

Deep Learning

Lecture 12: Diffusion models

Prof. Gilles Louppe
g.louppe@uliege.be



Today

- VAEs
- Variational diffusion models
- Score-based generative models

Caution: See also the side notes derived in class.

Applications

A few motivating examples.



Diffusion models have emerged as powerful generative models, beating previous state-of-the-art models (such as GANs) on a variety of tasks.

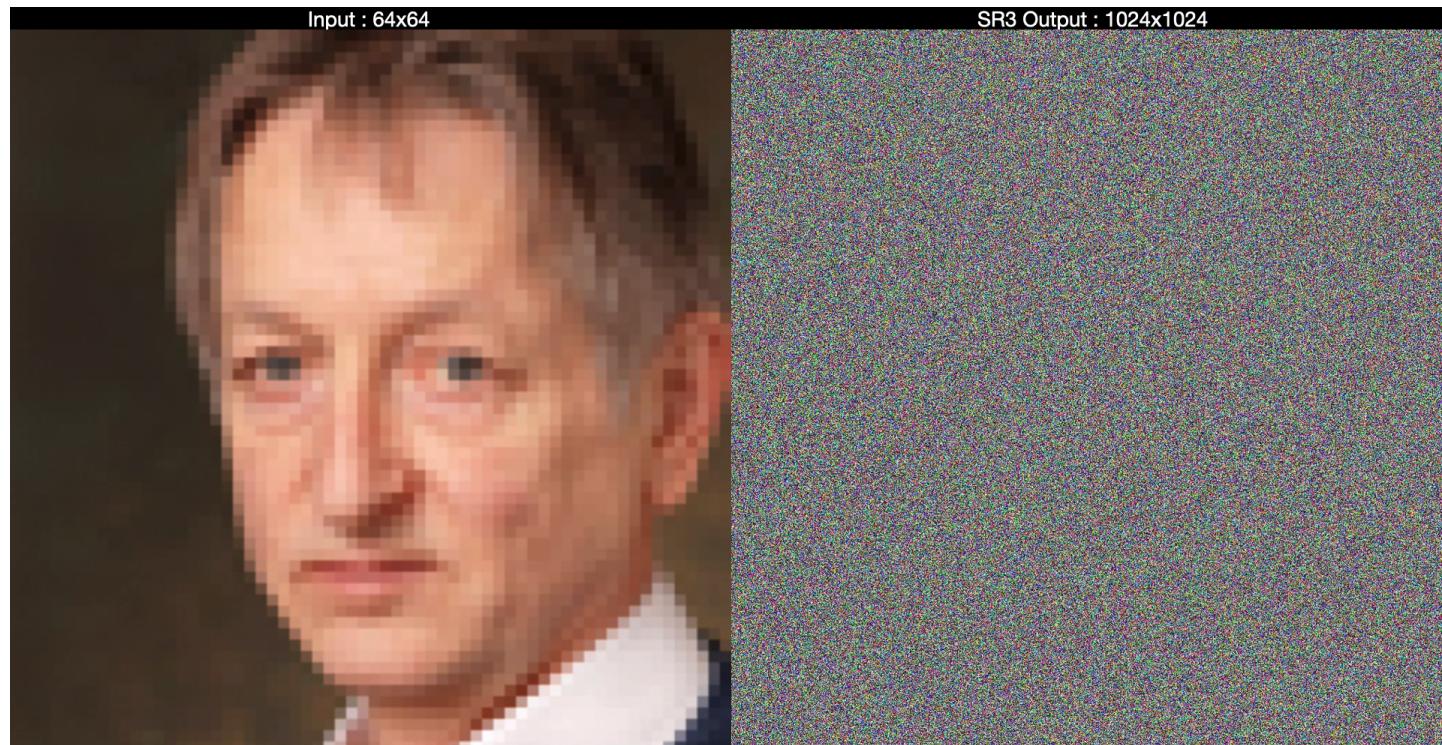
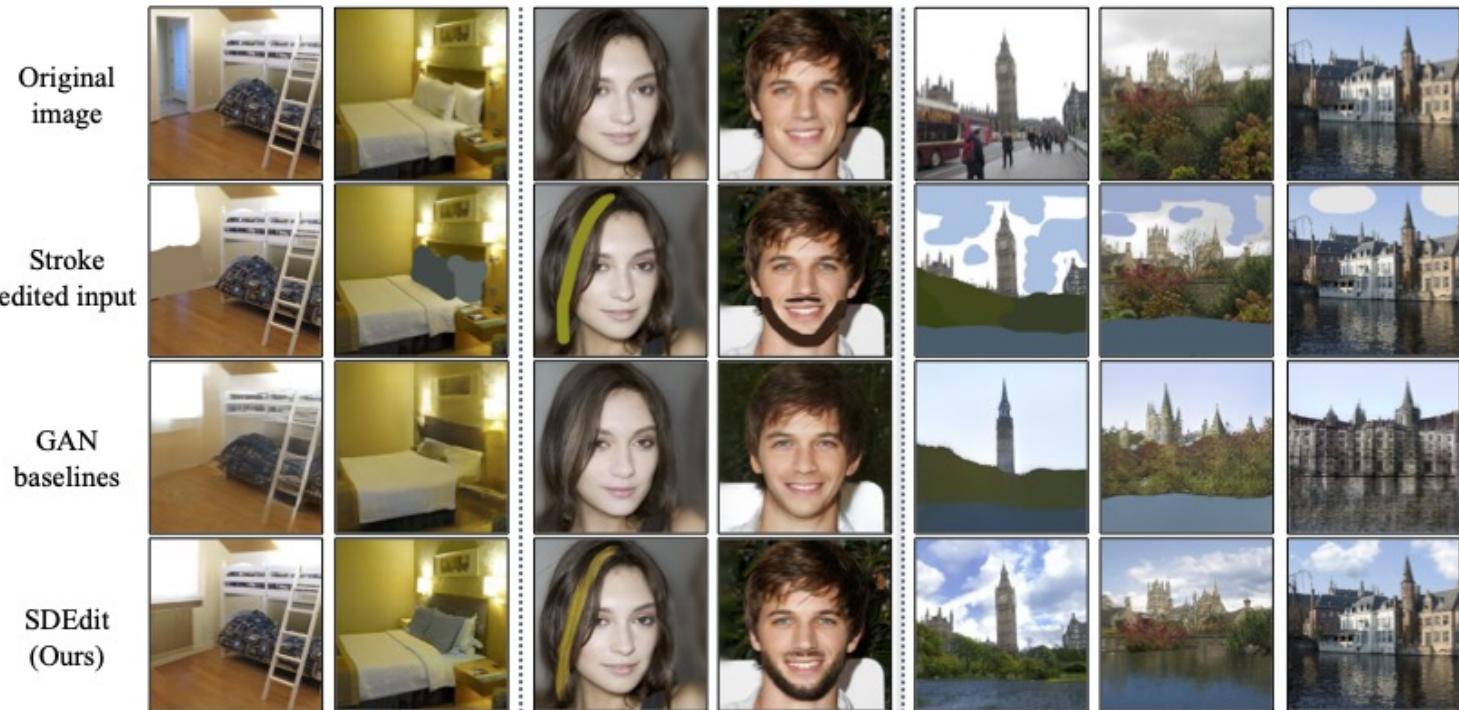


