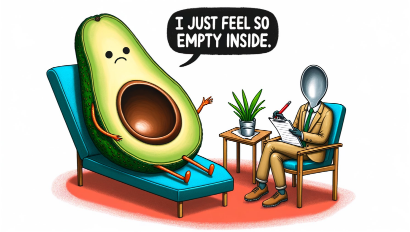
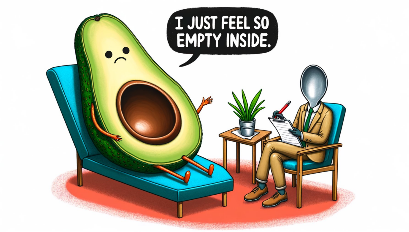
Diffusion

- Diffusion has become the most successful method for image generation, serving as backbone for DALL-E, Midjourney, Stable Diffusion



Diffusion

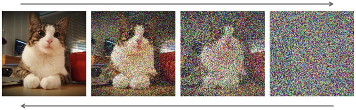
- Diffusion has become the most successful method for image generation, serving as backbone for DALL-E, Midjourney, Stable Diffusion



- Have a complicated distribution (say over images) q_0, would like to *learn* the distribution and then *sample* from it

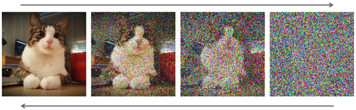
Forward and Reverse SDE

- **Idea:** Add *noise* to training images, learn how to *denoise*



Forward and Reverse SDE

- **Idea:** Add *noise* to training images, learn how to *denoise*



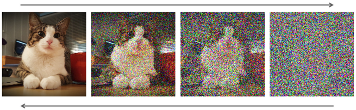
- Formally, consider the **forward SDE**

$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.

Forward and Reverse SDE

- **Idea:** Add *noise* to training images, learn how to *denoise*



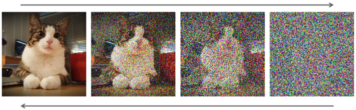
- Formally, consider the **forward SDE**

$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.

Forward and Reverse SDE

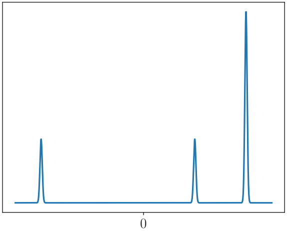
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

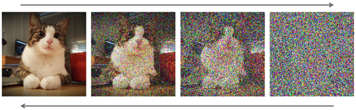
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

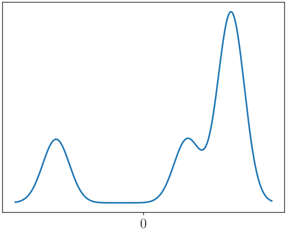
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

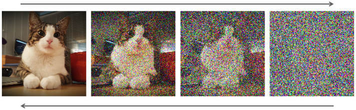
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

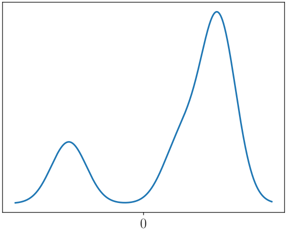
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

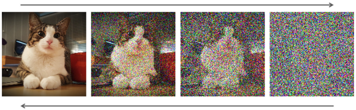
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

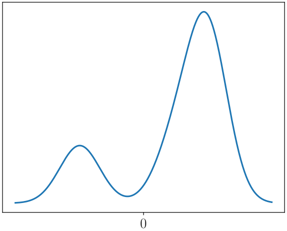
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

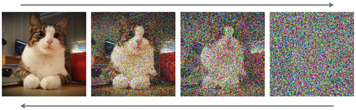
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

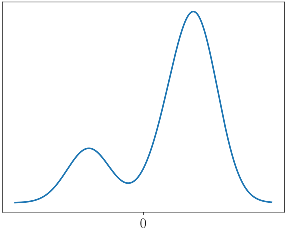
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

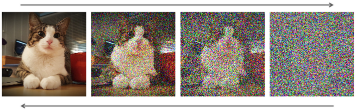
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

