Score-Based Generative Modeling through Stochastic Differential Equations

Yang Song*
Stanford University
yangsong@cs.stanford.edu

Jascha Sohl-Dickstein
Google Brain
jaschasd@google.com

Diederik P. Kingma
Google Brain
durk@google.com

Abhishek Kumar
Google Brain
abhishk@google.com

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

Ben Poole
Google Brain
pooleb@google.com

Content

Overview on generative modeling approaches

1. Likelihood-based methods (i.e. VAEs)
2. Implicit generative methods (i.e. GANs)
3. Score-based methods

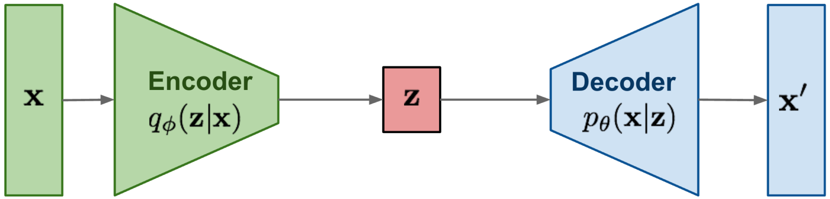
Langevin Dynamics

Score-based generative modeling with stochastic differential equations (SDEs)

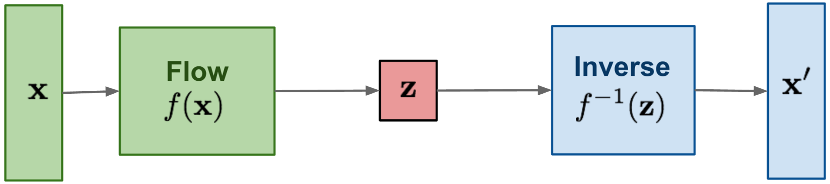
Generative Modeling

1. Likelihood-based methods,
directly learn the distribution's probability density via maximum likelihood. (Auto regressive models , normalizing flow models, energy-based models (EBMs), Variational Auto-Encoders (VAEs))

VAE: maximize
variational lower bound



Flow-based models:
Invertible transform of
distributions

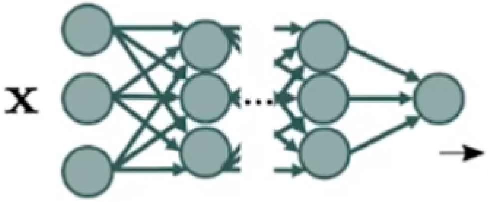


✗ likelihood-based models either have to use specialized architectures to build a normalized probability model (e.g., autoregressive models, flow models),
✗ or use of surrogate losses (e.g., the evidence lower bound used in variational auto-encoders).

Generative Modeling

1. Likelihood-based methods,

When using a parameterized model to approximate data distribution we should make sure that it is normalized.



$$f_{\boldsymbol{\theta}}(\mathbf{x})$$

$$\rightarrow p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{e^{f_{\boldsymbol{\theta}}(\mathbf{x})}}{Z_{\boldsymbol{\theta}}}$$

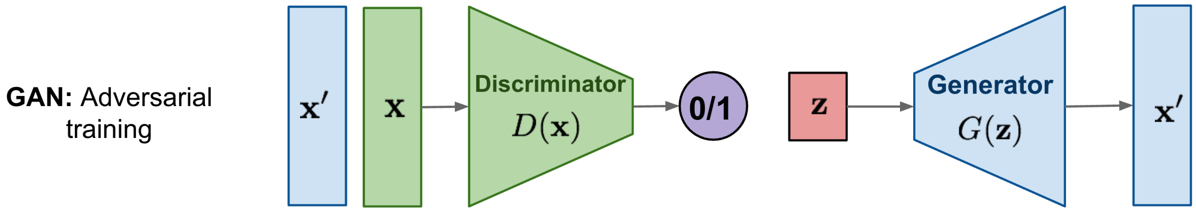
Normalizing constant
is generally intractable to compute.

$$\max_{\theta} \sum_{i=1}^{N} \log p_\theta(\mathbf{x}_i).$$

$$\int p_\theta(\mathbf{x})\mathrm{d}\mathbf{x} = 1.$$

Generative Modeling

2. Implicit generative methods,
Learn the sampling process, (i.e. generative adversarial networks (GANs), where new samples from the data distribution are synthesized by transforming a random Gaussian vector with a neural network).



