Optimization:
Stochastic Gradient Descent

Recap on Optimization

GD: simply follow the negative of the gradient

AdaGrad — each dim has its own learning rate, adapted based on the cumulation of the past squared derivatives — help make progress along all axises.

GD w/ momentum: think about gradient as “acceleration”, “velocity” is the exponential average of “acceleration” — help power through very flat region

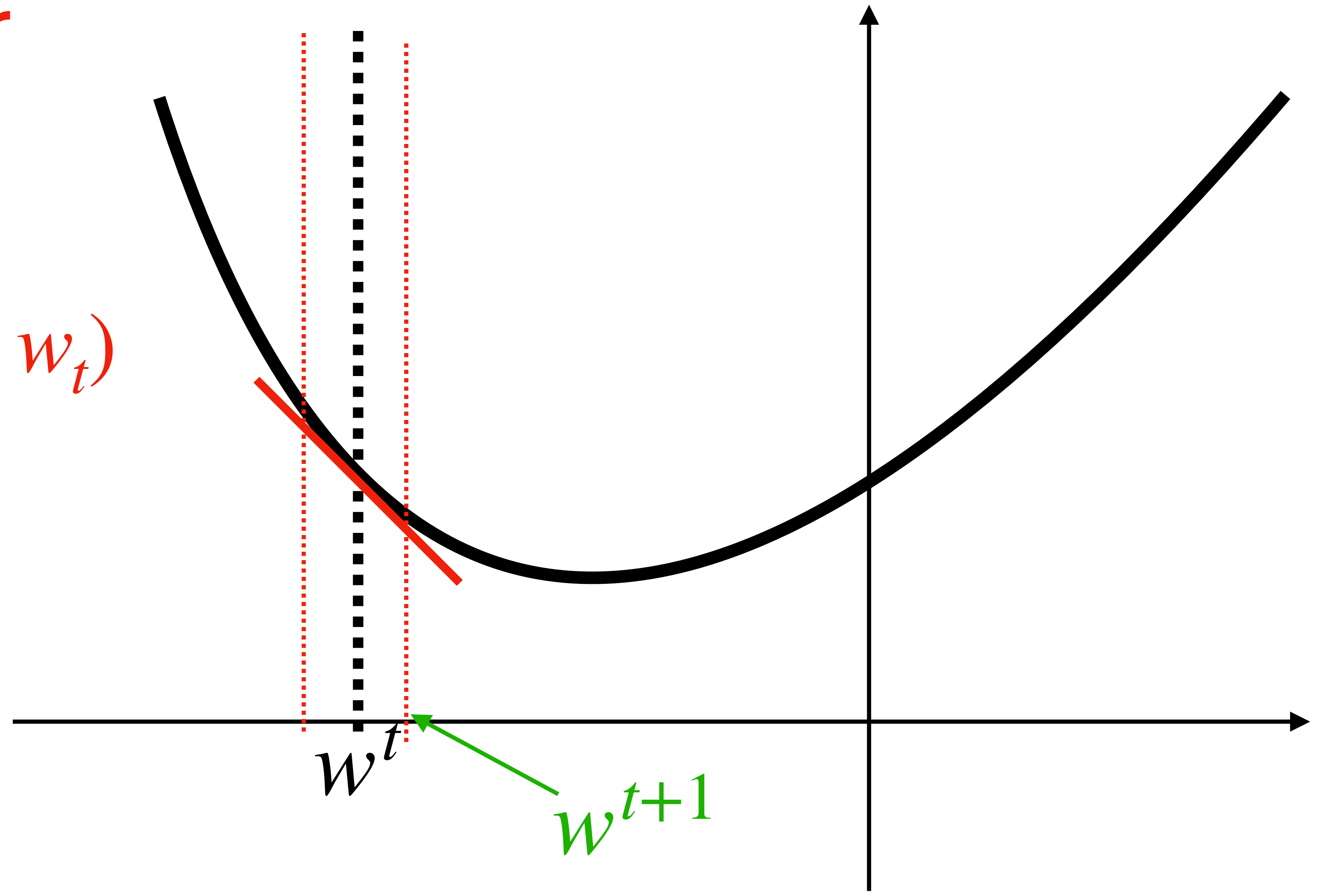
Recap on Gradient Descent

Gradient descent minimizes $\ell(w)$ iteratively:

$$w^{t+1} = w^t - \eta \nabla \ell(w) \,|_{w=w_t}$$

First-order Taylor expansion at w_t:

$\ell(w_t) + \nabla \ell(w_t)^\top (w - w_t)$



Objective

Understand the Stochastic GD algorithm, its convergence, and its benefits over GD

Outline for Today

1. Stochastic Gradient Descent

2. Mini-Batch SGD

Loss minimization in ML

In ML, the loss we minimize typically has some special form, e.g., in LR:

$$\ell(w) = \frac{1}{n}\sum_{i=1}^{n} \ln\left(1 + \exp(-y_i(w^\top x_i))\right)$$

Avg over n data points, i.e., $\sum_{i=1}^{n} \ell(x_i, y_i; w)/n$

To compute the gradient $\nabla \ell(w)$, we need to enumerate all n training data points

Can be very slow!

Stochastic GD to rescue

In ML, the loss we minimize typically has some special form, e.g., in LR:

$$\ell(w) = \frac{1}{n}\sum_{i=1}^{n} \ln\left(1 + \exp(-y_i(w^\top x_i))\right)$$

Avg over n data points, i.e., $\sum_{i=1}^{n} \ell(x_i, y_i; w)/n$

Idea: randomly sample a data point (x, y), use $\nabla \ell(x, y; w)$ to replace $\nabla \ell(w)$

Stochastic GD

Goal: minimize $\ell(w) = \frac{1}{n}\sum_{i=1}^{n} \ell(x_i, y_i; w)$

Initialize $w^0 \in \mathbb{R}^d$ randomly

Iterate until convergence:

1. Randomly sample a point (x_i, y_i) from the n data points
2. Compute noisy gradient $\tilde{g}^t = \nabla \ell(x_i, y_i; w)\,|_{w=w^t}$
3. Update (GD): $w^{t+1} = w^t - \eta \tilde{g}^t$

Intuition of why Stochastic GD can work

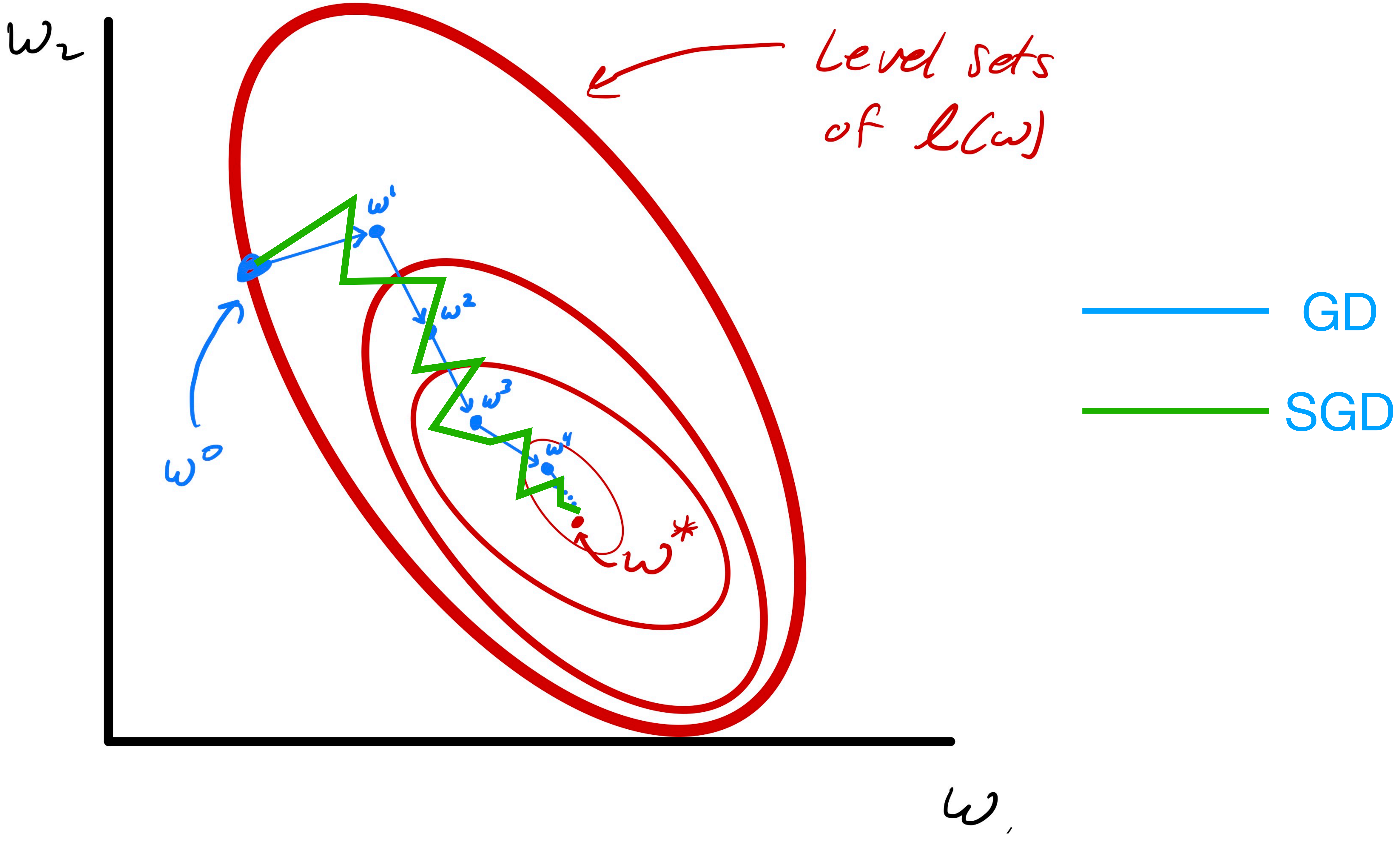
Claim: the random noisy gradient is an **unbiased** estimate of the true gradient