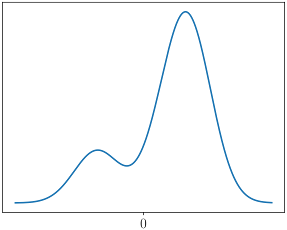
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

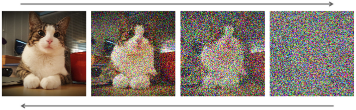
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t}x_0 + \mathcal{N}(0, (1 - e^{-2t})I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

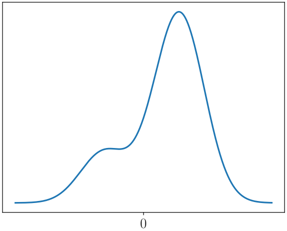
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

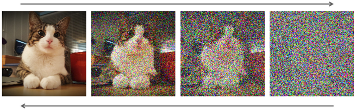
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

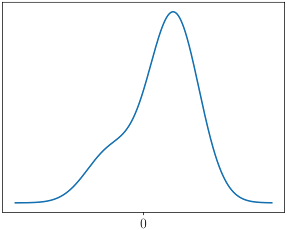
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

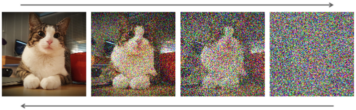
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t}x_0 + \mathcal{N}(0, (1 - e^{-2t})I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

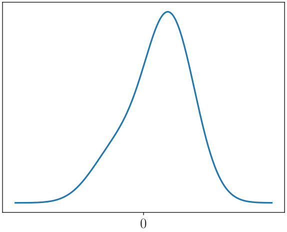
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

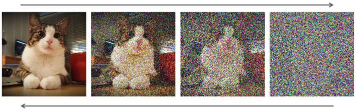
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

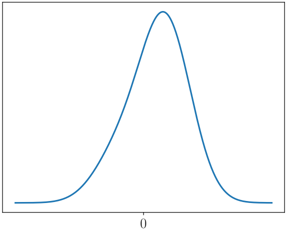
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

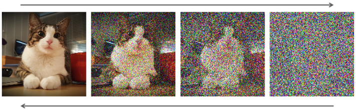
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

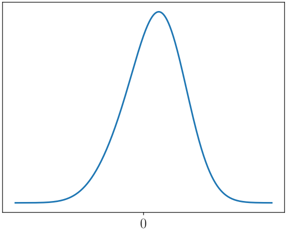
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

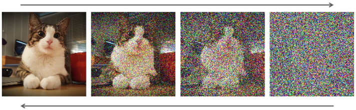
$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

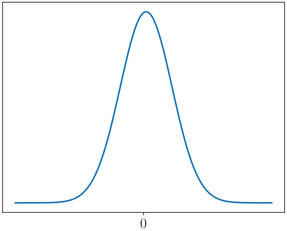
- **Idea:** Add *noise* to training images, learn how to *denoise*



- Formally, consider the **forward SDE**

$$dx_t = -x_t dt + \sqrt{2} dB_t, \quad x_0 \sim q_0$$

where B_t is Brownian motion.
Here $x_t \sim e^{-t} x_0 + \mathcal{N}(0, (1 - e^{-2t}) I_d)$. Converges to $\mathcal{N}(0, I_d)$.



Forward and Reverse SDE

- There is an associated reverse SDE – depends on **score** function $s_t = \nabla \log q_t$. Gives a way of sampling if s_t known.

Forward and Reverse SDE

- There is an associated reverse SDE – depends on **score** function $s_t = \nabla \log q_t$. Gives a way of sampling if s_t known.
- Strategy:
  1. **Train score model.** Done via ERM on the score-matching objective, which minimizes L^2 error with infinite samples.
  2. **Sample using score estimates.** Requires discretizing the reverse SDE.

