Custom Theorems and Lists in LaTeX

Part I

SDF

Chapter 1

Introduction

Go to this hidden section.

Theorem 1.3: Pythagoras Theorem

In a right-angled triangle, the square of the hypotenuse is equal to the sum of the squares of the other two sides.

$$a^2 = b^2 + c^2$$

Concept 2.1: Probability

The probability of an event is a measure of the likelihood that the event will occur.

Chapter 2

Images and Lists

2.1 Lists in LaTeX

2.1.1 Unordered List

- First item
- Second item
- Third item

2.1.2 Ordered List

1. First item
2. Second item
3. Third item

Chapter 3

Code Blocks

3.1 Simple Code (Verbatim)

```
printf("Hello World");
```

Chapter 4

Optimization Problem

4.1 Definition of Optimization Problem

Definition 1.1: Optimization Problem

In an **optimization problem**, we minimize or maximize a function value, possibly subject to constraints.

$$\begin{aligned} \underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad & f(\theta) \\ \text{subject to} \quad & h_1(\theta) = 0, \quad h_2(\theta) = 0, \quad \ldots, \quad h_m(\theta) = 0, \\ & g_1(\theta) \leq 0, \quad g_2(\theta) \leq 0, \quad \ldots, \quad g_n(\theta) \leq 0. \end{aligned}$$

- Decision variable: θ
- Objective function: f
- Equality constraint: $h_i(\theta) = 0$ for $i = 1, \ldots, m$
- Inequality constraint: $g_j(\theta) \leq 0$ for $j = 1, \ldots, n$

In machine learning (ML), we often minimize a "loss", but sometimes we maximize the "likelihood". In any case, minimization and maximization are equivalent since

$$\text{maximize } f(\theta) \quad \Leftrightarrow \quad \text{minimize } -f(\theta).$$

Definition 1.2: Feasible Point and Constraints

$\theta \in \mathbb{R}^p$ is a **feasible point** if it satisfies all constraints:

$$\begin{array}{cc} h_1(\theta) = 0 & g_1(\theta) \leq 0 \\ \vdots & \vdots \\ h_m(\theta) = 0 & g_n(\theta) \leq 0 \end{array}$$

Optimization problem is **infeasible** if there is no feasible point.

An optimization problem with no constraint is called an **unconstrained optimization problem**. Optimization problems with constraints is called a **constrained optimization problem**.

Definition 1.3: Optimal Value and Solution

Optimal value of an optimization problem is

$$p^{\star} = \inf \{ f(\theta) \mid \theta \in \mathbb{R}^n, \theta \text{ feasible } \}$$

- $p^{\star} = \infty$ if problem is infeasible
- $p^{\star} = -\infty$ is possible
- In ML, it is often a priori clear that $0 \leq p^{\star} < \infty$.

If $f(\theta^{\star}) = p^{\star}$, we say $\theta^{\star}$ is a **solution** or $\theta^{\star}$ is **optimal**. A solution may or may not exist, and a solution may or may not be unique.

4.2 Examples of Optimization Problem

Example 1.4: Curve Fitting

Consider setup with data $X_1, \ldots, X_N$ and corresponding labels $Y_1, \ldots, Y_N$ satisfying the relationship

$$Y_i = f_{\star}(X_i) + \text{ error}$$

for $i = 1, \ldots, N$. Hopefully, "error" is small. True function $f_{\star}$ is unknown. Goal is to find a function (curve) f such that $f \approx f_{\star}$.

Example 1.5: Least-Squares Minimization

- **Problem**

$$\underset{\theta \in \mathbb{R}^p}{\operatorname{minimize}} \quad \frac{1}{2} \|X\theta - Y\|^2$$

where $X \in \mathbb{R}^{N \times p}$ and $Y \in \mathbb{R}^N$. Equivalent to

$$\underset{\theta \in \mathbb{R}^p}{\operatorname{minimize}} \frac{1}{2} \sum_{i=1}^{N} \left(X_i^{\top}\theta - Y_i\right)^2$$

where $X = \begin{bmatrix} X_1^{\top} \\ \vdots \\ X_N^{\top} \end{bmatrix}$ and $Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_N \end{bmatrix}$.