Image super-resolution

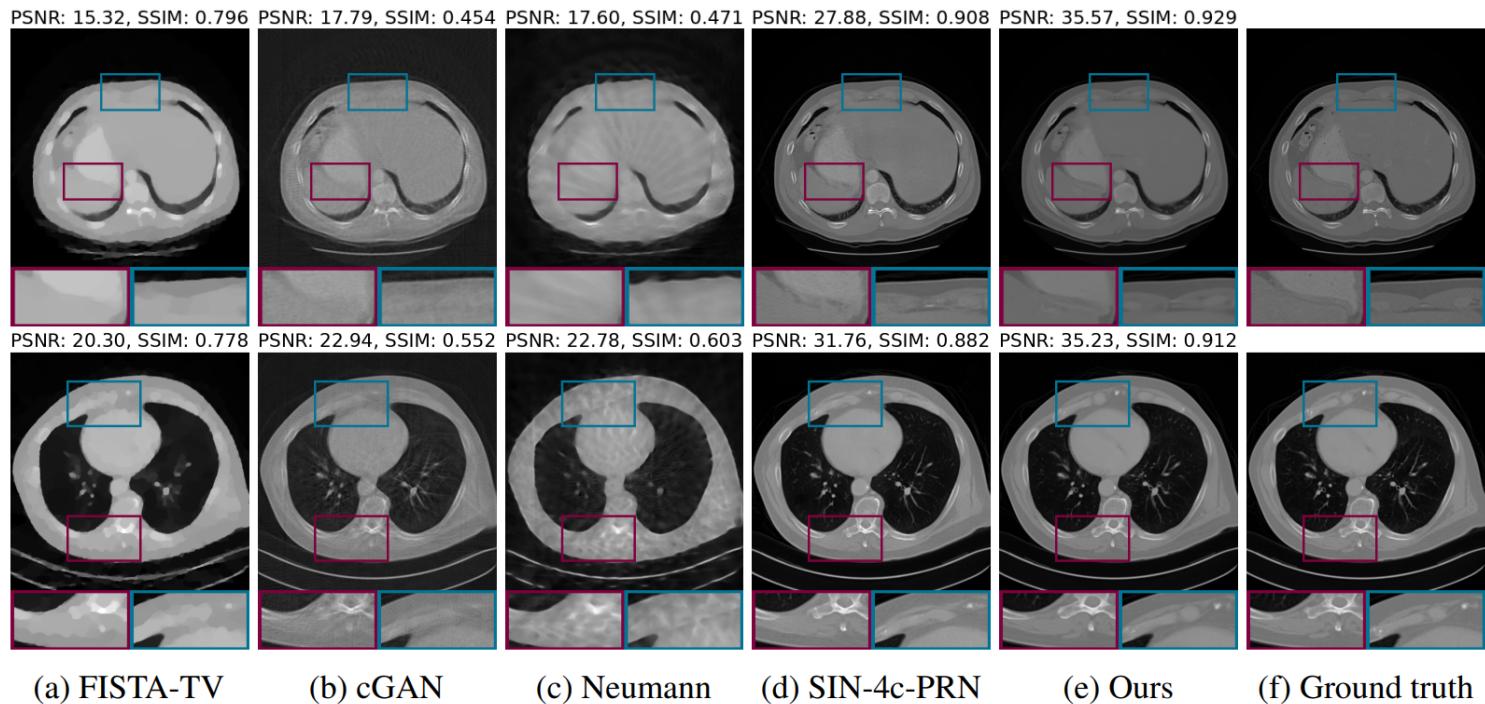
A group of teddy bears in suite in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.



Text-to-image generation



Artistic tools and image editing

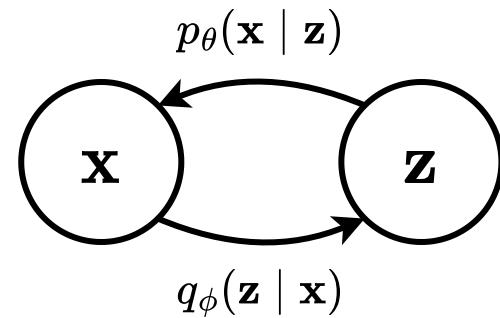


Solving inverse problems in medical imaging

VAEs

A short recap.

Variational autoencoders



Training

$$\begin{aligned}\theta^*, \phi^* &= \arg \max_{\theta, \phi} \mathbb{E}_{p(\mathbf{x})} \text{ELBO}(\mathbf{x}; \theta, \phi) \\ &= \arg \max_{\theta, \phi} \mathbb{E}_{p(\mathbf{x})} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \arg \max_{\theta, \phi} \mathbb{E}_{p(\mathbf{x})} \left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p(\mathbf{z})) \right].\end{aligned}$$

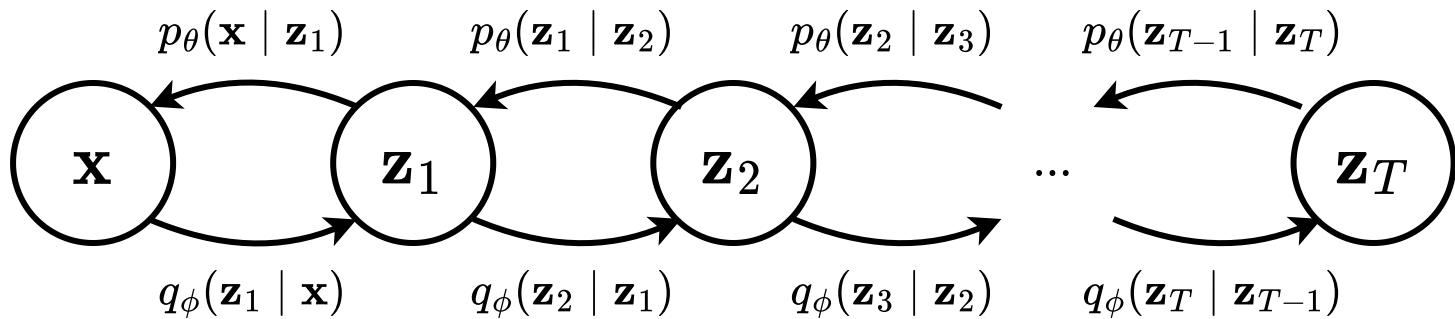
The prior matching term limits the expressivity of the model.

Solution: Make $p(\mathbf{z})$ a learnable distribution.



