

Iterative α -(de)Blending: a Minimalist Deterministic Diffusion Model

Eric Heitz, Laurent Belcour, Thomas Chambon
SIGGRAPH 2023



Nicolas Bonneel
@nbonneel

You probably have a vague idea of how diffusion models work: progressively add noise to images during training, and learn to remove that noise. To generate a new image, just denoise a "noisy" image. Let's provide some intuition.

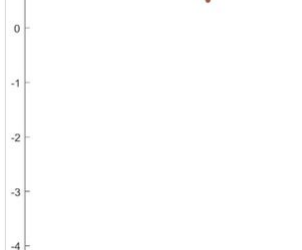
[Traduire le post](#)



Nicolas Bonneel · 20 nov. 2023
First there is the basic idea that for training, we add noise to images. It would be tempting to add noise to each pixel (gaussian) to each pixel to add noise. However, this destroys any signal.

1 6

Nicolas Bonneel · 20 nov. 2023
Indeed, ultimately we will sample from a known centered gaussian, and transform that noise into random values, you do not destroy the signal. Images of 2 grayscale pixels, each plotted as a point in a 2D space.



Nicolas Bonneel @nbonneel · 20 nov. 2023

Given an input image I^0 , without noise, we know how it may look at any time step t without performing all the intermediate iterations: blend the input image with a gaussian noise. Below, the α_t tends to 0, and I^0 ultimately disappears.

$$I^t = \sqrt{\alpha_t} I^0 + \sqrt{1 - \alpha_t} \mathcal{N}(0, 1)$$

$$\prod_{i=1}^t \alpha_i$$

7

Nicolas Bonneel · 20 nov. 2023
The image from our dataset, could have had (one of the) pixels which is a gaussian of mean and variance = 1 - α_t .



Nicolas Bonneel @nbonneel · 20 nov. 2023

Now we can train a neural network to remove noise, e.g., a U-Net. We feed it with a large dataset of images with random amount of noise generated by the previous formula, and try to predict the noise. We minimize a loss that is just the MSE between the input and predicted noise.

Nicolas Bonneel @nbonneel · 20 nov. 2023

So, if you have a noisy image I^{t+1} and want to denoise it to get I^t , just go "a little bit" towards I^0 . Ok, but we do not know I^0 , otherwise the problem would be solved already! The trick is to take the current estimate of I^0 given 2 tweets above (replace $t_{\max}$ with $t+1$).

1 5 1k

Nicolas Bonneel @nbonneel · 20 nov. 2023

Merely replacing I^0 in that formula gives the new iterates below. Given the noise predicted by the neural network ($\tilde{\epsilon}_{t+1}$), for the current image I^{t+1} , we can get a slightly denoised version I^t . Iterating this process yields a clean I^0 from pure noise $I^{t_{\max}}$.

$$I^t = \frac{I^{t+1} - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \tilde{\epsilon}_{t+1}}{\sqrt{\alpha_t}} + \mathcal{N}(0, \sigma)$$

1 7 1k

Nicolas Bonneel @nbonneel · 20 nov. 2023

[note: I did not carefully check the indices in formulas, which may be off by one; though I hope you got the intuition behind them]

1 6 1k

$$\mu = \frac{\sqrt{\alpha_{t+1}} (1 - \alpha_t)}{1 - \alpha_{t+1}} I^{t+1} + \frac{\sqrt{\alpha_t} (1 - \alpha_{t+1})}{1 - \alpha_{t+1}} I^0$$

$$\sigma = \frac{1 - \alpha_t}{1 - \alpha_{t+1}} (1 - \alpha_{t+1})$$



Nicolas Bonneel

In this post, we will see how to generate images with a diffusion model.

First, we will see how to generate images with a diffusion model.

Then, we will see how to generate images with a diffusion model.

Finally, we will see how to generate images with a diffusion model.

Let's start with the first part: how to generate images with a diffusion model.

We will see how to generate images with a diffusion model.

Let's start with the first part: how to generate images with a diffusion model.

Pourquoi une nouvelle théorie ?

