Exploring Novel Sampling Techniques in SDE-based Denoising Generative Models

CS-726 Course Project

Ninad Gandhi, Prabhat Reddy, Sachin Giroh, Jimut Bahan Pal

CMInDS, IIT Bombay

Table of contents

Motivation: Generative models using score functions

Score function

- **Score**: Removes the intractable normalizing constant.

Score function

- **Score**: Removes the intractable normalizing constant.
- Train score-based models by minimizing the Fisher divergence.

$$\mathbb{E}_{p(x)}[\|\nabla_x \log p(x) - s_\theta(x)\|_2^2] \tag{1}$$

Score function

- **Score**: Removes the intractable normalizing constant.
- Train score-based models by minimizing the Fisher divergence.

$$\mathbb{E}_{p(x)}[\|\nabla_x \log p(x) - s_\theta(x)\|_2^2] \tag{1}$$

- However we don't know the data distribution, we can not calculate the ground truth score directly.

Score function

- **Score**: Removes the intractable normalizing constant.
- Train score-based models by minimizing the Fisher divergence.

$$\mathbb{E}_{p(x)}[\|\nabla_x \log p(x) - s_\theta(x)\|_2^2] \tag{1}$$

- However we don't know the data distribution, we can not calculate the ground truth score directly.
- **Score/Sliced Score Matching** eliminates the data score using integration by parts.

Score function

- **Score**: Removes the intractable normalizing constant.
- Train score-based models by minimizing the Fisher divergence.

$$\mathbb{E}_{p(x)}[\|\nabla_x \log p(x) - s_\theta(x)\|_2^2] \tag{1}$$

- However we don't know the data distribution, we can not calculate the ground truth score directly.
- **Score/Sliced Score Matching** eliminates the data score using integration by parts.
- Score matching objectives can be optimized with stochastic gradient descent, analogous to the log-likelihood objective.

Score based generative models (discrete)

Langevin MCMC using estimated scores

After score matching, we can use **Langevin dynamics** to draw samples[3]. Langevin dynamics accesses $p(x)$ only through the score.

$$x_{i+1} \leftarrow x_i + \epsilon \nabla_x \log p(x) + \sqrt{2\epsilon} z_i, \qquad i = 0, 1, \dots, K \tag{2}$$

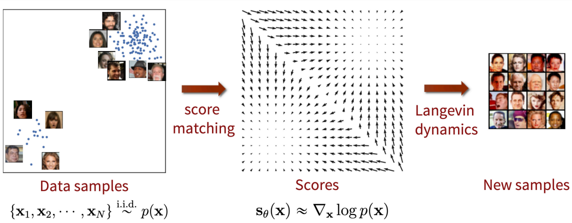


Figure 1: Score Matching with Langevin Dynamics (SMLD)[3]

[3]Song, Y., & Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution.

Langevin MCMC using estimated scores: Limitation

- Estimated score functions are inaccurate in low density regions where few data points are available for computing the score matching objective.
- Realistic data is often sparsely distributed, hence our initial sample is highly likely to be in low density regions.

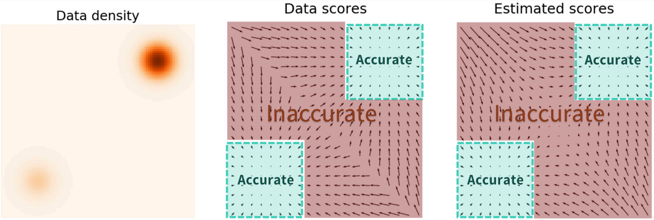


Figure 2: Pitfalls of Langevin MCMC: Score function[3]

[3]Song, Y., & Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution.

Langevin MCMC using estimated scores: Limitation

- Estimated score functions are inaccurate in low density regions where few data points are available for computing the score matching objective.
- Realistic data is often sparsely distributed, hence our initial sample is highly likely to be in low density regions.
- Inaccurate score-based model will derail Langevin dynamics from the very beginning.
- How to accurately estimate the score function in regions of low data density?

[3]Song, Y., & Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution.