Hierarchical VAEs

The prior $p(\mathbf{z})$ is itself a VAE, and recursively so for its own hyper-prior.

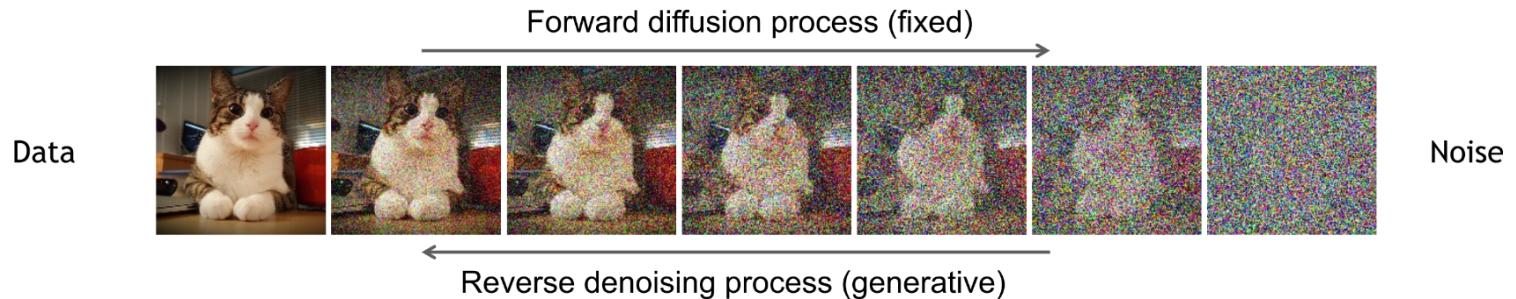


Similarly to VAEs, training is done by maximizing the ELBO.
(See side notes.)

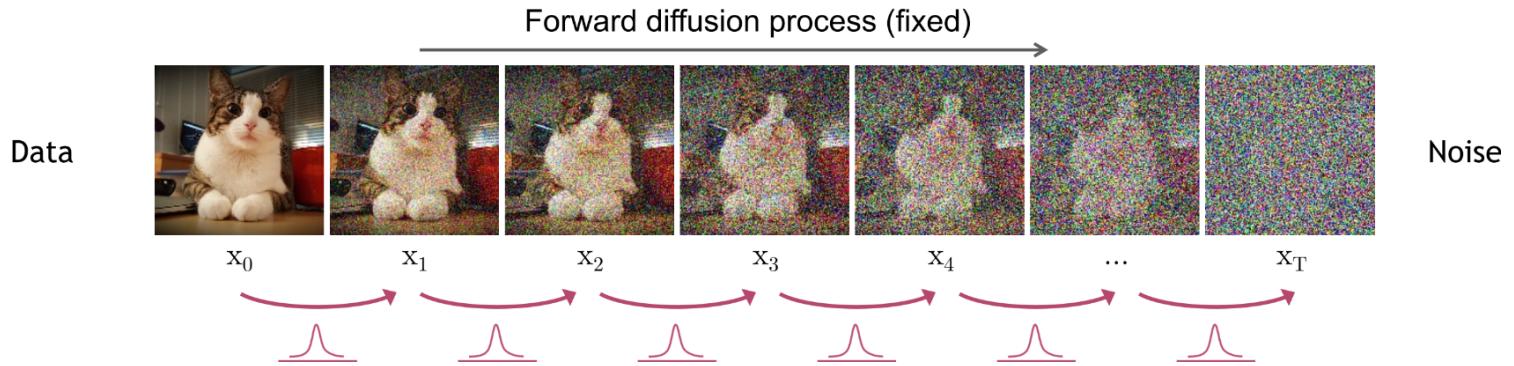
Variational diffusion models

Variational diffusion models are HVAEs with the following constraints:

- The latent dimension is the same as the data dimension.
- The encoder is fixed to linear Gaussian transitions $q(\mathbf{x}_t | \mathbf{x}_{t-1})$.
- The hyper-parameters are set such that $q(\mathbf{x}_T | \mathbf{x}_0)$ is a standard Gaussian.

