Optimization for Machine Learning

Qing-Yuan Jiang

The school of Computer Science and Engineering, NJUST

2026.09.18

Review of Previous Class

- What is machine learning?
- Difference between software and AI model
- Applications of machine learning
- The role of machine learning
- Machine learning terms

Why Optimization Matters in Machine Learning?

- The main process of machine learning:

Data Collection → Model → Prediction → Loss →
Objective Function → Optimization → Learned Parameters

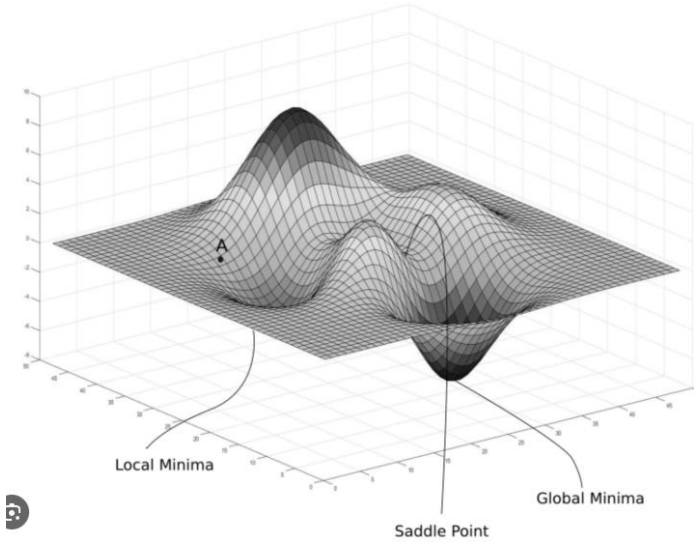
- The learned parameters are obtained through **optimization**.
- **Training a machine learning model** is often formulated as an optimization problem.

Convexity

Why Convexity?

- Optimization problems can be **very difficult** to solve

- Non-convex objective
- Multiple local minima
- Saddle points
- Difficult optimization landscape



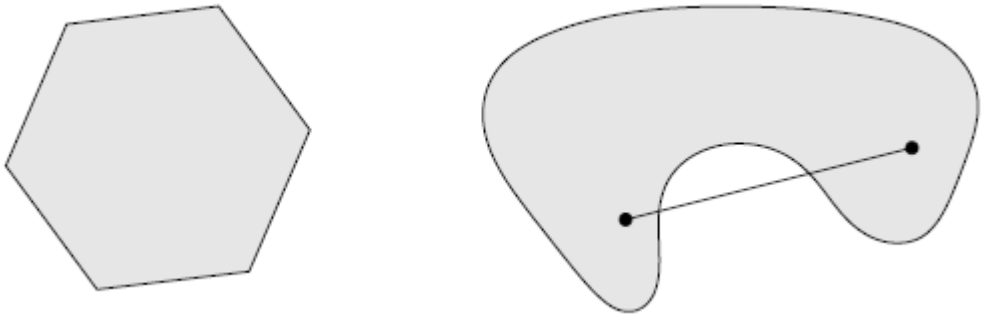
Convexity turns a difficult optimization problem into a more structured one.

Convex Set

- **Convex set**: $C \in \mathcal{R}^n$ is convex if:

$$\forall t \in [0,1], x, y \in C \Rightarrow tx + (1-t)y \in C$$

- Example:



- **Convex combination** of $x_1, x_2, \cdots, x_k \in \mathcal{R}^n$: any linear combination:

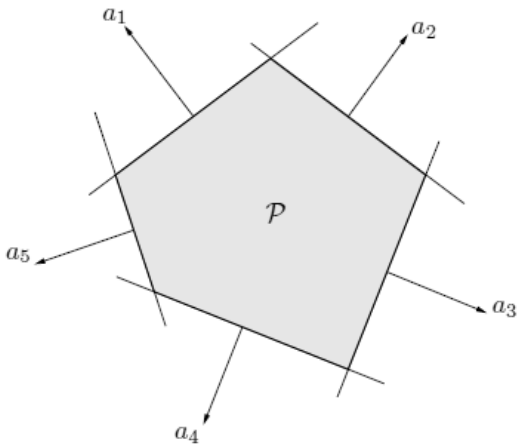
$$\theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_k x_k$$

with $\forall i \in \{1, 2, \cdots, k\}, \theta_i \geq 0$, and $\sum_{i=1}^{k} \theta_i = 1$.

- **Convex hull** $conv(C)$: all convex combinations of elements.

Examples of Convex Sets

- Empty set, point, line
- Norm ball: for given norm $\|\cdot\|$, radius r, $\{x : \|x\| \leq r\}$
- **Hyperplane**: for given a, b, $\{x : a^\top x = b\}$
- Halfspace: $\{x : a^\top x \leq b\}$
- Affine space: for given A, b, $\{x : A^\top x = b\}$
- **Polyhedron**: $\{x : A^\top x \leq b\}$



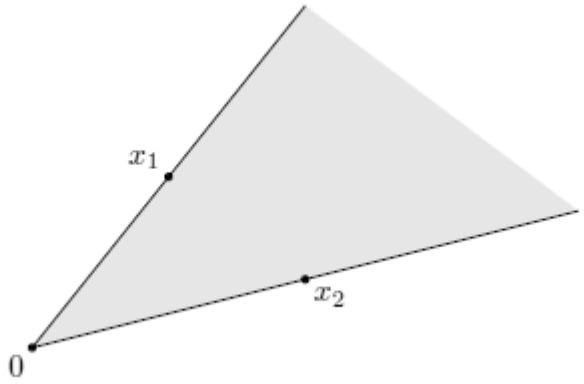
Cones

- **Cone**: $C \in \mathcal{R}^n$ is cone if:

$$\forall t \geq 0, x \in C \Rightarrow tx \in C$$

- **Convex cone**: cone that is also convex, i.e.,

$$\forall t_1, t_2 \geq 0, x_1, x_2 \in C \Rightarrow t_1 x_1 + t_2 x_2 \in C$$



- Conic combination:

$$x_1, \cdots, x_k \in \mathcal{R}^n, \theta_1 x_1 + \cdots + \theta_k x_k$$

with $\forall i \in \{1, 2, \cdots, k\}, \theta_i \geq 0$.

