SBM 2: Score-based Models

CS 598: Deep Generative and Dynamical Models

Instructor: Arindam Banerjee

November 16, 2021

Diffusion based Probabilistic Models

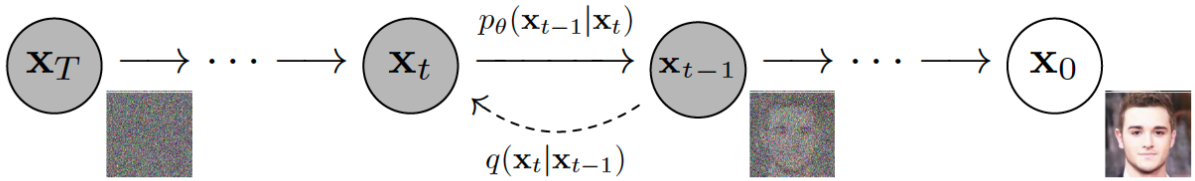


Figure 2: The directed graphical model considered in this work.

- Reverse process starts from x_T (noise)

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

  - Generative model $p_\theta(x_0) = \int_{x_{1:T}} p_\theta(x_{0:t}) dx_{1:T}$
- Forward process starts from x_0 (signal)

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_{1:T}|x_0)$$

Forward and Reverse Processes

- Reverse process is the generative model

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \quad p_\theta(x_{t+1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

- Forward process is the diffusion process, note $\beta_t < 1, \forall t$

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_{1:T}|x_0), \quad q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbb{I})$$

Variational Inference with Forward Process

- Recall the generative model: $p_\theta(x_0) = \int_{x_{1:T}} p_\theta(x_{0:T}) dx_{1:T}$
- Variational inference using the forward process q

$$-\log p_\theta(x_0) = \mathbb{E}_q \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] = \mathbb{E}_q \left[-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \right] =: L$$

- Inference with the forward process: Easy to sample from at any t

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\tilde{\alpha}_t} x_0, (1 - \tilde{\alpha}_t)\mathbb{I})$$

where

$$\alpha_t := 1 - \beta_t \ , \qquad \tilde{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

Optimizing the Variational Objective

- Variational objective can be written as

$$\mathbb{E}_q \left[\underbrace{D_{KL}(q(x_T|x_0)\|p(x_T))}_{L_T} + \sum_{t>1} \underbrace{D_{KL}(q(x_{t-1}|x_t,x_0)\|p_\theta(x_{t-1}\|x_t))}_{L_{t-1}} \underbrace{-\log p_\theta(x_0|x_1)}_{L_0} \right]$$

- First terms in the KL-divergences are conditional probabilities of the forward process

$$q(x_{t-1}|x_t,x_0) = \mathcal{N}(x_{t-1};\tilde{\mu}_t(x_t,x_0),\tilde{\beta}_t\mathbb{I})$$

where

$$\tilde{\mu}_t(x_t,x_0) := \frac{\sqrt{\tilde{\alpha}_{t-1}}\beta_t}{1-\tilde{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\tilde{\alpha}_{t-1})}{1-\tilde{\alpha}_t}x_t\ , \quad \tilde{\beta}_t := \frac{1-\tilde{\alpha}_{t-1}}{1-\tilde{\alpha}_t}\beta_t$$

Representation: Forward and Reverse Processes

- Forward process: Choose β_t to be constants, nothing to learn
  - Can be made learnable, gradient descent using reparameterization
- Reverse process: Choose $x_0 \sim \mathcal{N}(0, \mathbb{I})$ or a fixed point
  - Set $\Sigma_\theta(x_t, t) = \sigma_t^2 \mathbb{I}$ with $\sigma_t^2 = \beta_t$ or $\tilde{\beta}_t$
  - KL-divergence between Gaussians is (scaled) square loss
  - Choose $\mu_\theta(x_t, t)$ to minimize

$$L_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \left\| \tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t) \right\|_2^2 \right] + C$$

- The expression can be simplified by recalling

$$x_t(x_0, \epsilon) = \sqrt{\tilde{\alpha}_t} x_0 + \sqrt{1 - \tilde{\alpha}_t} \epsilon$$

and the form for $\tilde{\mu}_t(x_t, x_0)$

Learning the Reverse Process

- Plugging in $x_t(x_0, \epsilon)$ and $\tilde{\mu}_t(x_t, x_0)$

$$L_{t-1} = \mathbb{E}_{x_0,\epsilon}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\tilde{\alpha}_t}}\epsilon\right) - \mu_\theta(x_t(x_0,\epsilon), t)\right\|^2\right] + C$$