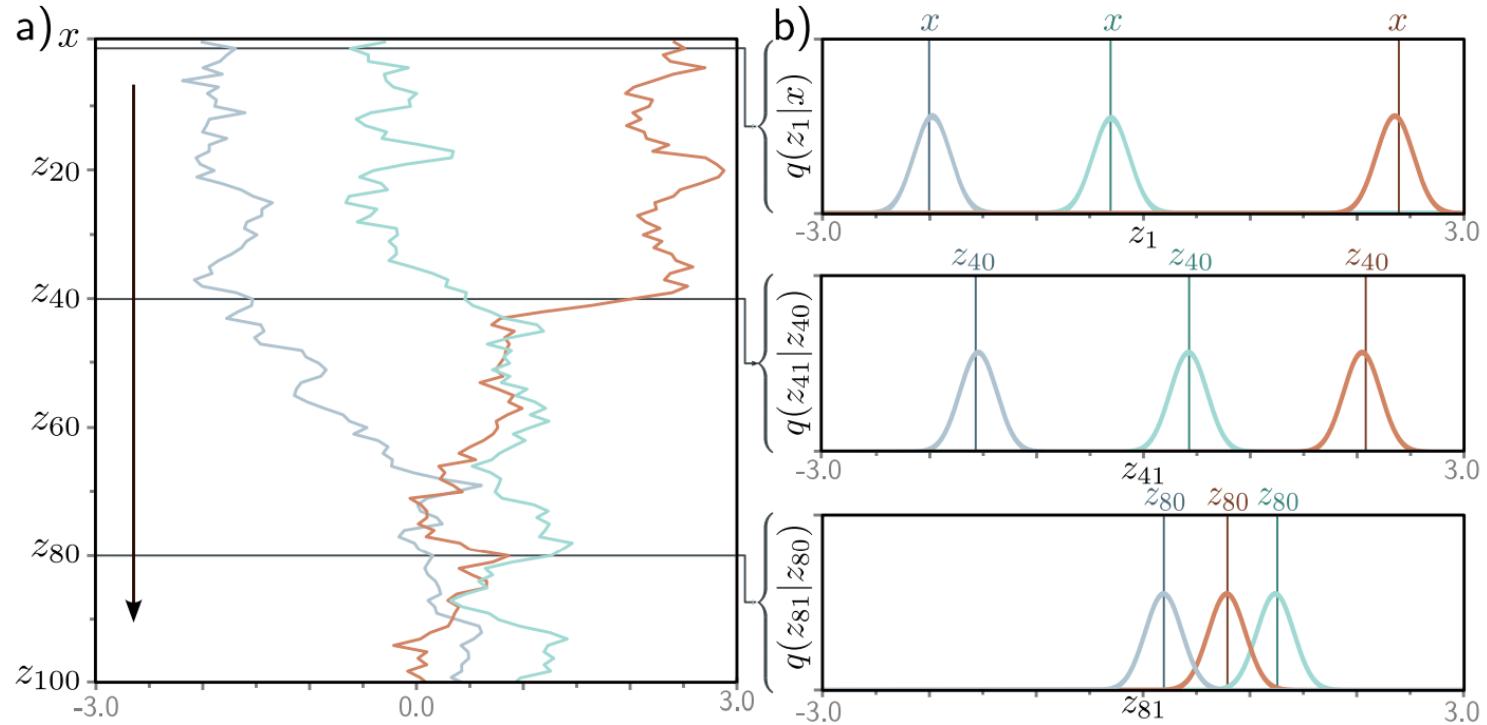


Forward diffusion process

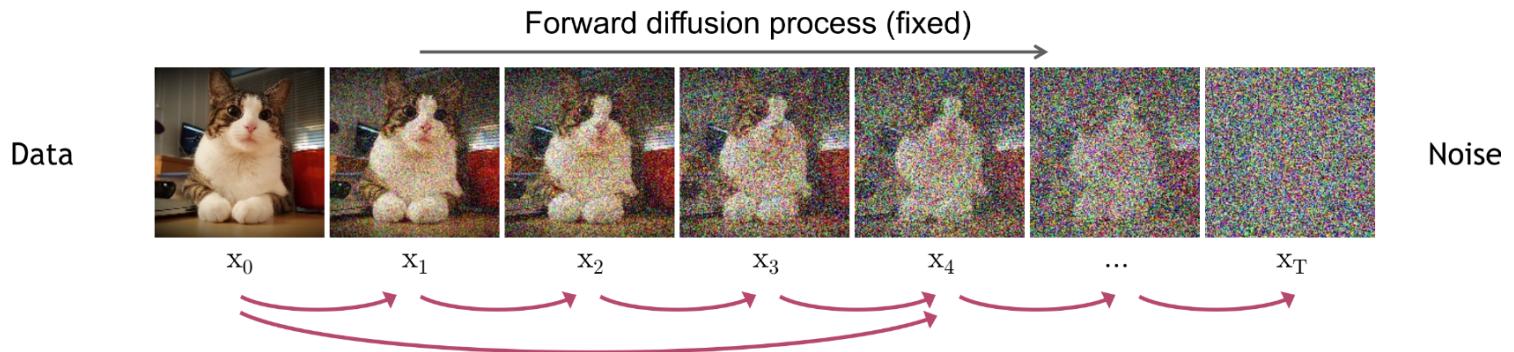


With $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we have

$$\mathbf{x}_t = \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \epsilon$$
$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t} \mathbf{x}_{t-1}, (1 - \alpha_t) \mathbf{I})$$
$$q(\mathbf{x}_{1:T} | \mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1})$$



