10-301/601: Introduction to Machine Learning Lecture 8 – Optimization for Machine Learning

Henry Chai

5/31/23

Front Matter

- Announcements:
  - PA2 released 5/25, due 6/01 at 11:59 PM
  - No new programming assignment this week!
- Recommended Readings:
  - None

Recall: Minimizing the Squared Error

$$\ell_{\mathcal{D}}(\boldsymbol{w}) = \sum_{n=1}^{N} \left(\boldsymbol{w}^T \boldsymbol{x}^{(n)} - y^{(n)}\right)^2 = \sum_{n=1}^{N} \left(\boldsymbol{x}^{(n)^T} \boldsymbol{w} - y^{(n)}\right)^2$$

$$= \|X\boldsymbol{w} - \boldsymbol{y}\|_2^2 \text{ where } \|\boldsymbol{z}\|_2 = \sqrt{\sum_{d=1}^{D} z_d^2} = \sqrt{\boldsymbol{z}^T \boldsymbol{z}}$$

$$= (X\boldsymbol{w} - \boldsymbol{y})^T (X\boldsymbol{w} - \boldsymbol{y})$$

$$= (\boldsymbol{w}^T X^T X \boldsymbol{w} - 2\boldsymbol{w}^T X^T \boldsymbol{y} + \boldsymbol{y}^T \boldsymbol{y})$$

$$\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\widehat{\boldsymbol{w}}) = (2X^T X \widehat{\boldsymbol{w}} - 2X^T \boldsymbol{y}) = 0$$

$$\rightarrow X^T X \widehat{\boldsymbol{w}} = X^T \boldsymbol{y}$$

$$\rightarrow \widehat{\boldsymbol{w}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

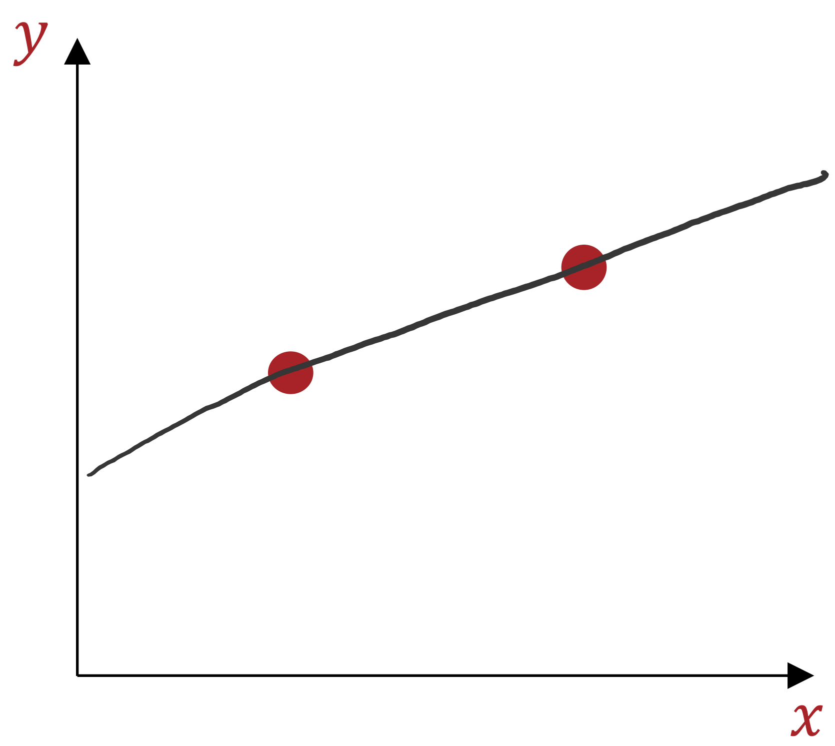
$$\hat{\boldsymbol{w}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

Recall: Closed Form Solution

1. Is $X^T X$ invertible?
   - When $N \gg D + 1$, $X^T X$ is (almost always) full rank and therefore, invertible!
   - If $X^T X$ is not invertible (occurs when one of the features is a linear combination of the others), what does that imply about our problem?
2. If so, how computationally expensive is inverting $X^T X$?
   - $X^T X \in \mathbb{R}^{D+1 \times D+1}$ so inverting $X^T X$ takes $O(D^3)$ time…
     - Computing $X^T X$ takes $O(ND^2)$ time
   - What alternative optimization method(s) can we use to minimize the mean squared error?

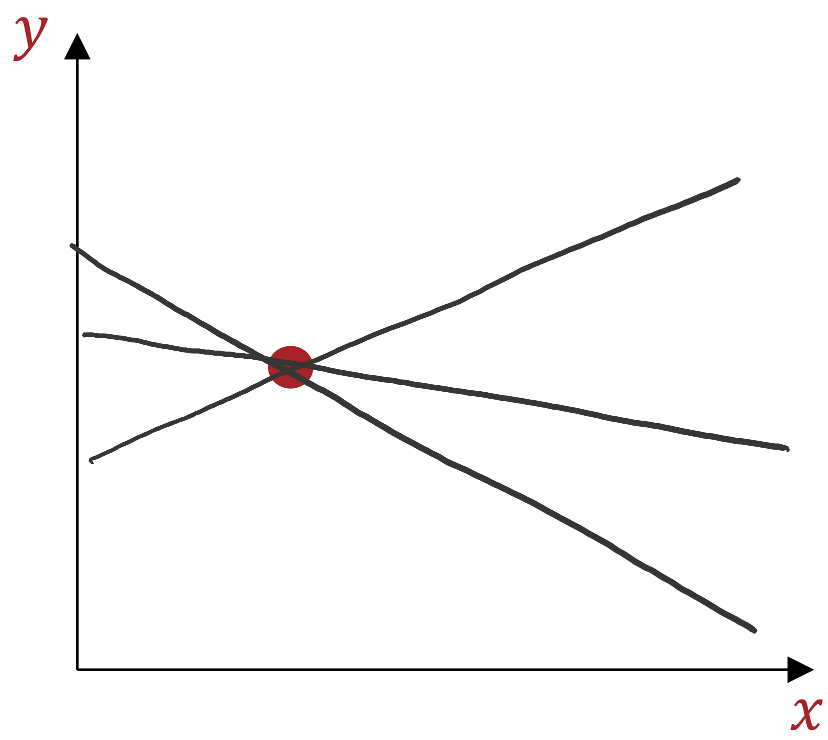
Linear Regression: Uniqueness

- Consider a 1D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?



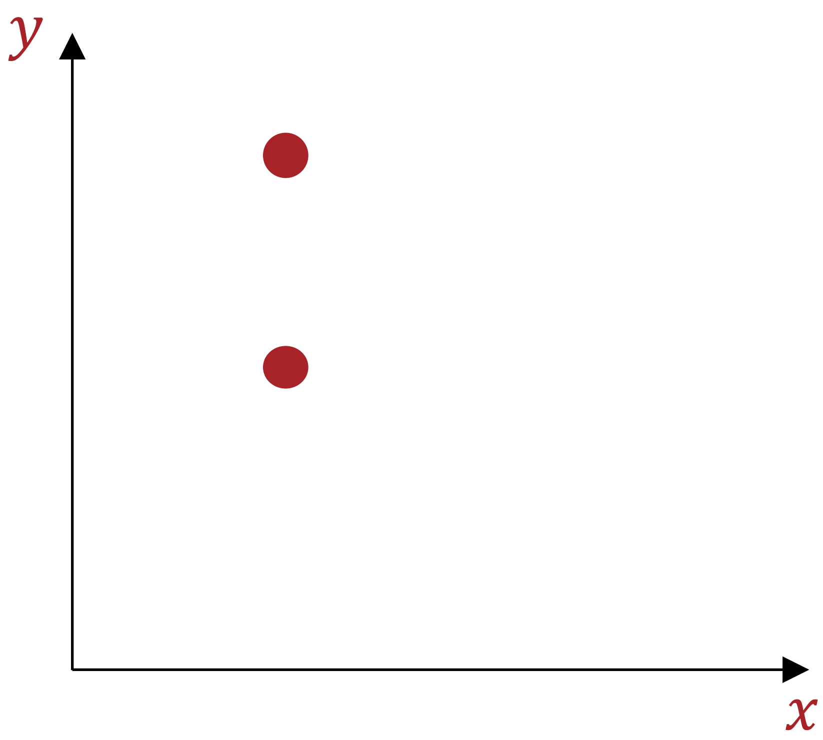
Linear Regression: Uniqueness

- Consider a 1D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?

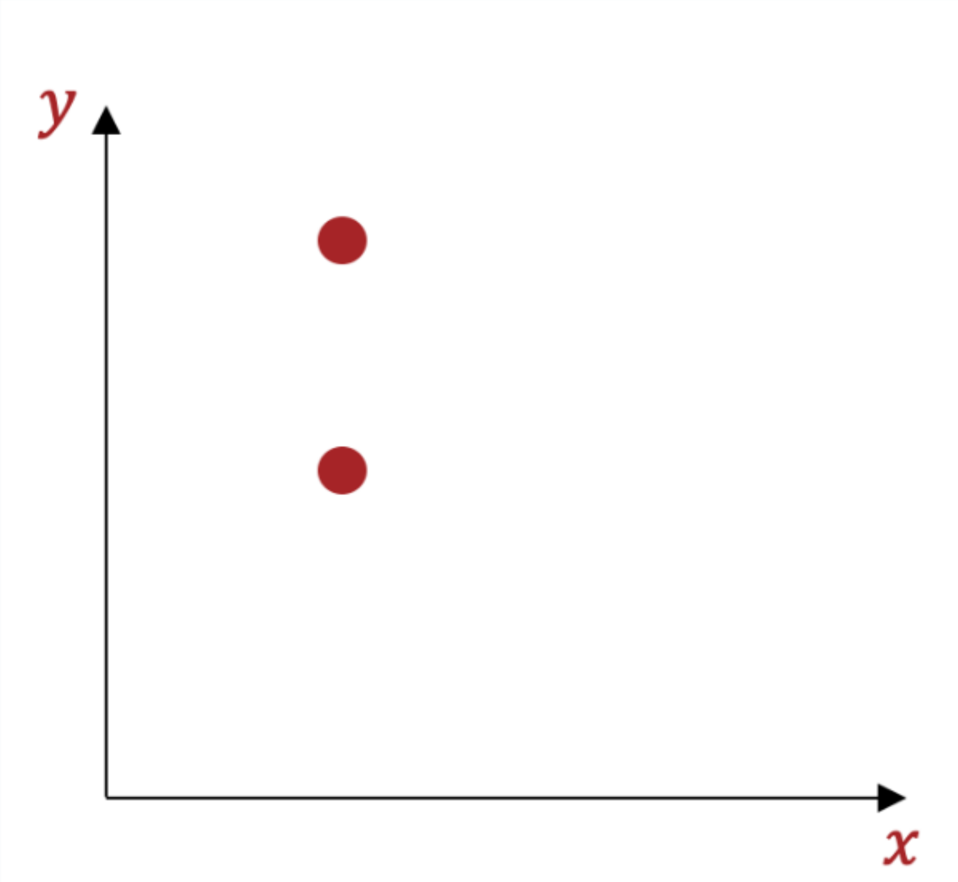


Linear Regression: Uniqueness

- Consider a 1D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?



How many solutions optimal solutions are there for the given dataset?



0

1

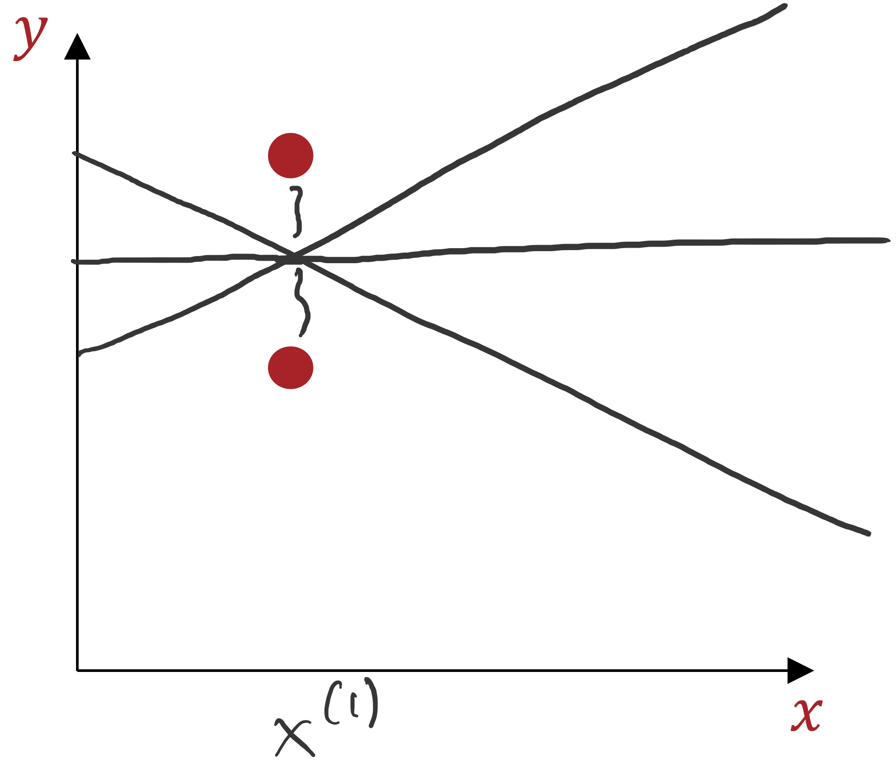
2

∞

if minimizing the absolute error there are even more solutions!