- **Solution**

To solve

$$\underset{\theta \in \mathbb{R}^p}{\operatorname{minimize}} \frac{1}{2}\|X\theta - Y\|^2$$

take gradient and set it to 0.

$$\nabla_\theta \frac{1}{2}\|X\theta - Y\|^2 = X^\top(X\theta - Y)$$

$$X^\top\left(X\theta^\star - Y\right) = 0$$
$$\theta^\star = \left(X^\top X\right)^{-1} X^\top Y$$

Here, we assume $X^\top X$ is invertible.

Concept 1.6: Least squares is an instance of curve fitting.

Define $f_\theta(x) = x^\top\theta$. Then LS becomes

$$\underset{\theta \in \mathbb{R}^p}{\operatorname{minimize}} \frac{1}{2}\sum_{i=1}^{N}\left(f_\theta\left(X_i\right) - Y_i\right)^2$$

and the solution hopefully satisfies

$$Y_i = f_\theta\left(X_i\right) + \text{ small.}$$

Since X_i and Y_i is assumed to satisfy

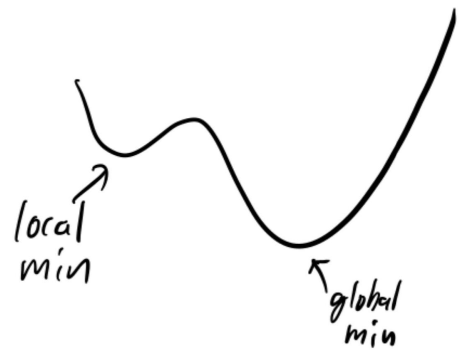
$$Y_i = f_\star\left(X_i\right) + \text{ error}$$

we are searching over linear functions (linear curves) f_θ that best fit (approximate) $f_\star$.

4.3 Local and Global Minimum

Definition 1.7: Local vs Global Minima

$\theta^\star$ is a **local minimum** if $f(\theta) \geq f\left(\theta^\star\right)$ for all feasible θ within a small neighborhood.
$\theta^\star$ is a **global minimum** if $f(\theta) \geq f\left(\theta^\star\right)$ for all feasible θ.



In the worst case, finding the global minimum of an optimization problem is difficult. However, in deep learning, optimization problems are often "solved" without any guarantee of global optimality.

Chapter 5

Basics of Monte Carlo

5.1 Monte Carlo Estimation

Definition 13.1: Monte Carlo Estimation

Consider IID data $X_1, \ldots, X_N \sim f$. Let $\phi(X) \geq 0$ be some function. (The assumption $\phi(X) \geq 0$ can be relaxed.) Consider the problem of estimating

$$I = \mathbb{E}_{X\sim f}[\phi(X)] = \int \phi(x)f(x)dx$$

One commonly uses

$$\hat{I}_N = \frac{1}{N}\sum_{i=1}^{N} \phi\left(X_i\right)$$

to estimate I, which is called **monte carlo estimation**. After all, $\mathbb{E}\left[\hat{I}_N\right] = I$ and $\hat{I}_N \to I$ by the law of large numbers. (Convergence in probability by weak law of large numbers and almost sure convergence by strong law of large numbers.)

Concept 13.2: Evidence of Convergence for Monte Carlo Estimation

We can quantify convergence with variance:

$$\mathrm{Var}_{X\sim f}\left(\hat{I}_N\right) = \sum_{i=1}^{N} \mathrm{Var}_{X_i\sim f}\left(\frac{\phi\left(X_i\right)}{N}\right) = \frac{1}{N}\mathrm{Var}_{X\sim f}(\phi(X))$$

In other words

$$\mathbb{E}\left[\left(\hat{I}_N - I\right)^2\right] = \frac{1}{N}\mathrm{Var}_{X\sim f}(\phi(X))$$

and

$$\mathbb{E}\left[\left(\hat{I}_N - I\right)^2\right] \to 0$$

as $N \to \infty$. So, $\hat{I}_N \to I$ in L^2 provided that $\mathrm{Var}_{X \sim f}(\phi(X)) < \infty$.