- With x_t denoting $x_t(x_0, \epsilon)$, choose

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\tilde{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$$

- Sampling $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$ is essentially

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\tilde{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z\ , \quad z \sim \mathcal{N}(0, \mathbb{I})$$

Diffusion Models and Denoising Score Matching

- Plugging in the form for $\mu_\theta(x_t, t)$, the objective is

$$L_{t-1} = \mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\tilde{\alpha}_t)}\left\|\epsilon - \epsilon_\theta(\sqrt{\tilde{\alpha}_t}x_0 + \sqrt{1-\tilde{\alpha}_t}\epsilon, t)\right\|^2\right] + C$$

- Resembles denoising score matching at multiple scales t
- Scaling of terms across t is inversely proportional to variance σ_t^2
- Training of $\epsilon_\theta(\cdot, t)$ at all scales t simultaneously

Algorithms: Training and Sampling

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
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
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** $\mathbf{x}_0$

Results: Log-likelihood

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	**10.06**	**2.67**	
Unconditional			
Diffusion (original) [53]			≤ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			**2.80**
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87 ± 0.12	25.32	
SNGAN [39]	8.22 ± 0.05	21.7	
SNGAN-DDLS [4]	9.09 ± 0.10	15.42	
StyleGAN2 + ADA (v1) [29]	$\mathbf{9.74} \pm 0.05$	3.26	
Ours (L, fixed isotropic $\boldsymbol{\Sigma}$)	7.67 ± 0.13	13.51	≤ 3.70 (3.69)
Ours (L_{simple})	9.46 ± 0.11	**3.17**	≤ 3.75 (3.72)

Results: Training Objective, Ablation

Table 2: Unconditional CIFAR10 reverse process parameterization and training objective ablation. Blank entries were unstable to train and generated poor samples with out-of-range scores.

Objective	IS	FID
$\tilde{\mu}$ prediction (baseline)		
L, learned diagonal Σ	7.28±0.10	23.69
L, fixed isotropic Σ	8.06±0.09	13.22
$\|\tilde{\mu} - \tilde{\mu}_\theta\|^2$	–	–
ϵ prediction (ours)		
L, learned diagonal Σ	–	–
L, fixed isotropic Σ	7.67±0.13	13.51
$\|\tilde{\epsilon} - \epsilon_\theta\|^2$ (L_{simple})	**9.46±0.11**	**3.17**

Results: Samples (CelebA, CIFAR10)

