+1}^j) / \sum_{k=1}^N \widetilde{\omega}_s(\xi_{s+1}^k)\}_{j=1}^N\right)$, $\overline{z}_s^i \sim \mathcal{N}(\mathbf{0}_{\mathsf{d}_y}, \mathrm{I}_{\mathsf{d}_y})$, $\underline{z}_s^i \sim \mathcal{N}(\mathbf{0}_{\mathsf{d}_x - \mathsf{d}_y}, \mathrm{I}_{\mathsf{d}_x - \mathsf{d}_y})$;

  $\overline{\xi}_s^i = \mathsf{K}_s \bar{\alpha}_s^{1/2} y + (1 - \mathsf{K}_s) \overline{\mathsf{m}}_{s+1}(\xi_{s+1}^i) + (1 - \alpha_s)^{1/2} \mathsf{K}_s^{1/2} \overline{z}_s^i$, $\underline{\xi}_s^i = \underline{\mathsf{m}}_{s+1}(\xi_{s+1}^i) + \sigma_{s+1} \underline{z}_s^i$;

  Set $\xi_s^i = \overline{\xi}_s^i \frown \underline{\xi}_s^i$;

Part III

Convergence

$$\mathsf{KL}(\phi_0^y \parallel \Phi_0^N) \leq \mathsf{C}_{0:n}^y (N-1)^{-1} + \mathsf{D}_{0:n}^y N^{-2}$$

- N number of samples
- n number of steps

Quadratic Convergence

General Linear Inverse Problem

Model:

$$Y = \mathrm{A}X + \sigma_y \varepsilon$$

$$\Big\downarrow \text{ SVD (} \mathrm{A} = \mathrm{U}\mathrm{S}\overline{\mathrm{V}}^T \text{)}$$

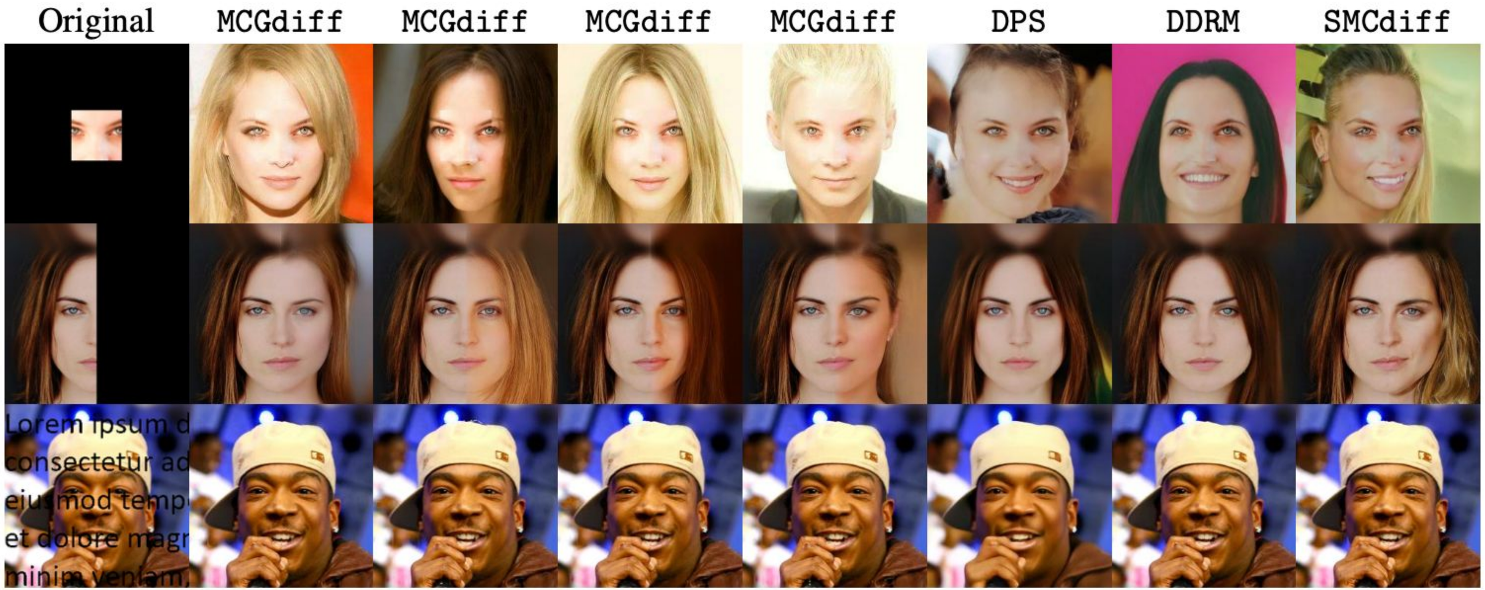
$$\mathbf{Y} = \overline{\mathbf{X}} + \sigma_y \mathrm{S}^{-1} \tilde{\varepsilon}$$