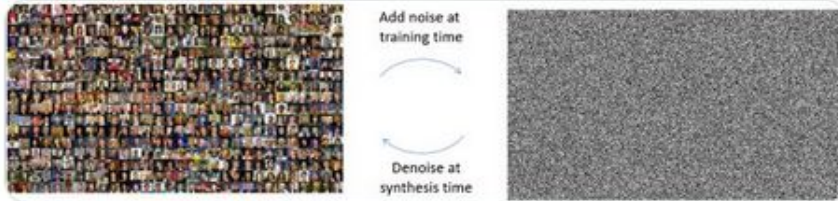


Nicolas Bonneel
@nbonneel

...

You probably have a vague idea of how diffusion models work: progressively add noise to images during training, and learn a model to remove that noise. To generate a new image, just denoise a "pure noise" image. Let's provide some intuition to the formulas involved. ▾

Traduire le post



5:15 PM · 20 nov. 2023 · 59,5 k vues

ABSTRACT

We derive a minimalist but powerful deterministic denoising-diffusion model. While denoising diffusion has shown great success in many domains, its underlying theory remains largely inaccessible to non-expert users. Indeed, an understanding of graduate-level concepts such as Langevin dynamics or score matching appears to be required to grasp how it works. We propose an alternative approach that requires no more than undergrad calculus and probability. We consider two densities and observe what happens when random samples from these densities are blended (linearly interpolated).

We show that iteratively blending and deblending samples produces random paths between the two densities that converge toward a deterministic mapping. This mapping can be evaluated with a neural network trained to deblend samples. We obtain a model that behaves like deterministic denoising diffusion: it iteratively maps samples from one density (e.g., Gaussian noise) to another (e.g., cat images). However, compared to the state-of-the-art alternative, our model is simpler to derive, simpler to implement, more numerically stable, achieves higher quality results in our experiments, and has interesting connections to computer graphics.

Pourquoi une nouvelle théorie ?

Diffusion models: how it works, why it works?

Benjamin Hoover, Nicolas Deschamps, 2023, 11:00:00

Modèles génératifs

Generative models

Generative models

Estimer une densité

1. Estimer la densité de la distribution de données
2. Estimer la densité de la distribution de données
3. Estimer la densité de la distribution de données

MCMC: Dynamique de Langevin

En physique

Equation de Langevin (SDE)

$$m \frac{dv}{dt} = -\gamma v + \sqrt{2\gamma k_B T} \xi(t)$$

1. L'ensemble de données est une suite de points
2. L'ensemble de données est une suite de points

Descente de gradient bruyante

En probabilité

Langevin Monte Carlo Sampling

$$dx_t = -\nabla \log p(x_t) dx + \sqrt{2} \sigma(x_t) dW_t$$

Le Score

En probabilité

Pourquoi le score ?

Equation de Fokker-Planck

$$\frac{\partial p}{\partial t} + \nabla \cdot (p \nabla \log p) = \frac{1}{2} \nabla^2 (p \sigma^2)$$

Pourquoi le score ?

Equation de Fokker-Planck

$$\frac{\partial p}{\partial t} + \nabla \cdot (p \nabla \log p) = \frac{1}{2} \nabla^2 (p \sigma^2)$$

Garanties théoriques de LMC

1. L'ensemble de données est une suite de points
2. L'ensemble de données est une suite de points

OU PROCESS

Processus d'Ornstein-Uhlenbeck (simplifié)

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Processus d'Ornstein-Uhlenbeck

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Processus d'Ornstein-Uhlenbeck

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Processus d'Ornstein-Uhlenbeck

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Peut-on faire l'inverse ?

Time-reversal formula

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Time-reversal formula

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Time-reversal formula

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Time-reversal formula

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Time-reversal formula

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

Problème résolu ?