✗ Unstable training due to the adversarial training procedure.

Generative Modeling

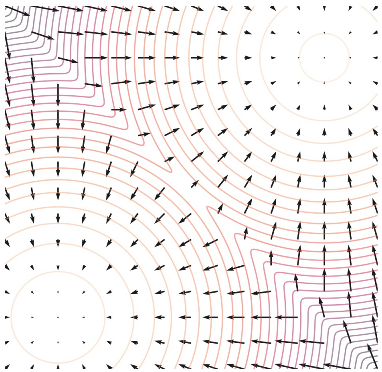
3. Score-based methods

Approximate $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ instead of approximating $p(\mathbf{x})$

(Stein) score function   Probability density function

$\nabla_{\mathbf{x}} \log p(\mathbf{x})$

(Stein) score function



Score function (the vector field) and density function (contours) of a mixture of two Gaussians.

$$p_\theta(\mathbf{x}) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}$$

$$\mathbf{s}_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z_\theta}_{=0} = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}).$$

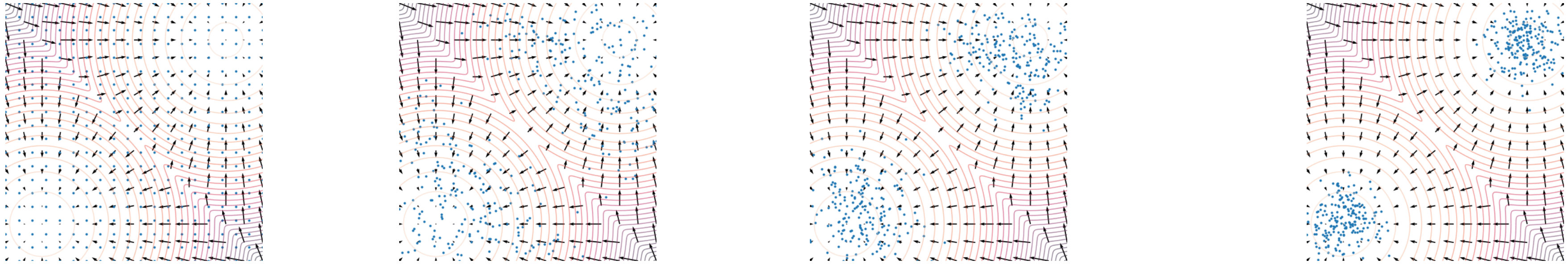
Langevin dynamics

Is an approach for mathematical modeling of dynamics of molecular systems.
Start from a random sample x_0 and iterate the following:

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \boxed{\sqrt{2\epsilon}\, \mathbf{z}_i,} \quad i = 0, 1, \cdots, K,$$

$$\mathbf{z}_i \sim \mathcal{N}(0, I).$$

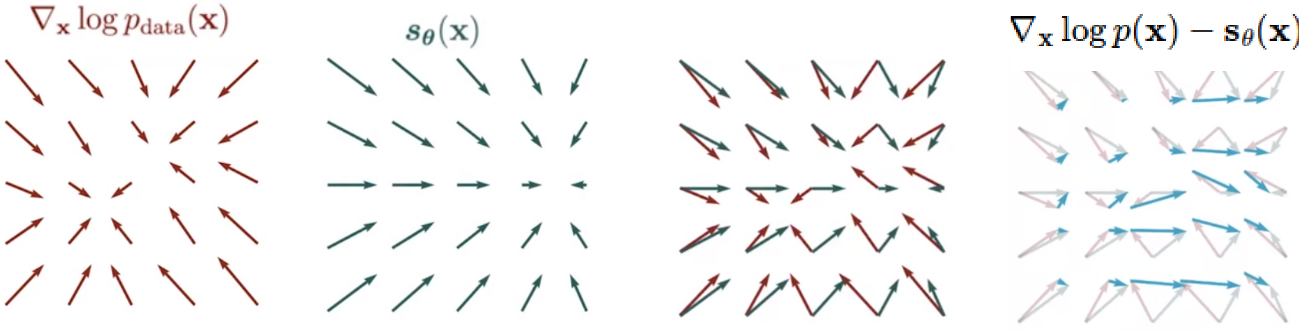
This extra term adds a bit of noise to avoid converging to one point.