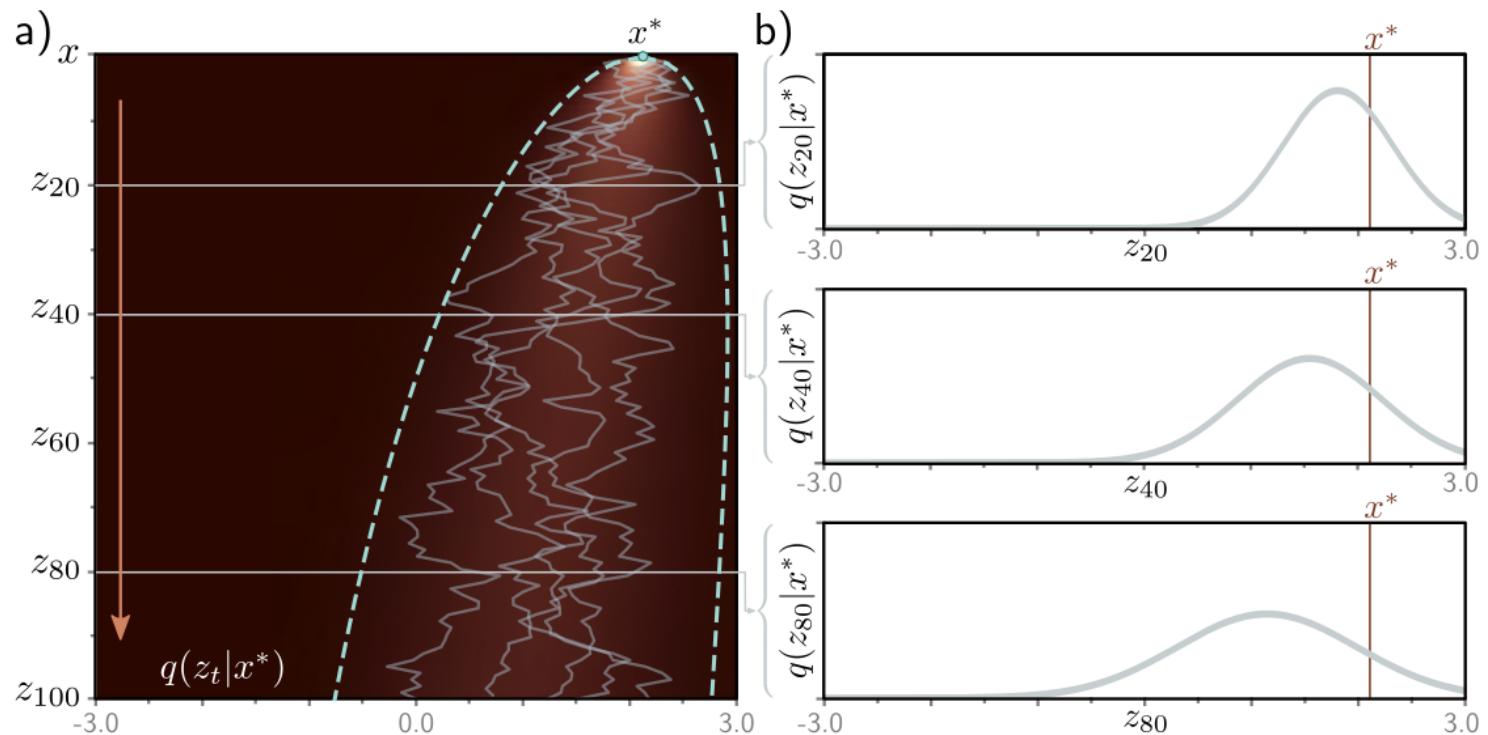
Diffusion kernel

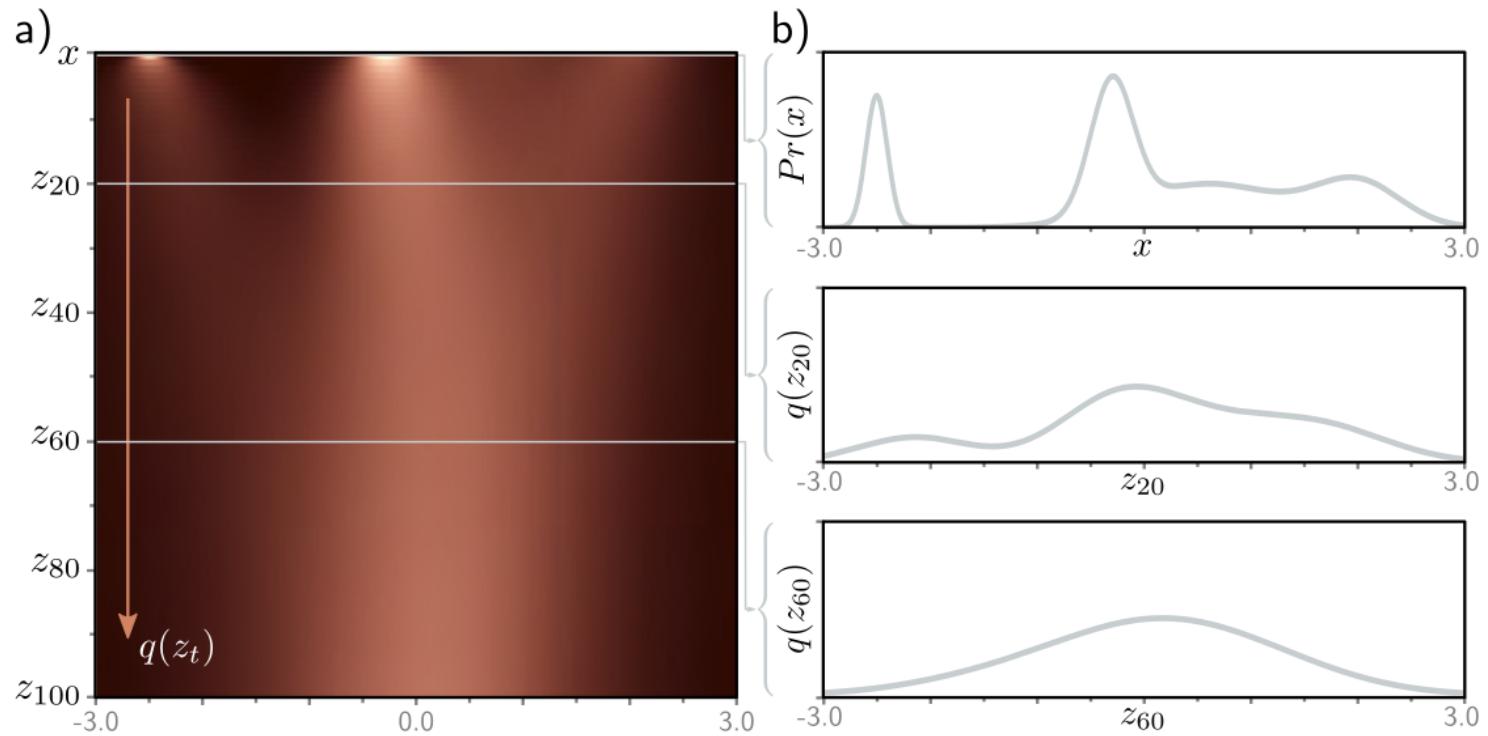


With $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we have

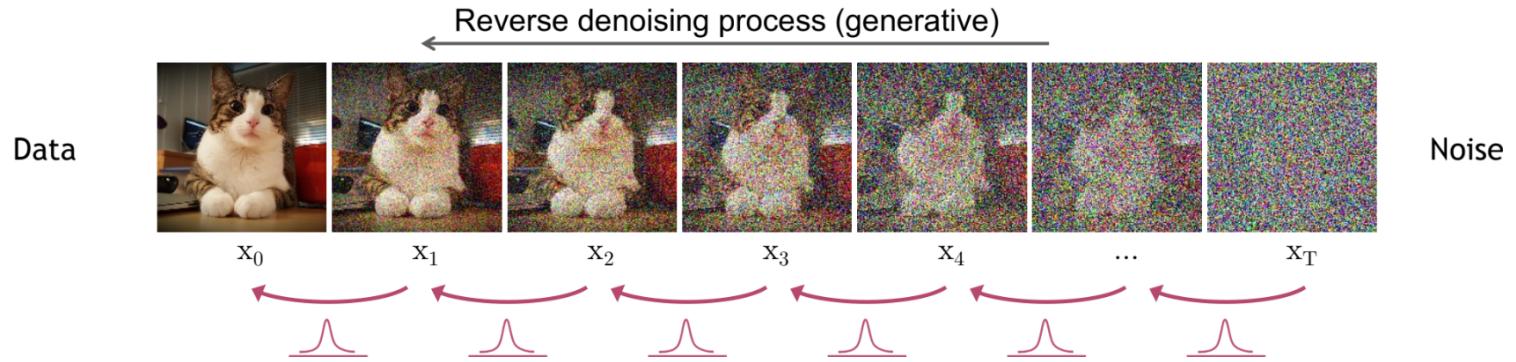
$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$$





Reverse denoising process



$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, I)$$

$$p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_{\theta}(\mathbf{x}_t), \Sigma_{\theta}(\mathbf{x}_t))$$

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)$$

Training

For learning the parameters θ of the reverse process, we can form a variational lower bound on the log-likelihood of the data as

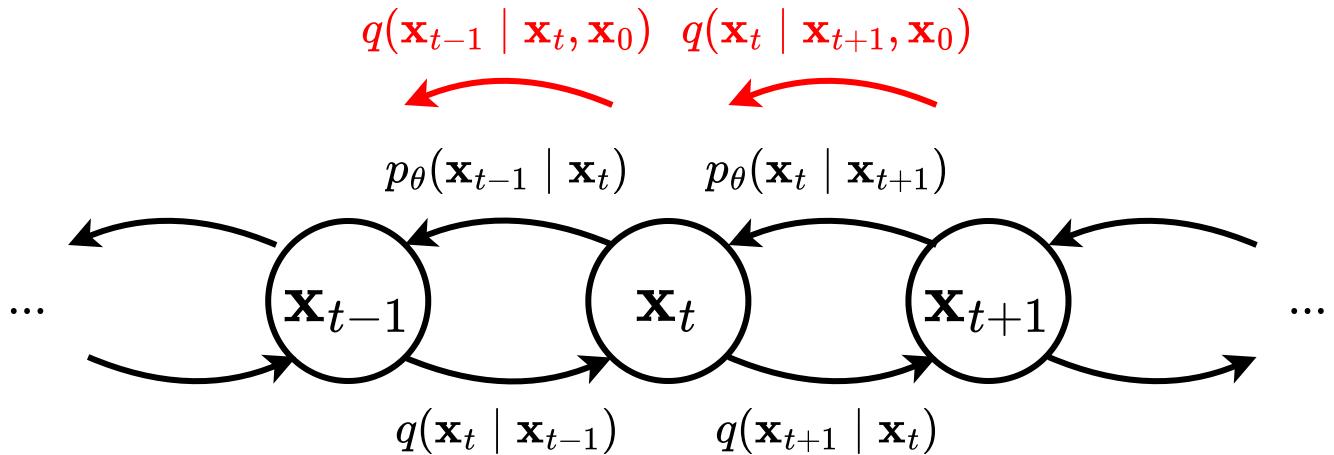
$$\mathbb{E}_{q(\mathbf{x}_0)} [\log p_\theta(\mathbf{x}_0)] \geq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] := L$$