Linear Regression: Uniqueness

- Consider a 1D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?

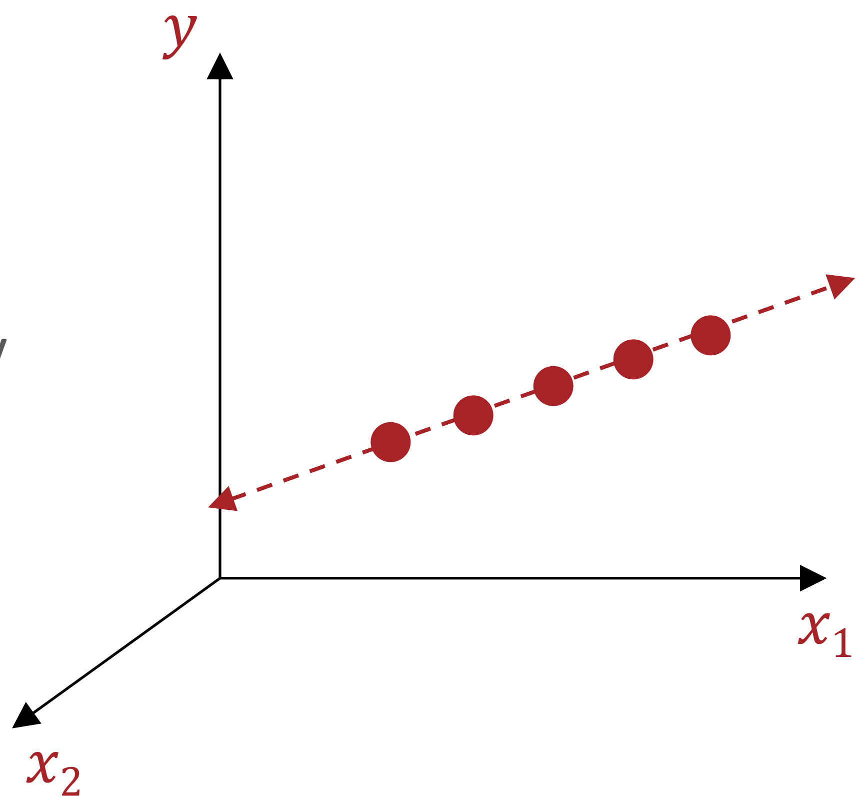


$x = x^{(1)}$

$y = wx + w_0$

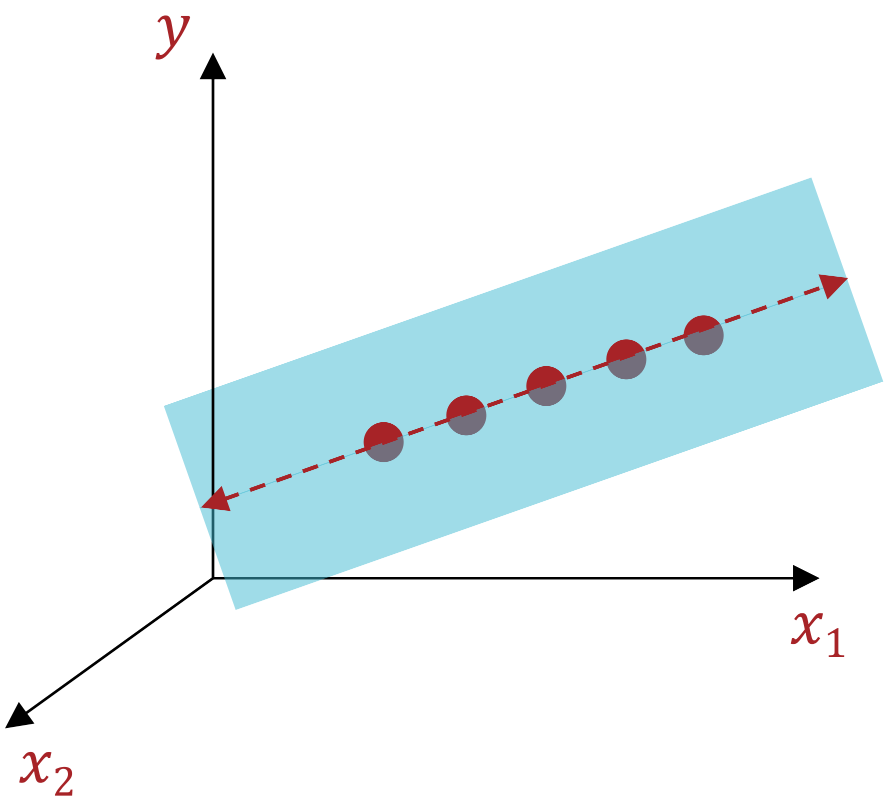
Linear Regression: Uniqueness

- Consider a 2D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?



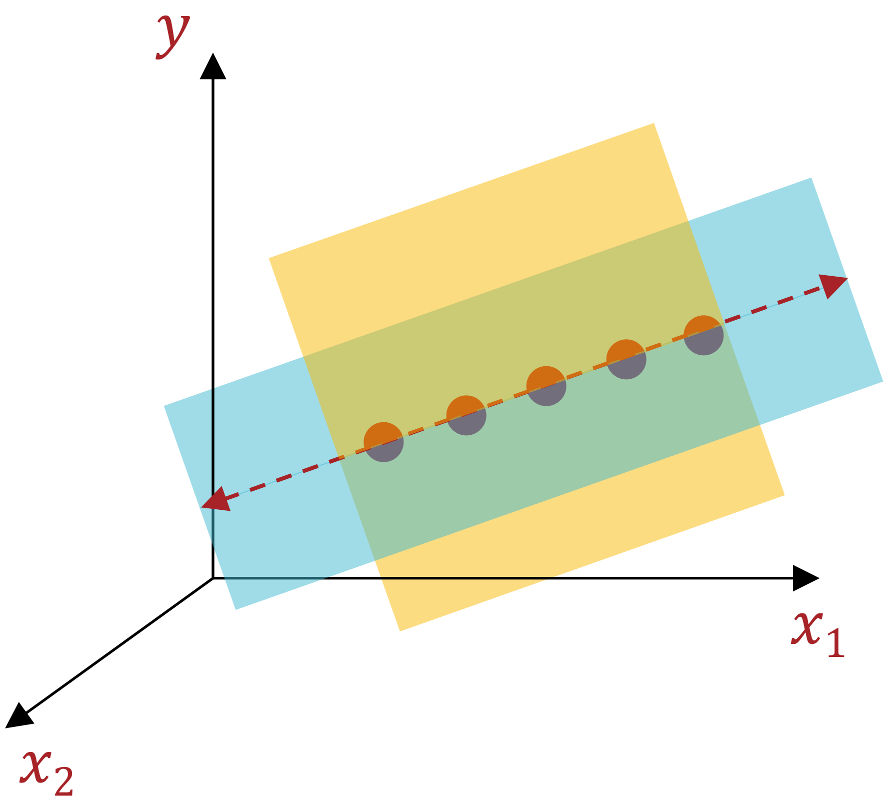
Linear Regression: Uniqueness

- Consider a 2D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?



Linear Regression: Uniqueness

- Consider a 2D linear regression model trained to minimize the mean squared error: how many optimal solutions (i.e., sets of parameters θ) are there for the given dataset?



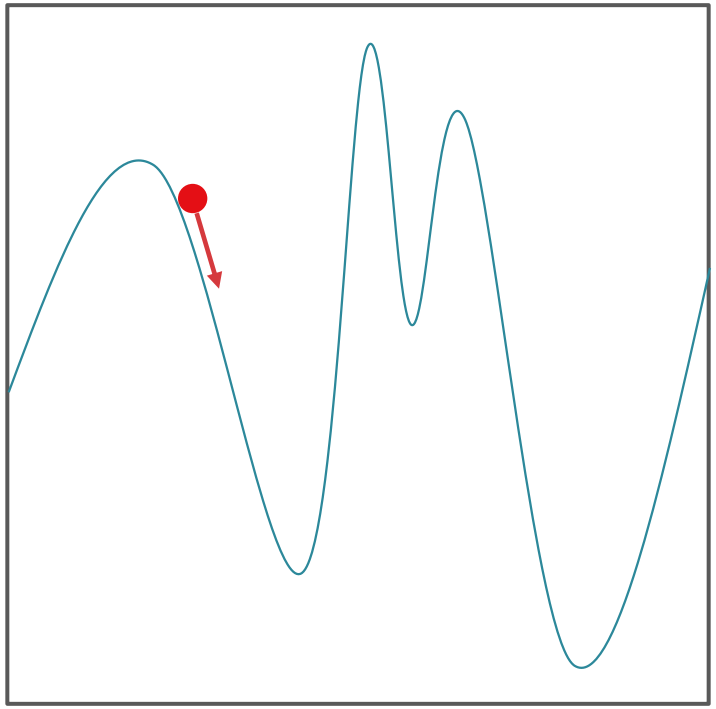
$$\hat{\boldsymbol{w}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

Closed Form Solution

1. Is $X^T X$ invertible?
   - When $N \gg D + 1$, $X^T X$ is (almost always) full rank and therefore, invertible!
   - If $X^T X$ is not invertible (occurs when at least one of the features is a linear combination of the others) then there are infinitely many solutions.
2. If so, how computationally expensive is inverting $X^T X$?
   - $X^T X \in \mathbb{R}^{D+1 \times D+1}$ so inverting $X^T X$ takes $O(D^3)$ time…
     - Computing $X^T X$ takes $O(ND^2)$ time
   - Can use gradient descent to (potentially) speed things up when N and D are large!

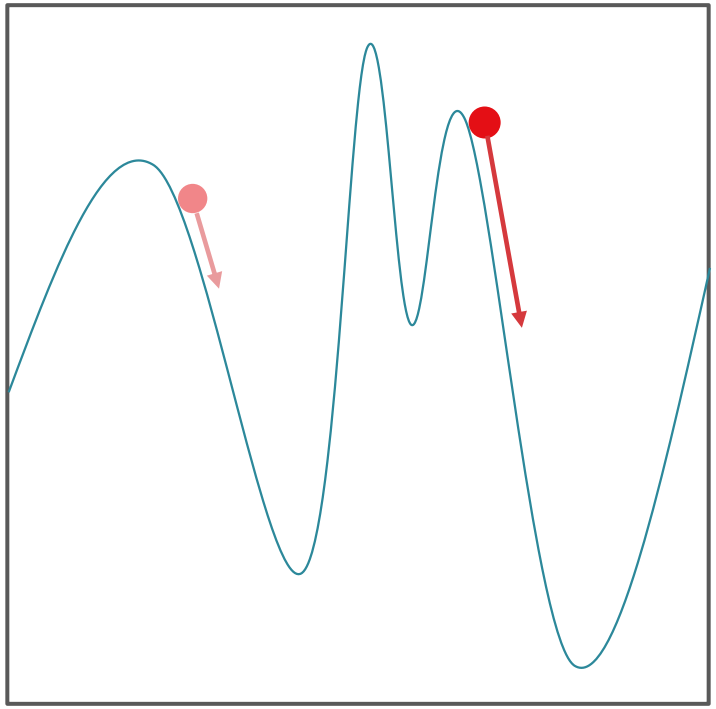
Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