Conic hull collects all conic combinations.

Operations Preserving Convexity

- **Intersection**: the intersection of convex sets is convex
- **Scaling and translation:** if C is convex, then:

  $$aC + b = \{ax + b : x \in C\}$$

  is convex for any a, b
- **Affine images and preimages:** if $f(x) = Ax + b$ and C is convex, then:

  $$f(C) = \{f(x) : x \in C\}$$

  is convex, and if D is convex then:

  $$f^{-1}(D) = \{x : f(x) \in D\}$$

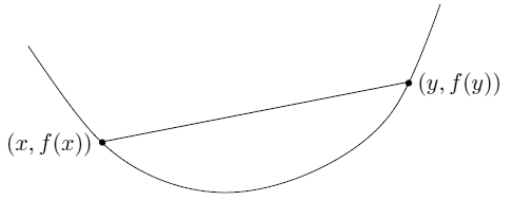
  is convex.

Convex Functions

- **Convex function**: $f : \mathcal{R}^n \mapsto \mathcal{R}$ is a convex function if $\mathrm{dom}(f) \subset \mathcal{R}^n$ is convex and:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y), \text{ for } 0 \leq t \leq 1$$

and all $x, y \in \mathrm{dom}(f)$.



- In words, functions lies below the line segment joining $f(x), f(y)$.
- **Concave function**: opposite inequality above, so that:

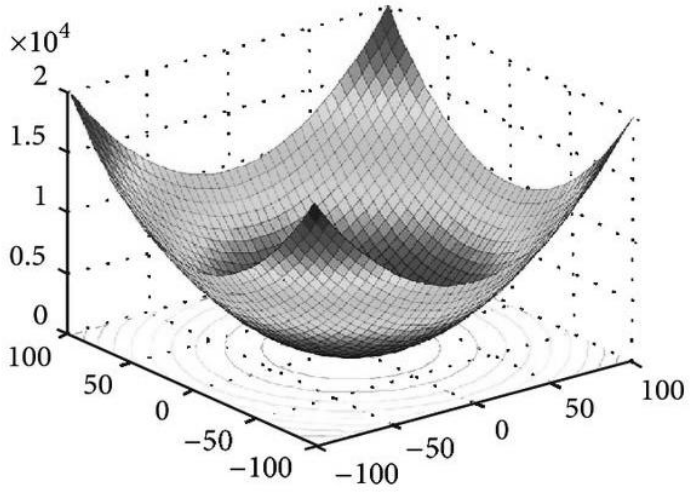
$$f \text{ is concave} \Leftrightarrow -f \text{ is convex}$$

Examples of Convex Functions

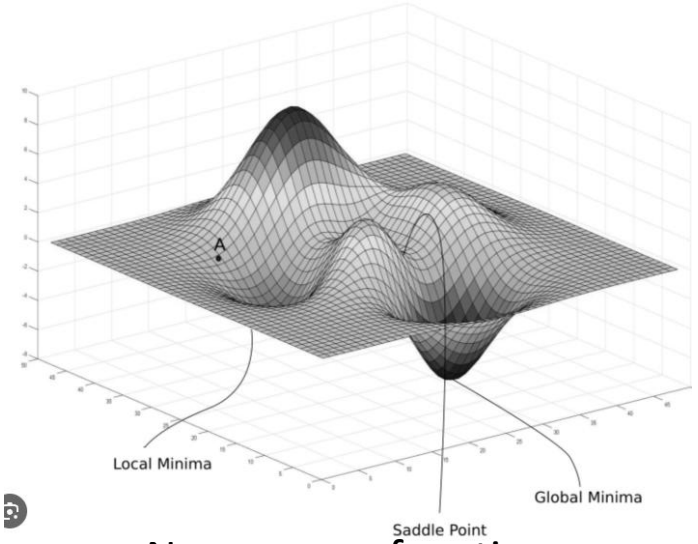
- Exponential function: $f(x) = e^{ax}$
- Power function: $f(x) = x^a$ is convex for $a \geq 1$ or $a \leq 0$ over $\mathcal{R}_+$
- **Logarithmic function** is concave over $\mathcal{R}_{++}$
- Affine function: $a^\top x + b$ is both convex and concave
- **Quadratic function**: $\frac{1}{2}x^\top Q x + b^\top x + c$ is convex provided that $Q \succeq 0$
- Least squares loss: $\|y - Ax\|_2^2$ is always convex

Why Convexity Matters?

- It gives us a much **simpler** optimization landscape.



Convex function



Non-convex function

- Every local minimum is a global minimum
- Easier to optimize

- Multiple local minima may exist
- Local minimum may not be global
- Optimization can be more challenging

Finding a local minimum is enough to find a global minimum.

