Announcements

Assignments

- HW3
  - Mon, 9/28, 11:59 pm

Midterm 1

- Mon, 10/5
- See Piazza for details
- Fill out swap-section / conflict form by Friday

Plan

Last time

- Regression
- Linear regression
- Optimization for linear regression

Today

- Optimization for linear regression
  - Linear and convex function
  - (Batch) Gradient descent
  - Closed-form solution
  - Stochastic gradient descent

Introduction to Machine Learning

Linear Regression and Optimization

Instructor: Pat Virtue

Linear Regression

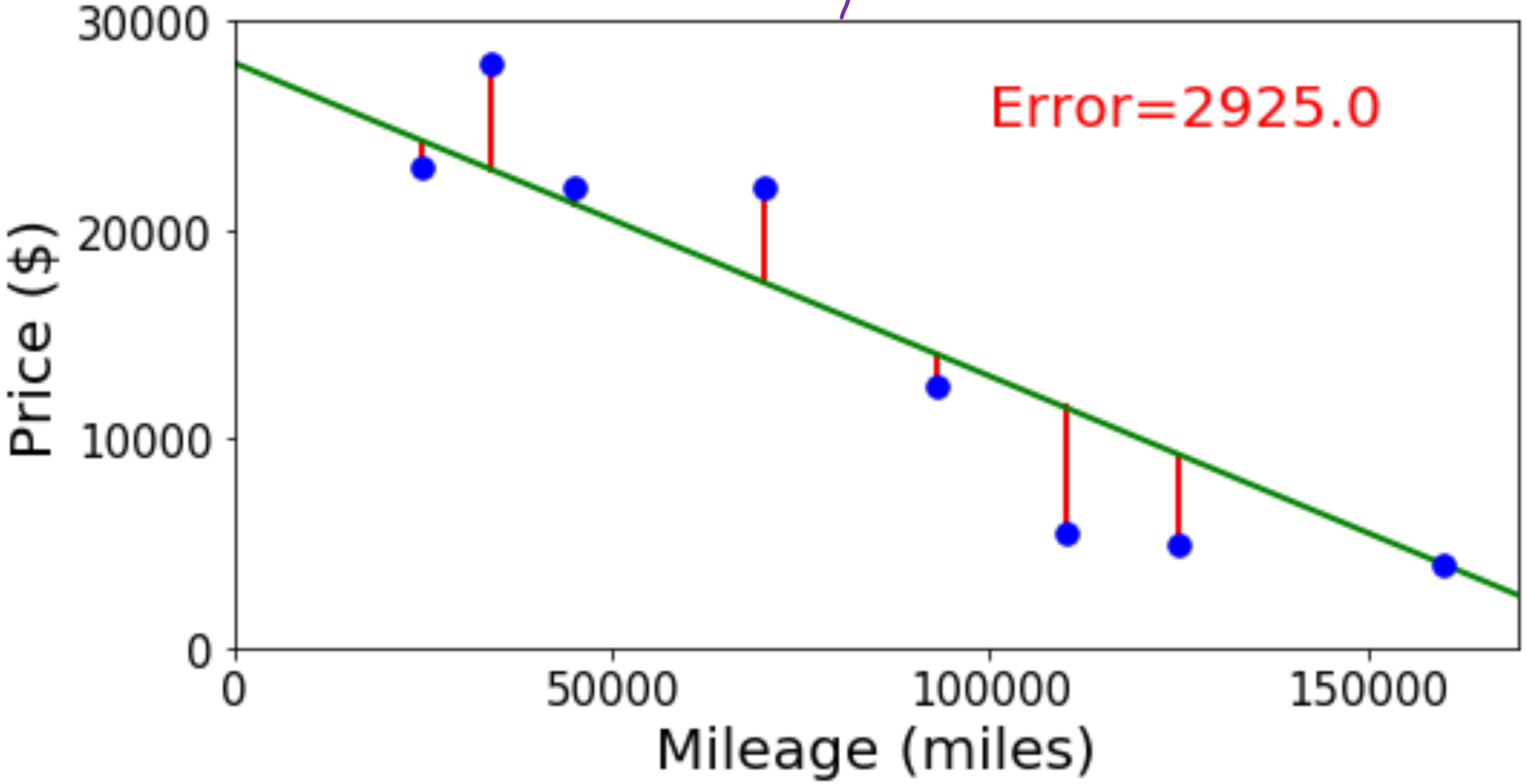
$y = mx + b$

$y = wx + b$

$y = w_1 x + w_0$

$y = \theta_1 x + \theta_0$

Selling my car



Linear in Higher Dimensions

1-D $y = wx + b$

2-D $y = w_1 x_1 + w_2 x_2 + b$

What are these linear shapes called for 1-D, 2-D, 3-D, M-D input?