Score Matching with Langevin Dynamics (SMLD)

- Solution: **Perturb the data points with noise** and train score-based models on the noisy data points.

Score Matching with Langevin Dynamics (SMLD)

- Solution: **Perturb the data points with noise** and train score-based models on the noisy data points.
- But how to choose the appropriate noise scale? Larger noise over-corrupts the data and smaller noise does not cover low density regions.

Score Matching with Langevin Dynamics (SMLD)

- Solution: **Perturb the data points with noise** and train score-based models on the noisy data points.
- But how to choose the appropriate noise scale? Larger noise over-corrupts the data and smaller noise does not cover low density regions.
- Way forward: Introduce multiple scales of noise perturbations simultaneously.

Score Matching with Langevin Dynamics (SMLD)

- Solution: **Perturb the data points with noise** and train score-based models on the noisy data points.
- But how to choose the appropriate noise scale? Larger noise over-corrupts the data and smaller noise does not cover low density regions.
- Way forward: Introduce multiple scales of noise perturbations simultaneously.
- **Annealed Langevin Dynamics**: Sample with langevin dynamics using decreasing noise scales. This works well!

Denoising Diffusion Probabilistic Model (DDPM)

- Forward noise is modeled by a directed chain graph (Bayesian Network). DDPM attempts to learn the reverse process, characterized by $s_\theta(x_i, i)$.

$$p_\theta(x_{i-1}|x_i) = \mathcal{N}\left(x_{i-1}; \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i, i)), \beta_i I\right) \qquad (3)$$

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Denoising Diffusion Probabilistic Model (DDPM)

- Forward noise is modeled by a directed chain graph (Bayesian Network). DDPM attempts to learn the reverse process, characterized by $s_\theta(x_i, i)$.

$$p_\theta(x_{i-1}|x_i) = \mathcal{N}\left(x_{i-1}; \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i, i)), \beta_i I\right) \tag{3}$$

- After training, samples are generated by starting from $x_N \sim \mathcal{N}(0, I)$ and sampling from the reverse bayesian network as follows.

$$x_{i-1} = \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i.i)) + \sqrt{\beta_i} z_i, \quad i = N, N-1, \ldots, 1 \tag{4}$$

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Denoising Diffusion Probabilistic Model (DDPM)

- Forward noise is modeled by a directed chain graph (Bayesian Network). DDPM attempts to learn the reverse process, characterized by $s_\theta(x_i, i)$.

$$p_\theta(x_{i-1}|x_i) = \mathcal{N}\left(x_{i-1}; \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i, i)), \beta_i I\right) \tag{3}$$

- After training, samples are generated by starting from $x_N \sim \mathcal{N}(0, I)$ and sampling from the reverse bayesian network as follows.

$$x_{i-1} = \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta(x_i.i)) + \sqrt{\beta_i} z_i, \quad i = N, N-1, \ldots, 1 \tag{4}$$

- Song et al.[4] call this method **ancestral sampling**, which is just another term for the well known **forward sampling** method.

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

DDPM as a score based model

- The objective for training DDPM is as follows.

$$\mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon-\epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0+\sqrt{1-\bar{\alpha}_t}\epsilon,t\right)\right\|^2\right] \tag{5}$$

- Song et al.[4] re-interpreted the above objective using scores in the following manner.

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} \sigma_i^2 \mathbb{E}_{p_{data}(x)} \mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \left[s_\theta(\tilde{x},\sigma_i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x} \mid x)_2^2 \right]. \tag{6}$$

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Score based SDE models (continuous)

Motivation for Score based SDE models

- Limitation of DDPM: the reverse process requires N steps to generate a sample, where N is generally large. Similarly, N steps are required to generate a sample using annealed langevin dynamics.