Note the point (x_i, y_i) is uniformly random sampled from n data points, we have:

$$\mathbb{E}\, \nabla \ell(x_i, y_i; w)$$

$$= \frac{1}{n}\sum_{i=1}^{n} \nabla \ell(x_i, y_i; w) = \nabla \left[\underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell(x_i, y_i; w)}_{\ell(w)} \right] = \nabla \ell(w)$$

Intuition of why Stochastic GD can work



Theoretical Guarantee of SGD

(Informal theorem and no proof)

Consider a function w/ β-Lipschitz gradient, i.e., $\|\nabla \ell(w) - \nabla \ell(w')\|_2 \leq \beta \|w - w'\|_2$.

Assume for all iteration t, $\tilde{g}^t$ is unbiased, and $\mathbb{E}\|\tilde{g}^t\|_2^2 \leq \sigma^2$,

then with $\eta = \sqrt{\frac{1}{\beta \sigma^2 T}}$, SGD satisfies:

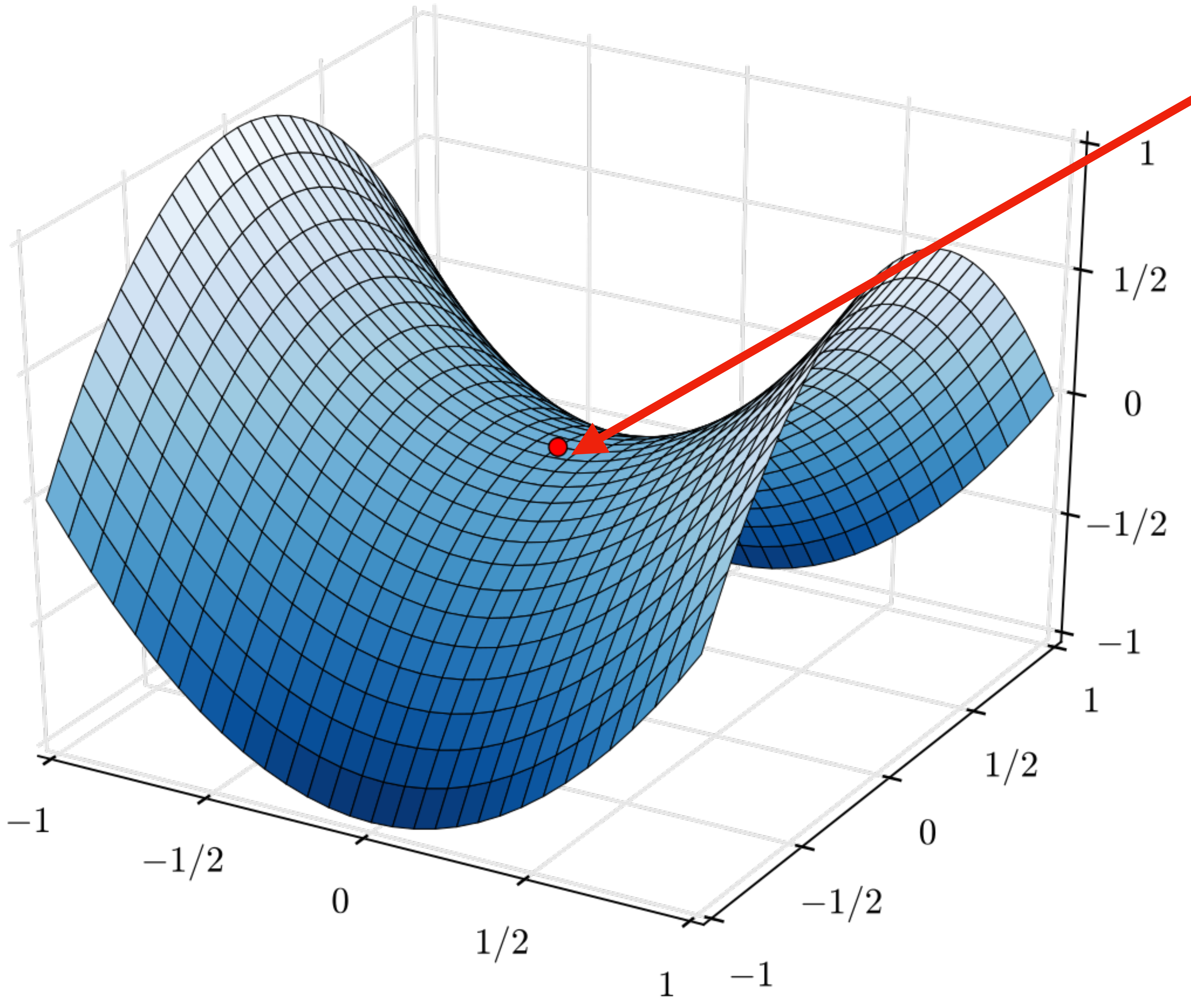
$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} \|\nabla \ell(w^t)\|_2\right] \leq 2\sqrt{\frac{\beta \sigma^2}{T}}$$

Larger variance can make SGD slower

Empirical Benefit of SGD on Non-Convex Optimization

e.g., saddle point $\ell(x, y) = x^2 - y^2$

GD can get stuck at the saddle point



Using a noisy gradient, we can escape this saddle point

Outline for Today

1. Stochastic Gradient Descent

2. Mini-Batch SGD