x_0 → x_T

Objective; Fisher divergence

Exact Score Approximated Score



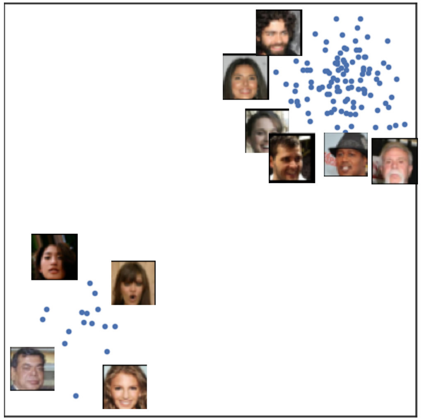
Objective

$$\mathbb{E}_{p(\mathbf{x})}[||\nabla_{\mathbf{x}} \log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2]$$

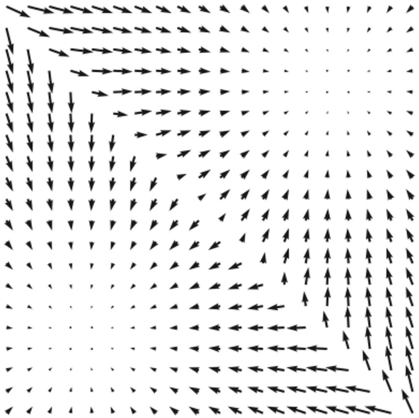
Unknown

Score matching
is the solution.

Score-based generative modeling procedure



score matching



Langevin dynamics

Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p(\mathbf{x})$$

Scores

$$\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

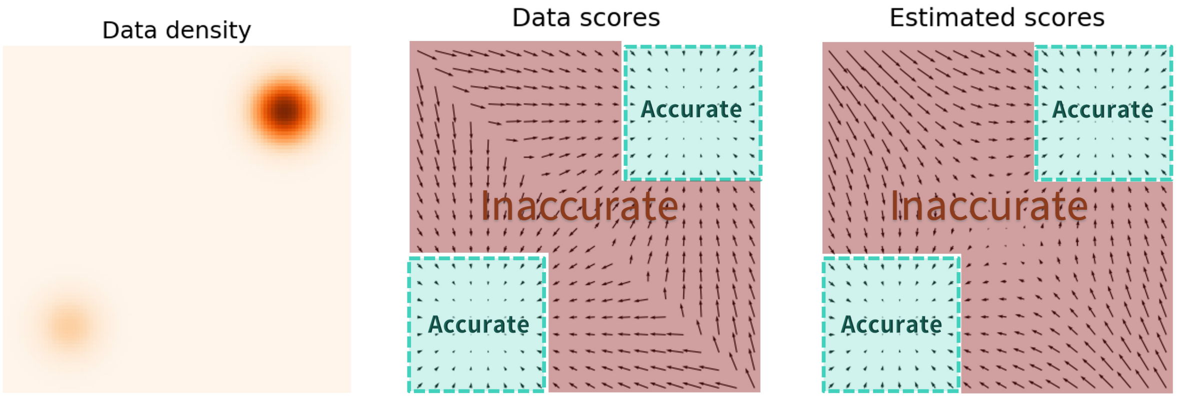
New samples

Is everything okay?