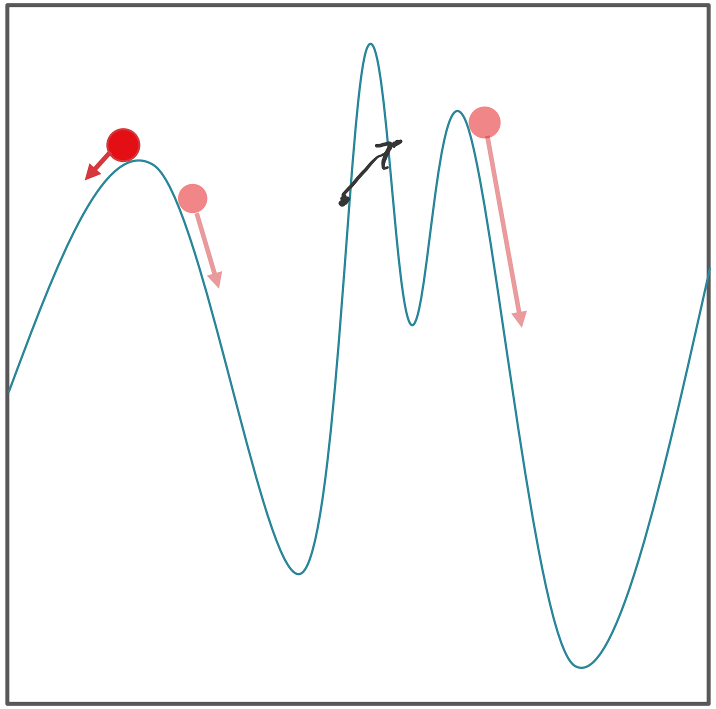
Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



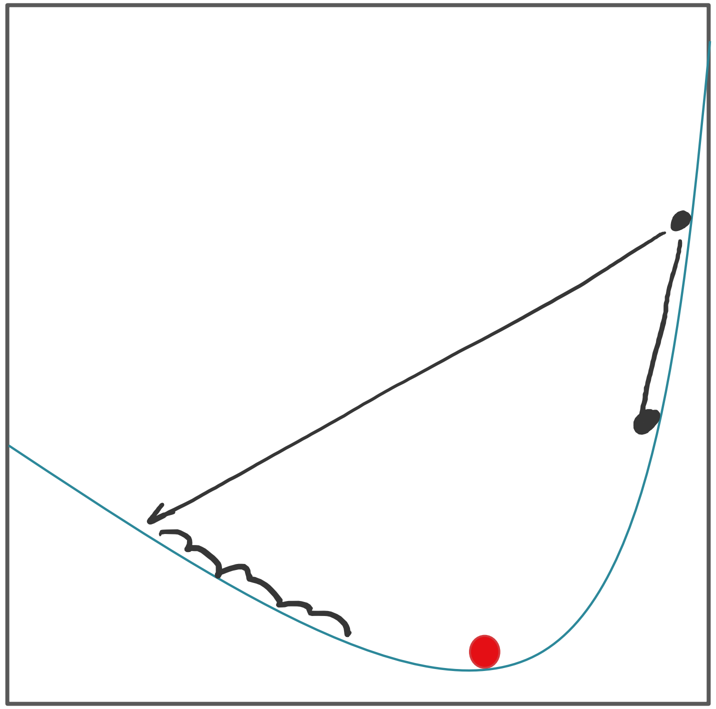
Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



- Good news: the squared error is convex!

Recall: Minimizing the Squared Error

$$\ell_{\mathcal{D}}(\boldsymbol{w}) = \sum_{n=1}^{N} \left(\boldsymbol{w}^T \boldsymbol{x}^{(n)} - y^{(n)}\right)^2 = \sum_{n=1}^{N} \left(\boldsymbol{x}^{(n)^T} \boldsymbol{w} - y^{(n)}\right)^2$$

$$= \|X\boldsymbol{w} - \boldsymbol{y}\|_2^2 \text{ where } \|\boldsymbol{z}\|_2 = \sqrt{\sum_{d=1}^{D} z_d^2} = \sqrt{\boldsymbol{z}^T \boldsymbol{z}}$$

$$= (X\boldsymbol{w} - \boldsymbol{y})^T (X\boldsymbol{w} - \boldsymbol{y})$$

$$= (\boldsymbol{w}^T X^T X \boldsymbol{w} - 2\boldsymbol{w}^T X^T \boldsymbol{y} + \boldsymbol{y}^T \boldsymbol{y})$$

$$\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}) = (2X^T X \boldsymbol{w} - 2X^T \boldsymbol{y})$$

$$H_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}) = 2X^T X$$

$H_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w})$ is positive semi-definite

Gradient Descent: Step Direction

- Suppose the current weight vector is $\boldsymbol{w}^{(t)}$
- Move some distance, η, in the "most downhill" direction, $\hat{\boldsymbol{v}}$:

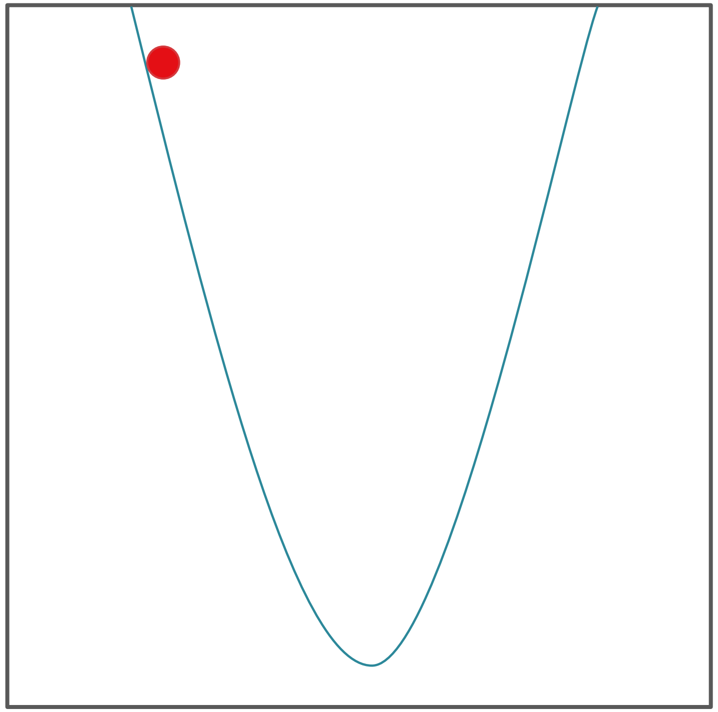
$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \eta\hat{\boldsymbol{v}}$$

- The gradient points in the direction of steepest *increase* …
- … so $\hat{\boldsymbol{v}}$ is a unit vector pointing in the opposite direction:

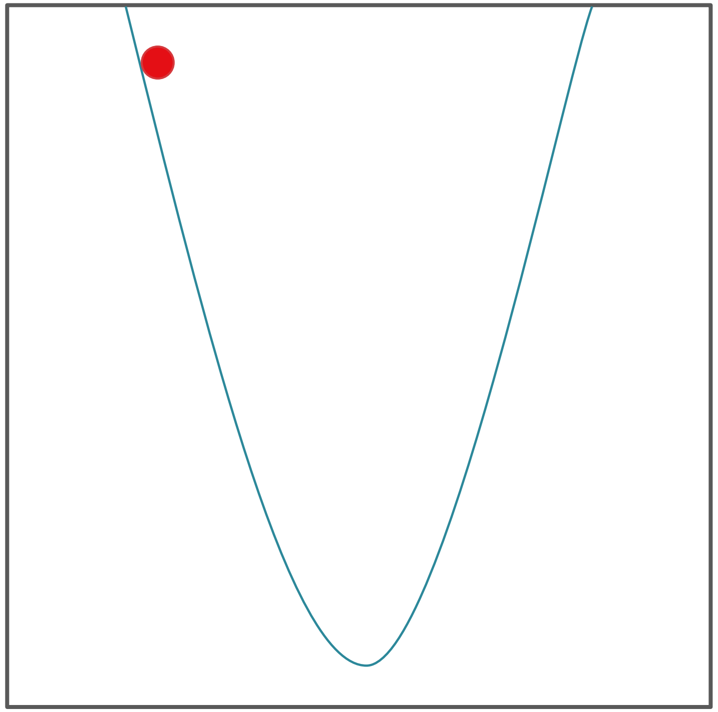
$$\hat{\boldsymbol{v}}^{(t)} = -\frac{\nabla_{\boldsymbol{w}}\ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)}{\left\|\nabla_{\boldsymbol{w}}\ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|}$$