Time-reversal formula

$$dx_t = -x_t dt + \sqrt{2} \sigma dt$$

APPRENDRE LE SCORE

Les réseaux de neurones

Score matching loss

Score matching loss

$$\mathbb{E} \left[\left\| \nabla \log p(x_t) - \nabla \log q(x_t) \right\|^2 \right]$$

Score matching loss

Score matching loss

$$\mathbb{E} \left[\left\| \nabla \log p(x_t) - \nabla \log q(x_t) \right\|^2 \right]$$

Score matching loss

Score matching loss

$$\mathbb{E} \left[\left\| \nabla \log p(x_t) - \nabla \log q(x_t) \right\|^2 \right]$$

Convergence des réseaux de diffusion

Convergence

$$\mathbb{E} \left[\left\| \nabla \log p(x_t) - \nabla \log q(x_t) \right\|^2 \right]$$

EN PRATIQUE

Forward process

Forward process

$$dx_t = x_t dt + \sqrt{2} \sigma dt$$

Training process

Training process

$$dx_t = x_t dt + \sqrt{2} \sigma dt$$

Le réseau utilisé

Sampling process

Sampling process

$$dx_t = x_t dt + \sqrt{2} \sigma dt$$

Sampling process

Sampling process

$$dx_t = x_t dt + \sqrt{2} \sigma dt$$

DIVERSES CHOSES

Conditionnement et espace latent

Respecter des contraintes géométriques

Respecter des contraintes géométriques

$$dx_t = x_t dt + \sqrt{2} \sigma dt$$

Plus rapide ???

Plus précis ?

Autres processus de diffusion

FIN

En fait, on peut faire plus simple



Nicolas Bonneel @nbonneel · 20 nov. 2023

Instead, to "transform" the "realistic images" manifold into a gaussian distribution, one needs to progressively blend each input image with a gaussian. The resulting iterate is just a linear combination of the image at the previous time step and a gaussian random value.

$$I^{t+1} = \sqrt{\alpha_t} I^t + \sqrt{1 - \alpha_t} \varepsilon_{t \rightarrow t+1}$$

with $\varepsilon_{t \rightarrow t+1} \sim \mathcal{N}(0,1)$

α_t some parameter (could be constant)



ABSTRACT

We derive a minimalist but powerful deterministic denoising-diffusion model. While denoising diffusion has shown great success in many domains, its underlying theory remains largely inaccessible to non-expert users. Indeed, an understanding of graduate-level concepts such as Langevin dynamics or score matching appears to be required to grasp how it works. We propose an alternative approach that requires no more than undergrad calculus and probability. We consider two densities and observe what happens when random samples from these densities are blended (linearly interpolated). We show that iteratively blending and deblending samples produces random paths between the two densities that converge toward a deterministic mapping. This mapping can be evaluated with a neural network trained to deblend samples. We obtain a model that behaves like deterministic denoising diffusion: it iteratively maps samples from one density (e.g., Gaussian noise) to another (e.g., cat images). However, compared to the state-of-the-art alternative, our model is simpler to derive, simpler to implement, more numerically stable, achieves higher quality results in our experiments, and has interesting connections to computer graphics.