Major pitfall of naive score-based generative modeling

Objective: $$\mathbb{E}_{p(\mathbf{x})}[||\nabla_{\mathbf{x}} \log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2] = \int p(\mathbf{x})||\nabla_{\mathbf{x}} \log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2 \mathrm{d}\mathbf{x}$$

✗ Estimated score functions are inaccurate in low density regions
And initial samples are more likely to be in the low density region.

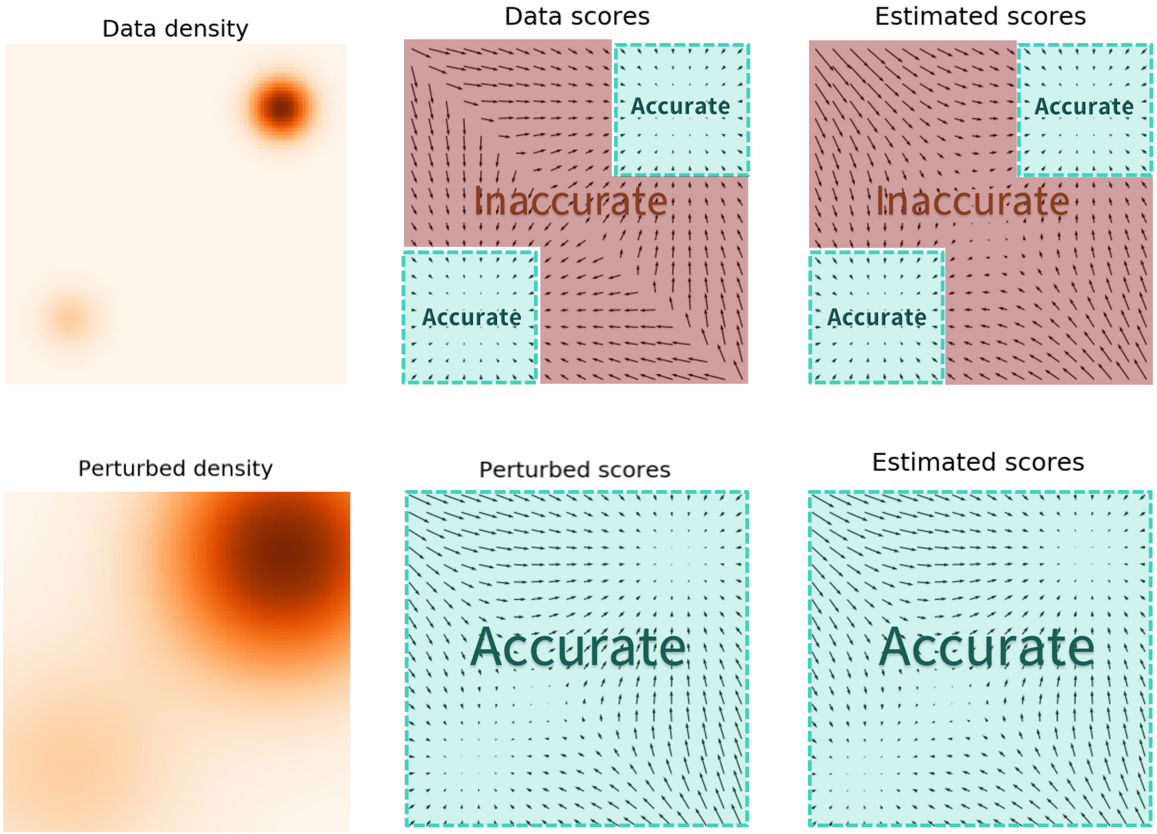


This prevents high quality sampling with Langevin dynamics.

Perturbations with noise

When the noise magnitude is sufficiently large, it can populate low data density regions to improve the accuracy of estimated scores.

How do we choose an appropriate noise scale?