Convex Optimization Problem

Optimization Problem

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0,\ i = 1, \cdots, m \\ & h_i(x) = 0,\ i = 1, \cdots, p \end{aligned}$$

- Objective function: $f_0(x)$
- Constraints: $f_i(x)$ and $h_i(x)$
- Optimal value p^* is defined as:

$$p^* = \inf\{f_0(x) | f_i(x) \leq 0, i = 1, \cdots, m, h_i(x) = 0, i = 1, \cdots, p\}$$

- If x^* is feasible and $f_0(x^*) = p^*$, x^* is an **optimal point**.
- If there is an $R > 0$ such that:

$$\begin{aligned} f_0(x) = & \inf\{f_0(z) | f_i(z) \leq 0, i = 1, \cdots, m, \\ & h_i(z) = 0, i = 1, \cdots, p, ||z - x||_2 \leq R\} \end{aligned}$$

x^* is **locally optimal point**.

Examples of Optimization Problem

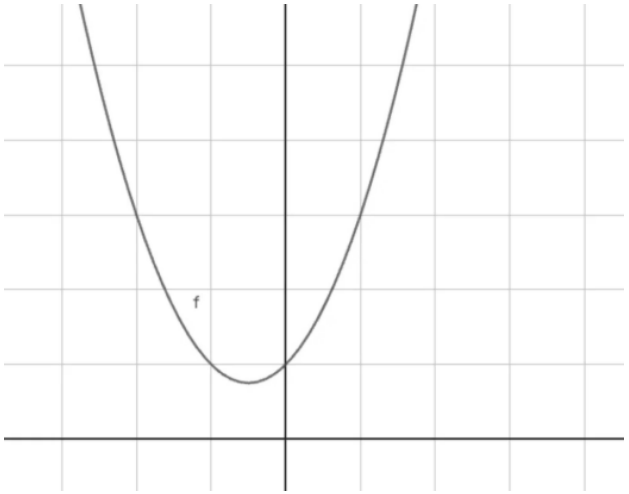
- Least-Norm Problem:

$$\min \quad ||Ax - b||_2$$

- Least-Norm Squared Problem:

$$\min \quad ||Ax - b||_2^2 = (Ax - b)^\top (Ax - b)$$

- These two problems are equivalent: the optimal points are the same



Convex Optimization Problem

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0,\ i = 1, \cdots, m \\ & a_i^\top x = b_i\ i = 1, \cdots, p \end{aligned}$$

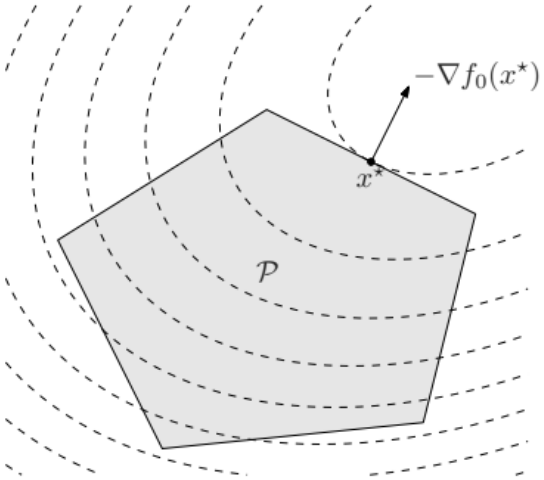
where $f_0, \cdots, f_m$ are convex functions.

- The objective function must be convex
- The inequality constraint functions must be convex
- The equality constraint function $h_i(x) = a_i^\top x - b_i$ must be affine
- Any **locally optimal point** of a convex optimization problem is also **optimal**

Examples: Quadratic Optimization Problems

$$\begin{array}{ll} \text{min} & \frac{1}{2}x^\top Px + q^\top x + r \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

where $P \in \mathbf{S}_+^n, G \in \mathcal{R}^{m\times n}, A \in \mathcal{R}^{p\times n}$



Stochastic Gradient Descent

Methods for Optimization the Objective

- Closed-form solution
- Grid search
- Random search
- Gradient descent
- Stochastic gradient descent (SGD)
- Minibatch SGD

Example: Linear Regression

- Learn a model to Project X to Y.

X				Y
1	1	1	1	8
2	1	1	1	10
1	2	1	1	11
1	1	2	1	7
1	1	1	2	12
2	2	1	1	13

- How to project? Solving the following problem:

$$XA = Y$$

- Closed-form Solutions:

$$A^* = (X^\top X)^{-1} X^\top Y$$

Optimal Point

A^*
2
3
-1
4

1	1	1	1
2	1	1	1
1	2	1	1
1	1	2	1
1	1	1	2
2	2	1	1
X			

8
10
11
7
12
13
$\hat{Y}$

8
10
11
7
12
13
Y

Example: Linear Regression

- Learn a model to Project X to Y.

X				Y
1	1	1	1	8
2	1	1	1	10
1	2	1	1	11
1	1	2	1	7
1	1	1	2	12
2	2	1	1	13

- Optimization problem:

$$\min \quad L = ||\hat{Y} - Y||_2^2 = ||AX - Y||_2^2$$

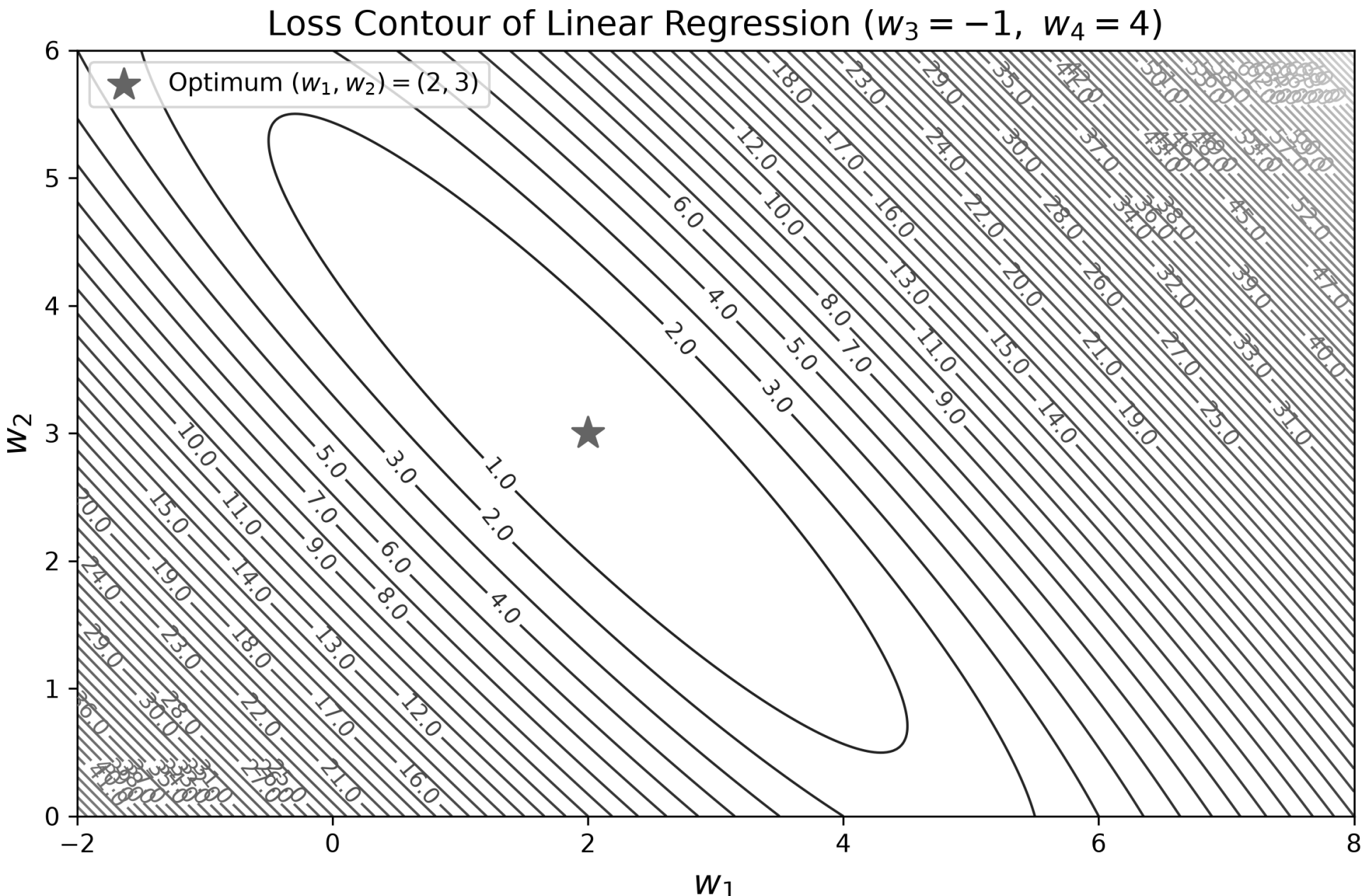
- Params: $A \in \mathcal{R}^4$

Optimal Point

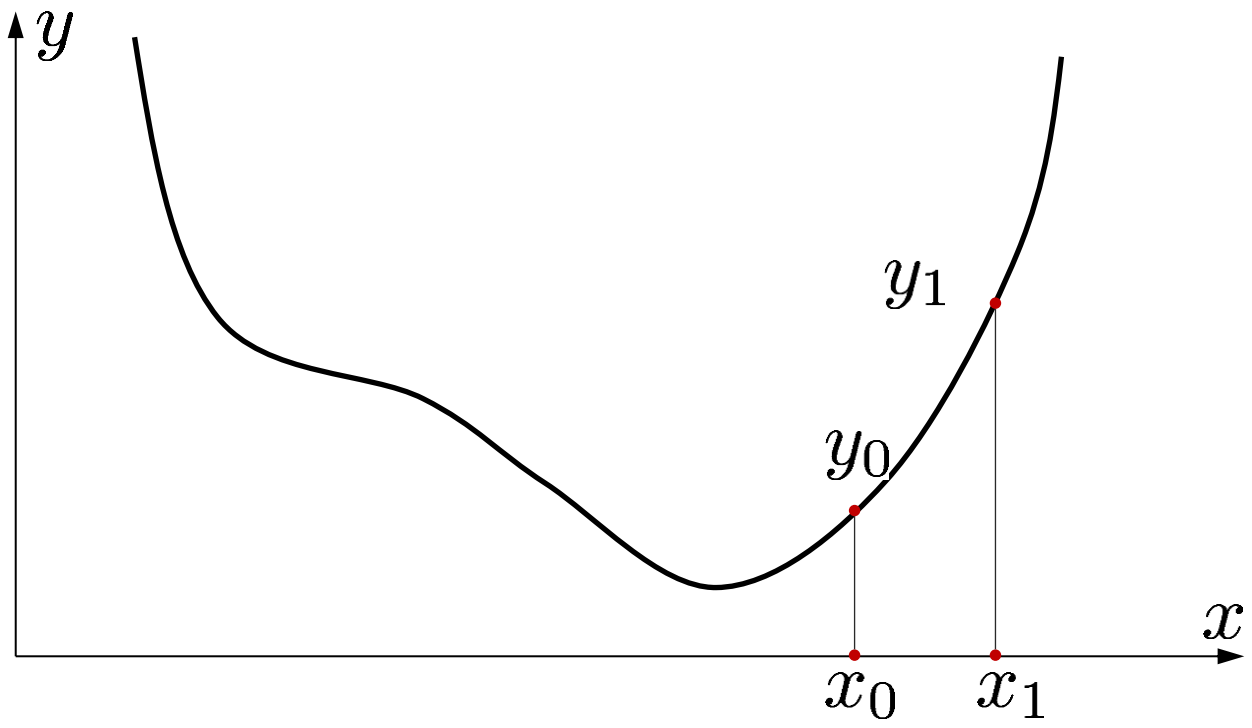
A^*
2
3
-1
4

1	1	1	1	8	8	0
2	1	1	1	10	10	0
1	2	1	1	11	11	0
1	1	2	1	7	7	0
1	1	1	2	12	12	0
2	2	1	1	13	13	0
X				$\hat{Y}$	Y	L

Loss Contour Curve

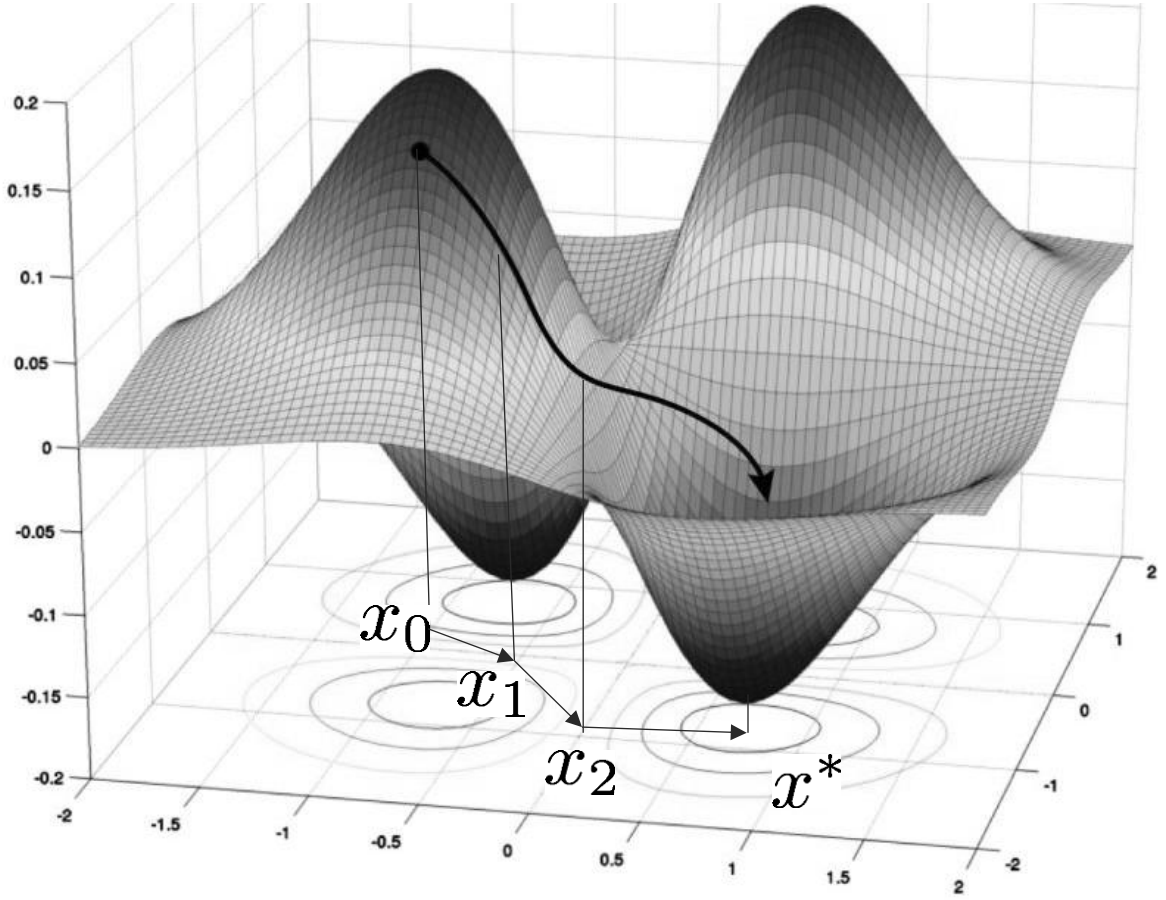


Gradient



The gradient points in the direction of the **steepest increase** of a function, while the negative gradient points in the direction of the **steepest decrease**.

Gradient



- The parameters x are updated **within the feasible set**.

Gradient

- Gradient of objective with respect to parameters:

$$
\begin{aligned}
L &= ||XA - Y||_2^2 \\
&= (XA - Y)^\top (XA - Y) \\
&= tr(XAA^\top X^\top) - 2tr(XAY^\top) + tr(YY^\top)
\end{aligned}
$$

$$
\frac{\partial L(X)}{\partial A} = 2X^\top XA - 2X^\top Y
$$

- How to calculate the gradient?
  - Matrix Cookbook

Gradient

- Derivatives of Matrices, Vectors:

$$\frac{\partial x^\top a}{\partial x} = a$$

$$\frac{\partial a^\top X b}{\partial X} = ab^\top$$

$$\frac{\partial a^\top X^\top b}{\partial X} = ba^\top$$

$$\frac{\partial b^\top X^\top X c}{\partial X} = X(bc^\top + cb^\top)$$

$$\frac{\partial x^\top B x}{\partial x} = (B + B^\top)x$$

Gradient

- Derivatives of Trace:

$$\frac{\partial tr(X)}{\partial X} = I$$

$$\frac{\partial tr(XA)}{\partial X} = A^\top$$

$$\frac{\partial tr(AXB)}{\partial X} = A^\top B^\top$$

$$\frac{\partial tr(AX^\top B)}{\partial X} = BA$$

$$\frac{\partial tr(X^\top A)}{\partial X} = A$$

$$\frac{\partial tr(A^\top X)}{\partial X} = A$$

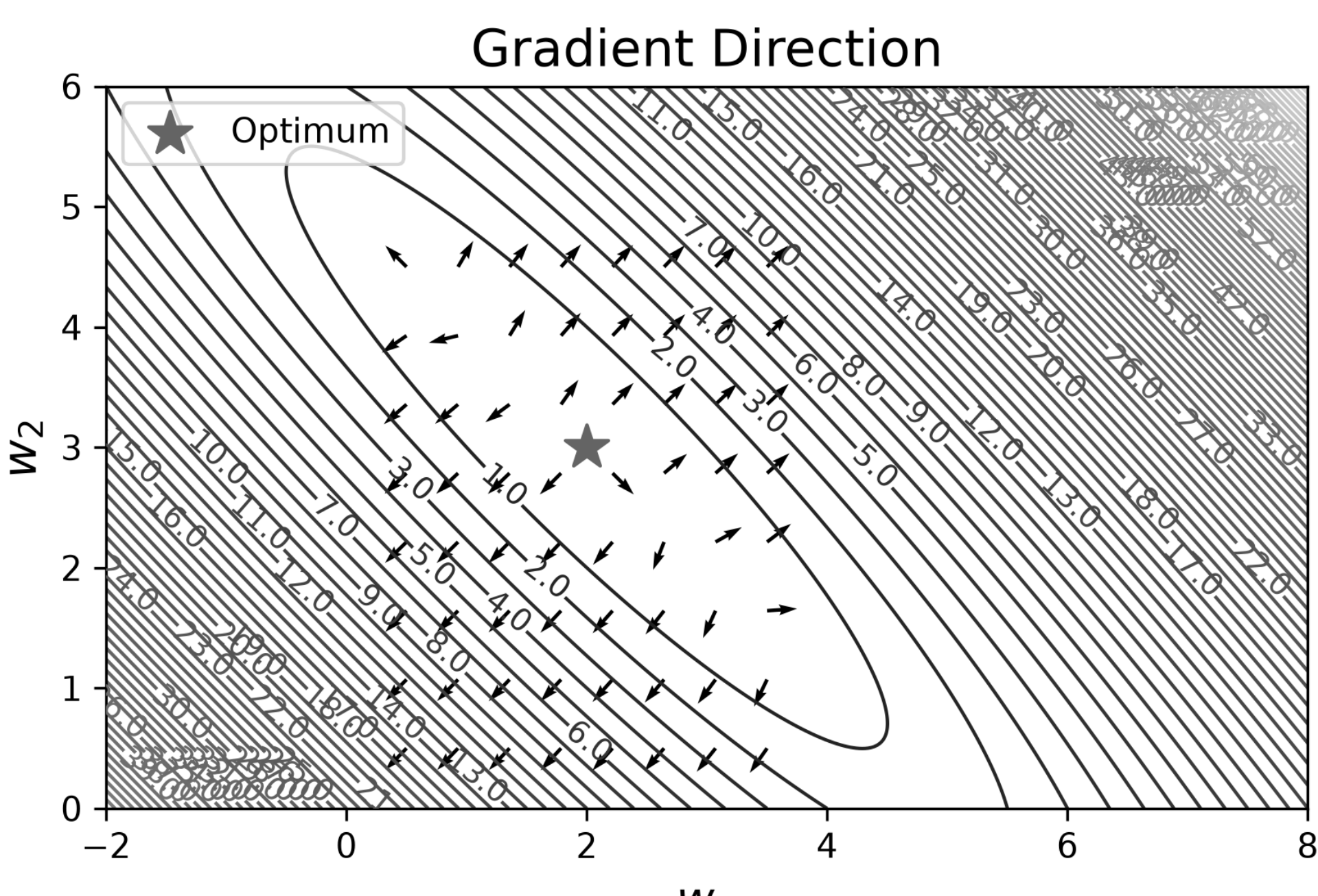
$$\frac{\partial tr(X^2)}{\partial X} = 2X^\top$$

$$\frac{\partial tr(X^\top BX)}{\partial X} = (B + B^\top)X$$

$$\frac{\partial tr(BX^\top X)}{\partial X} = (B + B^\top)X$$

$$\frac{\partial tr(X^\top XB)}{\partial X} = (B + B^\top)X$$

Gradient



Gradient Descent

- How can we find the **minimum** of an objective function using its gradient? Gradient descent:

$$A_t = A_{t-1} - \eta \frac{\partial L}{\partial A}$$

  - Iteration: t
  - Learning rate: η
- Move step by step toward the optimum along the negative gradient direction.