Definition 13.3: Empirical Risk Minimization (ERM)

In machine learning and statistics, we often wish to solve

$$\underset{\theta \in \Theta}{\operatorname{minimize}} \quad \mathcal{L}(\theta)$$

where the objective function

$$\mathcal{L}(\theta) = \mathbb{E}_{X \sim p_X}\left[\ell\left(f_\theta(X), f_\star(X)\right)\right]$$

Is the (true) risk. However, the evaluation of $\mathbb{E}_{X \sim p_X}$ is impossible (if p_X is unknown) or intractable (if p_X is known but the expectation has no closed-form solution). Therefore, we define the proxy loss function

$$\mathcal{L}_N(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell\left(f_\theta\left(X_i\right), f_\star\left(X_i\right)\right)$$

which we call the empirical risk, and solve

$$\underset{\theta \in \Theta}{\operatorname{minimize}} \quad \mathcal{L}_N(\theta)$$

This is called **empirical risk minimization (ERM)**. The idea is that

$$\mathcal{L}_N(\theta) \approx \mathcal{L}(\theta)$$

with high probability, so minimizing $\mathcal{L}_N(\theta)$ should be similar to minimizing $\mathcal{L}(\theta)$.

Concept 13.4: Evidence of Convergence for Empirical Risk Minimization

Technical note) The law of large numbers tells us that

$$\mathbb{P}\left(\left|\mathcal{L}_N(\theta) - \mathcal{L}(\theta)\right| > \varepsilon\right) = \text{ small}$$

for any given θ, but we need

$$\mathbb{P}\left(\sup_{\theta \in \Theta}\left|\mathcal{L}_N(\theta) - \mathcal{L}(\theta)\right| > \varepsilon\right) = \text{ small}$$

for all compact Θ in order to conclude that the argmins of the two losses to be similar. These types of results are established by a uniform law of large numbers.

5.2 Importance Sampling

Definition 13.5: Importance Sampling (IS)

Importance sampling (IS) is a technique for reducing the variance of a Monte Carlo estimator.
Key insight of important sampling:

$$I = \mathbb{E}_{X \sim f}[\phi(X)] = \int \phi(x) f(x) dx = \int \frac{\phi(x) f(x)}{g(x)} g(x) dx = \mathbb{E}_{X \sim g}\left[\frac{\phi(X) f(X)}{g(X)}\right]$$

(We do have to be mindful of division by 0.) Then

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} \phi\left(X_i\right) \frac{f\left(X_i\right)}{g\left(X_i\right)}$$

with $X_1, \ldots, X_N \sim g$ is also an estimator of I. Indeed, $\mathbb{E}\left[\hat{I}_N\right] = I$ and $\hat{I}_N \to I$. The weight $\frac{f(x)}{g(x)}$ is called the **likelihood ratio** or the **Radon-Nikodym derivative**.
So we can use samples from g to compute expectation with respect to f.

Example 13.6: IS Example

Consider the setup of estimating the probability

$$\mathbb{P}(X > 3) = 0.00135$$

where $X \sim \mathcal{N}(0, 1)$. If we use the regular Monte Carlo estimator

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{\{X_i > 3\}}$$

where $X_i \sim \mathcal{N}(0, 1)$, if N is not sufficiently large, we can have $\hat{I}_N = 0$. Inaccurate estimate.
If we use the IS estimator

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{\{Y_i > 3\}} \exp\left(\frac{\left(Y_i - 3\right)^2 - Y_i^2}{2}\right)$$

where $Y_i \sim \mathcal{N}(3, 1)$, having $\hat{I}_N = 0$ is much less likely. Estimate is much more accurate.

Concept 13.7: Optimal Sampling Distribution

Benefit of IS quantified by with variance:

$$\operatorname{Var}_{X\sim g}\left(\hat{I}_N\right)=\sum_{i=1}^{N}\operatorname{Var}_{X\sim g}\left(\frac{\phi\left(X_i\right)f\left(X_i\right)}{ng\left(X_i\right)}\right)=\frac{1}{N}\operatorname{Var}_{X\sim g}\left(\frac{\phi(X)f(X)}{g(X)}\right)$$

If $\operatorname{Var}_{X\sim g}\left(\frac{\phi(X)f(X)}{g(X)}\right)<\operatorname{Var}_{X\sim f}(\phi(X))$, then IS provides variance reduction. We call g the importance or sampling distribution. Choosing g poorly can increase the variance. What is the best choice of g ?

