EE364b
Convex Optimization II

Diffusion Models

Instructor : Mert Pilanci

Stanford University

May 30, 2023

Stochastic Gradient Descent

$$x_{t+1} = x_t - \alpha_t g_t$$

- g_t is an unbiased estimate of a subgradient at x_t

$$\mathbb{E}[g_t] \in \partial f(x_t)$$

- eventually iterates are near a global optimum of $f(x)$ when $f(x)$ is convex and the step-sizes α_t are chosen appropriately
- for non-convex differentiable functions, iterates are eventually near a stationary point $\nabla f(x) = 0$ under certain functional assumptions* on $f(x)$

*e.g, when the gradients are Lipschitz continuous

Langevin Diffusion (Langevin Monte Carlo)

Consider gradient descent steps function for a function $f(x)$ with additional Gaussian noise

$$x_{t+1} = x_t - \frac{\epsilon}{2}\nabla f(x_t) + \sqrt{\epsilon} z_t$$

where $z_t \sim \mathcal{N}(0, I)$

- random sample generation method
- noisy gradient descent updates
- the distribution of x_t converges to a distribution proportional to
$$e^{-f(x)}$$
as $t \to \infty$ and $\epsilon \to 0$ under certain assumptions* on $f(x)$
- known as the Langevin Monte Carlo method

*e.g., when $f(x)$ is convex

Comparing noisy and ordinary gradient descent

example: consider the quadratic function

- $f(x) = \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)$
  where $\mu \in \mathbb{R}^d$ is a known mean vector and $\Sigma \in \mathbb{S}^{d\times d}$ is a p.s.d. covariance matrix
- identical to the least squares objective $f(x) = \frac{1}{2}\|Ax-b\|_2^2$ where $A = \Sigma^{-1/2}$ and $b = \Sigma^{-1/2}\mu$
- ordinary gradient descent

$$\begin{aligned} x_{t+1} &= x_t - \epsilon\nabla f(x_t) \\ &= x_t - \epsilon\Sigma^{-1}(x_t-\mu) \end{aligned}$$

Comparing noisy and ordinary gradient descent

$$f(x) = \frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)$$

- ordinary gradient descent

$$x_{t+1} = x_t - \epsilon \Sigma^{-1} (x_t - \mu)$$

- noisy gradient descent (Langevin Diffusion)

$$x_{t+1} = x_t - \epsilon \Sigma^{-1} (x_t - \mu) + \sqrt{\epsilon} z_t$$
$$z_t \sim \mathcal{N}(0, I)$$

Comparing noisy and ordinary gradient descent

- ordinary gradient descent

$$x_{t+1} = x_t - \epsilon \Sigma^{-1}(x_t - \mu)$$

Comparing noisy and ordinary gradient descent

- noisy gradient descent (Langevin Diffusion)

$$
\begin{aligned}
x_{t+1} &= x_t - \epsilon \Sigma^{-1}(x_t - \mu) + \sqrt{\epsilon} z_t \\
z_t &\sim \mathcal{N}(0, I)
\end{aligned}
$$

Variants of the Langevin Sampler

- plain Langevin diffusion

$$x_{t+1} = x_t - \frac{\epsilon}{2}\nabla f(x_t) + \sqrt{\epsilon} z_t$$

- second-order (i.e., preconditioned) Langevin

$$x_{t+1} = x_t - \frac{\epsilon}{2}(\nabla^2 f(x))^{-1}\nabla f(x_t) + (\nabla^2 f(x))^{-1/2}\sqrt{\epsilon} z_t$$

- proximal Langevin for $e^{-f(x)-g(x)}$

$$x_{t+1} = \mathrm{prox}_{\lambda g}\left(x_t - \frac{\epsilon}{2}\nabla f(x_t) + \sqrt{\epsilon} z_t\right)$$

when g is non-differentiable

Variants of the Langevin Sampler

- primal-dual Langevin for $e^{-f(x)-g(Dx)}$

$$
\begin{aligned}
x_{t+1} &= \operatorname{prox}_{\lambda f}\Big(x_t - \frac{\epsilon}{2} D^T \tilde{u}_t + \sqrt{\epsilon} z_t\Big) \\
u_{t+1} &= \operatorname{prox}_{\lambda g}\Big(u_n + \lambda D(2x_{t+1} - x_t)\Big) \\
\tilde{u}_{t+1} &= \tilde{u}_t + \tau(u_{t+1} - u_t)
\end{aligned}
$$