Overview

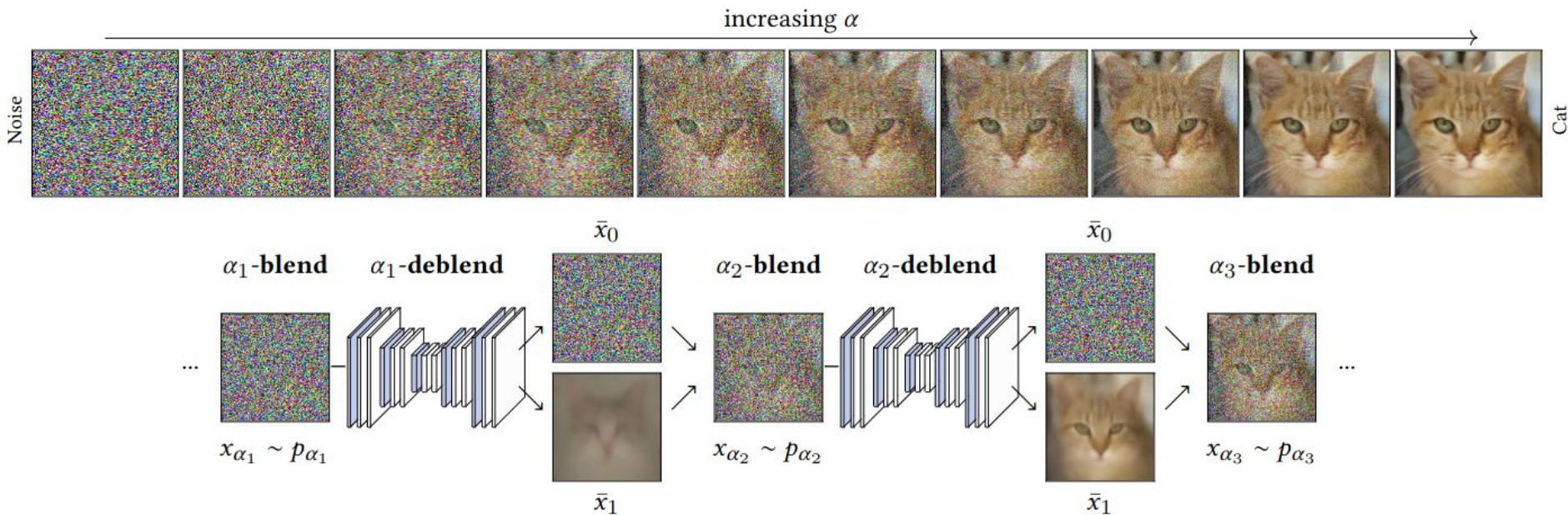


Fig. 1. **Iterative α -blending and deblending.** In this example, we map Gaussian noise to cat images. We use a neural network trained to deblend blended Gaussian noise and cats. By deblending and reblending iteratively, we obtain a mapping between the Gaussian and cat densities.

Blending and Deblending

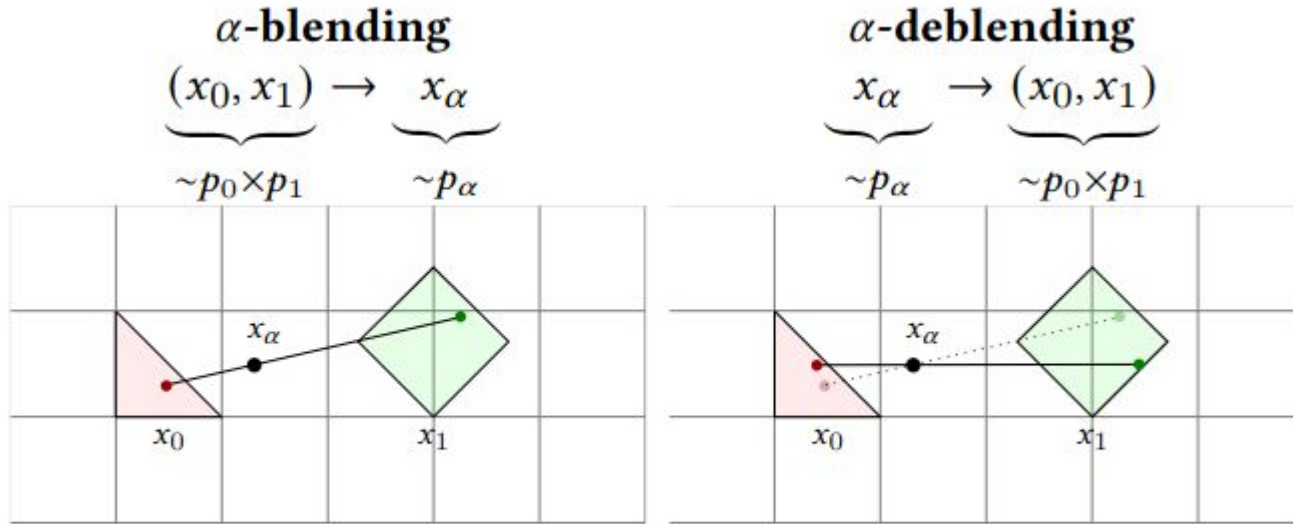


Fig. 2. Blending and deblending as sampling operations.