This objective can be rewritten as

$$\begin{aligned} L &= \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] \\ &= \mathbb{E}_q(\mathbf{x}_0) \left[L_0 - \sum_{t>1} L_{t-1} - L_T \right] \end{aligned}$$

where

- $L_0 = \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} [\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)]$ can be interpreted as a reconstruction term. It can be approximated and optimized using a Monte Carlo estimate.
- $L_{t-1} = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \text{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) || p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))$ is a denoising matching term. The transition $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ provides a learning signal for the reverse process, since it defines how to denoise the noisified input \mathbf{x}_t with access to the original input \mathbf{x}_0 .
- $L_T = \text{KL}(q(\mathbf{x}_T|\mathbf{x}_0) || p_\theta(\mathbf{x}_T))$ represents how close the distribution of the final noisified input is to the standard Gaussian. It has no trainable parameters.



The distribution $q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)$ is the tractable posterior distribution

$$\begin{aligned} q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) &= \frac{q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1} | \mathbf{x}_0)}{q(\mathbf{x}_t | \mathbf{x}_0)} \\ &= \mathcal{N}(\mathbf{x}_{t-1}; \mu_q(\mathbf{x}_t, \mathbf{x}_0, t), \sigma_t^2 I) \end{aligned}$$

where

$$\begin{aligned} \mu_q(\mathbf{x}_t, \mathbf{x}_0, t) &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \mathbf{x}_0 \\ \sigma_t^2 &= \frac{(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \end{aligned}$$

Interpretation 1: Denoising

To minimize the expected KL divergence L_{t-1} , we need to match the reverse process $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$ to the tractable posterior. Since both are Gaussian, we can match their means and variances.

By construction, the variance of the reverse process can be set to the known variance σ_t^2 of the tractable posterior.

For the mean, we reuse the analytical form of $\mu_q(\mathbf{x}_t, \mathbf{x}_0, t)$ and parameterize the mean of the reverse process using a **denoising network** as

$$\mu_\theta(\mathbf{x}_t) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \hat{\mathbf{x}}_\theta(\mathbf{x}_t, t).$$

Under this parameterization, the minimization of expected KL divergence L_{t-1} can be rewritten as

$$\begin{aligned} & \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \text{KL}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) || p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)) \\ &= \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \frac{1}{2\sigma_t^2} \|\mu_{\theta}(\mathbf{x}_t) - \mu_q(\mathbf{x}_t, \mathbf{x}_0, t)\|_2^2 \\ &= \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \frac{1}{2\sigma_t^2} \frac{\bar{\alpha}_{t-1}(1 - \alpha_t)^2}{(1 - \bar{\alpha}_t)^2} \|\hat{\mathbf{x}}_{\theta}(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2 \end{aligned}$$

Optimizing a VDM amounts to learning a neural network that predicts the original ground truth \mathbf{x}_0 from a noisy input \mathbf{x}_t .

Finally, minimizing the summation of the L_{t-1} terms across all noise levels t can be approximated by minimizing the expectation over all timesteps as

$$\arg \min_{\theta} \mathbb{E}_{t \sim U\{2, T\}} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \text{KL}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) || p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)).$$

Interpretation 2: Noise prediction

A second interpretation of VDMs can be obtained using the reparameterization trick. Using

$$\mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \epsilon}{\sqrt{\bar{\alpha}_t}},$$

we can rewrite the mean of the tractable posterior as

$$\begin{aligned}\mu_q(\mathbf{x}_t, \mathbf{x}_0, t) &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \mathbf{x}_0 \\ &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \epsilon}{\sqrt{\bar{\alpha}_t}} \\ &= \dots \\ &= \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{(1 - \bar{\alpha}_t)\alpha_t}} \epsilon\end{aligned}$$

Accordingly, the mean of the reverse process can be parameterized with a **noise-prediction network** as

$$\mu_\theta(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{(1 - \bar{\alpha}_t)\alpha_t}} \epsilon_\theta(\mathbf{x}_t, t).$$

Under this parameterization, the minimization of the expected KL divergence L_{t-1} can be rewritten as

$$\begin{aligned} & \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \text{KL}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) || p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)) \\ &= \arg \min_{\theta} \mathbb{E}_{\mathcal{N}(\epsilon; \mathbf{0}, I)} \frac{1}{2\sigma_t^2} \frac{(1 - \alpha_t)^2}{(1 - \bar{\alpha}_t)\alpha_t} \|\underbrace{\epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, t) - \epsilon}_\mathbf{x_t}\|_2^2 \end{aligned}$$

Optimizing a VDM amounts to learning a neural network that predicts the noise ϵ that was added to the original ground truth \mathbf{x}_0 to obtain the noisy \mathbf{x}_t .

Algorithms

Algorithm 1 Training

```
1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
      
$$\nabla_{\theta} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t)\|^2$$

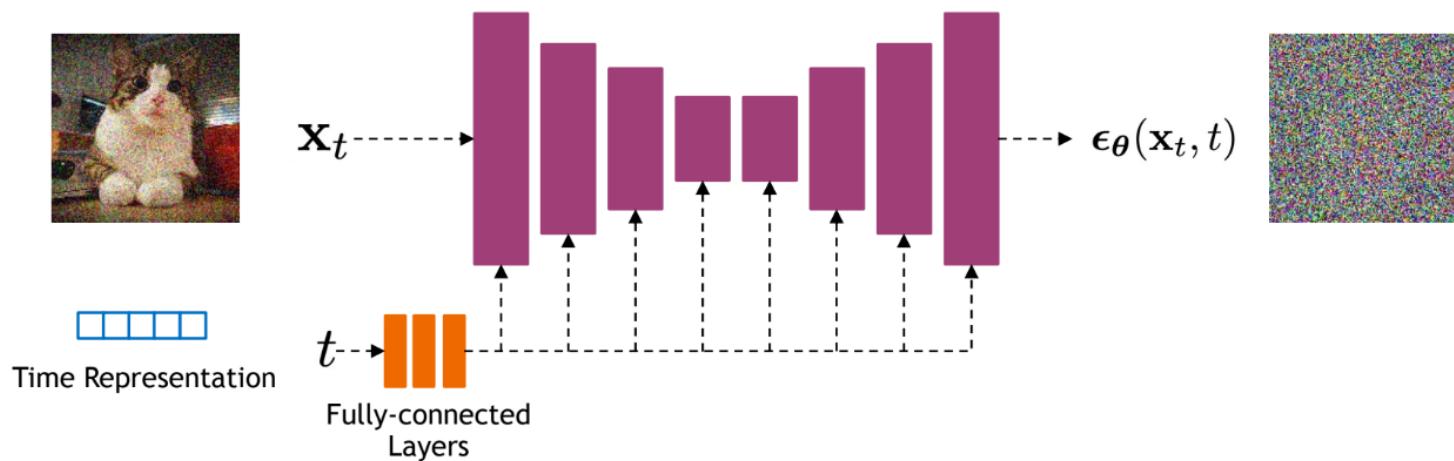
6: until converged
```

Algorithm 2 Sampling

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

Network architectures

Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$.