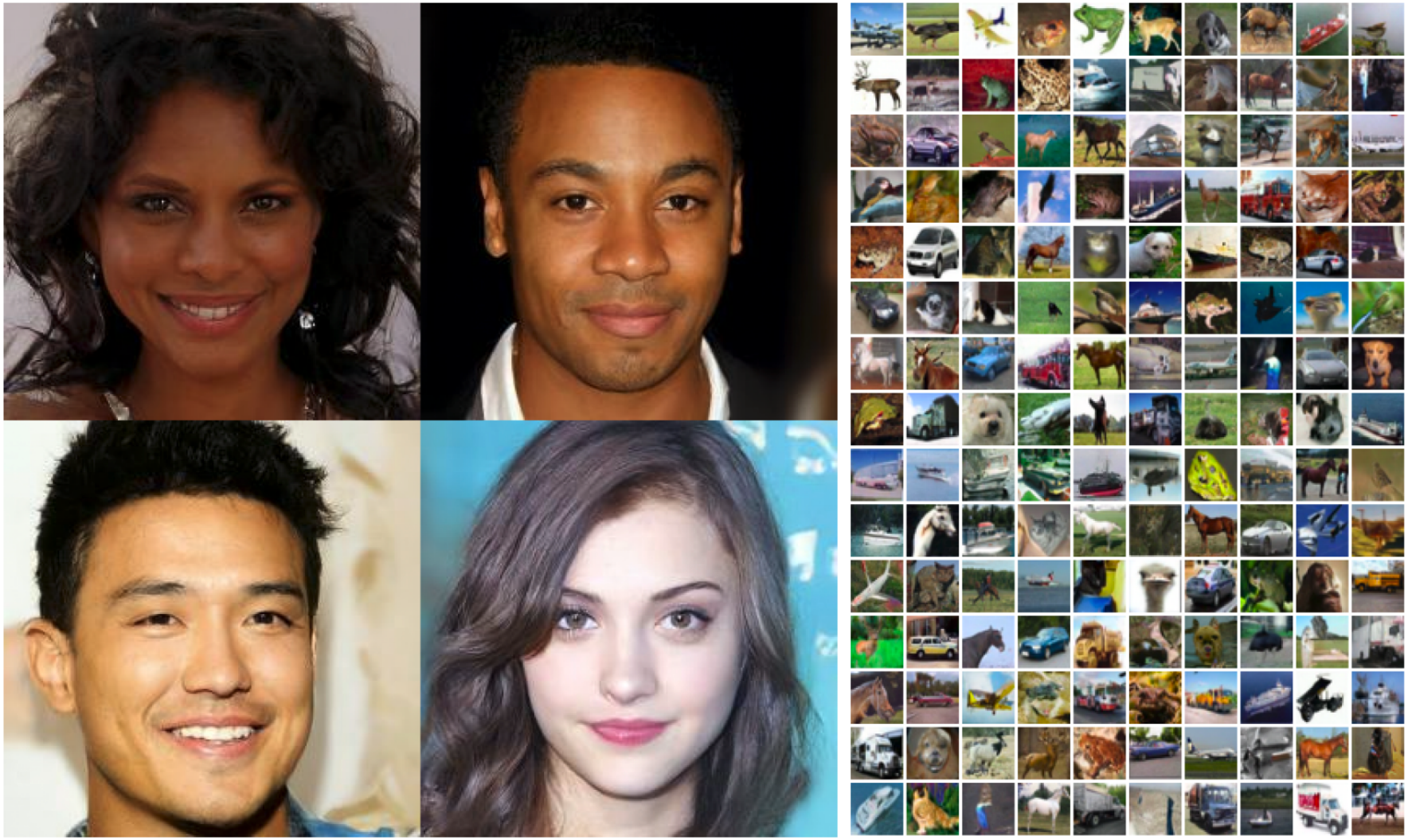


Figure 1: Generated samples on CelebA-HQ 256×256 (left) and unconditional CIFAR10 (right)

Results: Samples (LSUN)

Figure 3: LSUN Church samples. FID=7.89

Figure 4: LSUN Bedroom samples. FID=4.90

Results: Rate Distortion

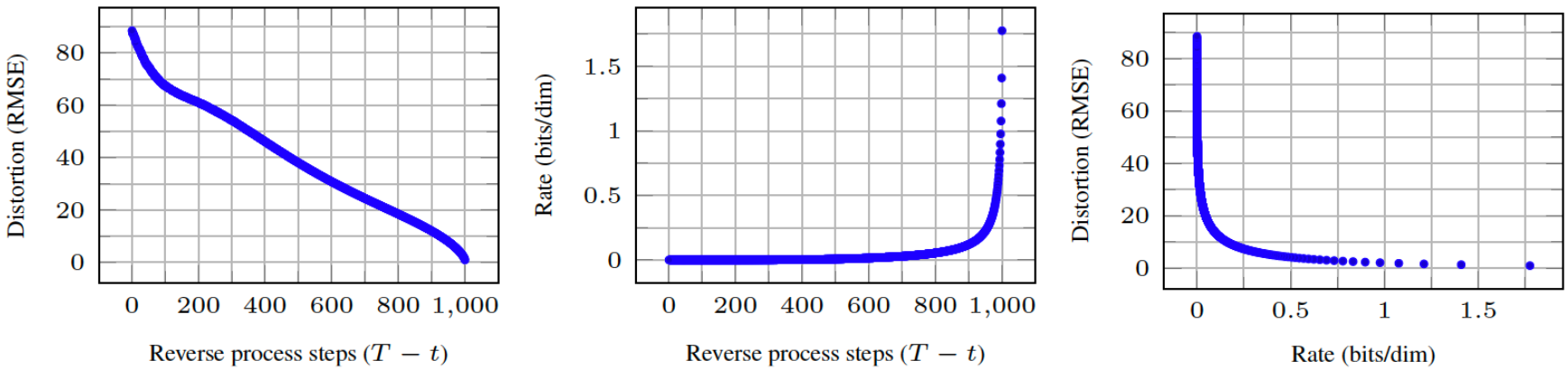


Figure 5: Unconditional CIFAR10 test set rate-distortion vs. time. Distortion is measured in root mean squared error on a $[0, 255]$ scale. See Table 4 for details.