	$\boldsymbol{x} \in \mathbb{R}$	$\boldsymbol{x} \in \mathbb{R}^2$	$\boldsymbol{x} \in \mathbb{R}^3$	$\boldsymbol{x} \in \mathbb{R}^M$
$y = \boldsymbol{w}^T \boldsymbol{x} + b$	line	plane	hyperplane	hyperplane
$\boldsymbol{w}^T \boldsymbol{x} + b = 0$	point	line	plane	hyperplane
$\boldsymbol{w}^T \boldsymbol{x} + b \geq 0$	halfline	halfplane	halfspace	halfspace

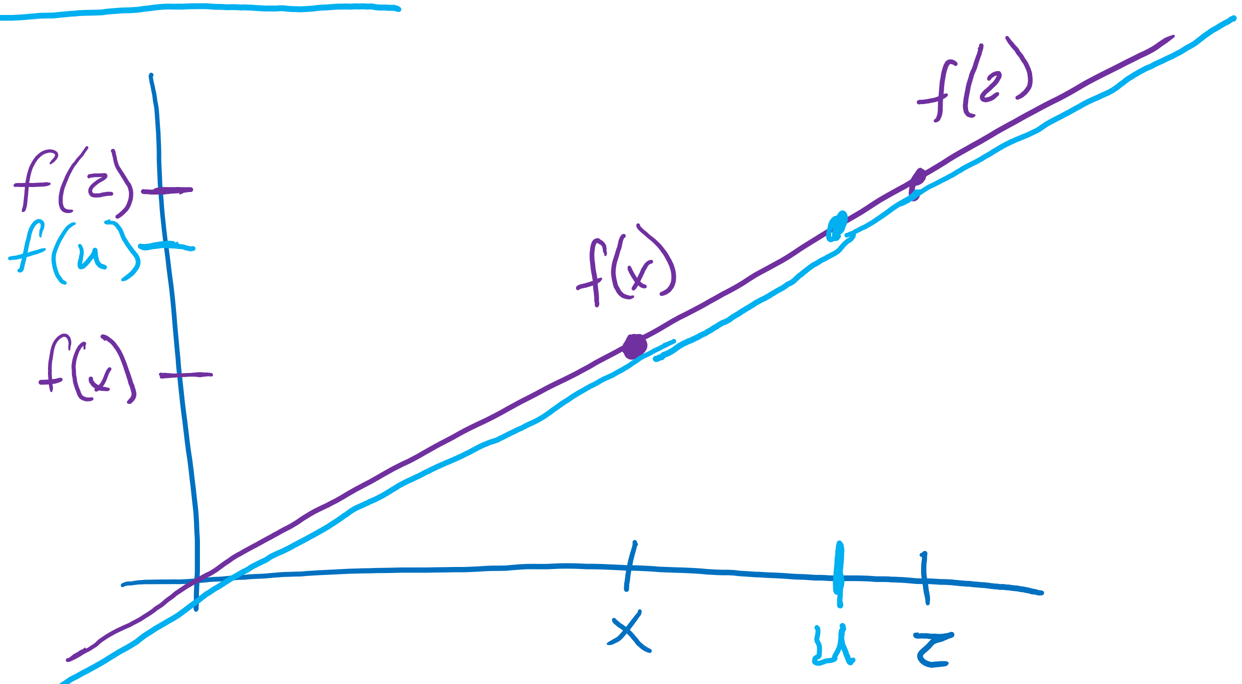
Linear Function

Linear function

If $f(\boldsymbol{x})$ is linear, then:

- $f(\boldsymbol{x} + \boldsymbol{z}) = f(\boldsymbol{x}) + f(\boldsymbol{z})$
- $f(\alpha\boldsymbol{x}) = \alpha f(\boldsymbol{x}) \quad \forall\alpha$
- $f(\alpha\boldsymbol{x} + (1 - \alpha)\boldsymbol{z}) = \alpha f(\boldsymbol{x}) + (1 - \alpha) f(\boldsymbol{z}) \quad \forall\alpha$

$\alpha = 0.25$



Piazza Poll 1

$$y = \underbrace{\underbrace{wx}_{\text{linear}} + b}_{\text{affine}}$$

Based on the following definition of a linear, is the equation for a line, $y = wx + b$, linear?   Example: $y = 3x + 5$

$$y = \underbrace{\underbrace{\vec{w}^T\vec{x}}_{\text{linear}} + b}_{\text{affine}}$$

$f(\boldsymbol{x})$ is linear if and only if:

- $f(\boldsymbol{x} + \boldsymbol{z}) = f(\boldsymbol{x}) + f(\boldsymbol{z})$ and
- $f(\alpha\boldsymbol{x}) = \alpha f(\boldsymbol{x}) \quad \forall\alpha$ ←