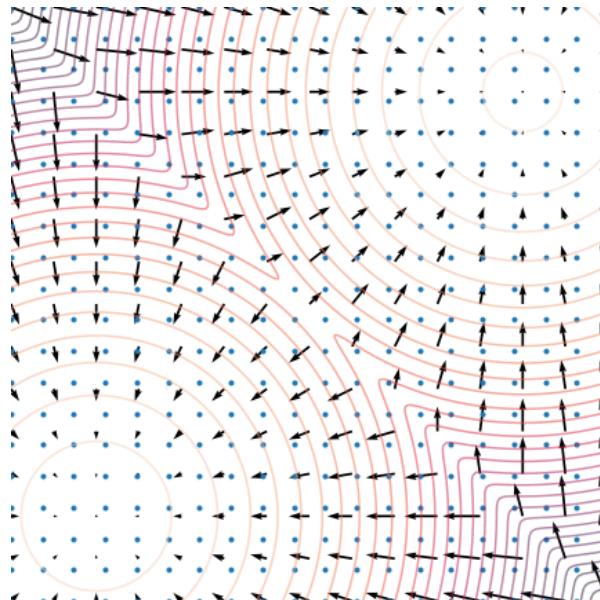


Score-based generative models

The score function

The score function $\nabla_{\mathbf{x}_0} \log q(\mathbf{x}_0)$ is a vector field that points in the direction of the highest density of the data distribution $q(\mathbf{x}_0)$.

It can be used to find modes of the data distribution or to generate samples by Langevin dynamics.



Interpretation 3: Denoising score matching

A third interpretation of VDMs can be obtained by reparameterizing \mathbf{x}_0 using Tweedie's formula, as

$$\mathbf{x}_0 = \frac{\mathbf{x}_t + (1 - \bar{\alpha}_t) \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0)}{\sqrt{\bar{\alpha}_t}},$$

which we can plug into the mean of the tractable posterior to obtain

$$\begin{aligned}\mu_q(\mathbf{x}_t, \mathbf{x}_0, t) &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \mathbf{x}_0 \\ &= \dots \\ &= \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0).\end{aligned}$$

The mean of the reverse process can be parameterized with a **score network** as

$$\mu_\theta(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} s_\theta(\mathbf{x}_t, t).$$

Under this parameterization, the minimization of the expected KL divergence L_{t-1} can be rewritten as

$$\begin{aligned} & \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \text{KL}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) || p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)) \\ &= \arg \min_{\theta} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \frac{1}{2\sigma_t^2} \frac{(1 - \alpha_t)^2}{\alpha_t} \|s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 \end{aligned}$$

Optimizing a score-based model amounts to learning a neural network that predicts the score $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0)$ of the tractable posterior.

Since $s_\theta(\mathbf{x}_t, t)$ is learned in expectation over the data distribution $q(\mathbf{x}_0)$, the score network will eventually approximate the score of the marginal distribution $q(\mathbf{x}_t)$, for each noise level t , that is

$$s_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t).$$

Ancestral sampling

Sampling from the score-based diffusion model is done by starting from $\mathbf{x}_T \sim p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and then following the estimated reverse Markov chain, as

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} s_\theta(\mathbf{x}_t, t) + \sigma_t \mathbf{z}_t,$$

where $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, for $t = T, \dots, 1$.

Conditional sampling

To turn a diffusion model $p_\theta(\mathbf{x}_{0:T})$ into a conditional model, we can add conditioning information y at each step of the reverse process, as

$$p_\theta(\mathbf{x}_{0:T}|y) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, y).$$

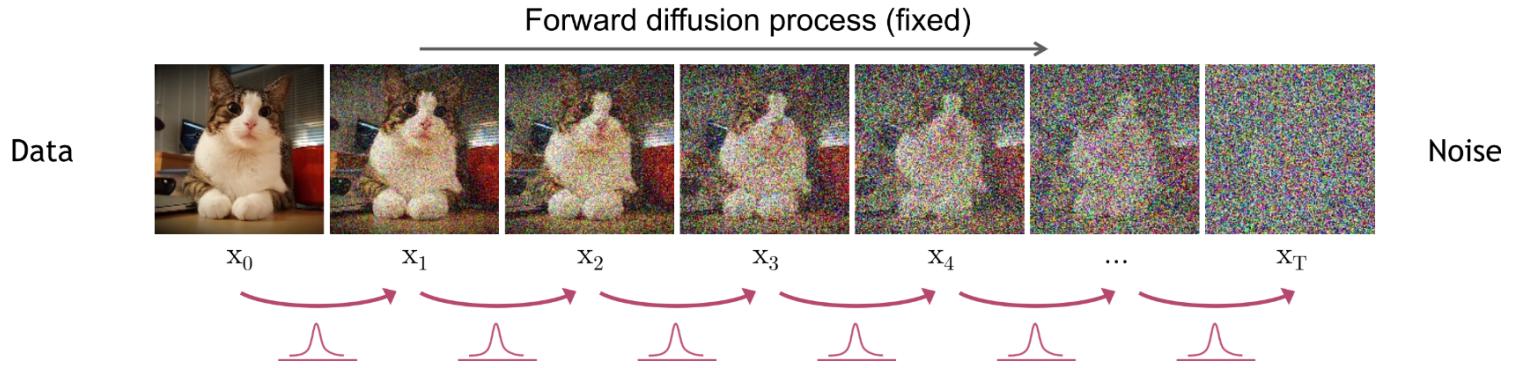
With a score-based model however, we can use the Bayes rule and notice that

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | y) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p(y | \mathbf{x}_t),$$

where we leverage the fact that the gradient of $\log p(y)$ with respect to \mathbf{x}_t is zero.

In other words, controllable generation can be achieved by adding a conditioning signal during sampling, without having to retrain the model. E.g., train an extra classifier $p(y | \mathbf{x}_t)$ and use it to control the sampling process by adding its gradient to the score.