The sampling distribution

$$g(x)=\frac{\phi(x)f(x)}{I}$$

makes $\operatorname{Var}_{X\sim g}\left(\frac{\phi(X)f(X)}{g(X)}\right)=\operatorname{Var}_{X\sim g}(I)=0$ and therefore is optimal. (I serves as the normalizing factor that ensures the density g integrates to 1.)
Problem: Since we do not know the normalizing factor I, the answer we wish to estimate, sampling from g is usually difficult.

Concept 13.8: Optimized / Trained Sampling Distribution

Instead, we consider the optimization problem

$$\underset{g\in\mathcal{G}}{\operatorname{minimize}}\quad D_{\mathrm{KL}}\left(g\|\frac{\phi f}{I}\right)$$

and compute a suboptimal, but good, sampling distribution within a class of sampling distributions $\mathcal{G}$. (In ML, $\mathcal{G}=\{g_\theta\mid\theta\in\Theta\}$ is parameterized by neural networks.)
Importantly, this optimization problem does not require knowledge of I.

$$\begin{aligned}
D_{\mathrm{KL}}\left(g_\theta\|\phi f/I\right)&=\mathbb{E}_{X\sim g_\theta}\left[\log\left(\frac{Ig_\theta(X)}{\phi(X)f(X)}\right)\right]\\
&=\mathbb{E}_{X\sim g_\theta}\left[\log\left(\frac{g_\theta(X)}{\phi(X)f(X)}\right)\right]+\log I\\
&=\mathbb{E}_{X\sim g_\theta}\left[\log\left(\frac{g_\theta(X)}{\phi(X)f(X)}\right)\right]+\text{ constant independent of }\theta
\end{aligned}$$

How do we compute stochastic gradients?

5.3 Log-Derivative Trick

Definition 13.9: Log-Derivative Trick

Generally, consider the setup where we wish to solve

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \mathbb{E}_{X \sim f_\theta}[\phi(X)]$$

with SGD. (Previous situation (Concept 13.8) had θ-dependence both on and inside the expectation. For now, let's simplify the problem so that ϕ does not depend on θ.)
Incorrect gradient computation:

$$\nabla_\theta \mathbb{E}_{X \sim f_\theta}[\phi(X)] \stackrel{?}{=} \mathbb{E}_{X \sim f_\theta}\left[\nabla_\theta \phi(X)\right] = \mathbb{E}_{X \sim f_\theta}[0] = 0$$

Correct gradient computation:

$$\begin{aligned}
\nabla_\theta \mathbb{E}_{X \sim f_\theta}[\phi(X)] &= \nabla_\theta \int \phi(x) f_\theta(x) dx = \int \phi(x) \nabla_\theta f_\theta(x) dx \\
&= \int \phi(x) \frac{\nabla_\theta f_\theta(x)}{f_\theta(x)} f_\theta(x) dx = \mathbb{E}_{X \sim f_\theta}\left[\phi(X) \frac{\nabla_\theta f_\theta(X)}{f_\theta(X)}\right] \\
&= \mathbb{E}_{X \sim f_\theta}\left[\phi(X) \nabla_\theta \log\left(f_\theta(X)\right)\right]
\end{aligned}$$

Therefore, $\phi(X)\nabla_\theta \log(f_\theta(X))$ with $X \sim f_\theta$ is a stochastic gradient of the loss function. This technique is called the log-derivative trick, the likelihood ratio gradient[#], or REINFORCE[⋆].
Formula with the log-derivative ($\nabla_\theta \log(\cdot)$) is convenient when dealing with Gaussians, or more generally exponential families, since the densities are of the form

$$f_\theta(x) = h(x) \exp(\text{function of } \theta)$$

([#]P. W. Glynn, Likelihood ratio gradient estimation for stochastic systems, Communications of the ACM, 1990.
[⋆]R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 1992.)