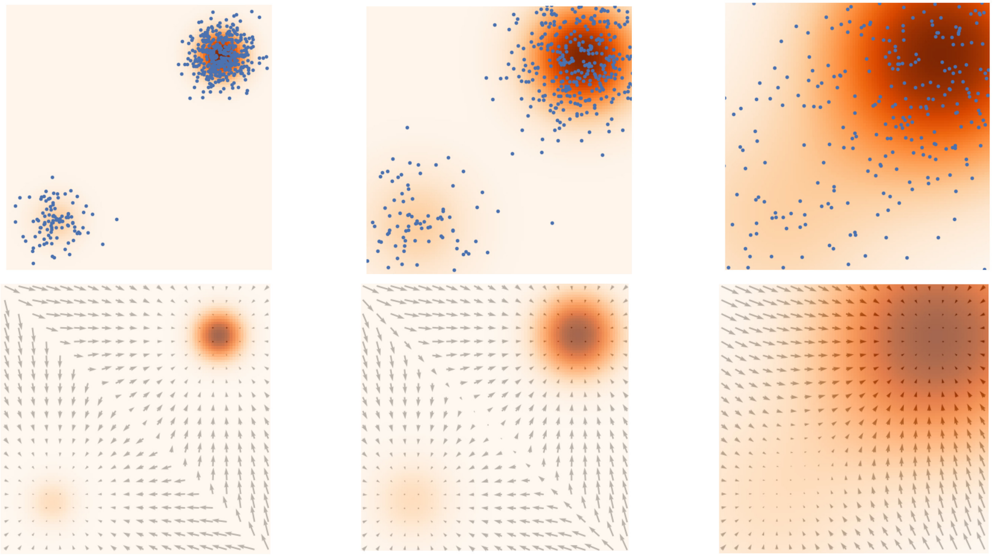


Multiple scales of noise perturbations

$$\mathbf{x} + \sigma_i \mathbf{z}, \text{ with } \mathbf{z} \sim \mathcal{N}(0, I).$$

Standard deviations of added Gaussian noise

$$\sigma_1 < \sigma_2 < \sigma_3$$



Noise Conditional Score-Based Model $\mathbf{s}_\theta(\mathbf{x}, i)$

$$\mathbf{s}_\theta(\mathbf{x}, i) \approx \nabla_\mathbf{x} \log p_{\sigma_i}(\mathbf{x})$$

A U-Net with skip connections
is used for $\mathbf{s}_\theta(\mathbf{x}, i)$

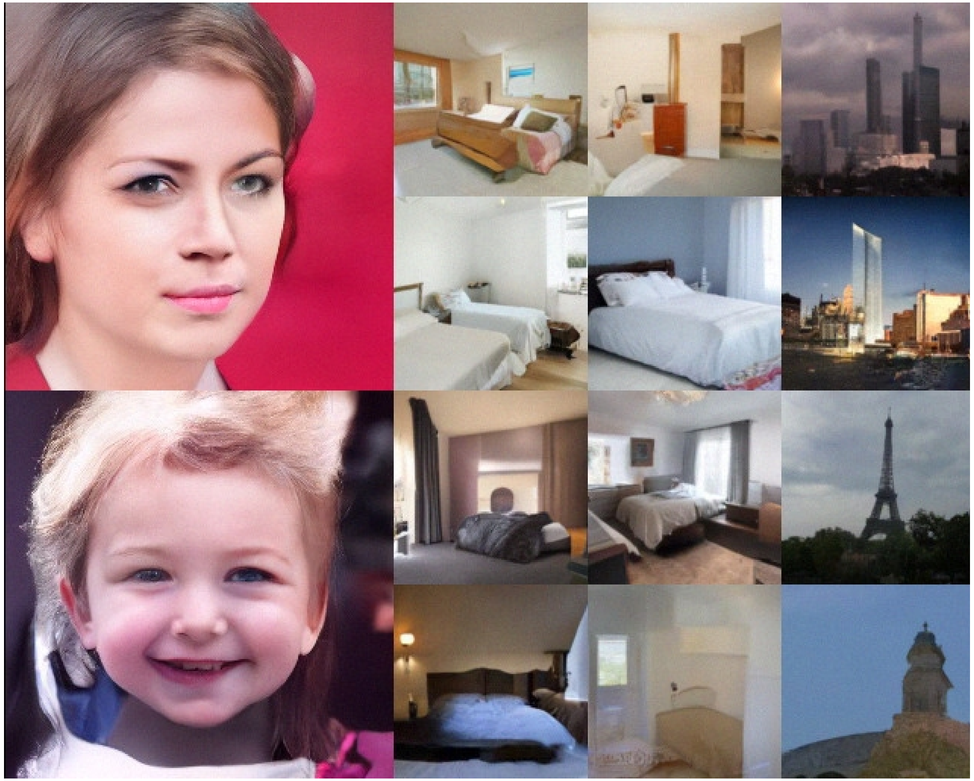
Perturbed image with multiple scales of noise.