Results: Progressive Generation

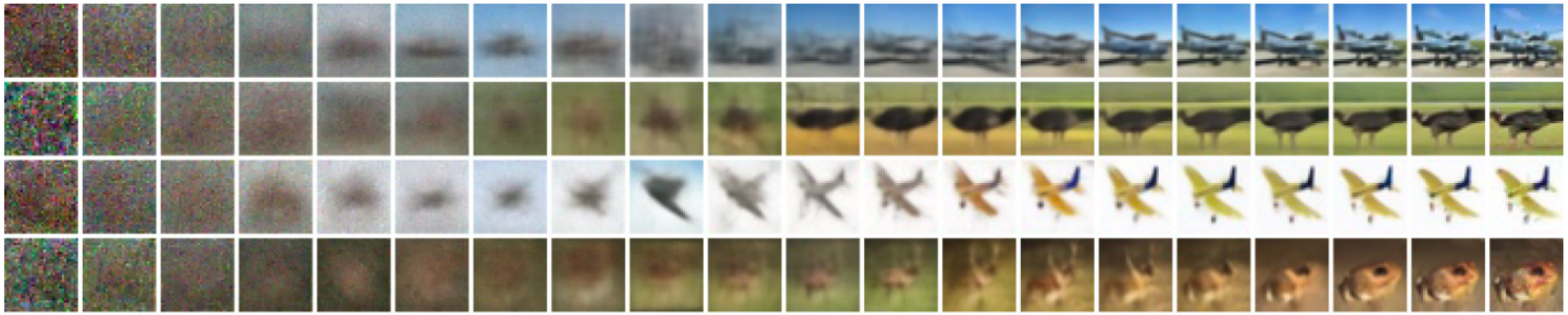


Figure 6: Unconditional CIFAR10 progressive generation ($\hat{\mathbf{x}}_0$ over time, from left to right). Extended samples and sample quality metrics over time in the appendix (Figs. 10 and 14).

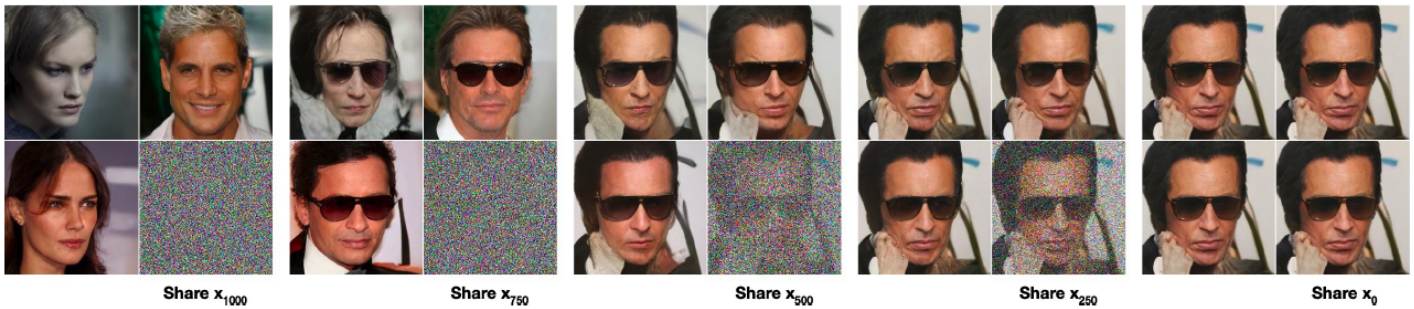
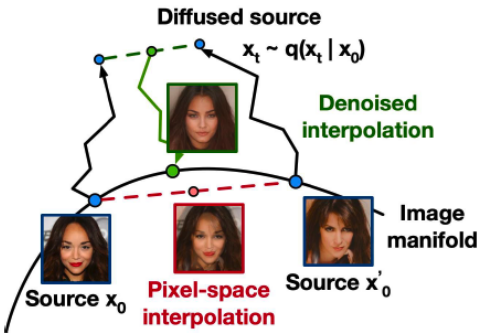


Figure 7: When conditioned on the same latent, CelebA-HQ 256×256 samples share high-level attributes. Bottom-right quadrants are $\mathbf{x}_t$, and other quadrants are samples from $p_\theta(\mathbf{x}_0|\mathbf{x}_t)$.