Gradient Descent

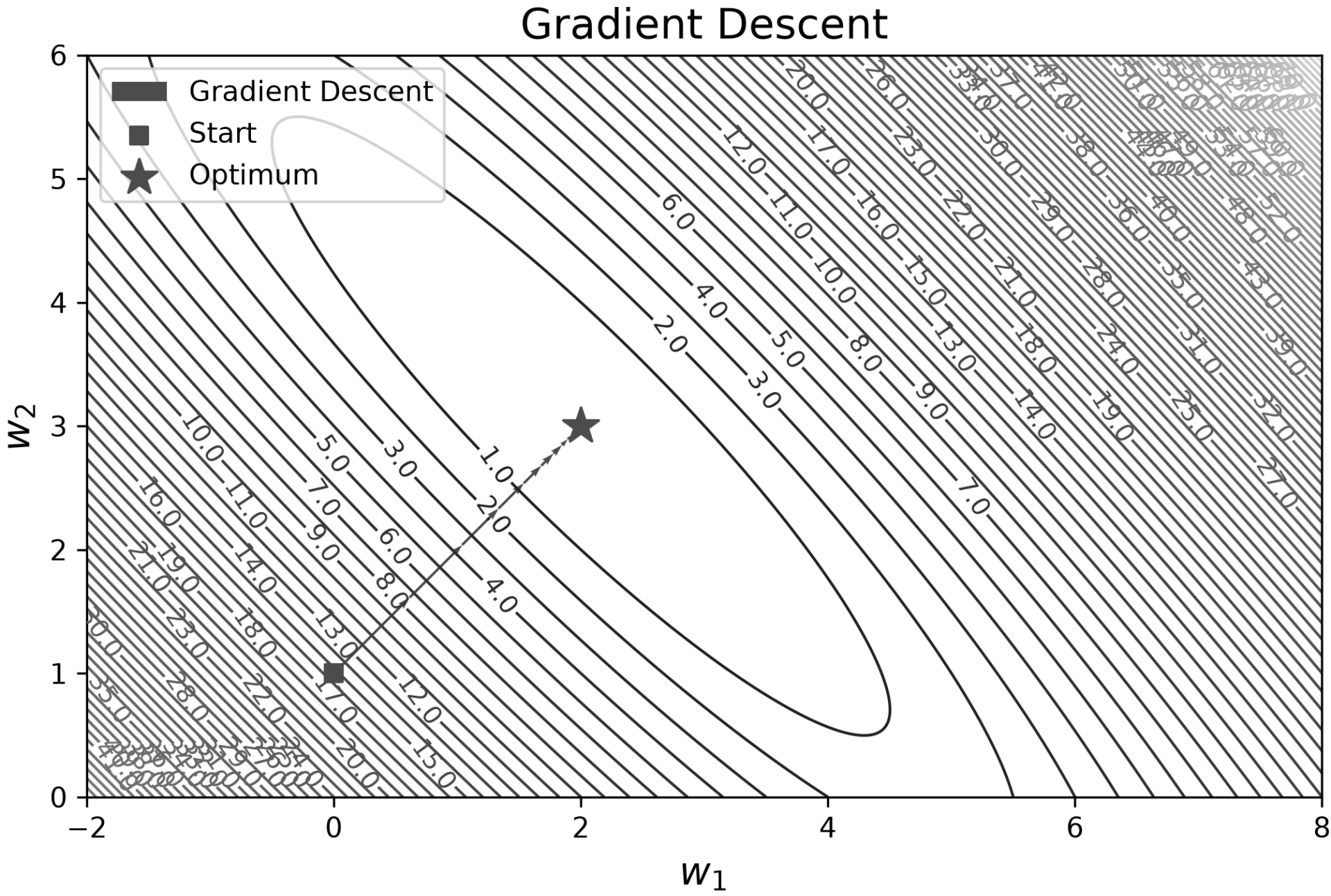
- Gradient descent algorithm:

Algorithm 1 Gradient descent algorithm.

Input: Training set X and labels $\boldsymbol{Y}$;
Output: The learned models;
INIT: Initialize model parameter A_0, learning rate η.

1: **for** $t = 1 \rightarrow Num\_Iteration$ **do**
2: Calculate gradient $\frac{\partial L}{\partial A}$.
3: Update parameters: $A_t = A_{t-1} - \eta \frac{\partial L}{\partial A_{t-1}}$.

Gradient Descent



Stochastic Gradient Descent

- Gradient calculation requires all data points, which is time consuming:

Algorithm 1 Gradient descent algorithm.

Input: Training set X and labels $\boldsymbol{Y}$;
Output: The learned models;
INIT: Initialize model parameter A_0, learning rate η.

1: **for** $t = 1 \rightarrow Num\_Iteration$ **do**
2: Calculate gradient $\frac{\partial L}{\partial A}$.
3: Update parameters: $A_t = A_{t-1} - \eta \frac{\partial L}{\partial A_{t-1}}$.

$$\frac{\partial L(X)}{\partial A} = 2X^\top XA - 2X^\top Y$$

Stochastic Gradient Descent

- SGD: employ the gradient calculated by a **single data point** as estimation of full gradient.