Iterative α -(de)blending

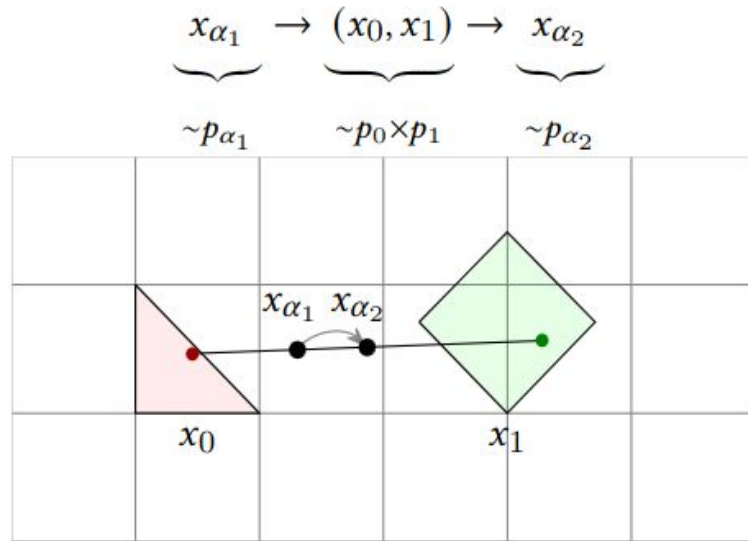


Fig. 4. α -(de)blending.

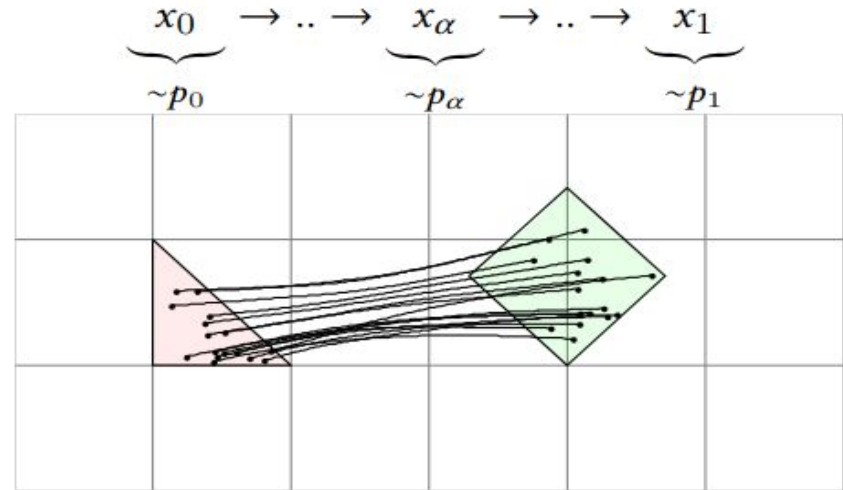


Fig. 5. **Iterative α -(de)blending**

Comment deblend : le learning

(a) learn $\bar{x}_0$ and $\bar{x}_1$	(b) learn only $\bar{x}_0$	(c) learn only $\bar{x}_1$	(d) learn $\bar{x}_1 - \bar{x}_0$
$(\bar{x}_0, \bar{x}_1) = D_\theta (x_{\alpha_t}, \alpha_t)$	$\bar{x}_0 = D_\theta (x_{\alpha_t}, \alpha_t)$	$\bar{x}_1 = D_\theta (x_{\alpha_t}, \alpha_t)$	$\bar{x}_1 - \bar{x}_0 = D_\theta (x_{\alpha_t}, \alpha_t)$
$x_{\alpha_{t+1}} =$ $(1 - \alpha_{t+1}) \bar{x}_0 + \alpha_{t+1} \bar{x}_1$	$x_{\alpha_{t+1}} = \bar{x}_0 +$ $\frac{\alpha_{t+1}}{\alpha_t} (x_{\alpha_t} - \bar{x}_0)$	$x_{\alpha_{t+1}} = \bar{x}_1 +$ $\frac{(1-\alpha_{t+1})}{(1-\alpha_t)} (x_{\alpha_t} - \bar{x}_1)$	$x_{\alpha_{t+1}} = x_{\alpha_t} +$ $(\alpha_{t+1} - \alpha_t) (\bar{x}_1 - \bar{x}_0)$
unstable	unstable when $\alpha_t \rightarrow 0$	unstable when $\alpha_t \rightarrow 1$	stable

Table 1. **Variant formulations of iterative α -(de)blending** (equivalent in theory but not in practice).