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Motivation for Score based SDE models

- Limitation of DDPM: the reverse process requires N steps to generate a sample, where N is generally large. Similarly, N steps are required to generate a sample using annealed langevin dynamics.
- If data is perturbed in a continuous-time stochastic process, we obtain[4]
  - Higher quality samples
  - Exact log-likelihood computation
  - Controllable generation for inverse problem solving

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Motivation for Score based SDE models

- Limitation of DDPM: the reverse process requires N steps to generate a sample, where N is generally large. Similarly, N steps are required to generate a sample using annealed langevin dynamics.
- If data is perturbed in a continuous-time stochastic process, we obtain[4]
  - Higher quality samples
  - Exact log-likelihood computation
  - Controllable generation for inverse problem solving
- How to represent a stochastic process?

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Motivation for Score based SDE models

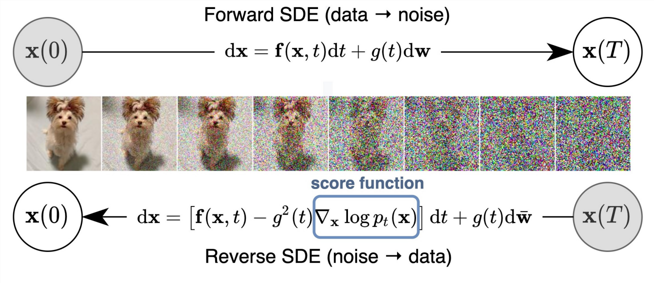
- Limitation of DDPM: the reverse process requires N steps to generate a sample, where N is generally large. Similarly, N steps are required to generate a sample using annealed langevin dynamics.
- If data is perturbed in a continuous-time stochastic process, we obtain[4]
  - Higher quality samples
  - Exact log-likelihood computation
  - Controllable generation for inverse problem solving
- How to represent a stochastic process? Using an SDE, because stochastic Processes are solutions of SDEs.

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Score based SDE

- The forward noising process can be modeled using an SDE.

$$\mathrm{d}x = f(x,t)\,\mathrm{d}t + g(t)\,\mathrm{d}w \tag{7}$$



[1]Anderson, B. D. (1982). Reverse-time diffusion equation models.

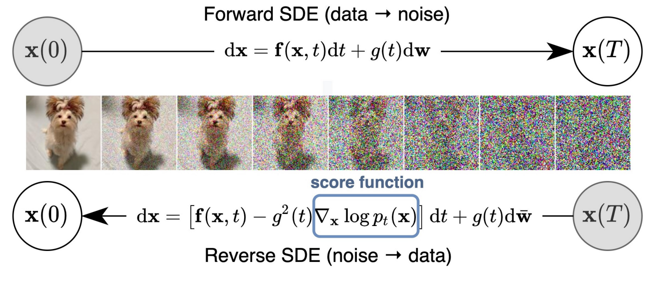
Score based SDE

- The forward noising process can be modeled using an SDE.

$$\mathrm{d}x = f(x,t)\,\mathrm{d}t + g(t)\,\mathrm{d}w \tag{7}$$

- Every forward process SDE has a corresponding reverse SDE[1].

$$\mathrm{d}x = [f(x,t) - g(t)^2 \nabla_x \log p_t(x)]\,\mathrm{d}t + g(t)\,\mathrm{d}\bar{w} \tag{8}$$



[1]Anderson, B. D. (1982). Reverse-time diffusion equation models.

Variance exploding (VE) and Variance preserving (VP) SDEs

Noise perturbations used in SMLD and DDPM can be interpreted as discretizations of two different SDEs.[4]

$$x_i = x_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} z_{i-1}$$

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

Continuous process corresponding to SMLD gives rise to VE SDE.

$$x_i = \sqrt{1-\beta_i} x_{i-1} + \sqrt{\beta_i} z_{i-1}$$

$$\mathrm{dx} = -\frac{1}{2}\beta(t) x\,\mathrm{dt} + \sqrt{\beta(t)}\,\mathrm{dw}$$

Continuous process corresponding to DDPM gives rise to VP SDE.

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Sampling methods in score based SDE models

Sampling from score based models

- Bottleneck for diffusion models. Sampling is slower in comparison with other generative models.