$$\frac{\partial L(x_i)}{\partial A} = 2x_i^\top x_i A - 2x_i^\top Y$$

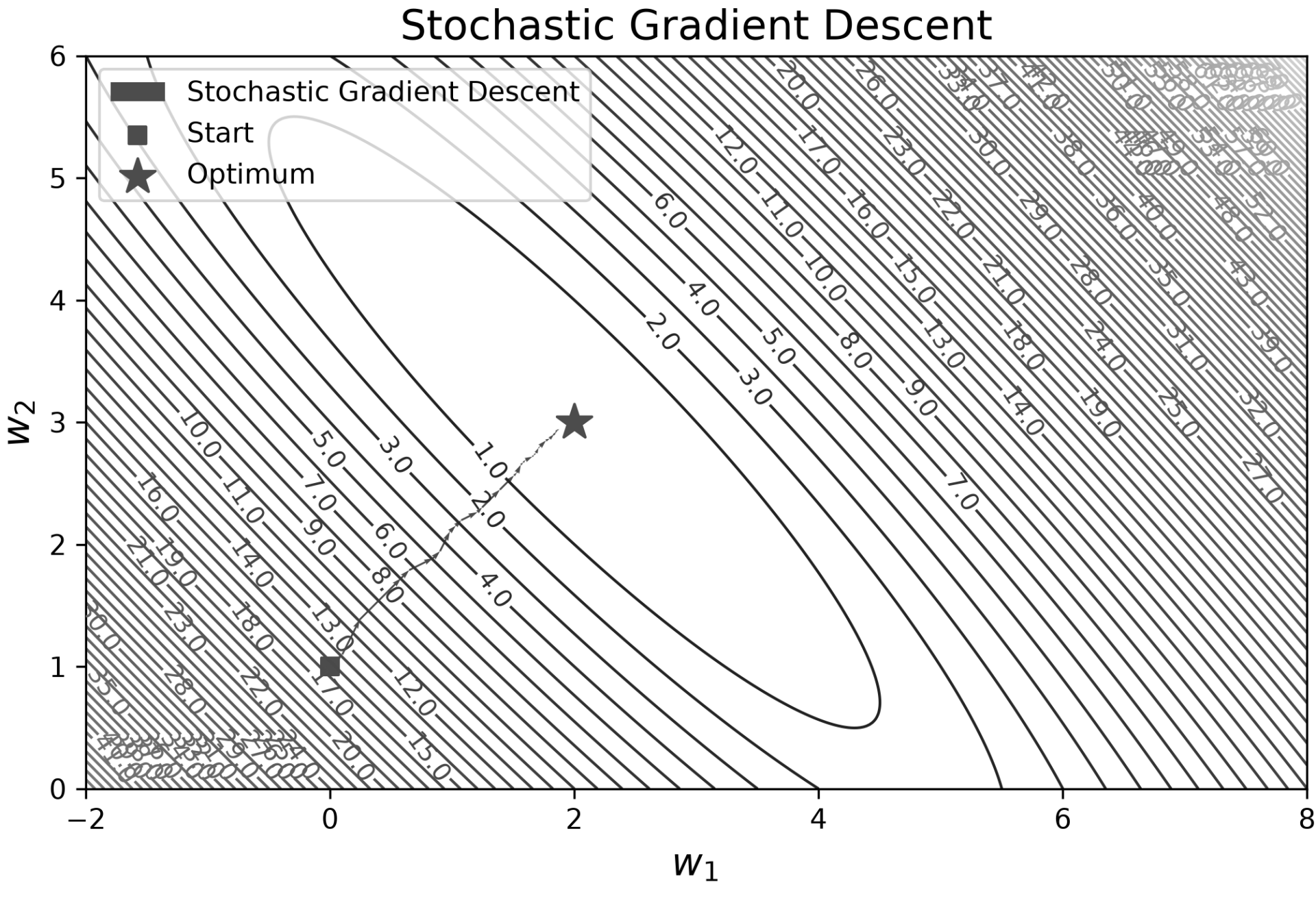
- Data point is sampled randomly.

Algorithm 2 Stochastic gradient descent algorithm.

Input: Training set X and labels $\boldsymbol{Y}$;
Output: The learned models;
INIT: Initialize model parameter A_0, learning rate η.

1: **for** $t = 1 \rightarrow Num\_Iteration$ **do**
2: Sample a data point x_i randomly.
3: Calculate gradient $\frac{\partial L(x_i)}{\partial A}$.
4: Update parameters: $A_t = A_{t-1} - \eta \frac{\partial L(x_i)}{\partial A_{t-1}}$.

Stochastic Gradient Descent



GD vs SGD

	Gradient Descent	**Stochastic Gradient Descent**
Gradient	Full dataset	One sample
Computation	Expensive	Cheap
Direction	Accurate	Noisy
Update frequency	Low	High
Memory	High	Low
Typical use	Small datasets	Large-scale learning

- Gradient descent: accurate but expensive
- SGD: noisy but efficient

What if we use a small batch of sample?

Why does SGD work?

- The full gradient is:

$$\frac{\partial L(X)}{\partial A} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial L(x_i)}{\partial A}$$

- SGD randomly samples a data point and uses:

$$g_i = \frac{\partial L(x_i)}{\partial A}$$

- If the data point is sampled uniformly, we have:

$$\mathbb{E}_{\xi : x_i \sim X} [\frac{\partial L(x_i)}{\partial A}] = \frac{\partial L(X)}{\partial A}$$

- Therefore:

The stochastic gradient is an **unbiased** estimator of the full gradient.

Mini-batch SGD: Accuracy or Efficiency?

- Mini-batch SGD: Accuracy or Efficiency?
- Instead of using **one sample (SGD)** or **the entire dataset (GD)**, we use a small batch of samples:

$$g_B = \frac{1}{B} \sum_{i \in \mathcal{B}_i} \frac{\partial L(x_i)}{\partial A}$$

Small batch	Large batch
More stochastic noise	More accurate gradient estimation
Cheaper update	More computation per update
More frequent updates	Less stochasticity

Mini-batch SGD provides a practical trade-off between gradient quality and computational efficiency.

Summary

- Learning = optimization of model parameters
- **Convexity** provides useful structure for optimization
- **Gradient** gives the steepest descent direction
- SGD improves scalability through stochastic estimation
- Mini-batch SGD balances efficiency and gradient quality

Optimization is the engine that turns a model into a learned model.

Thanks