Continuous-time diffusion models



With $\beta_t = 1 - \alpha_t$, we can rewrite the forward process as

$$\begin{aligned}\mathbf{x}_t &= \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &= \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &= \sqrt{1 - \beta(t)} \Delta_t \mathbf{x}_{t-1} + \sqrt{\beta(t)} \Delta_t \mathcal{N}(\mathbf{0}, \mathbf{I})\end{aligned}$$

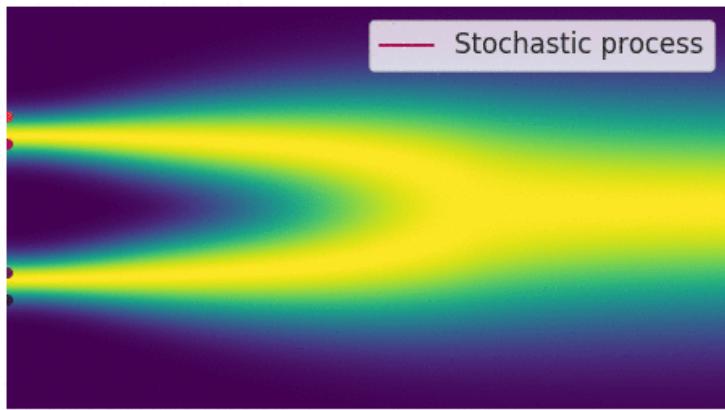
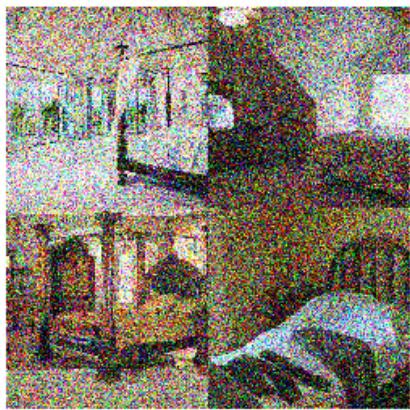
In the limit of many small steps, i.e. as $\Delta_t \rightarrow 0$, we can further rewrite the forward process as

$$\begin{aligned}\mathbf{x}_t &= \sqrt{1 - \beta(t)\Delta_t} \mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta_t} \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &\approx \mathbf{x}_{t-1} - \frac{\beta(t)\Delta_t}{2} \mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta_t} \mathcal{N}(\mathbf{0}, \mathbf{I}).\end{aligned}$$

This last update rule corresponds to the Euler-Maruyama discretization of the stochastic differential equation (SDE)

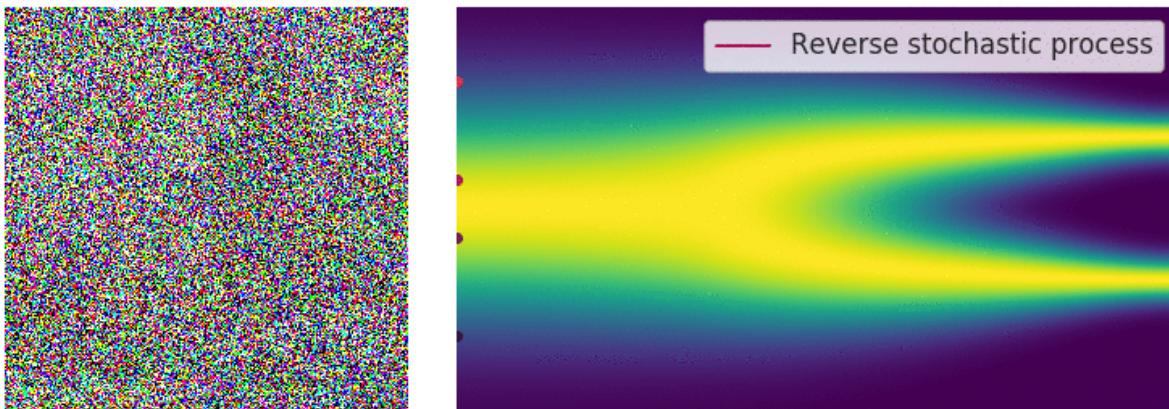
$$d\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t dt + \sqrt{\beta(t)} d\mathbf{w}_t$$

describing the diffusion in the infinitesimal limit.



The reverse process satisfies a reverse-time SDE that can be derived analytically from the forward-time SDE and the score of the marginal distribution $q(\mathbf{x}_t)$, as

$$d\mathbf{x}_t = \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) \right] dt + \sqrt{\beta(t)}dw_t.$$



The score $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$ of the marginal diffused density $q(\mathbf{x}_t)$ is not tractable, but can be estimated using denoising score matching (DSM) by solving

$$\arg \min_{\theta} \mathbb{E}_{t \sim U[0,T]} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \| s_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \|_2^2,$$

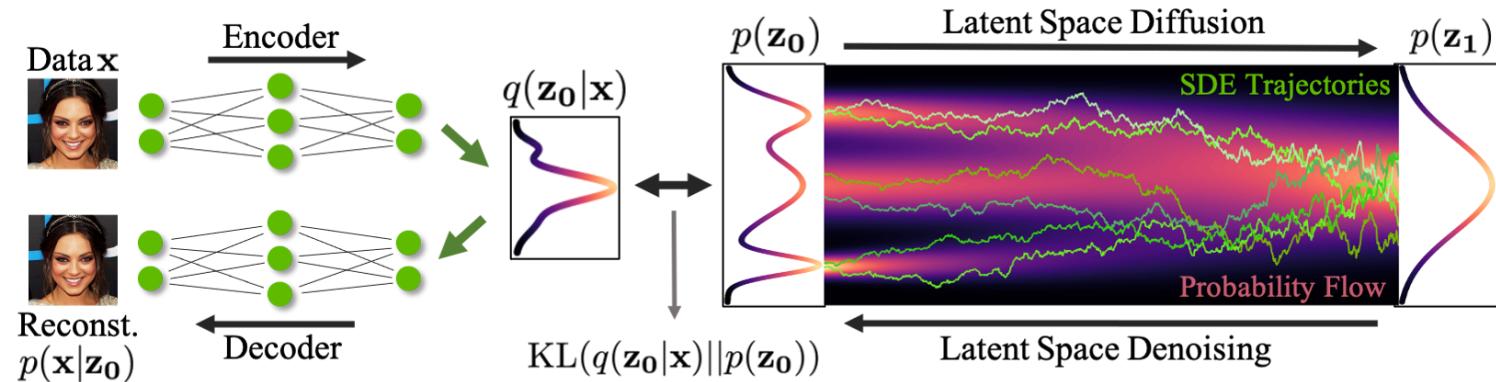
which will result in $s_{\theta}(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$ after expectation over $q(\mathbf{x}_0)$.

This is just the **same objective** as for VDMs!

Latent-space diffusion models

Directly modeling the data distribution can be make the denoising process difficult to learn. A more effective approach is to combine VAEs with a diffusion prior.

- The distribution of latent embeddings is simpler to model.
- Diffusion on non-image data is possible with tailored autoencoders.





The end.