  - the second term can represent non-differentiable regularizers, e.g., total variation $\|Dx\|_1$ via $g(\cdot) = \|\cdot\|_1$
  - analogue of Douglas-Rachford splitting and ADMM
- mirror-Langevin: analogue of mirror descent

Langevin Diffusion and Score Functions

- suppose we want to generate samples from a probability distribution $p(x)$
- let $f(x) := \log p(x)$ and apply plain Langevin diffusion

$$x_{t+1} = x_t - \epsilon \underbrace{\nabla \log p(x)}_{s(x)} + \sqrt{\epsilon} z_t$$

$$z_t \sim \mathcal{N}(0, I)$$

- $s(x)$ is the gradient of the log-likelihood
- $s(x)$ is called the score function
- typically we have parameters θ in our score function model $s(x) := s_\theta(x)$

Score functions

- the score function $s_\theta(x) = \nabla \log p_\theta(x)$ *scores* the values of x as it assumes values from the distribution $p(x)$
- scores near zero are good scores and scores different from zero are bad scores
- stationary points in the maximum likelihood objective

$$\arg\max_x p(x) = \arg\max \log p(x)$$

are given by the zeros of the score function

$$s_\theta(x) = \nabla \log p(x) = 0$$

Examples of score functions

- one-dimensional Gaussian density $p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

  $s(x) = \frac{\partial}{\partial x} \log p(x) = \frac{x-\mu}{\sigma^2}$
- multivariate Gaussian $s(x) = \Sigma^{-1}(x - \mu)$

  note $s(x) = 0$ at $x = \mu$

Examples of score functions

- mixture of non-overlapping densities

$$p(x) = \begin{cases} \pi_1 p_1(x) & x \in C_1 \\ \pi_2 p_2(x) & x \in C_2 \end{cases}$$

score function $s(x) = \nabla \log p(x)$ at the interior of these regions are given by*

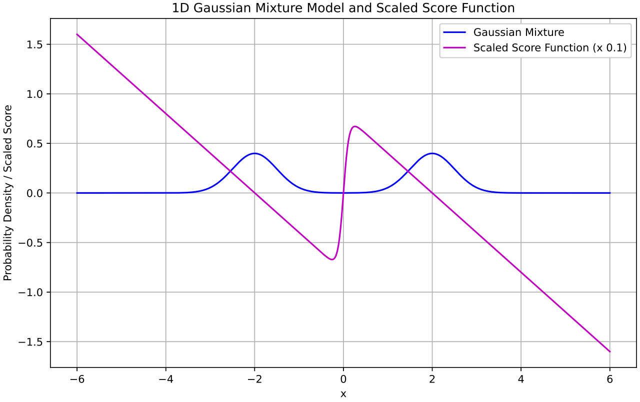
$$\begin{cases} \nabla \log p_1(x) & x \in \text{interior}(C_1) \\ \nabla \log p_2(x) & x \in \text{interior}(C_2) \end{cases}$$

- score function is a mixture of score functions
- note that the mixing weights π_1 and π_2 are lost!

*note that we need to be careful at the boundary: we are trying to differentiate a function with $0/1$ valued indicators, whose ordinary derivates do not exist. Clarke subdifferentials will exist.