= gradient / magnitude of gradient

Gradient Descent: Step Size

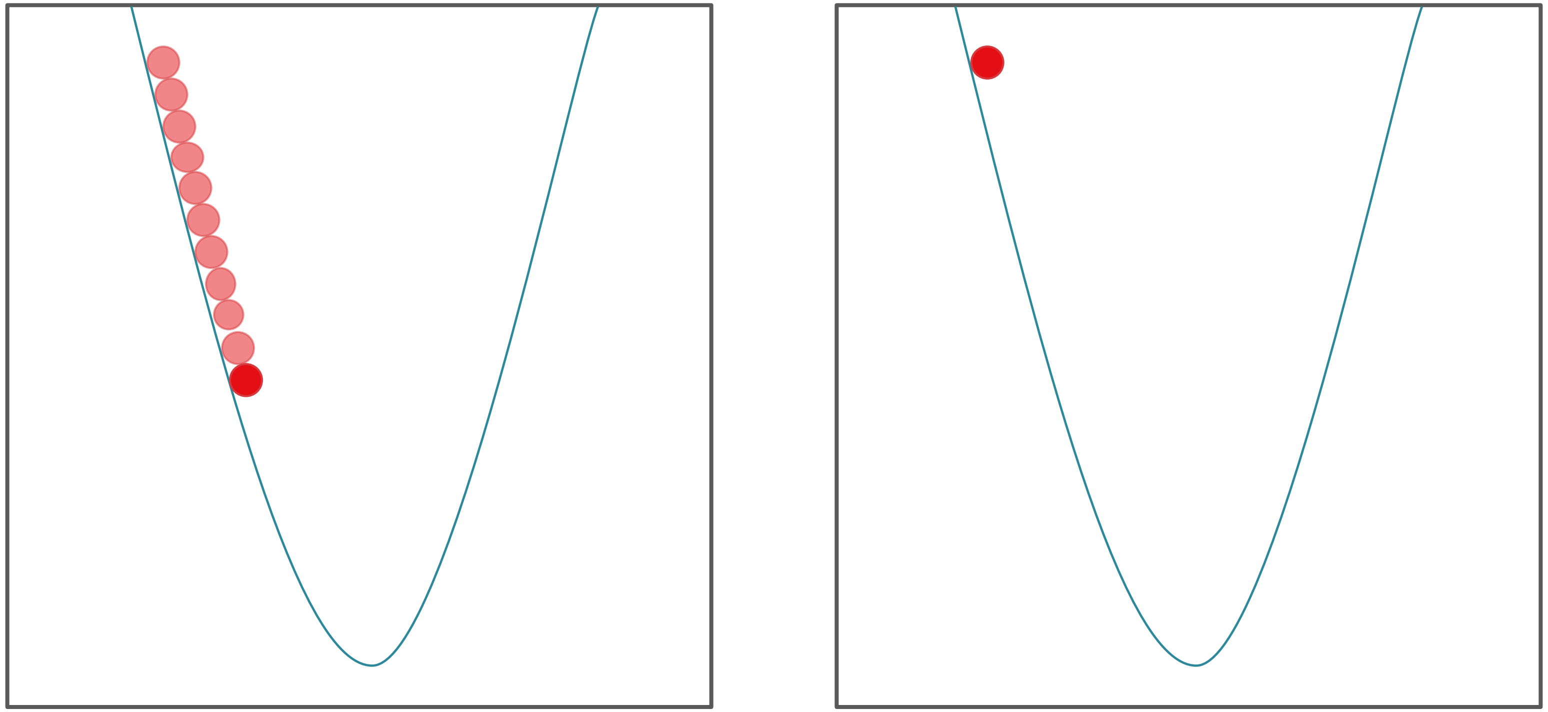


Small η



Large η

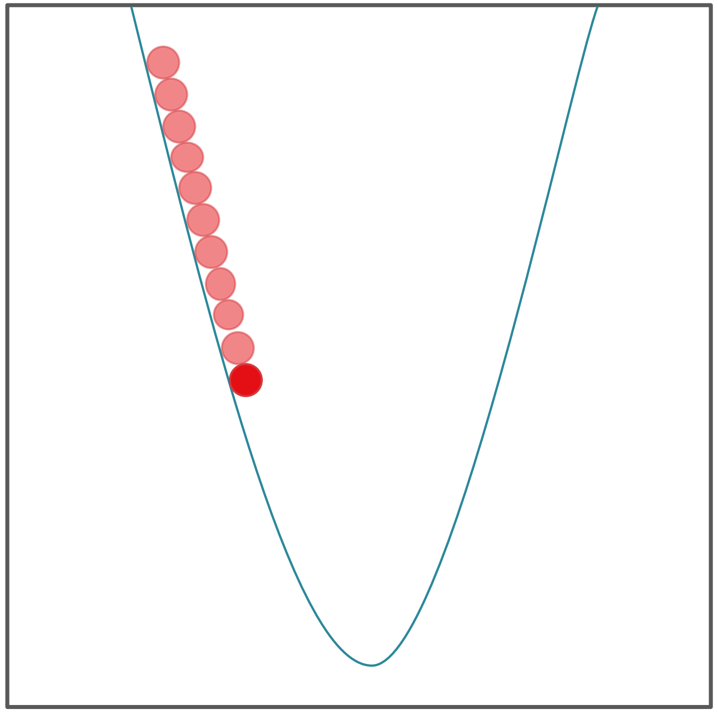
Gradient Descent: Step Size



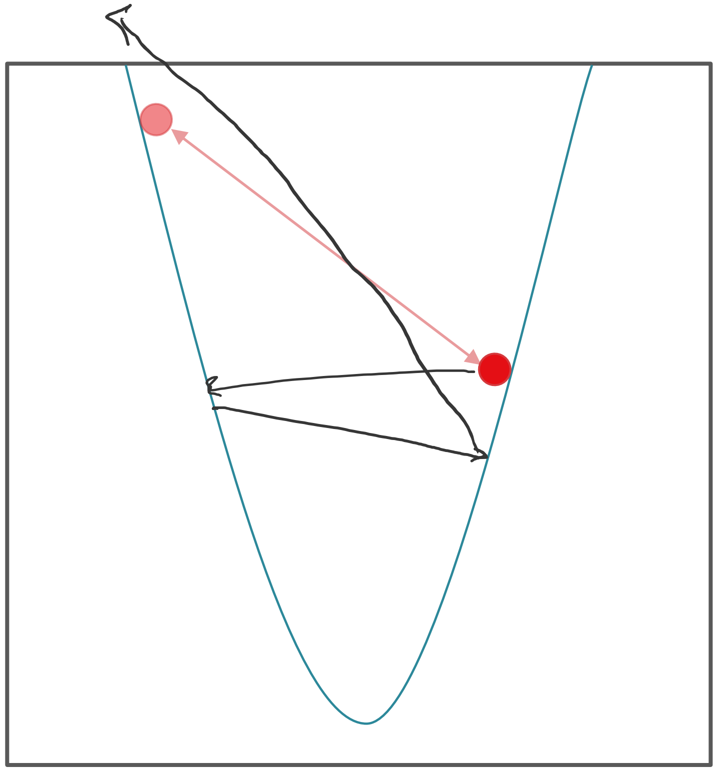
Small η

Large η

Gradient Descent: Step Size



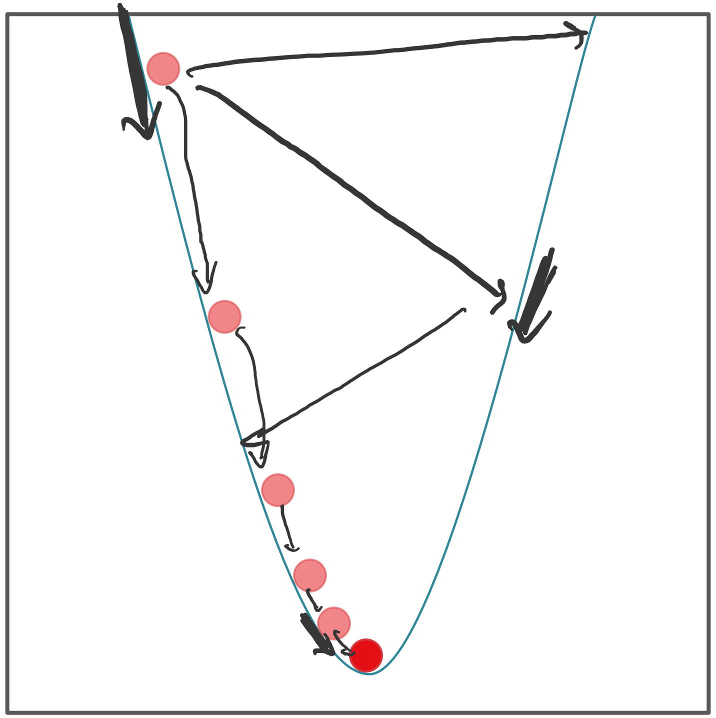
Small η



Large η

Gradient Descent: Step Size

- Use a variable $\eta^{(t)}$ instead of a fixed η!



- Set $\eta^{(t)} = \eta^{(0)} \|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}^{(t)})\|_2$ ← the magnitude or l2-norm of the gradient
- $\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}^{(t)})\|$ decreases as $\ell_{\mathcal{D}}$ approaches its minimum → $\eta^{(t)}$ (hopefully) decreases over time

Gradient Descent

- $\hat{\boldsymbol{v}}^{(t)} = -\frac{\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)}{\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|}$

test log-scales of $\eta^{(0)}$ so e.g., 10^1, 10^0, 10^{-1}, 10^{-2}, 10^{-3}, etc...

- $\eta^{(t)} = \eta^{(0)}\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|$

new location ↓ current location ↓

- $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \eta^{(t)}\hat{\boldsymbol{v}}^{(t)}$

$= w^{(t)} + \eta^{(0)} \|\nabla_w \ell_D(w^{(t)})\| \left(-\frac{\nabla_w \ell_D(w^{(t)})}{\|\nabla_w \ell_D(w^{(t)})\|}\right)$

$= w^{(t)} - \eta^{(0)} \nabla_w \ell_D(w^{(t)})$ where $\eta^{(0)}$ is the initial step size

Gradient Descent

- Input: $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}, \eta^{(0)}$

1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. Compute the gradient:
      $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}^{(t)})$ = $2X^TXw - X^Ty$ $O(ND^2)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta^{(0)} \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}(\boldsymbol{w}^{(t)})$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Gradient Descent

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta^{(0)}, \epsilon$

1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While $\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\| > \epsilon$
   a. Compute the gradient:
      $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta^{(0)} \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Gradient Descent

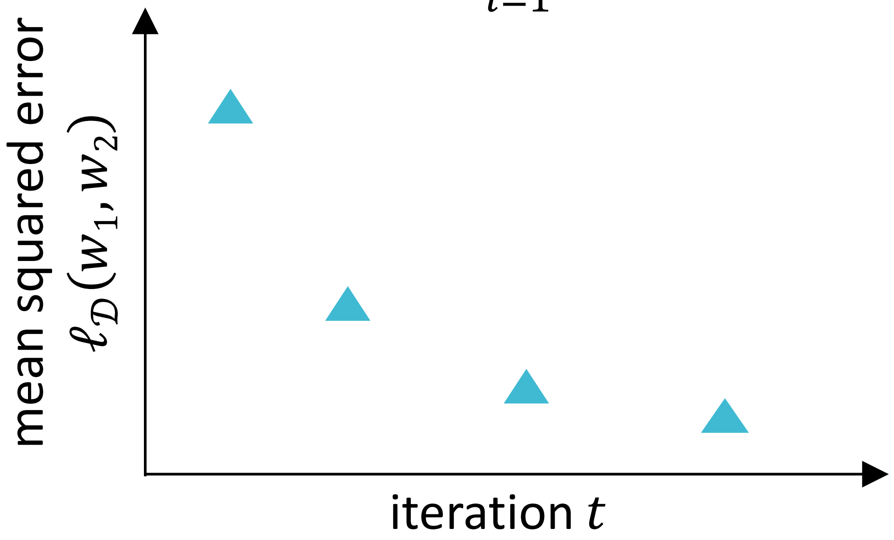
- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta^{(0)}, T$

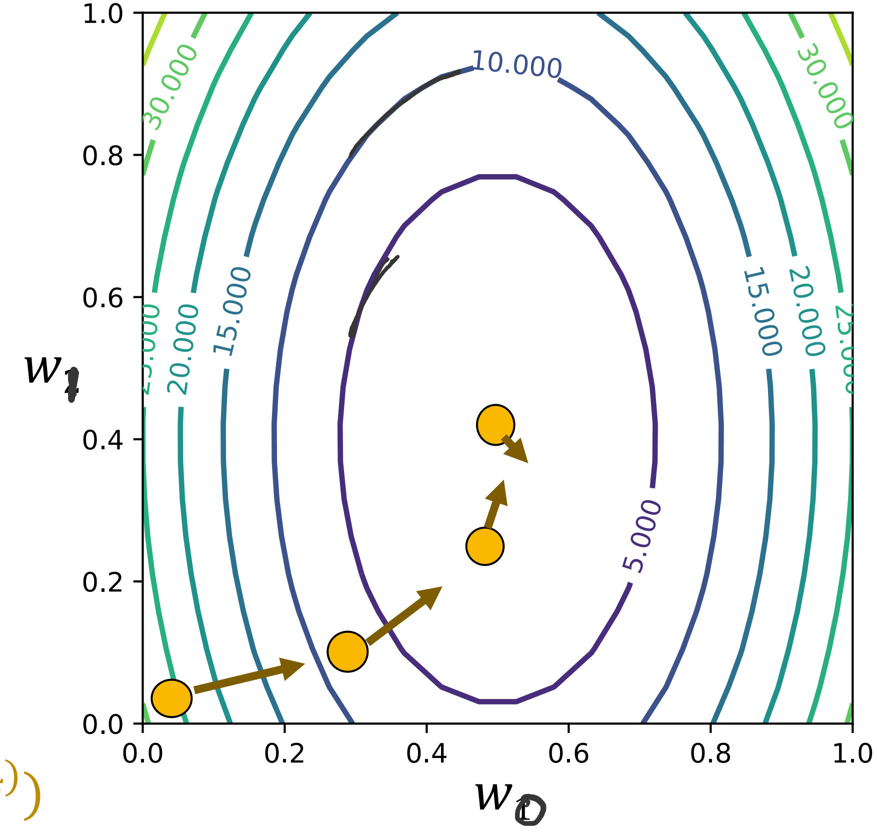
1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While $t < T$
   a. Compute the gradient: $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta^{(0)} \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

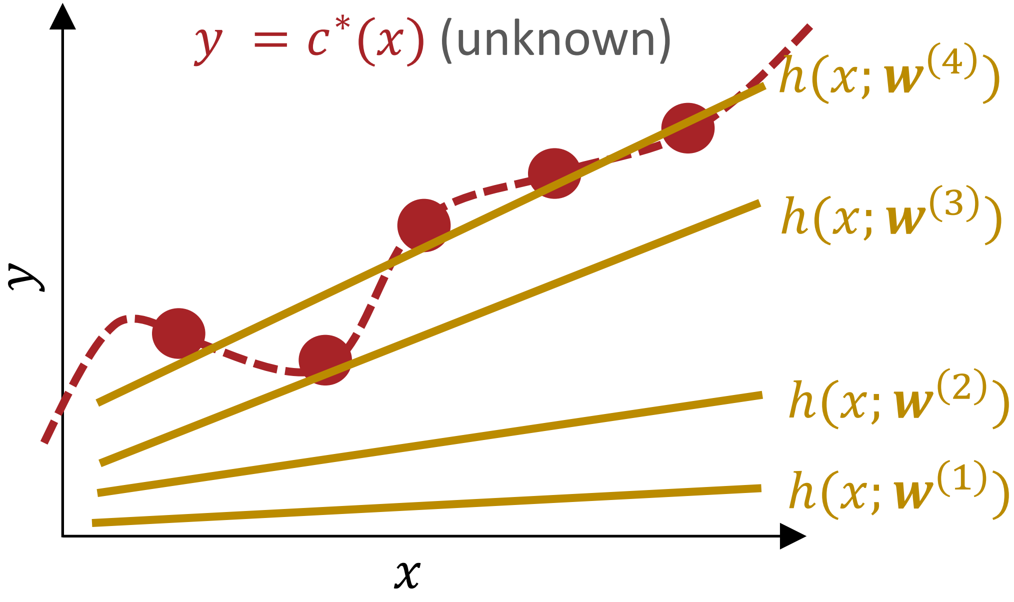
- Output: $\boldsymbol{w}^{(t)}$

Gradient Descent for Linear Regression

$$\ell_{\mathcal{D}}(w_1, w_2) = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^T\boldsymbol{x}^{(i)}\right)^2$$