$\alpha = 7$

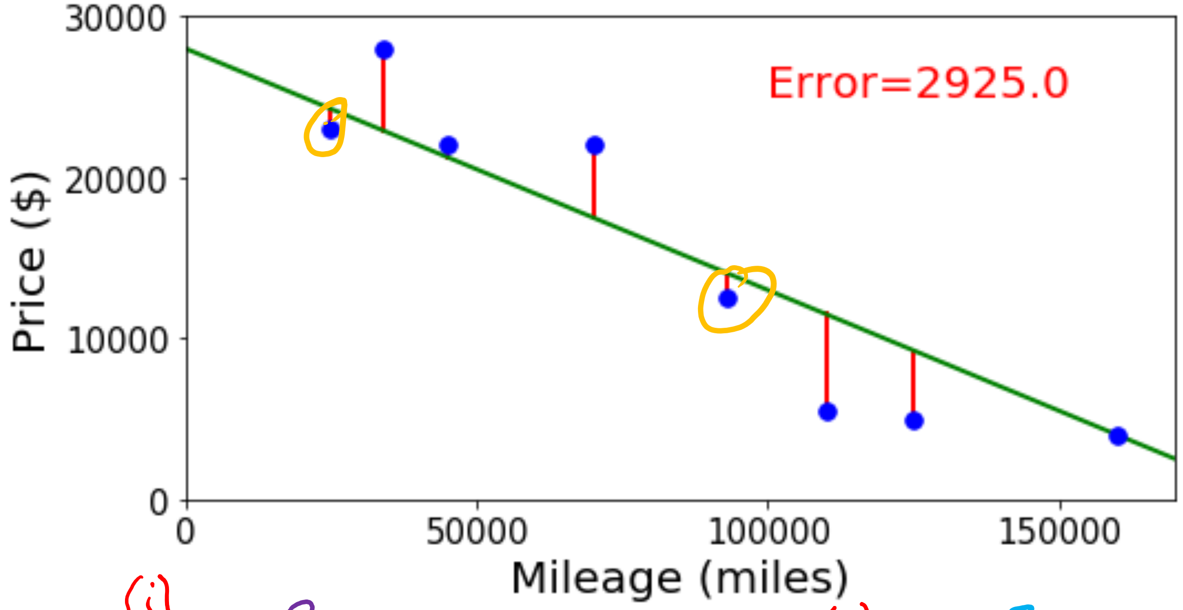
$$f(7x) = 3(7x) + 5$$
$$= 21x + 5$$
$$\neq 7(3x + 5)$$

Yes 62 → 32%

No 32 → 67%

Linear Regression

Linear algebra formulation



$$y = \vec{w}^T x_{orig} + b$$

$$y^{(i)} = \vec{w}^T x_{orig}^{(i)} + b$$

$$y^{(i)} = \vec{\theta}^T \vec{x} = \vec{x}^T \vec{\theta}$$

$$\begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(N)} \end{bmatrix} = \begin{bmatrix} - & x^{(1)T} & - \\ & \vdots & \\ - & x^{(N)T} & - \end{bmatrix} \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_M \end{bmatrix}$$

$$\vec{y} = X \vec{\theta}$$

Design Matrix X

$$\vec{x}_{orig}^{(i)} \in \mathbb{R}^2 \longrightarrow \vec{x}^{(i)} \in \mathbb{R}^3$$

$$\vec{x}_{orig}^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix} \longrightarrow \vec{x}^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}$$

$$\vec{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \longrightarrow \vec{\theta} = \begin{bmatrix} b \\ w_1 \\ w_2 \end{bmatrix}$$