Comment deblend : le learningd" ?

Blending:

$$x_{\alpha_t} = (1 - \alpha_t)x_0 + \alpha_t x_1$$

$$x_{\alpha_{t+1}} = (1 - \alpha_{t+1})x_0 + \alpha_{t+1}x_1$$

By subtracting :

$$x_{\alpha_t} - x_{\alpha_{t+1}} = (\alpha_t - \alpha_{t+1})(x_1 - x_0)$$

Therefore:

$$x_{\alpha_{t+1}} = x_{\alpha_t} - (\alpha_t - \alpha_{t+1})(x_1 - x_0)$$

Learning and Sampling

Algorithm 3 Training

Require: $x_0 \sim p_0, x_1 \sim p_1, \alpha \sim \mathcal{U}_{[0,1]}$

$$x_\alpha = (1 - \alpha) x_0 + \alpha x_1$$

$$l = \|D_\theta(x_\alpha, \alpha) - (x_1 - x_0)\|^2$$

backprop from l and update θ

Algorithm 4 Sampling

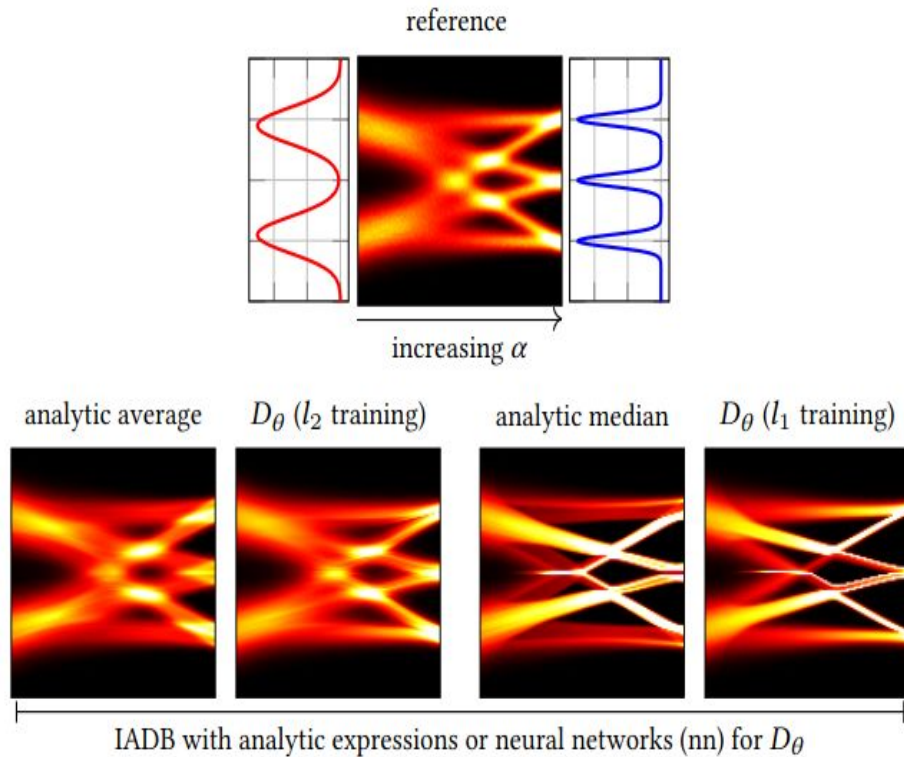
Require: $x_0 \sim p_0, T, \alpha_t := \frac{t}{T}$