Forward and Reverse SDE

- There is an associated reverse SDE – depends on **score** function $s_t = \nabla \log q_t$. Gives a way of sampling if s_t known.
- Strategy:
    1. **Train score model.** Done via ERM on the score-matching objective, which minimizes L^2 error with infinite samples.
    2. **Sample using score estimates.** Requires discretizing the reverse SDE.
- **Natural questions:** How many samples to train? How many discretization steps?

Forward and Reverse SDE

- There is an associated reverse SDE – depends on **score** function $s_t = \nabla \log q_t$. Gives a way of sampling if s_t known.
- Strategy:
    1. **Train score model.** Done via ERM on the score-matching objective, which minimizes L^2 error with infinite samples.
    2. **Sample using score estimates.** Requires discretizing the reverse SDE.
- **Natural questions:** How many samples to train? How many discretization steps?
- Remarkably, both these quantities have good theoretical bounds

Prior Work

- For γ-Wasserstein error, ε-TV error, and hypothesis class $\mathcal{H}$,

Samples to Train	**Steps to Sample**
poly $\left(d, \frac{1}{\varepsilon}, \frac{1}{\gamma}, \log \lvert\mathcal{H}\rvert\right)$ [BMR20][1]	poly $\left(d, \frac{1}{\gamma}, \frac{1}{\varepsilon}\right)$ [CCL+22][2]

[1] Adam Block, Youssef Mroueh, Alexander Rakhlin (2020)
[2] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, Anru R. Zhang (2022)
[3] Joe Benton, Valentin De Bortoli, Arnaud Doucet, George Deligiannidis (2023)

Prior Work

- For γ-Wasserstein error, ε-TV error, and hypothesis class $\mathcal{H}$,

Samples to Train	**Steps to Sample**
$\text{poly}\left(d, \frac{1}{\varepsilon}, \frac{1}{\gamma}, \log\lvert\mathcal{H}\rvert\right)$ [BMR20][1]	$\text{poly}\left(d, \frac{1}{\gamma}, \frac{1}{\varepsilon}\right)$ [CCL+22][2]
???	$\widetilde{O}\left(\frac{d \log^2 \frac{1}{\gamma}}{\varepsilon^2}\right)$ [BBDD23][3]

[1] Adam Block, Youssef Mroueh, Alexander Rakhlin (2020)
[2] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, Anru R. Zhang (2022)
[3] Joe Benton, Valentin De Bortoli, Arnaud Doucet, George Deligiannidis (2023)

Prior Work

- For γ-Wasserstein error, ε-TV error, and hypothesis class $\mathcal{H}$,

Samples to Train	**Steps to Sample**
$\text{poly}\left(d, \frac{1}{\varepsilon}, \frac{1}{\gamma}, \log\lvert\mathcal{H}\rvert\right)$ [BMR20][1]	$\text{poly}\left(d, \frac{1}{\gamma}, \frac{1}{\varepsilon}\right)$ [CCL+22][2]
???	$\widetilde{O}\left(\frac{d\log^2\frac{1}{\gamma}}{\varepsilon^2}\right)$ [BBDD23][3]

Can we **train** using a number of **samples** scaling polylogarithmically in $\frac{1}{\gamma}$ just like the number of steps?

[1] Adam Block, Youssef Mroueh, Alexander Rakhlin (2020)
[2] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, Anru R. Zhang (2022)
[3] Joe Benton, Valentin De Bortoli, Arnaud Doucet, George Deligiannidis (2023)

Our Results

- Prior works on sampling assume score learned in L^2. We show that learning the score in L^2 *requires* $\text{poly}(\frac{1}{\gamma})$ samples.

Our Results

- Prior works on sampling assume score learned in L^2. We show that learning the score in L^2 *requires* $\text{poly}(\frac{1}{\gamma})$ samples.
- Nevertheless, we show that it is possible to learn the score in a *weaker* sense using $\text{poly}\log\left(\frac{1}{\gamma}\right)$ samples. This suffices for efficient sampling.