Score function of a Gaussian Mixture

- for a Gaussian mixture whose components are almost disjoint, we expect the score function to be locally linear



Modeling score functions

- neural networks provide an expressive family of functions to model the score function

$$s_\theta(x) \approx W_L ... \sigma(W_2 \sigma(W_1 x))$$

  where $\theta = \{W_1, ..., W_L\}$ are learned using gradient descent
- after learning $s_\theta(x)$, we can sample using Langevin diffusion

$$x_{t+1} = x_t - \epsilon s_\theta(x_t) + \sqrt{\epsilon} z_t$$
$$z_t \sim \mathcal{N}(0, I)$$

Sampling using the learned score function

$$x_{t+1} = x_t - \epsilon s_\theta(x_t) + \sqrt{\epsilon} z_t$$
$$z_t \sim \mathcal{N}(0, I)$$

- for concave $\log p(x)$, empirical samples converges to $p(x)$ such that $s_\theta = \nabla \log p(x)$ as $t \to \infty$ and $\epsilon \to 0$ in terms of Wasserstein distance. Example: log-concave densities, e.g., multivariate Gaussian
  - concave: $\sqrt{\mathrm{KL}(\cdot||p)} \leq \epsilon$ in $O(\frac{d}{\epsilon^4})$ iterations
  - strongly concave: $\sqrt{\mathrm{KL}(\cdot||p)} \leq \epsilon$ in $O(\frac{\kappa d}{\epsilon^2} \log(\frac{1}{\epsilon}))$ iterations
- for non-convex $\log p(x)$, we converge near stationarity, i.e., the score function $s_\theta(x_t)$ almost vanishes

Challenges in fitting score models

- we can consider fitting a score model $s_\theta(x)$ via

$$\min_\theta \mathbb{E}_{x\sim p(x)} \|s_\theta(x) - \nabla \log p(x)\|_2^2$$

- for natural signals like images and audio, the density $p(x)$ is zero for most of the space
- we can smooth signals by adding Gaussian noise:

$$x + n \quad \text{where } n \sim \mathcal{N}(0, \sigma^2 I)$$

which makes the density better behaved

Denoising Score Matching

- we fit a model the the score function of a noise-perturbed distribution
- $$p(x) \to q_\sigma(\tilde{x}|x) \to q_\sigma(\tilde{x})$$

  the conditional distribution $q_\sigma(\tilde{x}|x)$ is an additive Gaussian corruption channel
- when $q_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}|x, \sigma^2 I)$ we have

  $$\tilde{x} = x + n \quad \text{where } n \sim \mathcal{N}(0, \sigma^2 I)$$

  the score function $\nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x) = -\frac{\tilde{x}-x}{\sigma^2}$ since $\tilde{x} \sim \mathcal{N}(x, \sigma^2 I)$

Denoising Score Matching

- score function fitting problem for noise-perturbed data

$$\min_\theta \mathbb{E}_{x\sim p(x)} \|s_\theta(\tilde{x}) - \nabla \log q_\sigma(x)\|_2^2$$

- is identical to (requires a short derivation)

$$\min_\theta \mathbb{E}_{x\sim p(x)} \mathbb{E}_{\tilde{x}\sim q_\sigma(\tilde{x}|x)} \|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q(\tilde{x}|x)\|_2^2$$

- for the Gaussian corruption, we have

$$\min_\theta \mathbb{E}_{x\sim p(x)} \mathbb{E}_{\tilde{x}\sim \mathcal{N}(x,\sigma^2 I)} \|s_\theta(\tilde{x}) - \frac{\tilde{x}-x}{\sigma^2}\|_2^2$$

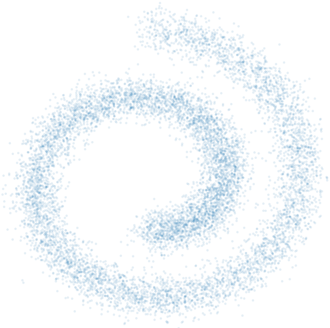
Denoising Score Matching

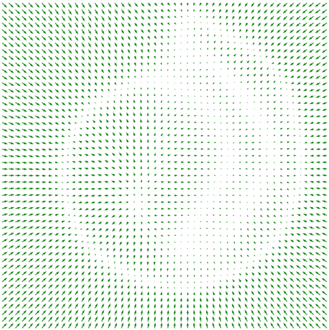
- alternatively

$$\min_\theta \mathbb{E}_{x\sim p(x)}\mathbb{E}_{n\sim\mathcal{N}(0,I)}\left\|s_\theta(x+\sigma n)-\frac{n}{\sigma}\right\|_2^2$$