Sampling from score based models

- Bottleneck for diffusion models. Sampling is slower in comparison with other generative models.
- Sampling efficiency metric: number of **network function evaluations (NFE)** required to generate an image of desired quality.
    - Eg: If a DDPM takes 1000 steps to generate 1 sample, and each step requires evaluating the learnt neural network once, then NFE=1000.

Sampling from score based models

- Bottleneck for diffusion models. Sampling is slower in comparison with other generative models.
- Sampling efficiency metric: number of **network function evaluations (NFE)** required to generate an image of desired quality.
    - Eg: If a DDPM takes 1000 steps to generate 1 sample, and each step requires evaluating the learnt neural network once, then NFE=1000.
- Sample quality metrics: **FID**, **inception score**.

Sampling from score based models

- Bottleneck for diffusion models. Sampling is slower in comparison with other generative models.
- Sampling efficiency metric: number of **network function evaluations (NFE)** required to generate an image of desired quality.
    - Eg: If a DDPM takes 1000 steps to generate 1 sample, and each step requires evaluating the learnt neural network once, then NFE=1000.
- Sample quality metrics: **FID**, **inception score**.

We will look at various sampling techniques in the upcoming slides.

Predictor-Corrector Sampler

- DDPM uses ancestral sampling, but doesn't use a langevin-based step. SMLD samples according to langevin dynamics only.

Predictor-Corrector Sampler

- DDPM uses ancestral sampling, but doesn't use a langevin-based step. SMLD samples according to langevin dynamics only. What if we use both sampling techniques? Will it improve sample quality?

Predictor-Corrector Sampler

- DDPM uses ancestral sampling, but doesn't use a langevin-based step. SMLD samples according to langevin dynamics only. What if we use both sampling techniques? Will it improve sample quality?
- **Predictor-Corrector samplers** first make a prediction step (solve the DE) followed by a corrector step (gradient ascent).
    - Predictor is any numerical SDE solver that predicts $x_{t+\Delta t}$.
    - Corrector is any MCMC approach that uses score function.

Predictor-Corrector Sampler

- DDPM uses ancestral sampling, but doesn't use a langevin-based step. SMLD samples according to langevin dynamics only. What if we use both sampling techniques? Will it improve sample quality?
- **Predictor-Corrector samplers** first make a prediction step (solve the DE) followed by a corrector step (gradient ascent).
  - Predictor is any numerical SDE solver that predicts $x_{t+\Delta t}$.
  - Corrector is any MCMC approach that uses score function.
- This makes sampling more efficient and improves sampling quality, when compared to using predictor-only or corrector-only samplers.

Probability Flow ODE

- Probability Flow ODE is obtained by converting SDE to ODE without changing marginal distribution.

$$\mathrm{d}\mathrm{x} = [f(x,t) - \frac{1}{2}g(t)^2 \nabla_x \log p_t(x)]\,\mathrm{d}t \tag{9}$$

Probability Flow ODE

- Probability Flow ODE is obtained by converting SDE to ODE without changing marginal distribution.

$$\mathrm{dx} = [f(x,t) - \frac{1}{2}g(t)^2\nabla_x \log p_t(x)]\,\mathrm{dt} \tag{9}$$

- By solving this ODE, we can sample from the same distributions as the reverse SDE. The probability flow ODE becomes a special case of a neural ODE.

Probability Flow ODE

- Probability Flow ODE is obtained by converting SDE to ODE without changing marginal distribution.

$$\mathrm{dx} = [f(x,t) - \frac{1}{2}g(t)^2\nabla_x \log p_t(x)]\,\mathrm{dt} \tag{9}$$

- By solving this ODE, we can sample from the same distributions as the reverse SDE. The probability flow ODE becomes a special case of a neural ODE.
- State of the art ODE solvers can be used to generate samples efficiently.

Probability Flow ODE

In particular, PF is an example of **continuous normalizing flows** since the probability flow ODE converts a data distribution to a prior noise distribution (since it shares the same marginal distributions as the SDE) and is fully invertible.