Results: Interpolation



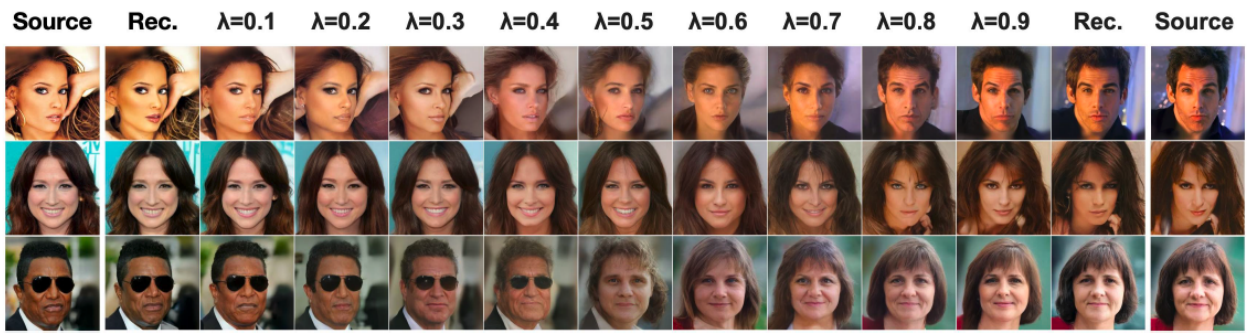


Figure 8: Interpolations of CelebA-HQ 256x256 images with 500 timesteps of diffusion.

Recap: Denoising Score Matching

- Consider data-conditioned noisy $p_\sigma(\tilde{x}|x) := \mathcal{N}(\tilde{x}; x, \sigma^2 \mathbb{I})$
  - Marginal $p_\sigma(\tilde{x}) = \int_x p(x) p(\tilde{x}|x) dx$
  - Sequence $\sigma_1 < \sigma_2 < \cdots < \sigma_N$
- Weighted sum of denoising score matching

$$\theta^* = \text{argmin}_\theta \sum_{i=1}^{N} \sigma_i^2 \mathbb{E}_{p(x)} \mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \left[\| s_\theta(\tilde{x}, \sigma_i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x}|x) \|_2^2 \right]$$

- Learned score function $s_{\theta^*}(x, \sigma)$ matches $\nabla_x \log p_\sigma(x)$
- Sampling based on Langevin dynamics, with $z_i^t \sim \mathcal{N}(0, \mathbb{I})$

$$x_i^t = x_i^{t-1} + \epsilon_i s_{\theta^*}(x_i^{t-1}, \sigma_i) + \sqrt{2\epsilon_i} z_i^t$$

Recap: Denoising Diffusion Probabilistic Models

- Choose a sequence of positive noise scales $0 < \beta_1, \ldots, \beta_N < 1$
- Forward process $x_0 \sim p(x), p(x_i|x_{i-1}) = \mathcal{N}(x_i; \sqrt{1-\beta_i}x_{i-1}, \beta_i \mathbb{I})$

$$p_{\alpha_i}(x_i|x_0) = \mathcal{N}(x_i; \sqrt{\alpha_i}x_0, (1-\alpha_i)\mathbb{I}) \ , \quad \alpha_i = \prod_{j=1}^{i}(1-\beta_j)$$

  - Marginal $p_{\alpha_i}(x) = \int p(x)p_{\alpha_i}(\tilde{x}|x)dx$
- Forward process does variational inference on generative model

$$p_\theta(x_{i-1}|x_i) = \mathcal{N}\left(x_{i-1}; \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i, i)), \beta_i \mathbb{I}\right)$$

- Training based on reweighted ELBO

$$\theta^* = \text{argmin}_\theta \sum_{i=1}^{n}(1-\alpha_i)\mathbb{E}_{p(x)}\mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)}\left[\|s_\theta(\tilde{x}, \sigma_i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x}|x)\|_2^2\right]$$

- Ancestral sampling using the reverse process

$$x_i^{t-1} = x_i^t + \epsilon_i s_{\theta^*}(x_i^t, \sigma_i) + \sqrt{2\epsilon_i}z_i^t$$

SBMs with Stochastic Differential Equations (SDEs)

- Instead of discrete steps, make the processes continuous
  - Forward process: data $\mathrm{x}_0 \sim p(\mathrm{x})$ to prior x_T
  - Reverse process is the generative model: $x_T \sim p_T(\mathrm{x})$ to data x_0
- Forward process is a SDE with drift-diffusion

$$d\mathrm{x} = f(\mathrm{x}, t)dt + g(t)d\mathrm{w}$$

  - $f(\mathrm{x}, t)$ is the drift, $g(t)$ determines variance of the Wiener process w
- Samples can be generated by reversing the SDE

$$d\mathrm{x} = [f(\mathrm{x}, t) - g(t)^2 \nabla_{\mathrm{x}} \log p_t(\mathrm{x})]dt + g(t)d\bar{\mathrm{w}}$$