Reduce the variance via mini-batch

SGD's convergence typically depend on the second moment of $\tilde{g}$, i.e., $\mathbb{E}||\tilde{g}||_2^2$

Larger variance implies slower convergence

Solution: we can reduce the variance using a **mini-batch**

Reduce the variance via mini-batch

Randomly sample m data points from the dataset, denoted as $\mathscr{B}$

$$\tilde{g} = \frac{1}{m} \sum_{(x,y) \in \mathscr{B}} \nabla \ell(w; x, y)$$

Averaging over m points reduce the variance

Claim: $\tilde{g}$ is still unbiased, and variance of $\tilde{g}$ decreases as m increases

Mini-batch SGD

Goal: minimize $\ell(w) = \frac{1}{n}\sum_{i=1}^{n} \ell(x_i, y_i; w)$

Initialize $w^0 \in \mathbb{R}^d$ randomly

Iterate until convergence:

1. Randomly sample m points, denoted as mini-batch $\mathscr{B}$

2. Compute gradient $\tilde{g} = \frac{1}{m} \sum_{(x,y)\in\mathscr{B}} \nabla \ell(w; x_i, y_i)\,|_{w=w^t}$

3. Update (GD): $w^{t+1} = w^t - \eta \tilde{g}^t$

Batch size m & learning rate η are very important hyper-parameters!

Closing Remarks on Optimization

1. Min-batch SGD is the foundation of today's deep learning

2. Can use Stochastic gradients together w/ AdaGrad, GD w/ Momemtum, and Adam