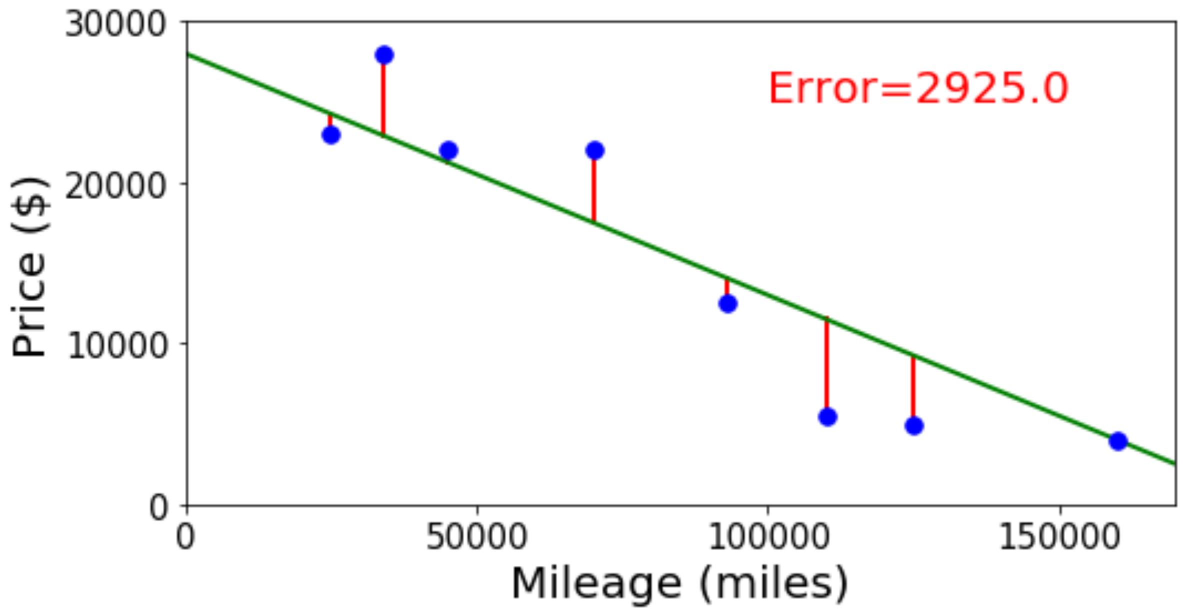
b

Linear Regression

Error and objectives

$$J(w, b) = \frac{1}{N} \sum_{i=1}^{N} \left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

$$\hat{y}^{(i)} = w x^{(i)} + b$$



$$J(w_1, w_2, b) = \frac{1}{N} \sum \left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

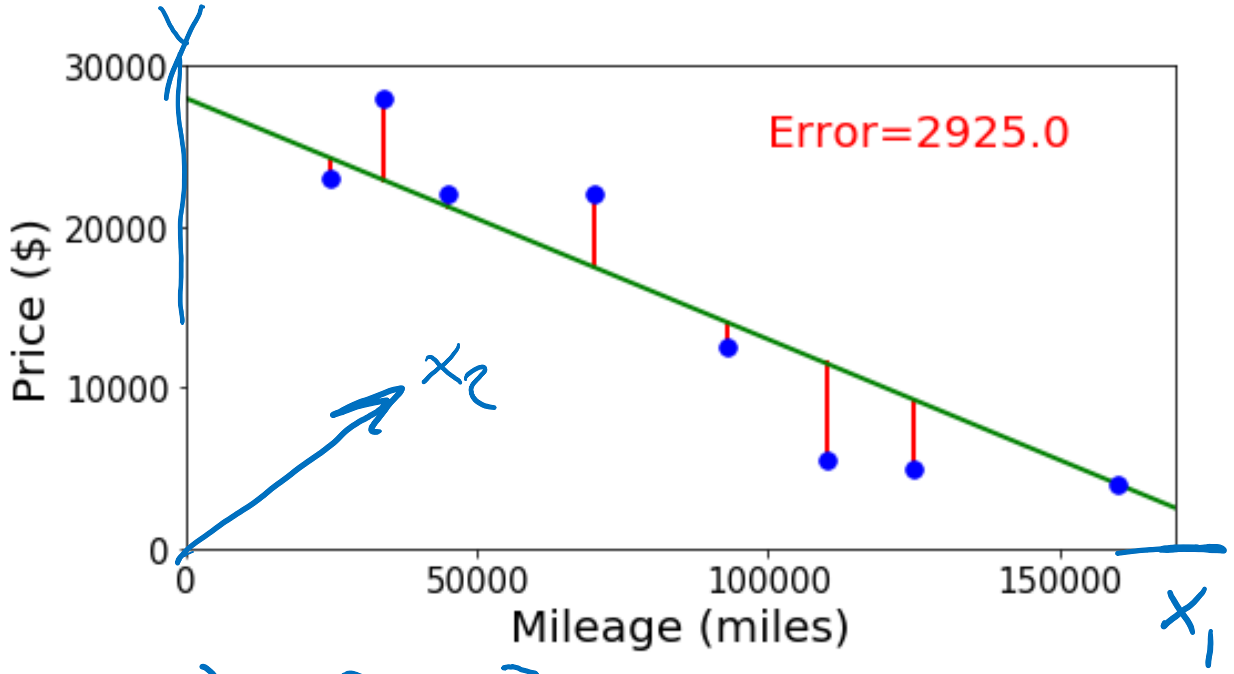
$$\hat{y}^{(i)} = w_1 x_1^{(i)} + w_2 x_2^{(i)} + b$$

$$J(w_1, \ldots w_m, b) =$$

$$\hat{y}^{(i)} = \sum_{j=1}^{m} w_j x_j^{(i)} + b$$

Linear Regression

Linear algebra formulation



$$\vec{\theta} = \begin{bmatrix} b \\ w_1 \\ w_2 \end{bmatrix} \qquad \vec{x}^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \\ & \vdots & \\ 1 & x_1^{(N)} & x_2^{(N)} \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

$$J(\vec{\theta}) = \frac{1}{N} \|\vec{y} - \hat{\vec{y}}\|_2^2 = \frac{1}{N} \|\vec{y} - X\vec{\theta}\|_2^2$$