Annealed Langevin dynamics.

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T.$

Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$ ▷ α_i is the step size.
  for $t \leftarrow 1$ to T **do**
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \dfrac{\alpha_i}{2} \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\, \mathbf{z}_t$
  end for
  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
end for
return $\tilde{\mathbf{x}}_T$



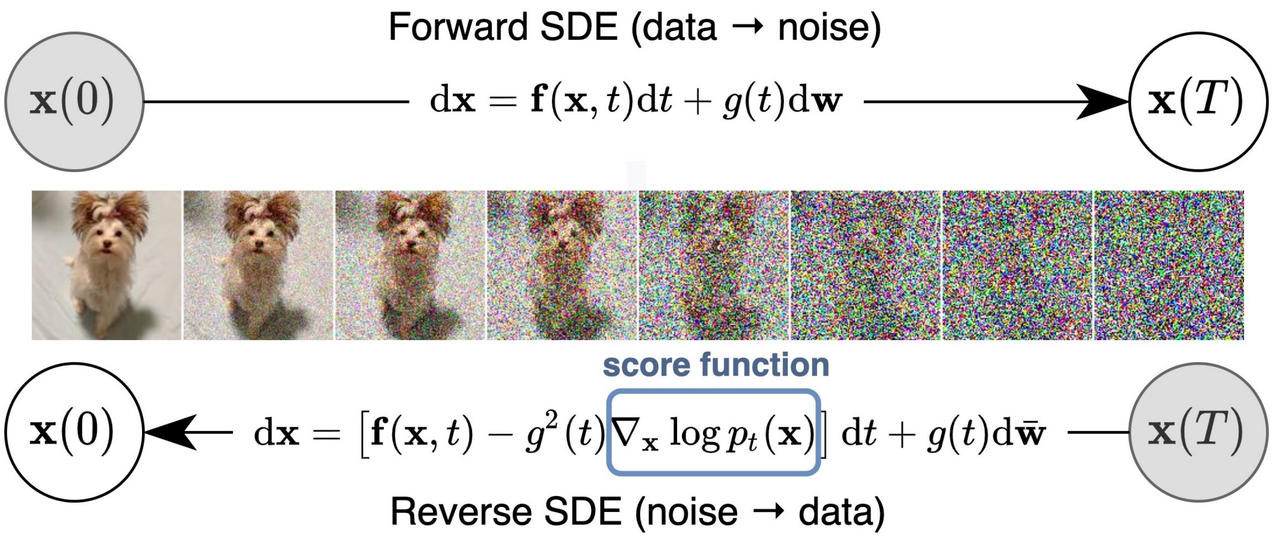
Generated Samples

Generative modeling with Stochastic Differential Equations (SDEs)

Generalize the number of noise scales to infinity and perturb data with an SDE

An SDE with known hyper parameters converts data distribution into a Gaussian noise.

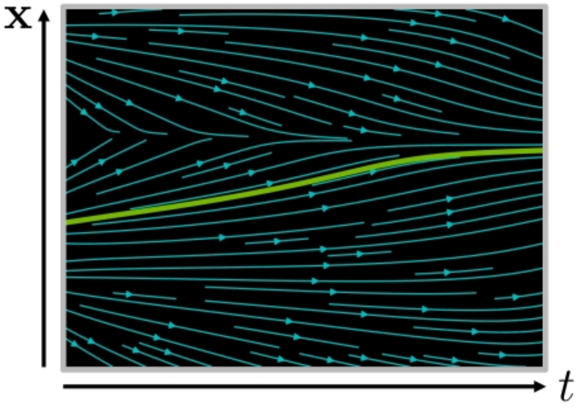
For creating new samples we reverse it with an SDE simmilar to Langevin dynamics.