for $t = 0, \dots, T - 1$ **do**

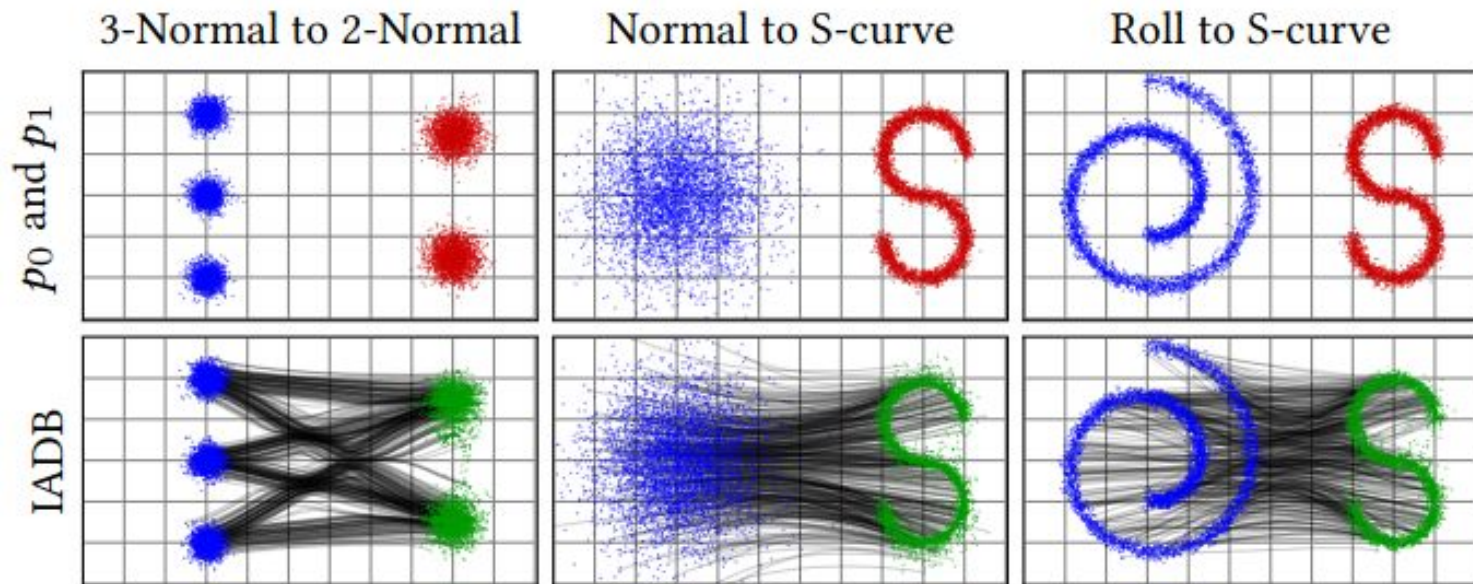
$$x_{\alpha_{t+1}} = x_{\alpha_t} + (\alpha_{t+1} - \alpha_t) D_\theta(x_{\alpha_t}, \alpha_t)$$