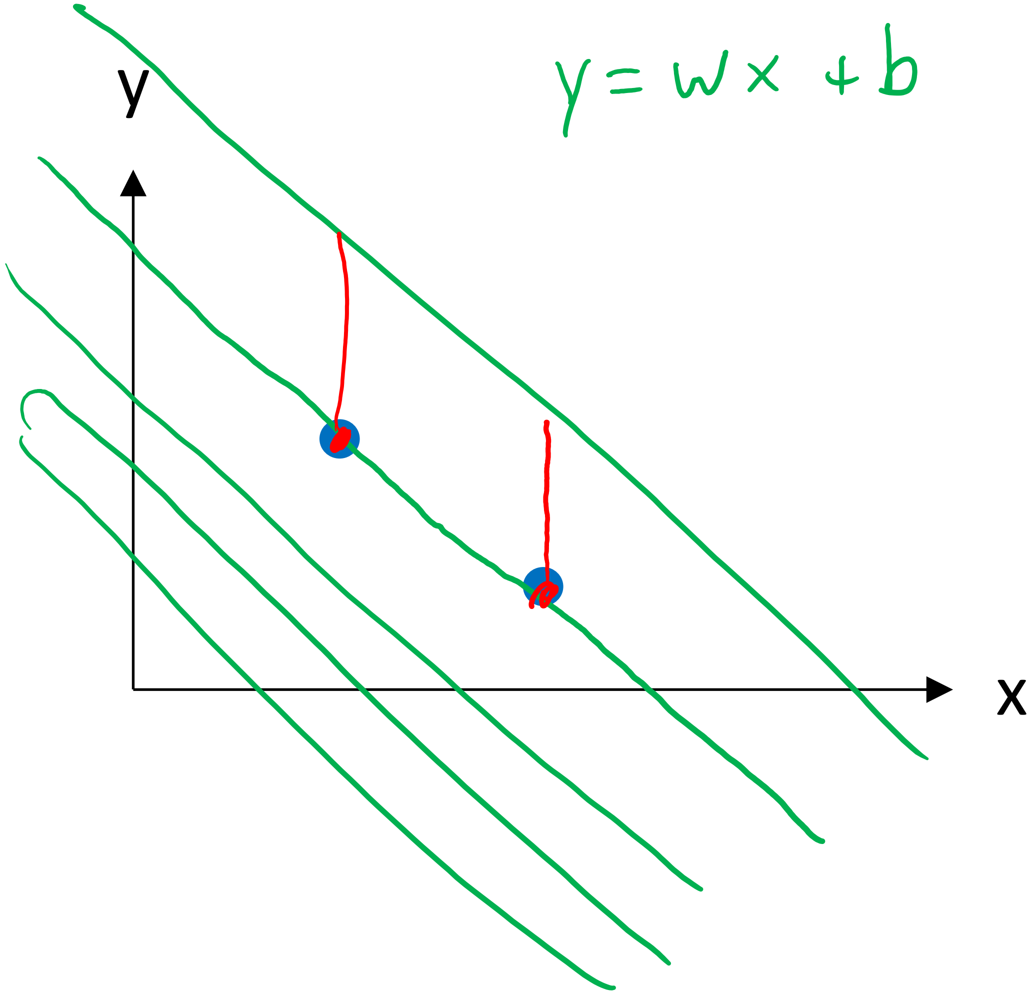
ℓ_2-norm squared

$$\|\vec{z}\|_2^2 = \sum_{i=1}^{N} (z_i)^2$$

Previous Piazza Poll

For fixed data and fixed slope, w, what shape do we get by plotting MSE objective vs intercept, b?