	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**

Figure 2: Comparing different reverse time SDE solvers on the CIFAR10 dataset[4]

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Fast Sampling: DEIS and a more general SDE

Faster Sampling

- Zhang et al.[5] discuss two main ways to improve sampling efficiency:
  - Optimize the forward process so that backward process is more efficient. E.g. DDIM[2] uses a non-markovian noising process.
  - Speed up the numerical solver for SDEs or ODEs.

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Faster Sampling

- Zhang et al.[5] discuss two main ways to improve sampling efficiency:
  - Optimize the forward process so that backward process is more efficient. E.g. DDIM[2] uses a non-markovian noising process.
  - Speed up the numerical solver for SDEs or ODEs.
- Diffusion Exponential Integrator Sampler (DEIS)[5] attempts the latter approach, and can be used on any pretrained diffusion model.

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Faster Sampling

- Zhang et al.[5] discuss two main ways to improve sampling efficiency:
  - Optimize the forward process so that backward process is more efficient. E.g. DDIM[2] uses a non-markovian noising process.
  - Speed up the numerical solver for SDEs or ODEs.
- Diffusion Exponential Integrator Sampler (DEIS)[5] attempts the latter approach, and can be used on any pretrained diffusion model.
- DEIS achieves SOTA performance when NFE is small: **4.17 FID** with **10 NFEs**, **2.86 FID** with **20 NFEs** on CIFAR10 dataset.

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Diffusion Exponential Integrator Sampler (DEIS)

DEIS assumes a forward diffusion process with a linear drift coefficient.

$$\mathrm{dx} = F_t x \, \mathrm{dt} + G_t \, \mathrm{dw} \tag{10}$$

A family of reverse SDEs is given by

$$d\hat{x} = \left[F_t \hat{x} - \frac{1+\lambda^2}{2} G_t G_t^T s_\theta(\hat{x}, t) \right] \mathrm{dt} + \lambda G_t \, \mathrm{dw} \, . \tag{11}$$

When $\lambda = 0$ the above equation corresponds to the probability flow ODE, while $\lambda = 1$ gives us the approximated reverse SDE for the above forward SDE.

Diffusion Exponential Integrator Sampler (DEIS)

The error of the generative model is defined as the difference between $p_0(x)$ and $\hat{p}_0(x)$. Two error sources:

Fitting Error Difference between the learned score network and ground truth score.

Discretization Error Errors introduced in the discretization process to solve reverse SDE equations numerically.

Objective: to **minimize the above two errors** so as to **increase the step-size** in the discretization process without compromising on the sample quality.

Minimizing the discretization error

Consider the case of probability flow ODE.

$$d\hat{x} = \left[F_t\hat{x} - \frac{1}{2}G_tG_t^T s_\theta(\hat{x}, t)\right] \mathrm{dt}\,. \tag{12}$$

The exact solution to the above ODE is given by

$$\hat{x}_t = \Psi(t,s)\hat{x}_s + \int_s^t \Psi(t,\tau)\left[-\frac{1}{2}G_tG_t^T s_\theta(\hat{x}_\tau, \tau)\right] d\tau \tag{13}$$

where $\Psi(t,s)$ satisfying $\frac{\partial}{\partial t}\Psi(t,s) = F_t\Psi(t,s)$ and $\Psi(s,s) = I$ is known as the *transition matrix* from time s to t associated with F_τ.

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Minimizing the discretization error

Consider the case of probability flow ODE.

$$d\hat{x} = \left[F_t\hat{x} - \frac{1}{2}G_tG_t^T s_\theta(\hat{x}, t)\right] \mathrm{dt}\,. \tag{12}$$

The exact solution to the above ODE is given by

$$\hat{x}_t = \Psi(t, s)\hat{x}_s + \int_s^t \Psi(t, \tau)\left[-\frac{1}{2}G_tG_t^T s_\theta(\hat{x}_\tau, \tau)\right] d\tau \tag{13}$$

If we have $s_\theta(\mathrm{x}_t, t) = \nabla_x \log p_t(\mathrm{x}) \quad \forall \mathrm{x}, t$ and $\hat{p}_T^* = \pi$, then it has been shown by [5] that the estimated $\hat{p}_t^*$ exactly matches the actual $p_t \quad \forall 0 \leq t \leq T$. The problem is that this assumption is not true for most choices of x, t. We discretize the above and get an approximate solution.