- score function predicts the noise from noisy samples

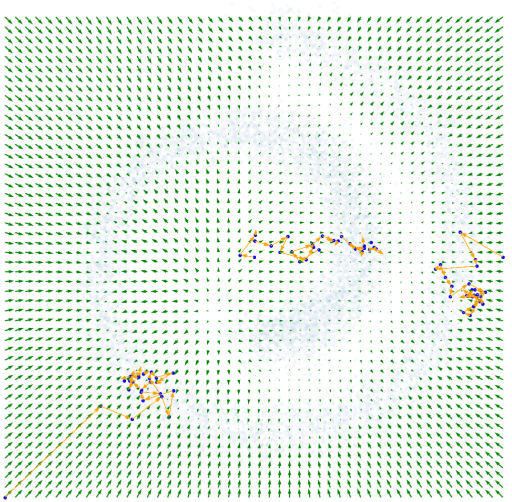
Numerical Example of Score Matching





swiss roll dataset (left) and learned score function via a ReLU NN (right) (slide credit: Kevin Murphy)

Numerical Example of Score Matching



trajectories generated by Langevin diffusion (3 trials) (slide credit: Kevin Murphy)

Challenges in Denoising Score Matching

- score function fit is less accurate over low density regions of $p(x)$ since we observe few samples
- increasing additive noise variance helps estimating a better score function, however we learn a noisier perturbed distribution
- sampling can get stuck at isolated modes

Challenges in Denoising Score Matching

- score function fit is less accurate over low density regions of $p(x)$ since we observe few samples
- increasing additive noise variance helps estimating a better score function, however we learn a noisier perturbed distribution
- sampling can get stuck at isolated modes
- **idea:** use multiple scales of noise (Song and Ermon, Generative Modeling by Estimating Gradients of the Data Distribution, 2019)

Multiple scales of noise

- we perturb data by adding Gaussian noise of standard deviation $\sigma_1, \sigma_2, ..., \sigma_L$ such that $\sigma_1 \leq \sigma_2 \leq ... \leq \sigma_L$
  i.e, given a data sample x, we generate $x + \mathcal{N}(0, \sigma_1 I)$, $x + \mathcal{N}(0, \sigma_2 I)$, ... , $x + \mathcal{N}(0, \sigma_L I)$

- we fit a score function model $s_\theta(x, \sigma)$ which is also a function of the noise level σ, e.g., a neural network with inputs x and σ

Multiple scales of noise

- we perturb data by adding Gaussian noise of standard deviation $\sigma_1, \sigma_2, ..., \sigma_L$ such that $\sigma_1 \leq \sigma_2 \leq ... \leq \sigma_L$
  i.e, given a data sample x, we generate $x + \mathcal{N}(0, \sigma_1 I)$, $x + \mathcal{N}(0, \sigma_2 I)$, ... , $x + \mathcal{N}(0, \sigma_L I)$

- we fit a score function model $s_\theta(x, \sigma)$ which is also a function of the noise level σ

Fitting a noise conditional score function

- we minimize a weighted combination of denoising score matching losses over L noise scales

$$\min_\theta \sum_{i=1}^{L} \lambda_i \mathbb{E}_{x\sim p(x)} \mathbb{E}_{n\sim\mathcal{N}(0,I)} \left\| s_\theta(x+\sigma_i n, \sigma_i) - \frac{n}{\sigma_i} \right\|_2^2$$

Fitting a noise conditional score function

- we minimize a weighted combination of denoising score matching losses over L noise scales

$$\min_\theta \sum_{i=1}^{L} \lambda_i \mathbb{E}_{x\sim p(x)} \mathbb{E}_{n\sim\mathcal{N}(0,I)} \left\| s_\theta(x+\sigma_i n, \sigma_i) - \frac{n}{\sigma_i} \right\|_2^2$$