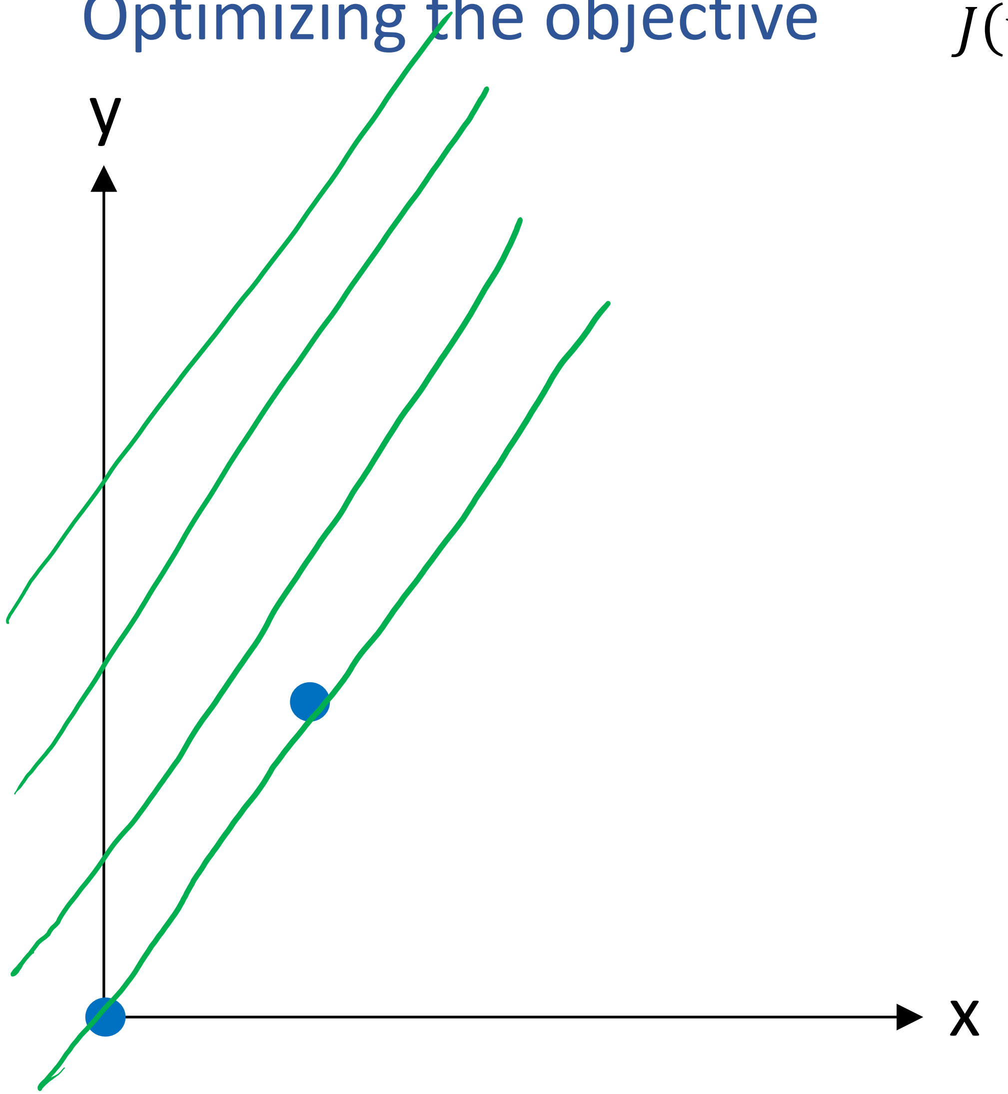
A. Line
B. Plane
C. Half-plane
D. Convex Parabola (U-shape)
E. Concave parabola (up-side-down U)
F. None of the above