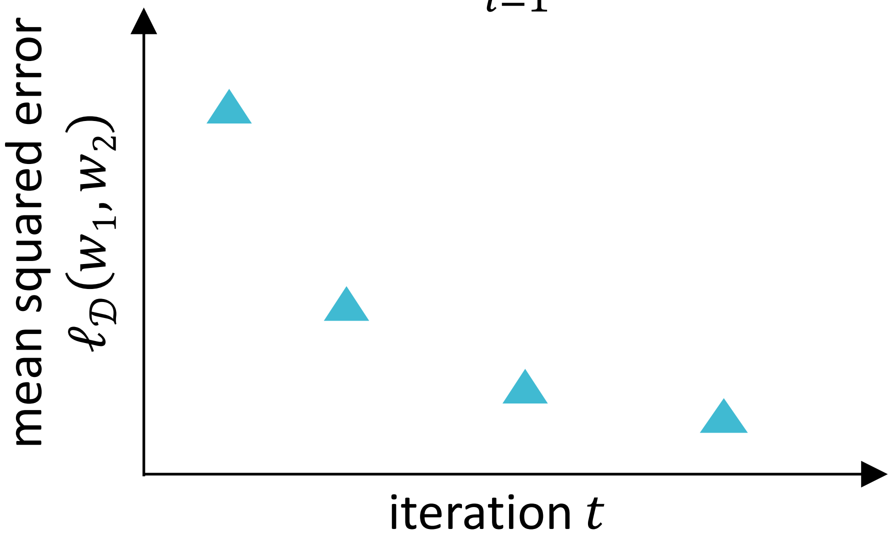


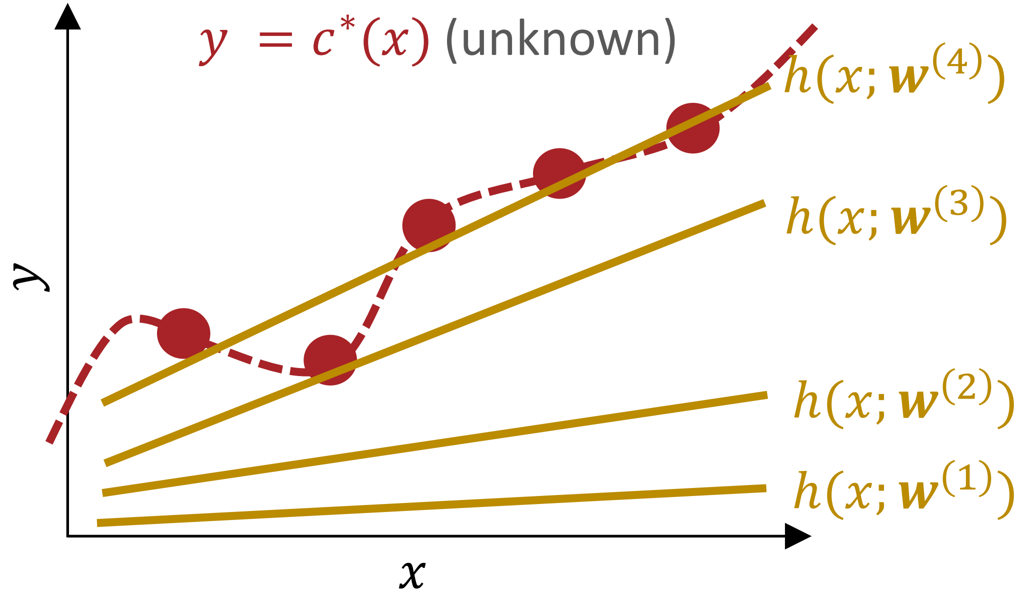


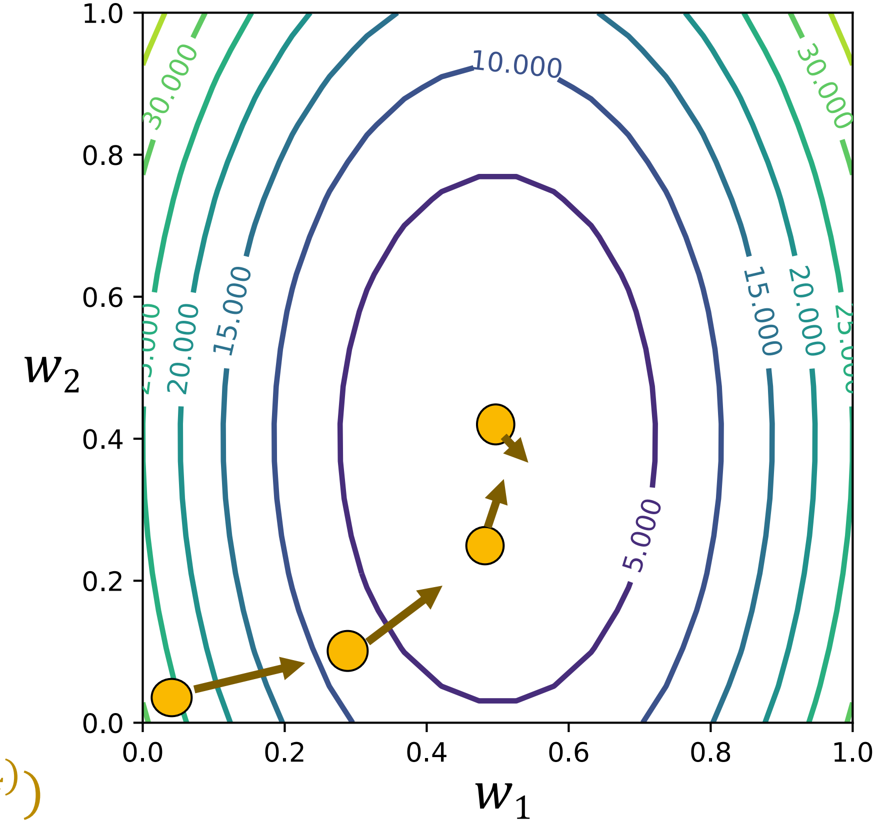
t	w_0	w_1	$\ell_{\mathcal{D}}(w_0, w_1)$

Why Gradient Descent for Linear Regression?

$$\ell_{\mathcal{D}}(w_1, w_2) = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^T\boldsymbol{x}^{(i)}\right)^2$$





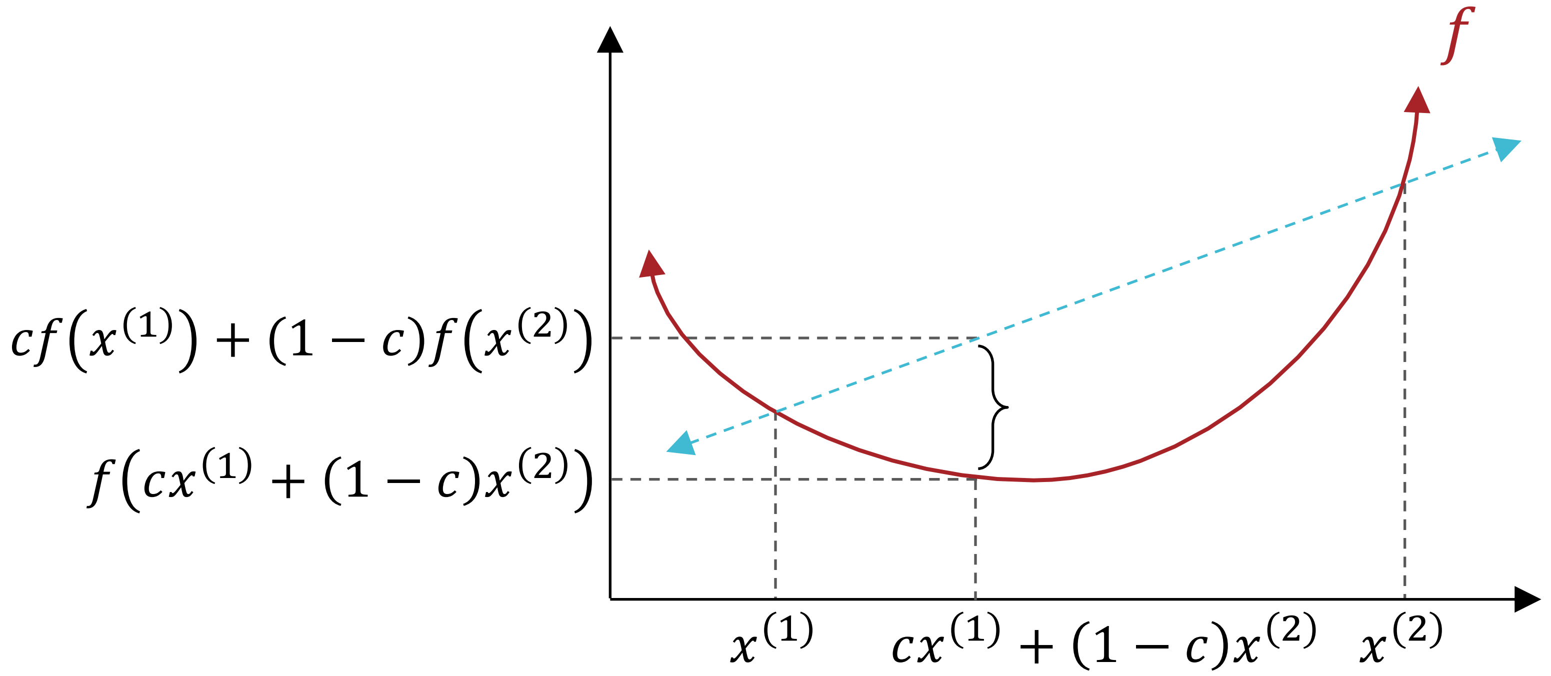


t	w_1	w_2	$\ell_{\mathcal{D}}(w_1, w_2)$
1	0.01	0.02	25.2
2	0.30	0.12	8.7
3	0.51	0.30	1.5
4	0.59	0.43	0.2

Convexity

- A function $f: \mathbb{R}^D \to \mathbb{R}$ is convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 \leq c \leq 1$

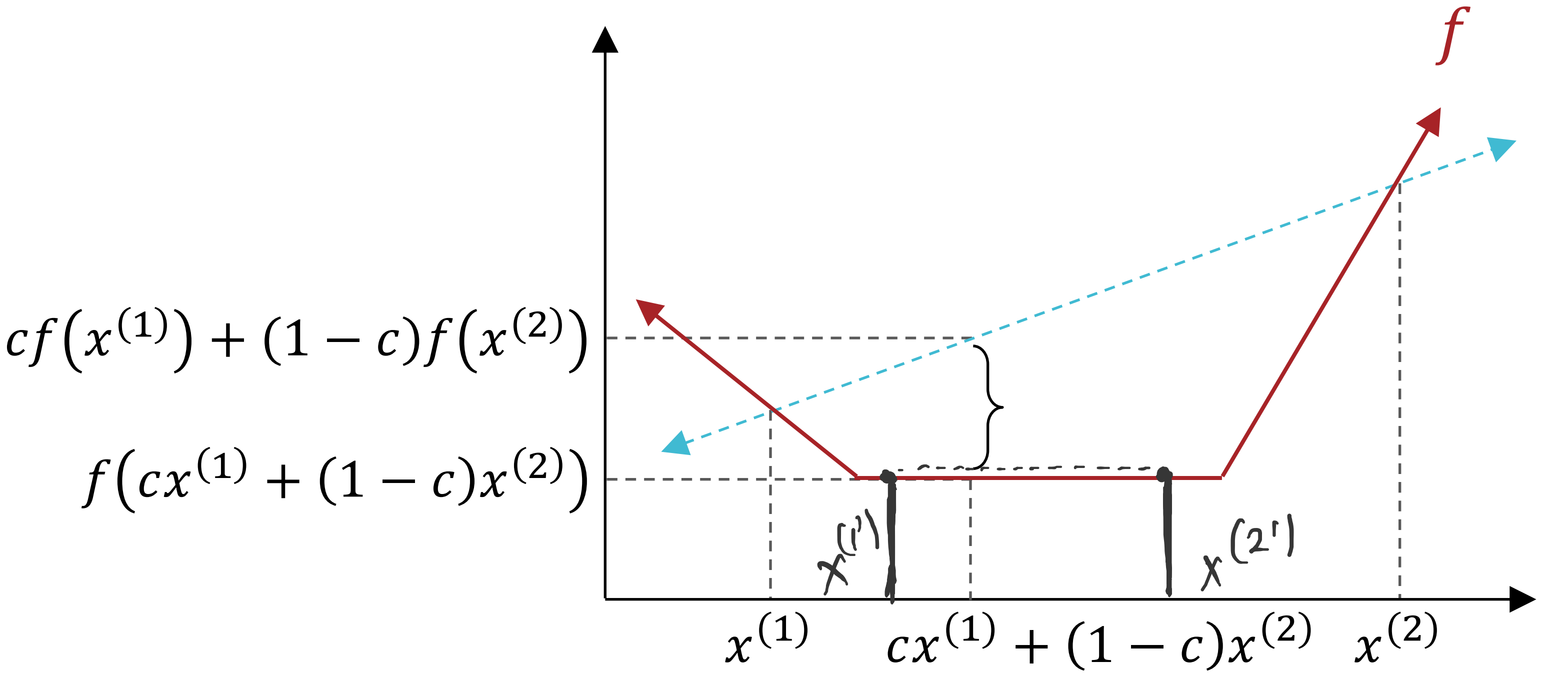
$$f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) \leq cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$$