Example 13.10: Log-Derivative Trick Example

Learn $\mu \in \mathbb{R}^2$ to minimize the objective below.

$$\underset{\mu \in \mathbb{R}^2}{\text{minimize}} \mathbb{E}_{X \sim \mathcal{N}(\mu, I)} \left\| X - \begin{pmatrix} 5 \\ 5 \end{pmatrix} \right\|^2$$

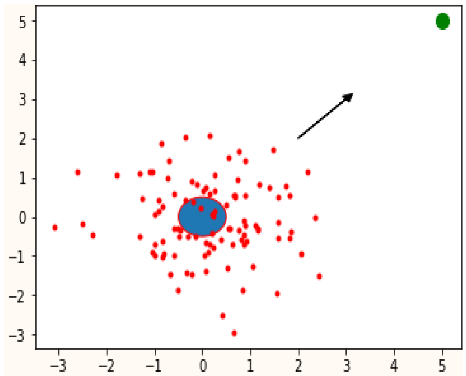
Then the loss function is

$$\mathcal{L}(\mu) = \mathbb{E}_{X \sim \mathcal{N}(\mu, I)} \left\| X - \begin{pmatrix} 5 \\ 5 \end{pmatrix} \right\|^2 = \int \left\| x - \begin{pmatrix} 5 \\ 5 \end{pmatrix} \right\|^2 \frac{1}{2\pi} \exp\left(-\frac{1}{2}\|x - \mu\|^2\right) dx$$

And, using $X_1, \ldots, X_B \sim \mathcal{N}(\mu, I)$, we have stochastic gradients

$$\nabla_\mu \mathcal{L}(\mu) = \mathbb{E}_{X\sim\mathcal{N}(\mu,I)}\left[\left\|x-\begin{pmatrix}5\\5\end{pmatrix}\right\|^2 \nabla_\mu\left(-\frac{1}{2}\|x-\mu\|^2\right)\right] \approx \frac{1}{B}\sum_{i=1}^{B}\left\|X_i-\begin{pmatrix}5\\5\end{pmatrix}\right\|^2 (X_i-\mu)$$

These stochastic gradients have large variance and thus SGD is slow.



5.4 Reparameterization Trick

Definition 13.11: Reparameterization Trick

The reparameterization trick (RT) or the pathwise derivative (PD) relies on the key insight.

$$\mathbb{E}_{X\sim\mathcal{N}(\mu,\sigma^2)}[\phi(X)] = \mathbb{E}_{Y\sim\mathcal{N}(0,1)}[\phi(\mu+\sigma Y)]$$

Gradient computation:

$$\nabla_{\mu,\sigma}\mathbb{E}_{X\sim\mathcal{N}(\mu,\sigma^2)}[\phi(X)] = \mathbb{E}_{Y\sim\mathcal{N}(0,1)}\left[\nabla_{\mu,\sigma}\phi(\mu+\sigma Y)\right] = \mathbb{E}_{Y\sim\mathcal{N}(0,1)}\left[\phi'(\mu+\sigma Y)\begin{bmatrix}1\\Y\end{bmatrix}\right]$$

$$\approx \frac{1}{B}\sum_{i=1}^{B}\phi'\left(\mu+\sigma Y_i\right)\begin{bmatrix}1\\Y_i\end{bmatrix}, \quad Y_1,\ldots,Y_B\sim\mathcal{N}(0,I)$$

RT is less general than log-derivative trick, but it usually produces stochastic gradients with lower variance.

Example 13.12: Reparameterization Trick Example

Consider the same example as before

$$\mathcal{L}(\mu) = \mathbb{E}_{X\sim\mathcal{N}(\mu,I)}\left\|X-\begin{pmatrix}5\\5\end{pmatrix}\right\|^2 = \mathbb{E}_{Y\sim\mathcal{N}(0,I)}\left\|Y+\mu-\begin{pmatrix}5\\5\end{pmatrix}\right\|^2$$

Gradient computation:

$$\begin{aligned}\nabla_\mu \mathcal{L}(\mu) &= \mathbb{E}_{Y\sim\mathcal{N}(0,I)} \nabla_\mu \left\| Y + \mu - \begin{pmatrix}5\\5\end{pmatrix} \right\|^2 = 2\mathbb{E}_{Y\sim\mathcal{N}(0,I)} \left(Y + \mu - \begin{pmatrix}5\\5\end{pmatrix} \right) \\ &\approx \frac{2}{B} \sum_{i=1}^{B} \left(Y_i + \mu - \begin{pmatrix}5\\5\end{pmatrix} \right), \quad Y_1, \ldots, Y_B \sim \mathcal{N}(0, I)\end{aligned}$$

These stochastic gradients have smaller variance and thus SGD is faster.

Example 13.13: Log Derivative Trick vs Reparameterization Trick

The image below is the result of SGD with the computed gradients by Example 13.10 and Example 13.12.