SDE based Modeling

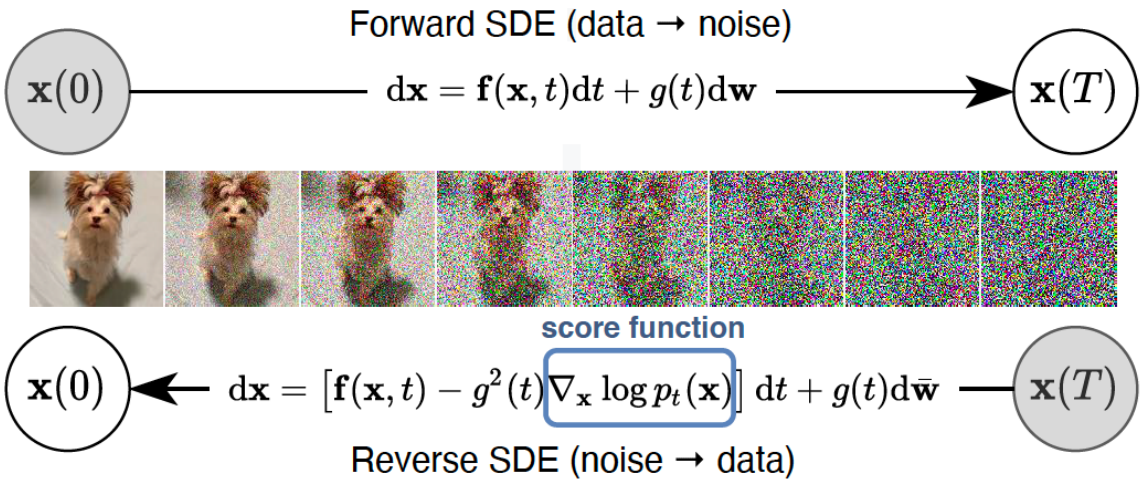


Figure 1: **Solving a reverse-time SDE yields a score-based generative model.** Transforming data to a simple noise distribution can be accomplished with a continuous-time SDE. This SDE can be reversed if we know the score of the distribution at each intermediate time step, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$.

Forward and Reverse SDE

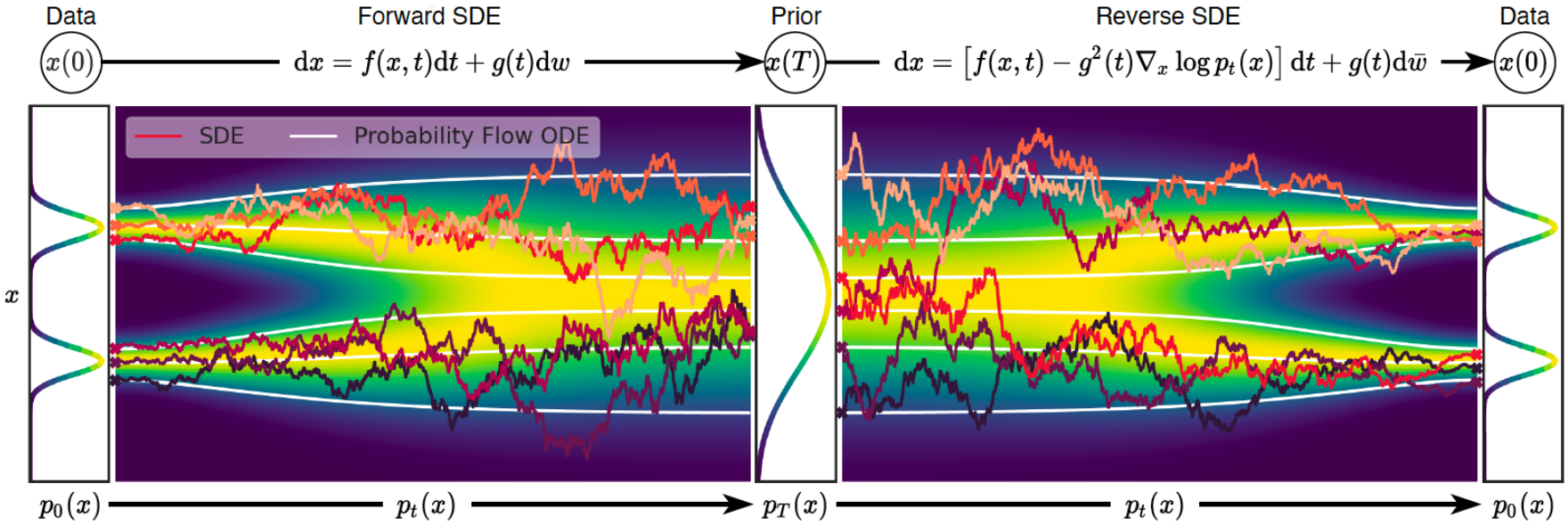


Figure 2: **Overview of score-based generative modeling through SDEs.** We can map data to a noise distribution (the prior) with an SDE (Section 3.1), and reverse this SDE for generative modeling (Section 3.2). We can also reverse the associated probability flow ODE (Section 4.3), which yields a deterministic process that samples from the same distribution as the SDE. Both the reverse-time SDE and probability flow ODE can be obtained by estimating the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ (Section 3.3).