Convexity

- A function $f: \mathbb{R}^D \to \mathbb{R}$ is convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 \leq c \leq 1$

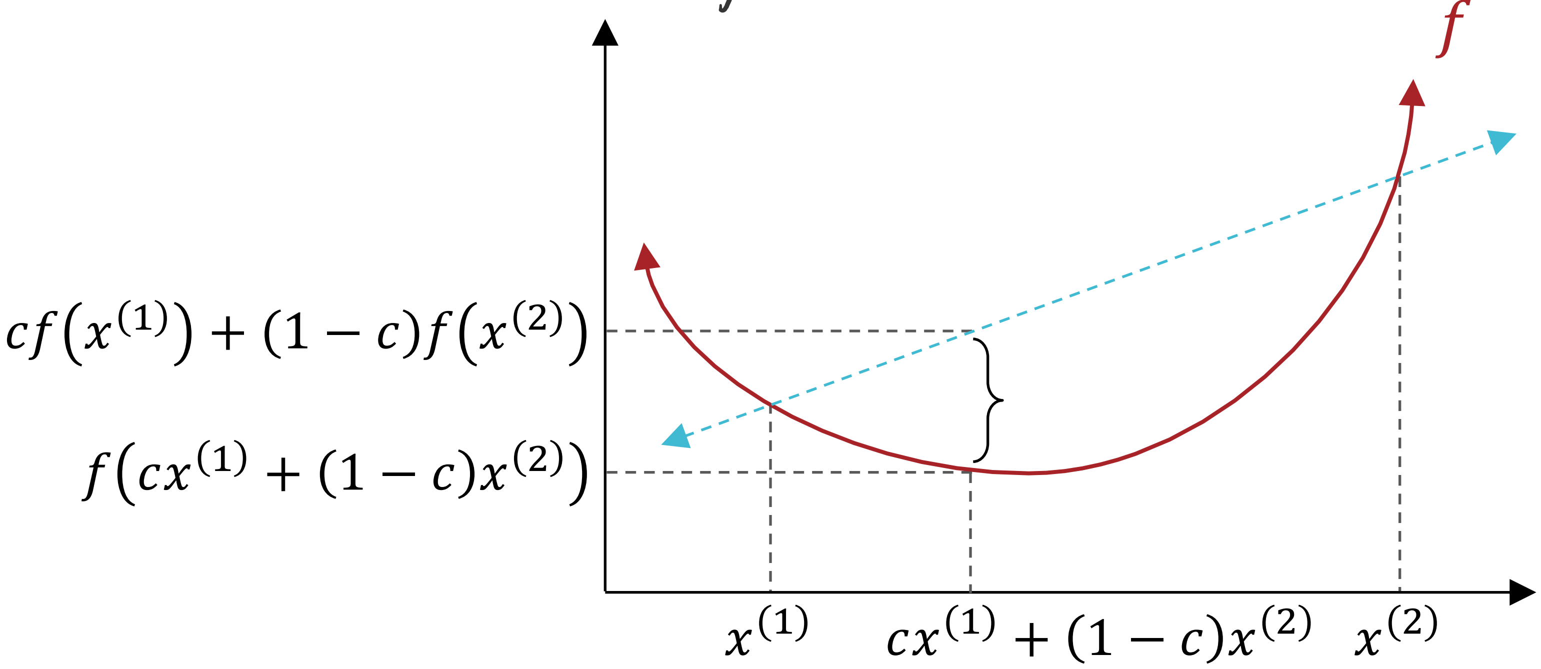
$$f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) \leq cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$$



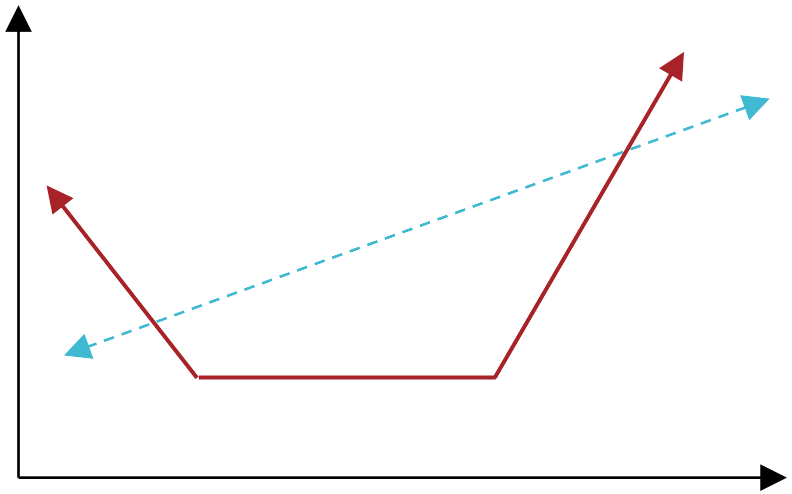
Convexity

- A function $f: \mathbb{R}^D \rightarrow \mathbb{R}$ is *strictly* convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 < c < 1$

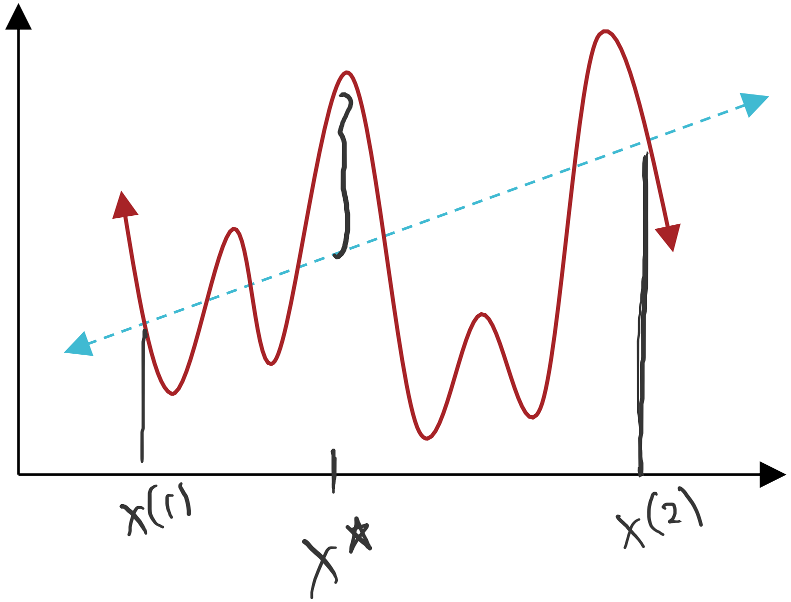
$$f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) < cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$$



Convexity

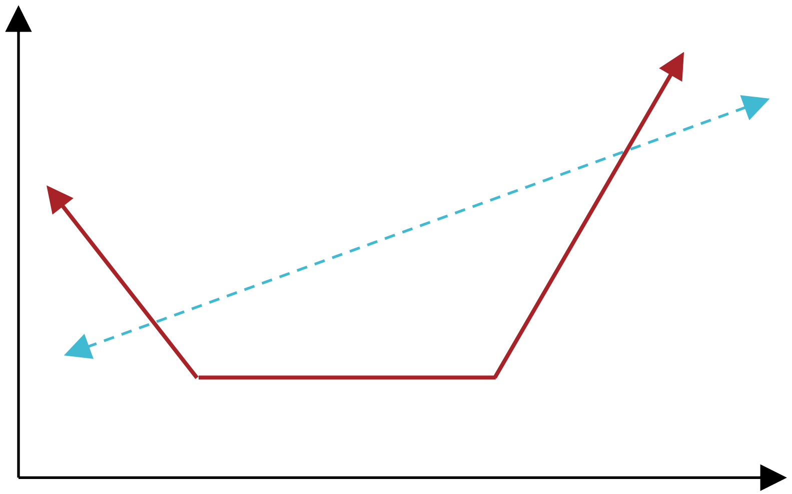


Convex functions



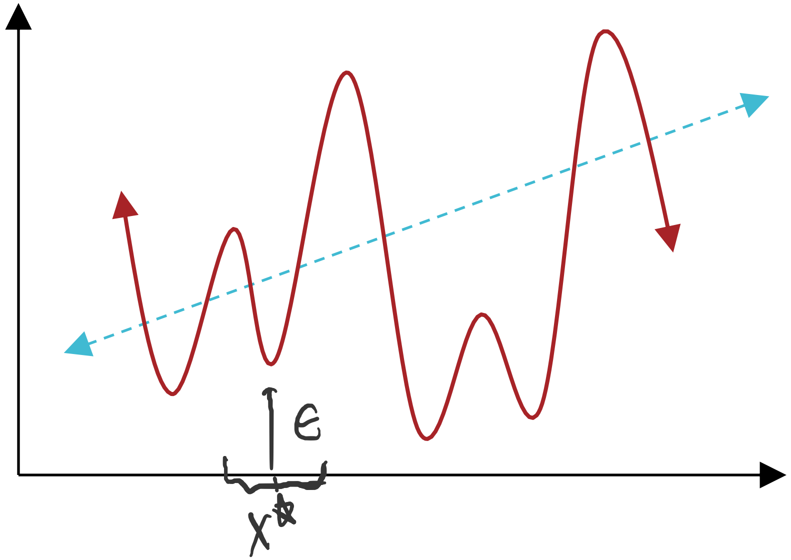
Non-convex functions

Convexity



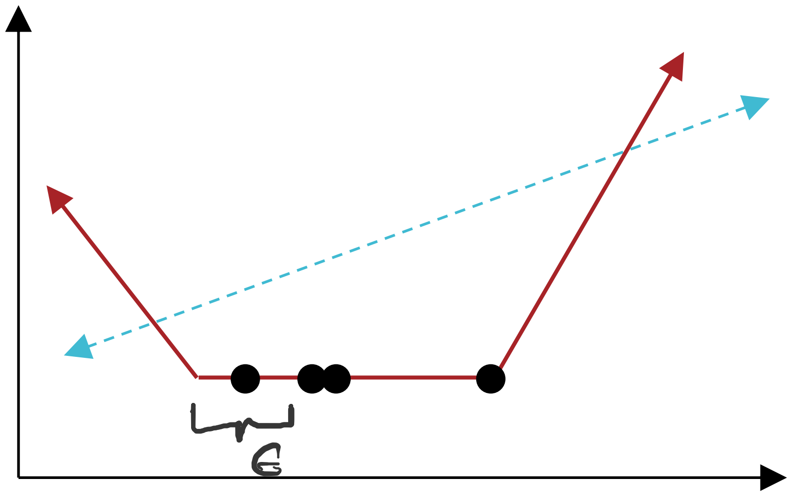
Given a function $f: \mathbb{R}^D \rightarrow \mathbb{R}$

- $\boldsymbol{x}^*$ is a global minimum iff $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \; \forall \; \boldsymbol{x} \in \mathbb{R}^D$



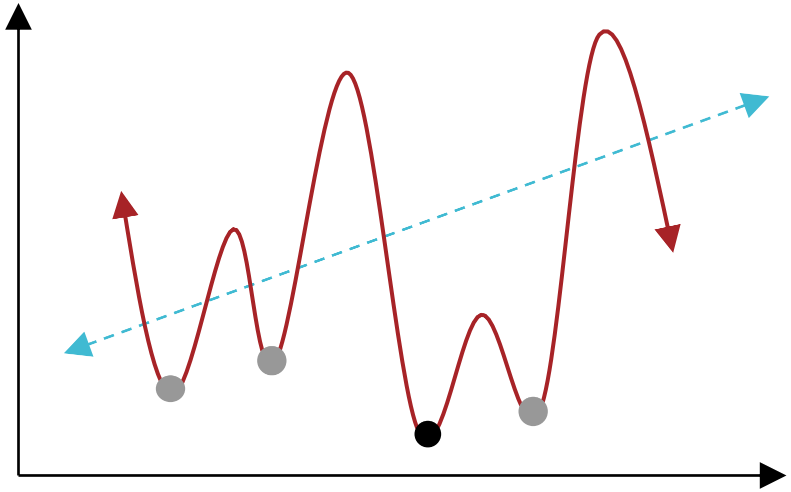
- $\boldsymbol{x}^*$ is a local minimum iff $\exists \; \epsilon$ s.t. $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \; \forall \; \boldsymbol{x}$ s.t. $\|\boldsymbol{x} - \boldsymbol{x}^*\|_2 < \epsilon$