Differential Equations

Ordinary Differential Equation (ODE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) \quad \text{or} \quad \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t$$



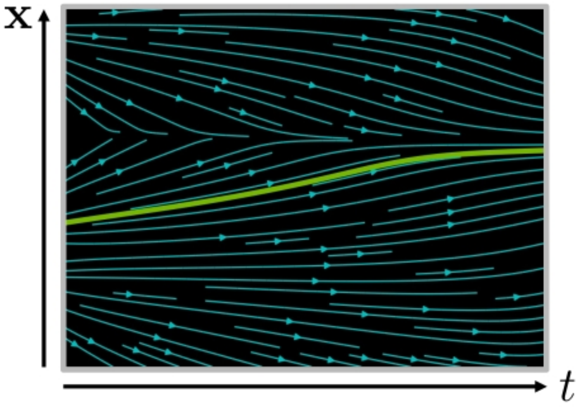
Analytical Solution: $$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{f}(\mathbf{x}, \tau)\mathrm{d}\tau$$

Iterative Numerical Solution: $$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t$$

Differential Equations

Ordinary Differential Equation (ODE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) \quad \text{or} \quad \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t$$



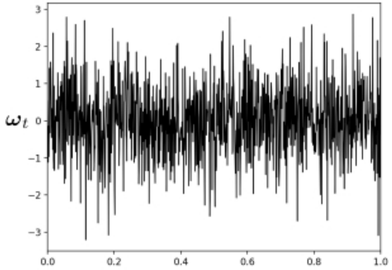
Analytical Solution: $\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{f}(\mathbf{x}, \tau)\mathrm{d}\tau$

Iterative Numerical Solution: $\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t$

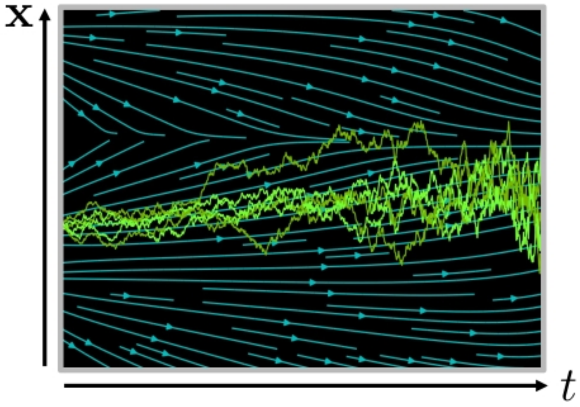
Stochastic Differential Equation (SDE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \underbrace{\mathbf{f}(\mathbf{x}, t)}_{\text{drift coefficient}} + \underbrace{\sigma(\mathbf{x}, t)}_{\text{diffusion coefficient}}\boldsymbol{\omega}_t$$

Wiener Process (Gaussian White Noise)