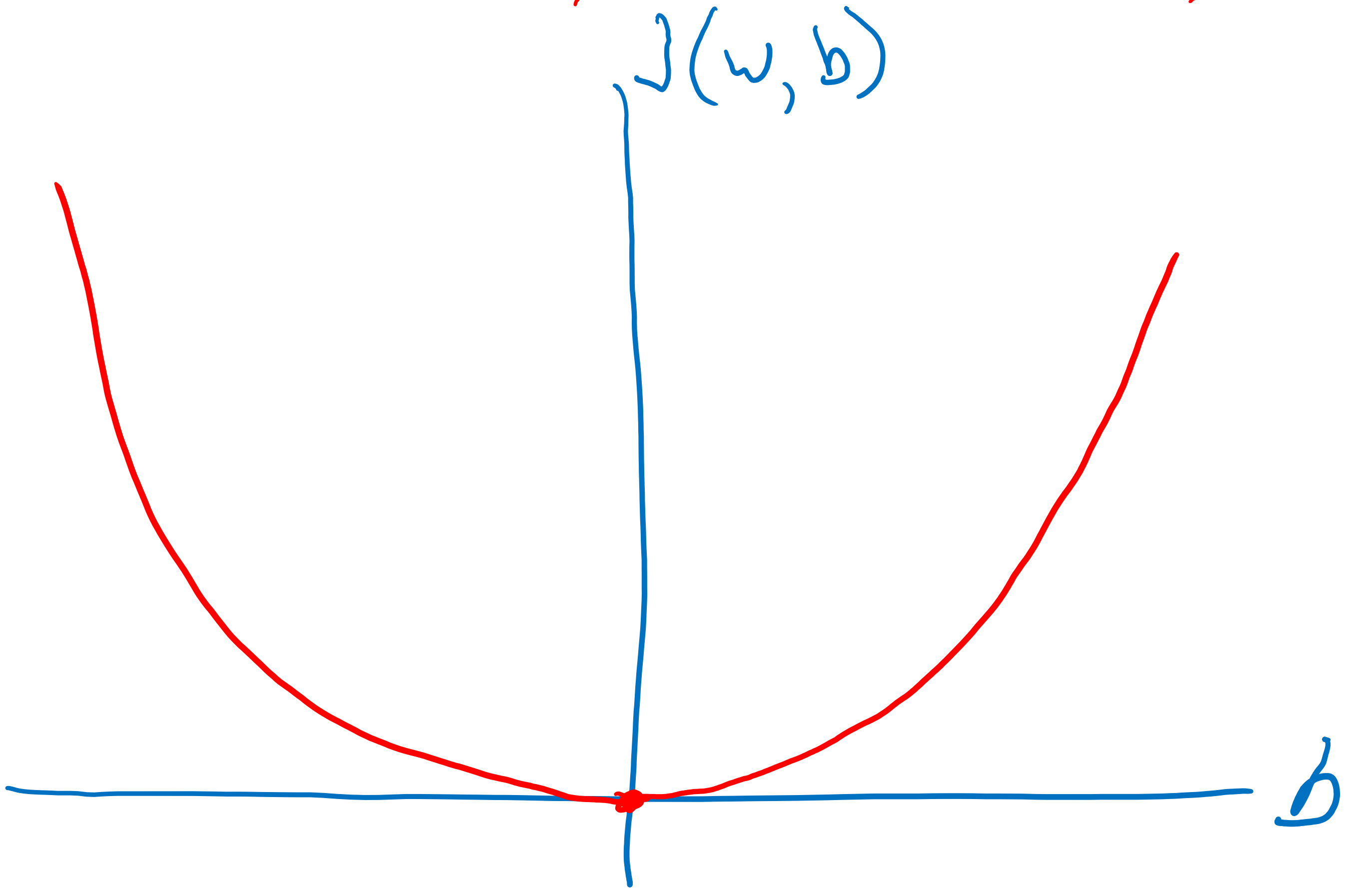


Linear Regression

Optimizing the objective

$$J(w,b) = \frac{1}{2}\left[\left(y^{(1)} - \left(wx^{(1)} + b\right)\right)^2 + \left(y^{(2)} - \left(wx^{(2)} + b\right)\right)^2\right]$$