Convexity



Convex functions:

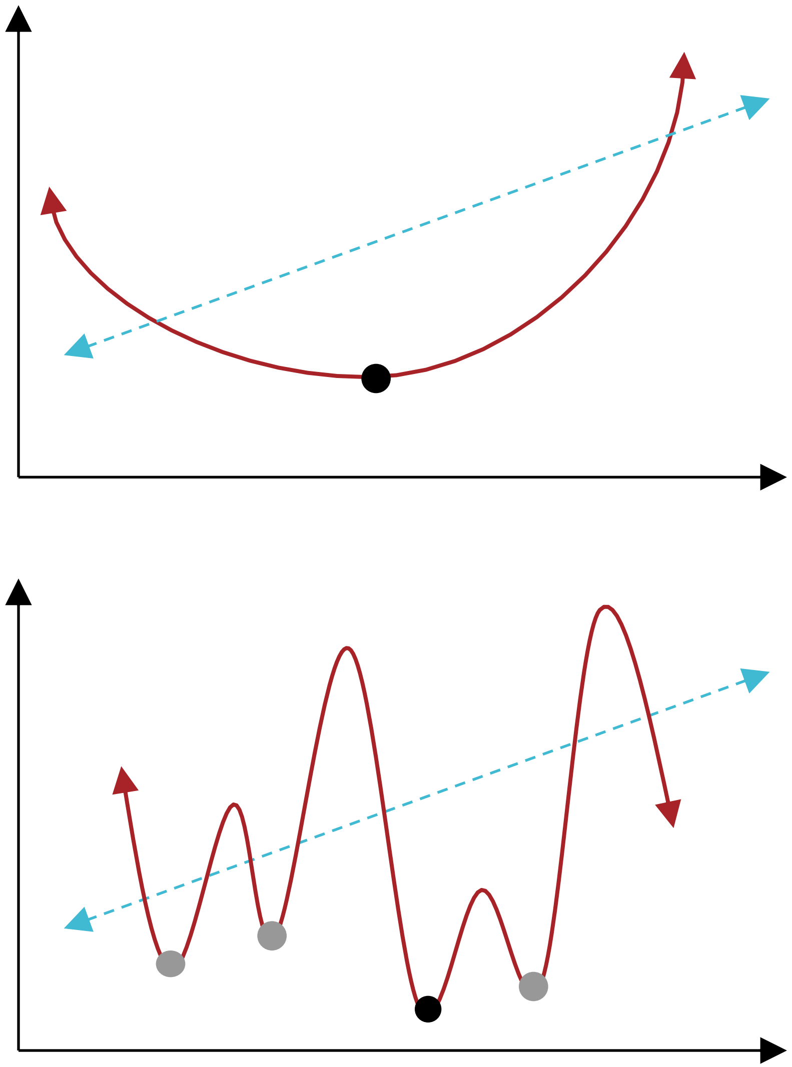
Each local minimum is a global minimum!



Non-convex functions:

A local minimum may or may not be a global minimum…

Convexity



Strictly convex functions:

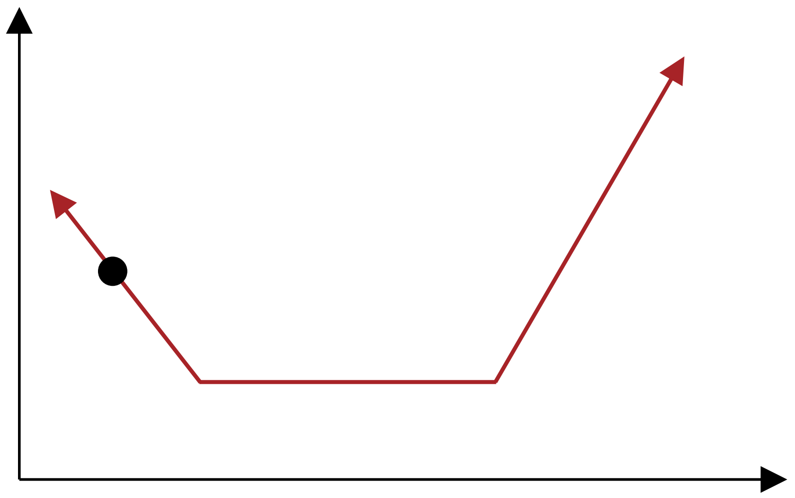
There exists a unique global minimum!

Non-convex functions:

A local minimum may or may not be a global minimum…

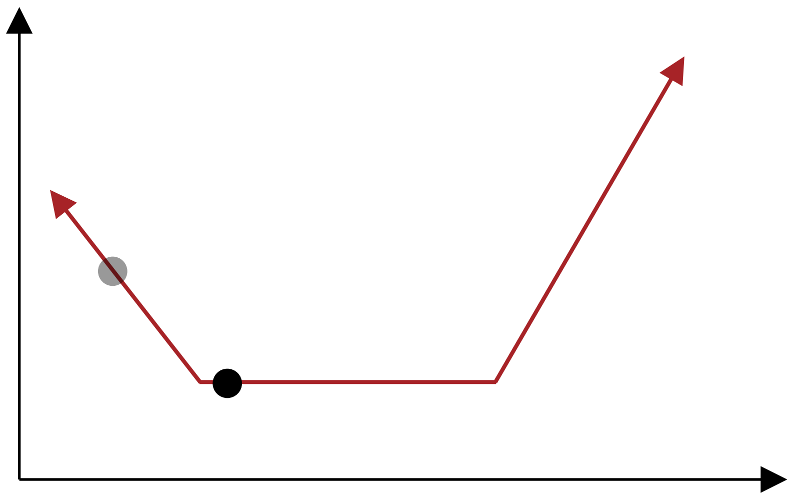
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
    - Works great if the objective function is convex!



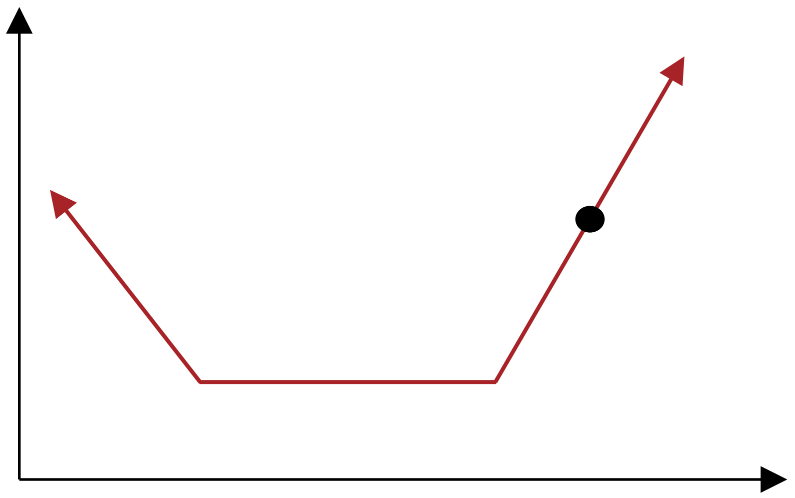
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



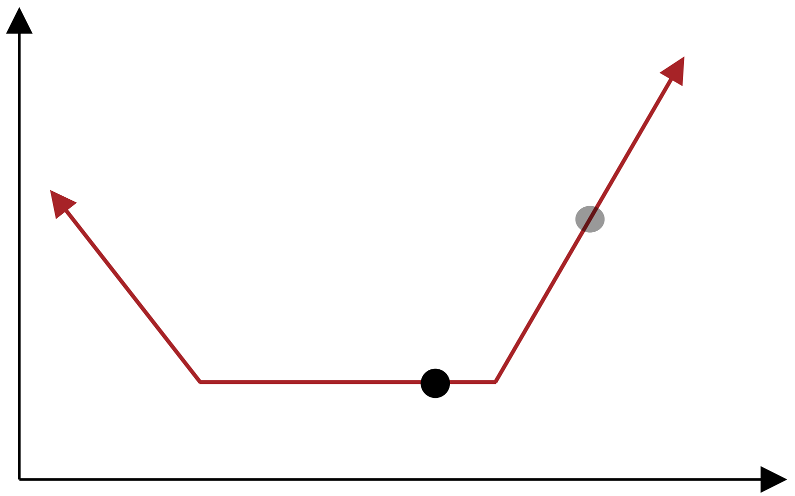
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



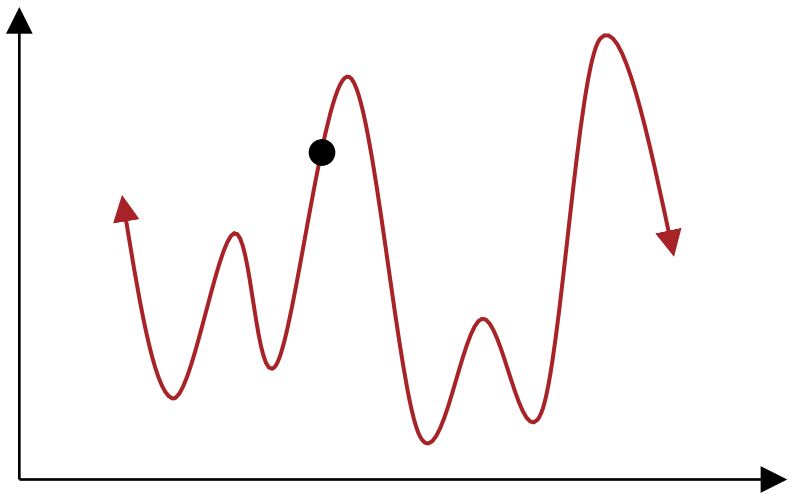
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



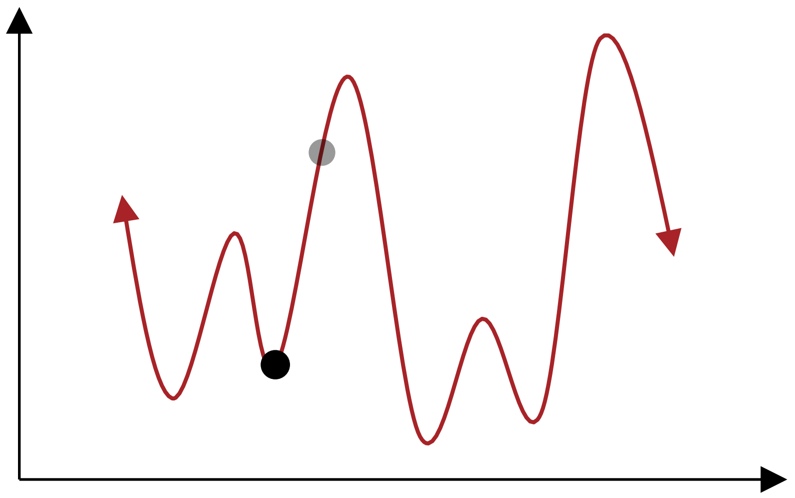
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



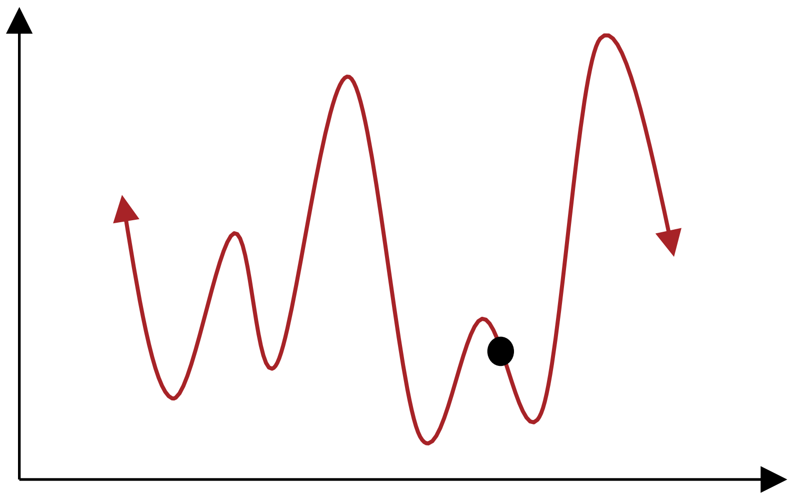
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