Estimating Scores for the SDE

- Train a score based model for all scales

$$\theta^* = \text{argmin}_\theta \, \mathbb{E}_t \left[\lambda(t) \mathbb{E}_{\mathrm{x}(0)} \mathbb{E}_{\mathrm{x}(t)|\mathrm{x}(0)} \left[\left\| s_\theta(\mathrm{x}(t), t) - \nabla_{\mathrm{x}(t)} \log p_{0t}(\mathrm{x}(t)|\mathrm{x}(0)) \right\|_2^2 \right] \right]$$

- $\lambda : [0, T] \mapsto \mathbb{R}_{++}$ is a positive weighting function
  - Typically choose $\lambda(t) \propto 1.\mathbb{E}[\|\nabla_{\mathrm{x}(t)} \log p_{0t}(\mathrm{x}(t)|\mathrm{x}(0))\|^2]$
- Need to know the transition kernel $p_{0t}(\mathrm{x}(t)|\mathrm{x}(0))$
  - Affine drift $f(\mathrm{x}, t)$ keeps the kernel Gaussian
  - In general, solve Kolmogorov forward equation

Example SDEs

- Variance exploding (VE) SDE, variance 'explodes' as $t \to \infty$

$$dx = \sqrt{\frac{d[\sigma^2(t)]}{dt}} dw$$

- Variance Preserving (VP) SDE, process with fixed variance

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dw$$

- Sub-VP SDE, variance bounded by VP SDE

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)ds})}dw$$

Reverse SDE Solvers

- General purpose numerical SDE solvers
- Predictor-Corrector samplers
  - Score based MCMC, e.g., Langevin MCMC, or Hamiltonian MC
  - Predictor: Numerical SDE first gives a numerical estimate
  - Corrector: Score-based MCMC corrects the marginal distribution
- Author's claim: Deterministic process with same marginal as SDE

$$d\mathrm{x} = \left[f(\mathrm{x}, t) - \frac{1}{2} g^2(t) \nabla_x \log p_t(\mathrm{x}) \right] dt$$

  - Called "probability flow"
  - Score modeled by neural model, example of neural ODE
  - Allows exact likelihood computation, embedding

Results: Reverse SDE Solver

Table 1: Comparing different reverse-time SDE solvers on CIFAR-10. Shaded regions are obtained with the same computation (number of score function evaluations). Mean and standard deviation are reported over five sampling runs. "P1000" or "P2000": predictor-only samplers using 1000 or 2000 steps. "C2000": corrector-only samplers using 2000 steps. "PC1000": Predictor-Corrector (PC) samplers using 1000 predictor and 1000 corrector steps.

	Variance Exploding SDE (SMLD)				Variance Preserving SDE (DDPM)			
FID↓ Sampler / Predictor	P1000	P2000	C2000	PC1000	P1000	P2000	C2000	PC1000
ancestral sampling	4.98 ± .06	4.88 ± .06		**3.62 ± .03**	3.24 ± .02	3.24 ± .02		**3.21 ± .02**
reverse diffusion	4.79 ± .07	4.74 ± .08	20.43 ± .07	**3.60 ± .02**	3.21 ± .02	3.19 ± .02	19.06 ± .06	**3.18 ± .01**
probability flow	15.41 ± .15	10.54 ± .08		**3.51 ± .04**	3.59 ± .04	3.23 ± .03		**3.06 ± .03**

Results: NLL and FID on CIFAR-10

Table 2: NLLs and FIDs (ODE) on CIFAR-10.

Model	NLL Test ↓	FID ↓
RealNVP (Dinh et al., 2016)	3.49	-
iResNet (Behrmann et al., 2019)	3.45	-
Glow (Kingma & Dhariwal, 2018)	3.35	-
MintNet (Song et al., 2019b)	3.32	-
Residual Flow (Chen et al., 2019)	3.28	46.37
FFJORD (Grathwohl et al., 2018)	3.40	-
Flow++ (Ho et al., 2019)	3.29	-
DDPM (L) (Ho et al., 2020)	$\leqslant 3.70^{*}$	13.51
DDPM (L_{simple}) (Ho et al., 2020)	$\leqslant 3.75^{*}$	3.17
DDPM	3.28	3.37
DDPM cont. (VP)	3.21	3.69
DDPM cont. (sub-VP)	3.05	3.56
DDPM++ cont. (VP)	3.16	3.93
DDPM++ cont. (sub-VP)	3.02	3.16
DDPM++ cont. (deep, VP)	3.13	3.08
DDPM++ cont. (deep, sub-VP)	**2.99**	**2.92**

Results: FID and IS in CIFAR-10

Table 3: CIFAR-10 sample quality.