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Ingredients for fast sampling

The *EI approach* gives the following discretized solution.

$$\hat{x}_{t-\Delta t} = \Psi(t-\Delta t, t)\hat{x}_t + \int_t^{t-\Delta t} \Psi(t-\Delta t, t)\left[-\frac{1}{2}G_\tau G_\tau^T d\tau\right] s_\theta(\hat{x}_t, t) \quad (14)$$

- If we assume $s_\theta(\hat{x}_t, t)$ is constant over the time interval $[t-\Delta t, t]$, then the score approximation error increases, and EI approach perform worse than the Euler method.
- To address the above issue, a different parameterization of the score network is used.

$$\nabla \log p_t(x) \approx -L_t^{-T}\epsilon_\theta(x, t), \quad L_t L_t^T = \Sigma_t \qquad (15)$$

In this case, we assume that $\epsilon_\theta(x, \tau)$ is constant over the time interval $\tau \in [t-\Delta t, t]$.

Ingredients for fast sampling

- The score approximation error Δ_s reduces now!
- However, the polynomial extrapolation of ϵ_θ gives even better results as compared to the above zeroth order approximation.

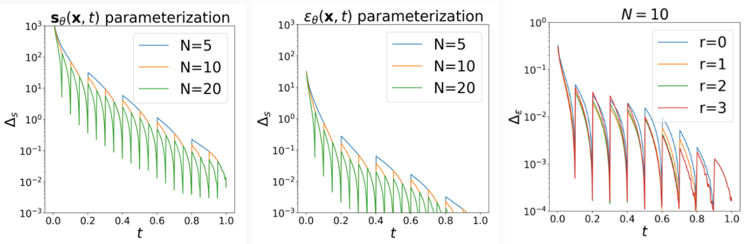


Figure 3: Improving upon score approximation.[5]

[5]Zhang, Q., & Chen, Y. (2022). Fast sampling of diffusion models with exponential integrator.

Limitations of DEIS

- Assumes linear drift coefficient.
- No theoretical proofs given. Arguments in the paper come from intuition and empirical results.

Experiments and Discussion

Experiments

- Set up and run Score-SDE and DEIS code bases locally.
- Adapt DEIS (PyTorch) and Score-SDE (PyTorch) codebases to work with each other.
- Identify improvements to sampling techniques and implement them.
- Test the improvements.

Experiments

- Set up and run Score-SDE and DEIS code bases locally. Done!
- Adapt DEIS (PyTorch) and Score-SDE (PyTorch) codebases to work with each other. Done!
- Identify improvements to sampling techniques and implement them. **Dormund-Prince method** is higher order method than can work better than Adams-Bashforth method used in a variant of DEIS.
- Test the improvements. Couldn't do that - resource constraint!

Challenges Faced

- Lack of ease of reproducibility and reusability of code and programming environment.
- Unavailability of powerful hardware that can experiment with code (for storage, processing and efficiency).
- Notational inconsistency across literature.
- Involved mathematics and nonintuitive concepts.

Summary

We have gone through lots of literature in the domain of score-based generative models and familiarized ourselves with various sampling techniques and existing state of the art models.

We have tried to compile all that information in this project presentation and tried to make it easier to digest.

Questions?

References i

B. D. Anderson.
Reverse-time diffusion equation models.
Stochastic Processes and their Applications, 12(3):313–326, 1982.

J. Song, C. Meng, and S. Ermon.
Denoising diffusion implicit models.
arXiv preprint arXiv:2010.02502, 2020.

Y. Song and S. Ermon.
Generative modeling by estimating gradients of the data distribution.
Advances in neural information processing systems, 32, 2019.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole.
Score-based generative modeling through stochastic differential equations.
arXiv preprint arXiv:2011.13456, 2020.

Q. Zhang and Y. Chen.
Fast sampling of diffusion models with exponential integrator.
arXiv preprint arXiv:2204.13902, 2022.