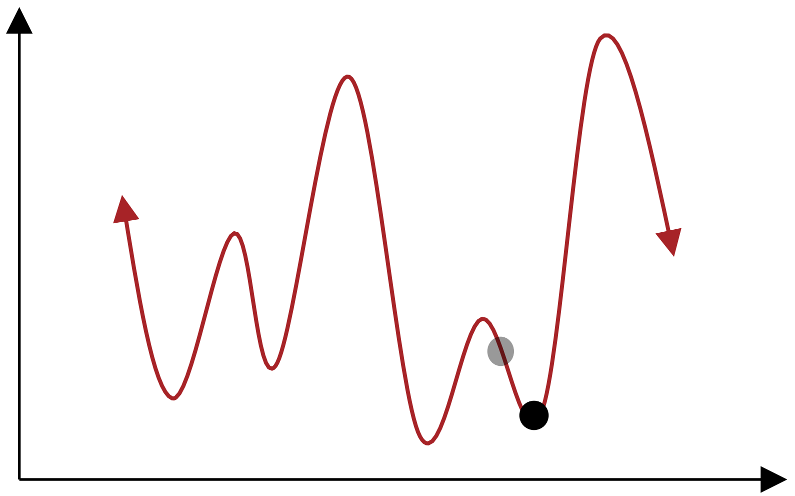
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



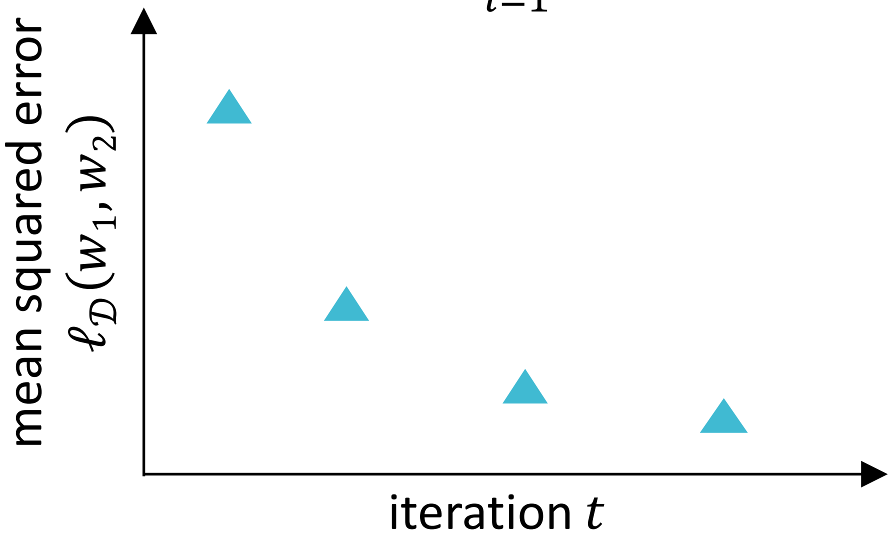
Gradient Descent & Convexity

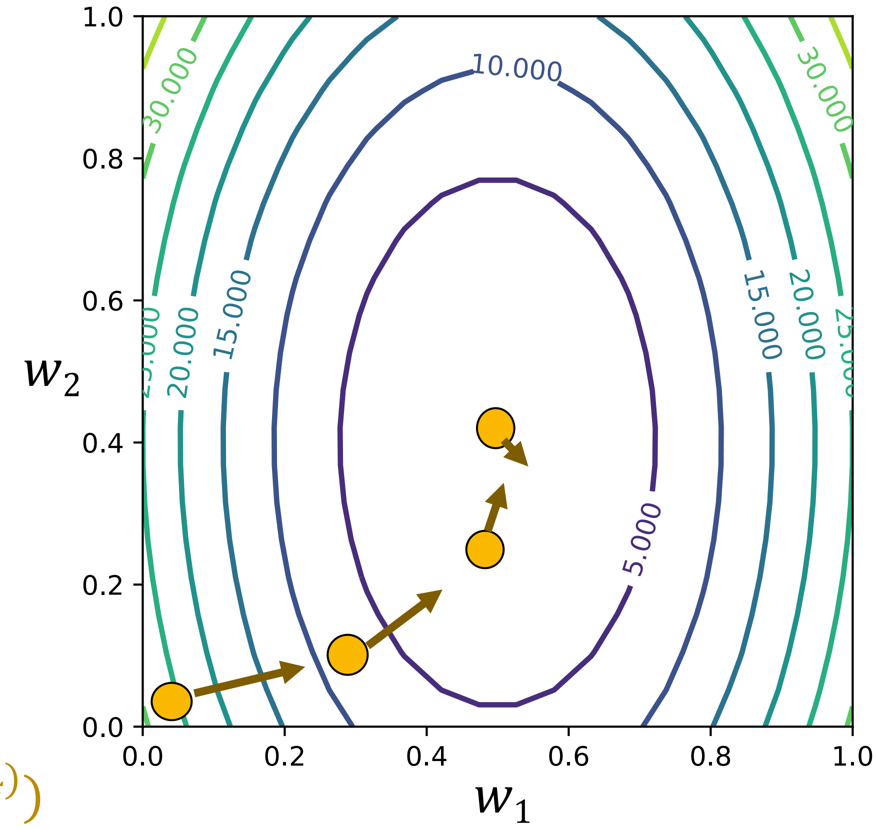
- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…

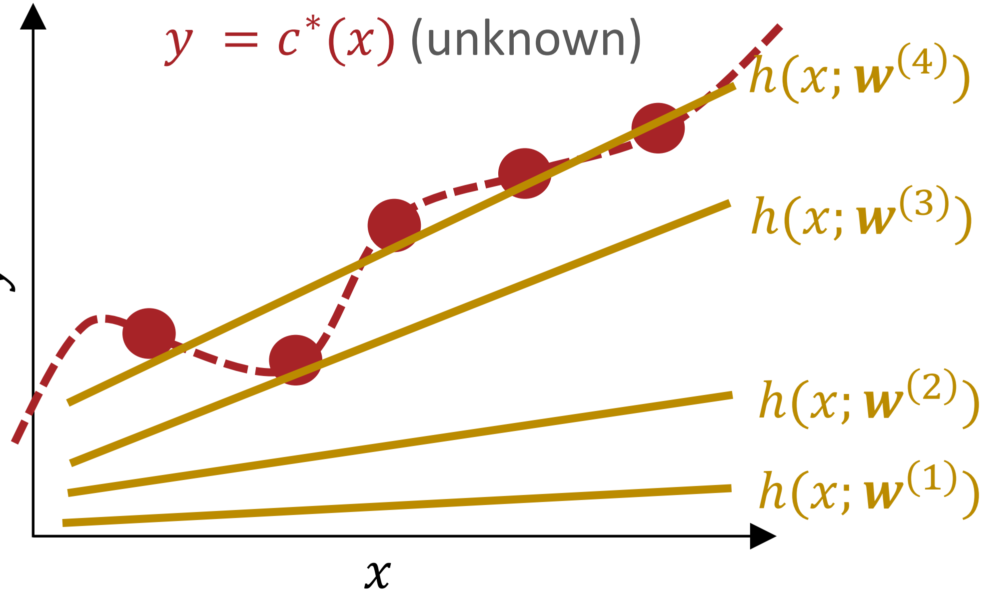


The mean squared error is convex (but not always strictly convex)

$$\ell_{\mathcal{D}}(w_1, w_2) = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^T\boldsymbol{x}^{(i)}\right)^2$$







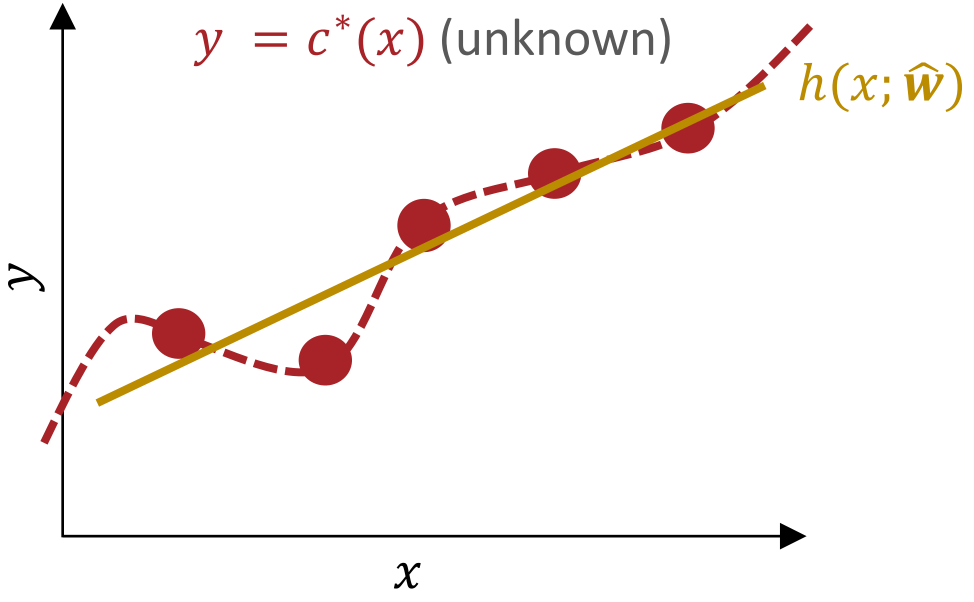
t	w_1	w_2	$\ell_{\mathcal{D}}(w_1, w_2)$
1	0.01	0.02	25.2
2	0.30	0.12	8.7
3	0.51	0.30	1.5
4	0.59	0.43	0.2

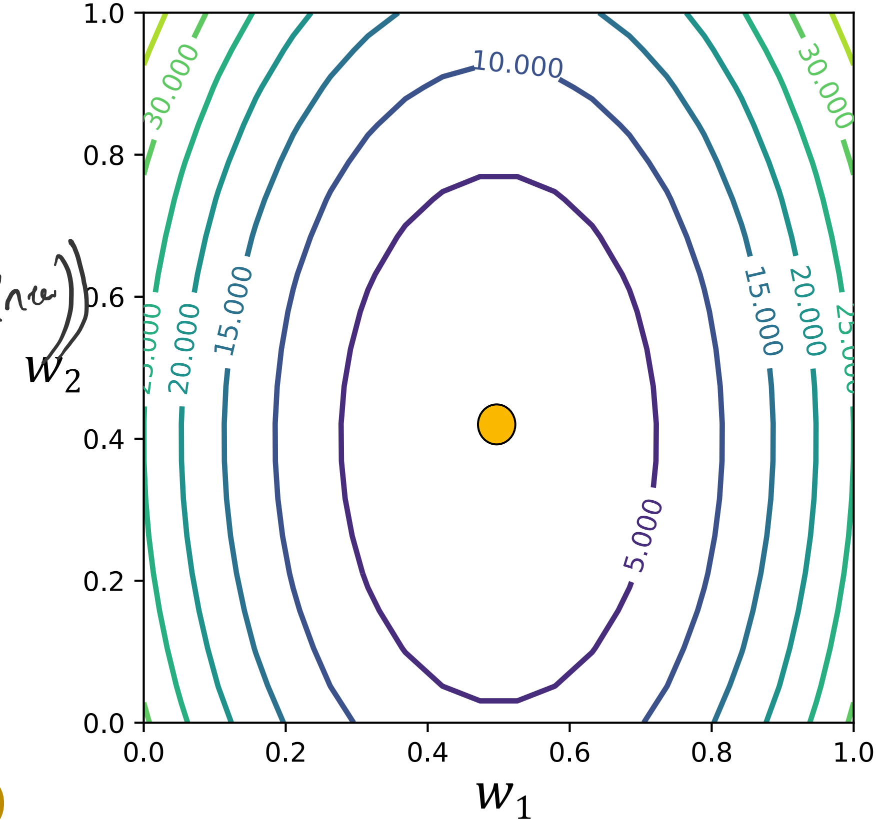
Closed Form Optimization

$$\hat{\boldsymbol{w}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

$\hat{w} = f(\hat{w}, x^{(new)}, y^{(new)})$

rank-1 update





t	w_1	w_2	$\ell_{\mathcal{D}}(w_1, w_2)$
1	0.59	0.43	0.2

Key Takeaways

- Convexity vs. non-convexity
  - Strong vs. weak convexity
  - Implications for local, global and unique optima
- Gradient descent
  - Effect of step size
  - Termination criteria