Model	FID↓	IS↑
Conditional		
BigGAN (Brock et al., 2018)	14.73	9.22
StyleGAN2-ADA (Karras et al., 2020a)	**2.42**	**10.14**
Unconditional		
StyleGAN2-ADA (Karras et al., 2020a)	2.92	9.83
NCSN (Song & Ermon, 2019)	25.32	$8.87 \pm .12$
NCSNv2 (Song & Ermon, 2020)	10.87	$8.40 \pm .07$
DDPM (Ho et al., 2020)	3.17	$9.46 \pm .11$
DDPM++	2.78	9.64
DDPM++ cont. (VP)	2.55	9.58
DDPM++ cont. (sub-VP)	2.61	9.56
DDPM++ cont. (deep, VP)	2.41	9.68
DDPM++ cont. (deep, sub-VP)	2.41	9.57
NCSN++	2.45	9.73
NCSN++ cont. (VE)	2.38	9.83
NCSN++ cont. (deep, VE)	**2.20**	**9.89**

Results: Adaptive Sampling

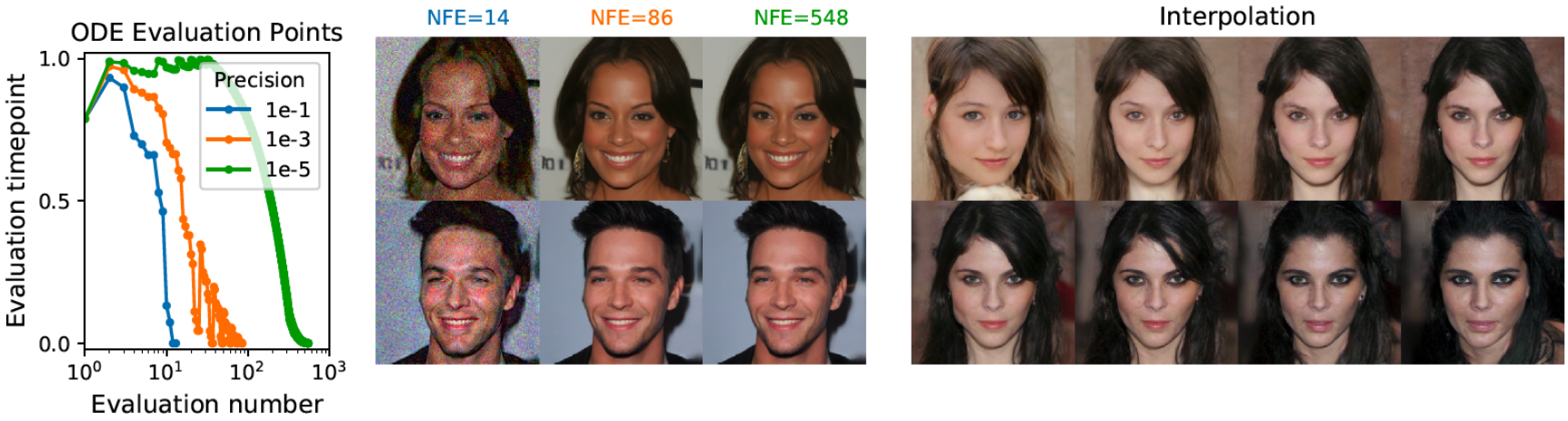


Figure 3: **Probability flow ODE enables fast sampling** with adaptive stepsizes as the numerical precision is varied (*left*), and reduces the number of score function evaluations (NFE) without harming quality (*middle*). The invertible mapping from latents to images allows for interpolations (*right*).

Results: Sampling, Inpainting

Figure 4: *Left*: Class-conditional samples on 32×32 CIFAR-10. Top four rows are automobiles and bottom four rows are horses. *Right:* Inpainting (top two rows) and colorization (bottom two rows) results on 256×256 LSUN. First column is the original image, second column is the masked/gray-scale image, remaining columns are sampled image completions or colorizations.

Results: Samples

Results: Samples

References

- J. Ho, A. Jain, P. Abbeel. Denoising diffusion probabilistic models, NeurIPS, 2020.
- Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon, B. Poole. Score-based generative modeling through stochastic differential equations, ICLR, 2021.