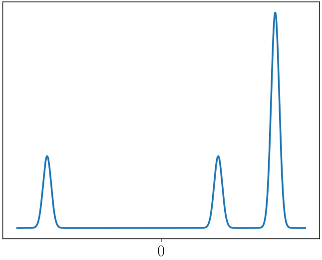
Our Results

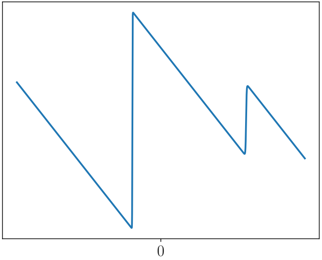
- Prior works on sampling assume score learned in L^2. We show that learning the score in L^2 *requires* $\mathrm{poly}(\frac{1}{\gamma})$ samples.
- Nevertheless, we show that it is possible to learn the score in a *weaker* sense using $\mathrm{poly}\log\left(\frac{1}{\gamma}\right)$ samples. This suffices for efficient sampling.

Samples to Train	**Steps to Sample**
$\mathrm{poly}\left(d, \frac{1}{\varepsilon}, \frac{1}{\gamma}, \log\lvert\mathcal{H}\rvert\right)$ [BMR20]	$\mathrm{poly}\left(d, \frac{1}{\gamma}, \frac{1}{\varepsilon}\right)$ [CCL+22]
$\widetilde{O}\left(\frac{d^2}{\varepsilon^5}\log^3\frac{1}{\gamma}\log\lvert\mathcal{H}\rvert\right)$ **[Ours]**	$\widetilde{O}\left(\frac{d\log^2\frac{1}{\gamma}}{\varepsilon^2}\right)$ [BBDD23]

Only require second moment to be between $\frac{1}{\mathrm{poly}(d)}$ and $\mathrm{poly}(d)$.

Intuition



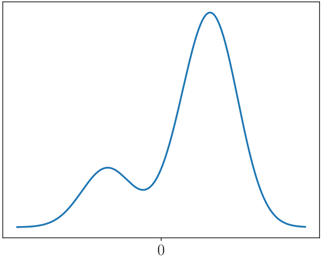


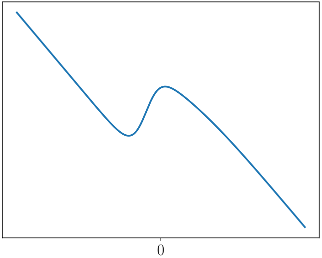
- To get $\operatorname{poly}\log\frac{1}{\gamma}$ dependence for **sampling**, [CLL23][4], [BBDD23] observe that score function better behaved with increasing noise. Can tolerate larger score error for small t, proportional to $\frac{1}{\min(1,t)}$.

[4]Hongrui Chen, Holden Lee, Jianfeng Lu (2023)

*In a weaker sense than L^2, but sufficient for sampling

Intuition

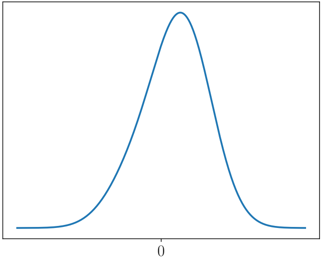


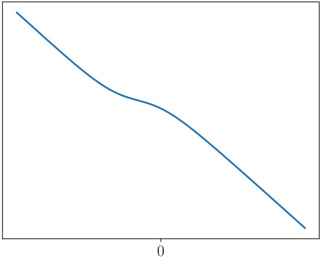


- To get $\operatorname{poly}\log\frac{1}{\gamma}$ dependence for **sampling**, [CLL23][4], [BBDD23] observe that score function better behaved with increasing noise. Can tolerate larger score error for small t, proportional to $\frac{1}{\min(1,t)}$.