where $\mathfrak{p}_0(\mathbf{x}_0) := \mathrm{p}_0(\mathrm{V}\mathbf{x}_0)$

$\mathbf{X} := \mathrm{V}^T X$

MCGdiff Algorithm can be extended in this case.

Article Results for Image Inpainting



Model and Results

Model:

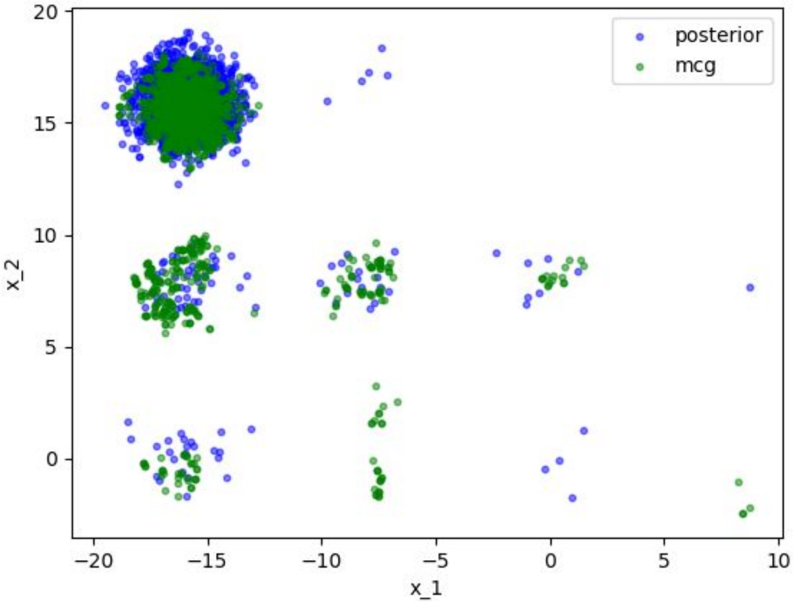
- GMM of 25 components
- Total Dimension and Masked Dimension
- Create a linear inverse problems

Advantage of GMM: **Exact** Posterior is known

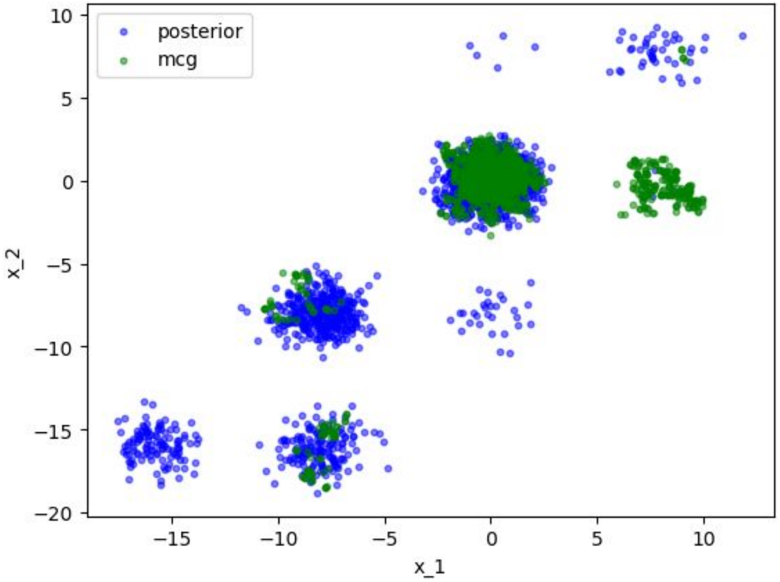
For **Numerics**, change the inputs and evaluate a metrics for each case:

NUMERICS

With Total dimension fixed



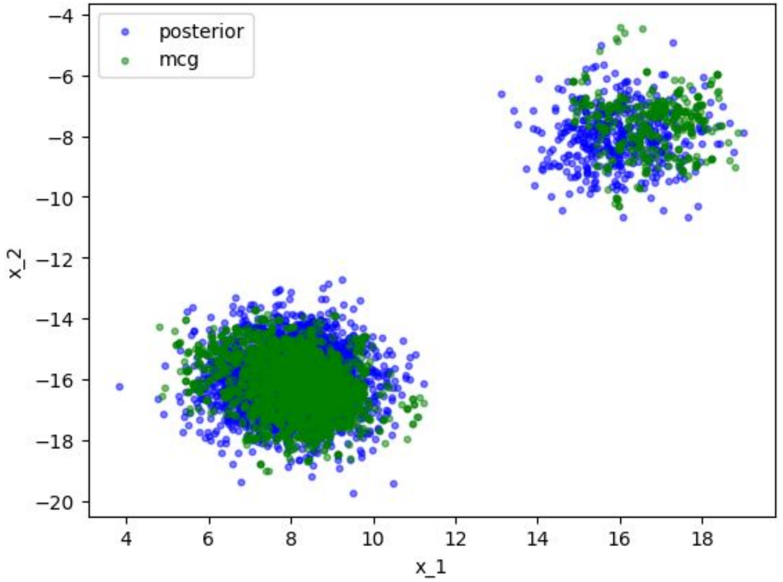
$d_x = 80, d_y = 4$



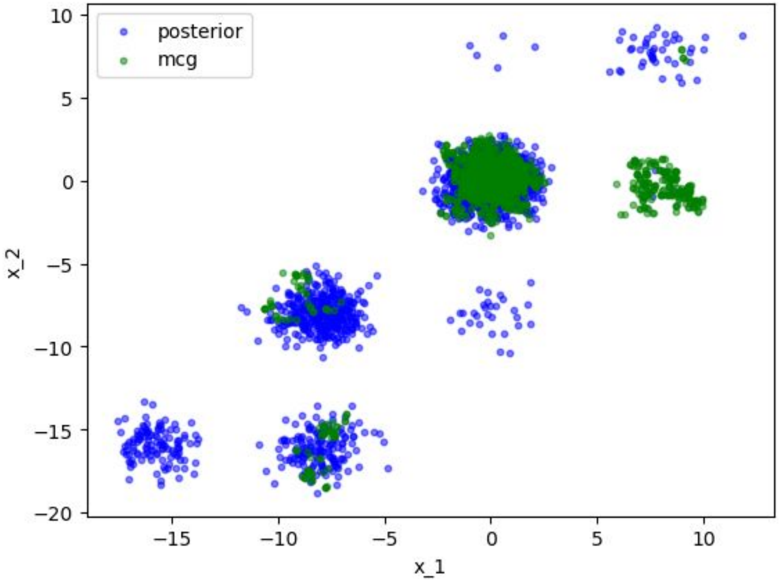
$d_x = 80, d_y = 2$

NUMERICS

With Masked dimension fixed



$d_x = 400, d_y = 2$



$d_x = 80, d_y = 2$

d_x	d_y	steps	SW
8	2	20	1.59
8	4	20	2.1
80	2	20	3.9
80	4	20	2.3
400	2	20	1.7
80	4	100	1.8

Sliced Wasserstein Score for MCGDiff model with 100 sliced

Thanks