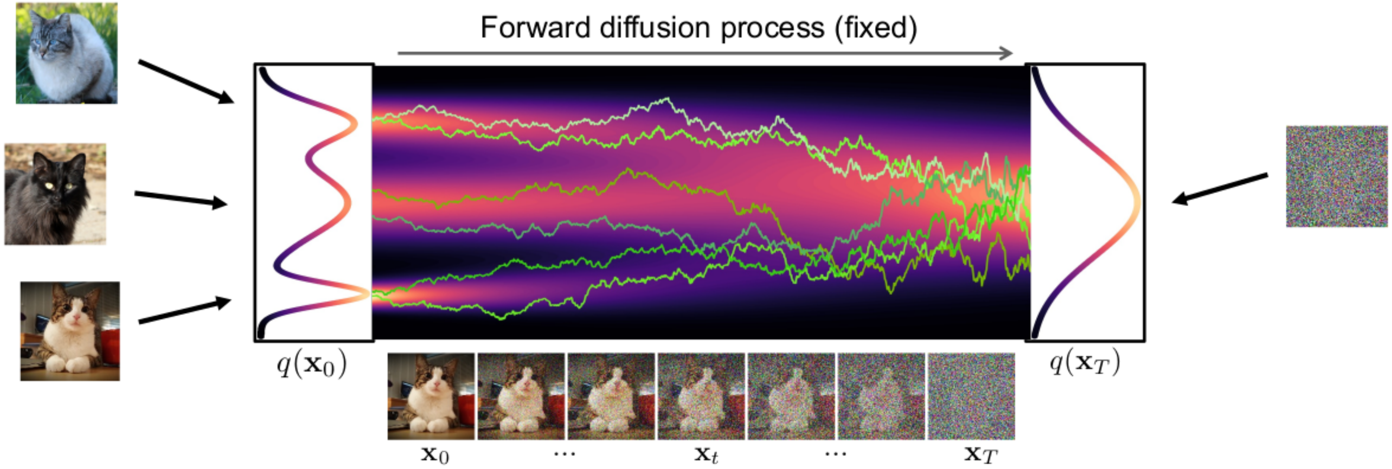


$$\left(\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + \sigma(\mathbf{x}, t)\mathrm{d}\boldsymbol{\omega}_t \right)$$



$$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t + \sigma(\mathbf{x}(t), t)\sqrt{\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

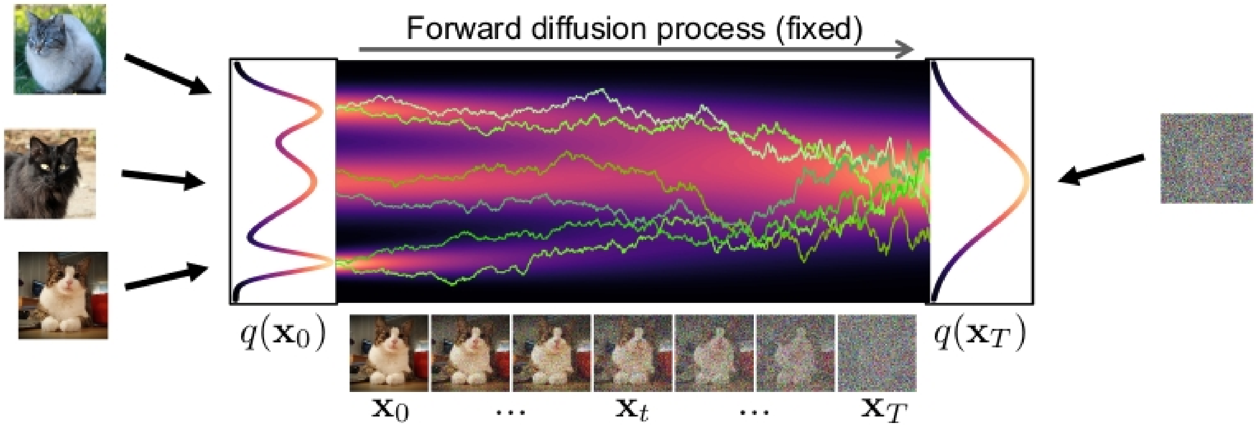
Forward Diffusion with Stochastic Differential Equation



Forward Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \underbrace{-\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t}_{\substack{\text{drift term}\\ \text{(pulls towards mode)}}} + \underbrace{\sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t}_{\substack{\text{diffusion term}\\ \text{(injects noise)}}}$$

The Generative Reverse Stochastic Differential Equation



Forward Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t$$

Reverse Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \overbrace{\left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\underbrace{\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)}_{\text{“Score Function”}}\right]}^{\text{drift term}}\mathrm{d}t + \overbrace{\sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t}^{\text{diffusion term}}$$

Thank you for your attention!