- we can pick weights proportional to variance, $\lambda_i = \sigma_i^2$

$$\min_\theta \sum_{i=1}^{L} \sigma_i \mathbb{E}_{x\sim p(x)} \mathbb{E}_{n\sim\mathcal{N}(0,I)} \left\| s_\theta(x+\sigma_i n, \sigma_i) - \frac{n}{\sigma_i} \right\|_2^2$$

which simplifies to

$$\min_\theta \sum_{i=1}^{L} \mathbb{E}_{x\sim p(x)} \mathbb{E}_{n\sim\mathcal{N}(0,I)} \left\| \sigma_i s_\theta(x+\sigma_i n, \sigma_i) - n \right\|_2^2$$

Generative Modeling by Estimating Gradients of the Data Distribution Y. Song, S. Ermon. NeurIPS 2019

Algorithm for fiting a conditional score function

- choose a sequence of decaying noise standard deviations, e.g., $\sigma_1 = 1, \sigma_2 = 0.5, ..., \sigma_{10} = 0.01$ for $L = 10$ noise levels (a standard deviation of 0.01 is almost indistinguishable to human eyes for images)
- sample a batch of data points $x_1, ..., x_N \sim p(x)$
- sample a batch of Gaussian noise $n_1, ..., n_N$
- sample a batch of noise scale indices $i_1, ..., i_N \sim$Uniform$\{1, 2, ..., L\}$

  fit a noise conditional score model $s_\theta(x + \sigma n, \sigma) \approx n$, e.g., a DNN, to

$$\min_\theta \frac{1}{N} \sum_{k=1}^{N} \left\| \sigma_{i_k} s_\theta(x + \sigma_{i_k} n, \sigma_{i_k}) - n_k \right\|_2^2$$

Annealed Langevin Dynamics

- sample using noise levels $\sigma_1, \sigma_2, ..., \sigma_L$ sequentially as follows
- begin by sampling using the Langevin process using the smallest noise scale
  - anneal down the noise level
  - use the generated sample as initialization for the next level
  - repeat the Langevin sampling process

Annealed Langevin Dynamics

for $i \in \{1, ..., L\}$

$\epsilon_i = \epsilon_0 \, \sigma_i^2 / \sigma_L^2$

for $t \in \{1, ..., T\}$

$z_{t-1} \sim N(0, I)$

$x_t = x_{t-1} - \frac{\epsilon_i}{2} s_\theta(x_{t-1}, \sigma_i) + \sqrt{\epsilon_i} z_{t-1}$

end

$x_0 \leftarrow x_T$

end

Denoising Diffusion Models

- Annealed sampling process can be simplified by taking $T = 1$

  for $i \in \{1, ..., L\}$

  $$z_{i-1} \sim N(0, I)$$

  $$x_i = x_{i-1} - \frac{\epsilon_i}{2} s_\theta(x_{i-1}, \sigma_i) + \sqrt{\epsilon_i} z_{i-1}$$

  end

- each step of applying the score function can be viewed as denoising, i.e., reversing the noise corruption process

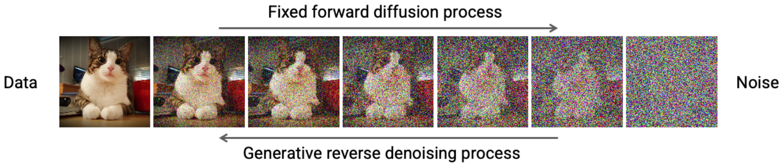


Image generation

CelebA-HQ 256x256 samples.

LSUN 256x256 Church, Bedroom, and Cat samples. Notice that our models occasionally generate dataset watermarks.

References

- Stanford CS236 Deep Generative Models course

  https://deepgenerativemodels.github.io/
- Log-Concave Sampling, Sinho Chewi

  https://chewisinho.github.io/main.pdf
- Kevin Murphy "Probabilistic Machine Learning: Advanced Topics" (2023)

  https://probml.github.io/pml-book/
- Tim Tsz-Kit Lau, Han Liu,Thomas Pock, Non-Log-Concave and Nonsmooth Sampling via Langevin Monte Carlo Algorithms, 2023.