end for

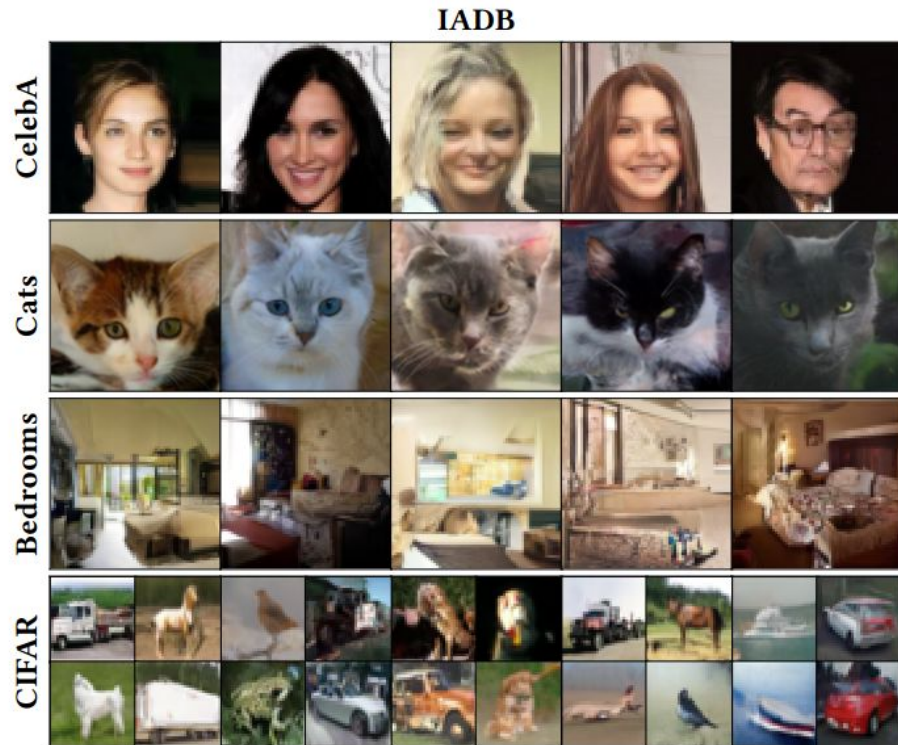
Results



Results



Results et comparaisons

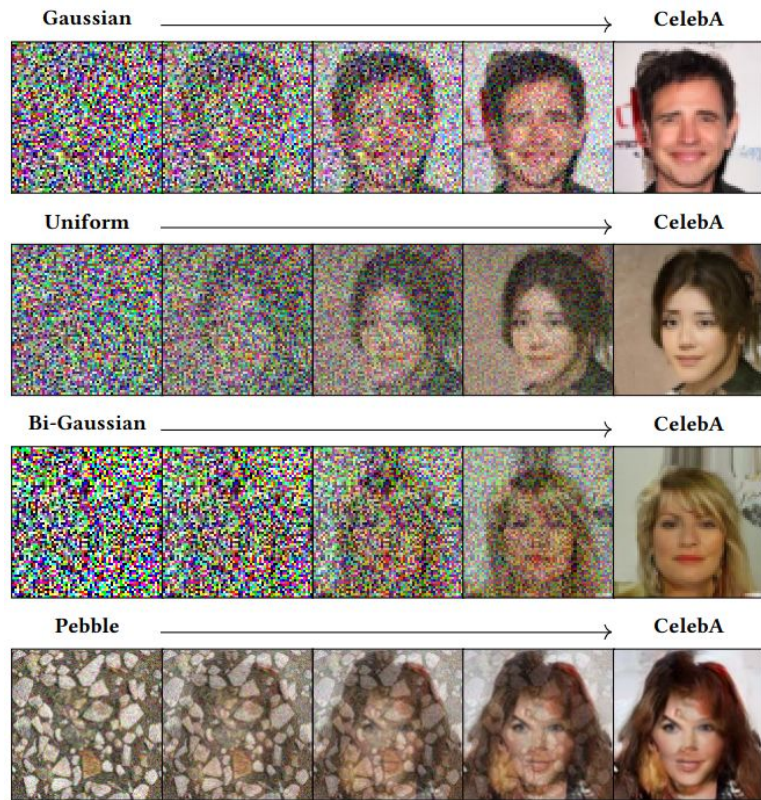


Results et comparaisons

		$T = 2$	$T = 4$	$T = 8$	$T = 16$	$T = 32$	$T = 128$
Cats	DDIM	200.89	39.32	22.98	18.12	13.66	9.26
	IADB	86.68	39.40	17.50	11.15	8.06	6.69
CelebA	DDIM	177.86	46.58	19.50	11.91	9.23	6.93
	IADB	108.13	51.79	22.68	12.15	7.52	5.56
Bedrooms	DDIM	307.76	104.55	38.58	24.26	20.34	16.82
	IADB	238.45	57.60	18.55	14.12	14.57	15.93
CIFAR	DDIM	174.54	58.53	20.92	11.01	7.74	6.20
	IADB	157.32	50.69	16.74	8.73	6.45	5.61

Fig. 12. **Comparing IADB and DDIM.** We use the same Gaussian noise to sample images with IADB and DDIM. We obtain very close images because the underlying theoretical mappings are the same. IADB achieves better FID scores w.r.t. the number of steps T than DDIM most of the time.

Results et comparaisons



FIN