[4] Hongrui Chen, Holden Lee, Jianfeng Lu (2023)
*In a weaker sense than L^2, but sufficient for sampling

Intuition



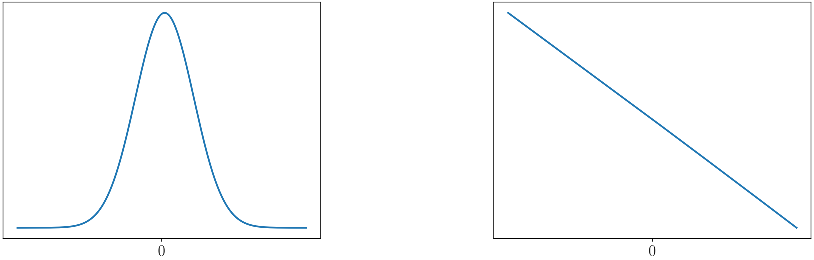


- To get $\operatorname{poly}\log \frac{1}{\gamma}$ dependence for **sampling**, [CLL23][4], [BBDD23] observe that score function better behaved with increasing noise. Can tolerate larger score error for small t, proportional to $\frac{1}{\min(1,t)}$.

[4]Hongrui Chen, Holden Lee, Jianfeng Lu (2023)

*In a weaker sense than L^2, but sufficient for sampling

Intuition

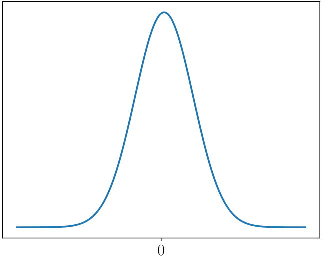


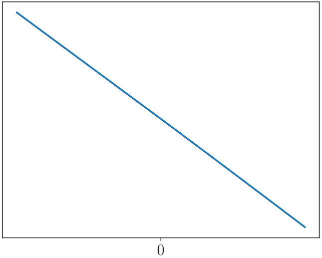
- To get poly $\log \frac{1}{\gamma}$ dependence for **sampling**, [CLL23][4], [BBDD23] observe that score function better behaved with increasing noise. Can tolerate larger score error for small t, proportional to $\frac{1}{\min(1,t)}$.

[4] Hongrui Chen, Holden Lee, Jianfeng Lu (2023)

*In a weaker sense than L^2, but sufficient for sampling

Intuition





- To get $\text{poly}\log\frac{1}{\gamma}$ dependence for **sampling**, [CLL23][4], [BBDD23] observe that score function better behaved with increasing noise. Can tolerate larger score error for small t, proportional to $\frac{1}{\min(1,t)}$.
- We exploit this for **training** – show $\text{poly}\log\frac{1}{\gamma}$ dependence for sample complexity to learn score*.

[4]Hongrui Chen, Holden Lee, Jianfeng Lu (2023)

*In a weaker sense than L^2, but sufficient for sampling

Conclusion

- Learning the score in L^2 *requires* $\mathrm{poly}\left(\frac{1}{\gamma}\right)$ samples, can learn score in weaker sense with $\mathrm{poly}\log\frac{1}{\gamma}$ dependence for fast sampling

Conclusion

- Learning the score in L^2 *requires* $\operatorname{poly}\left(\frac{1}{\gamma}\right)$ samples, can learn score in weaker sense with $\operatorname{poly}\log\frac{1}{\gamma}$ dependence for fast sampling
- Proof exploits the fact that we need weaker approximations for small t, and that scores are better behaved as t increases.

Conclusion

- Learning the score in L^2 *requires* poly $\left(\frac{1}{\gamma}\right)$ samples, can learn score in weaker sense with poly $\log \frac{1}{\gamma}$ dependence for fast sampling
- Proof exploits the fact that we need weaker approximations for small t, and that scores are better behaved as t increases.

Samples to Train	**Steps to Sample**
poly $\left(d, \frac{1}{\varepsilon}, \frac{1}{\gamma}, \log \lvert\mathcal{H}\rvert\right)$ [BMR20]	poly $\left(d, \frac{1}{\gamma}, \frac{1}{\varepsilon}\right)$ [CCL+22]
$\widetilde{O}\left(\frac{d^2}{\varepsilon^5} \log^3 \frac{1}{\gamma} \log \lvert\mathcal{H}\rvert\right)$ **[Ours]**	$\widetilde{O}\left(\frac{d \log^2 \frac{1}{\gamma}}{\varepsilon^2}\right)$ [BBDD23]