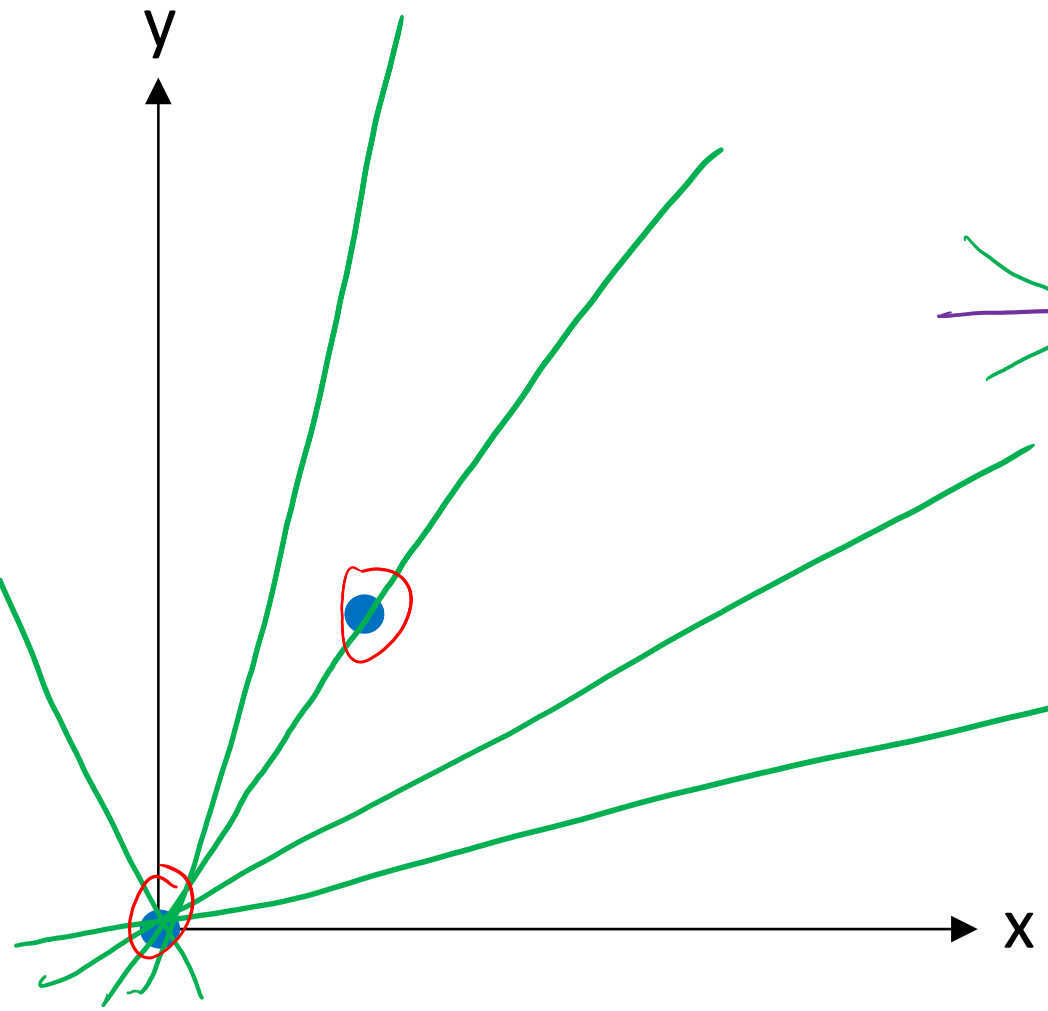


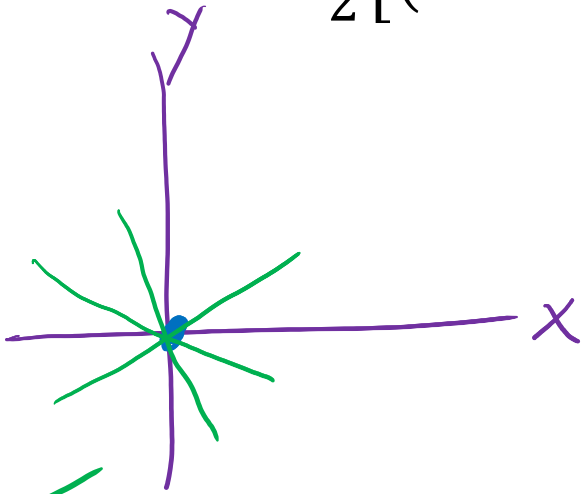


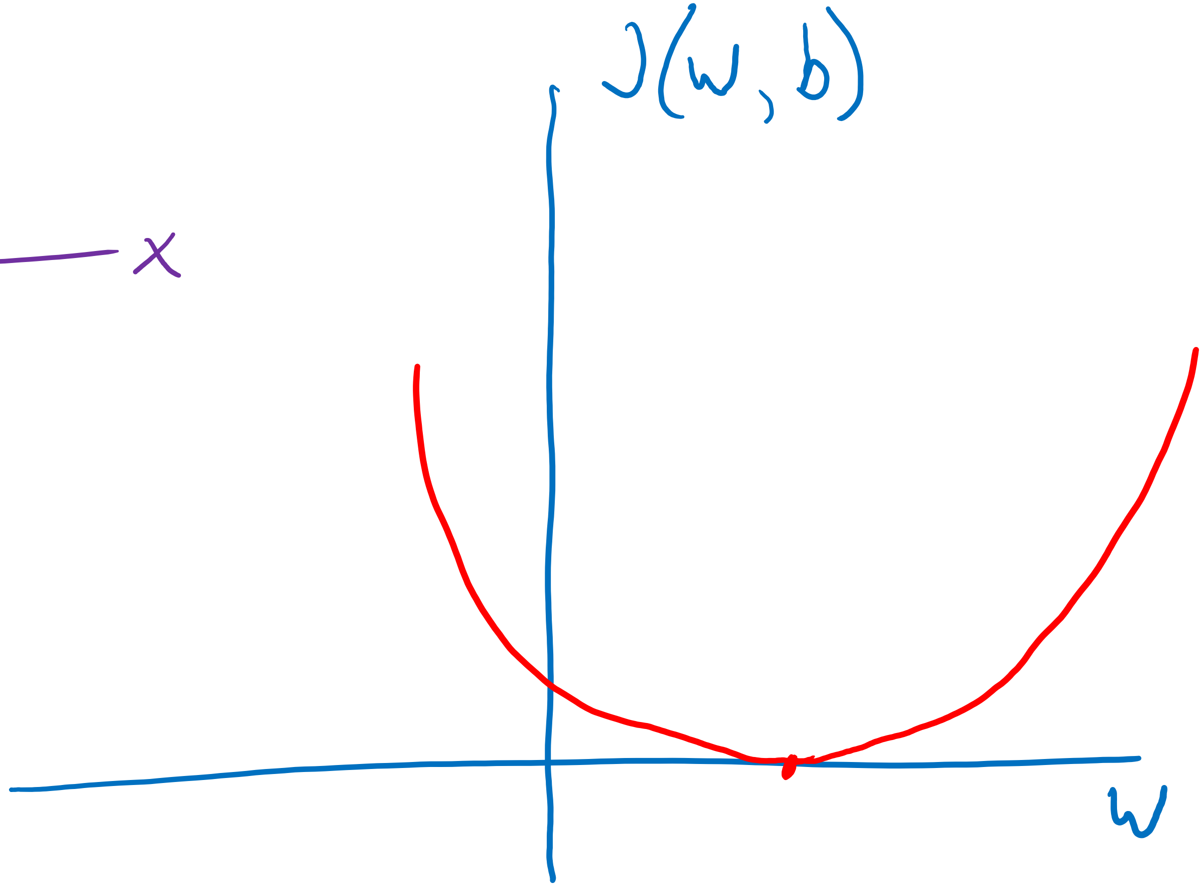
Linear Regression

Optimizing the objective

$$J(w,b) = \frac{1}{2}\left[\left(y^{(1)} - \left(wx^{(1)} + b\right)\right)^2 + \left(y^{(2)} - \left(wx^{(2)} + b\right)\right)^2\right]$$

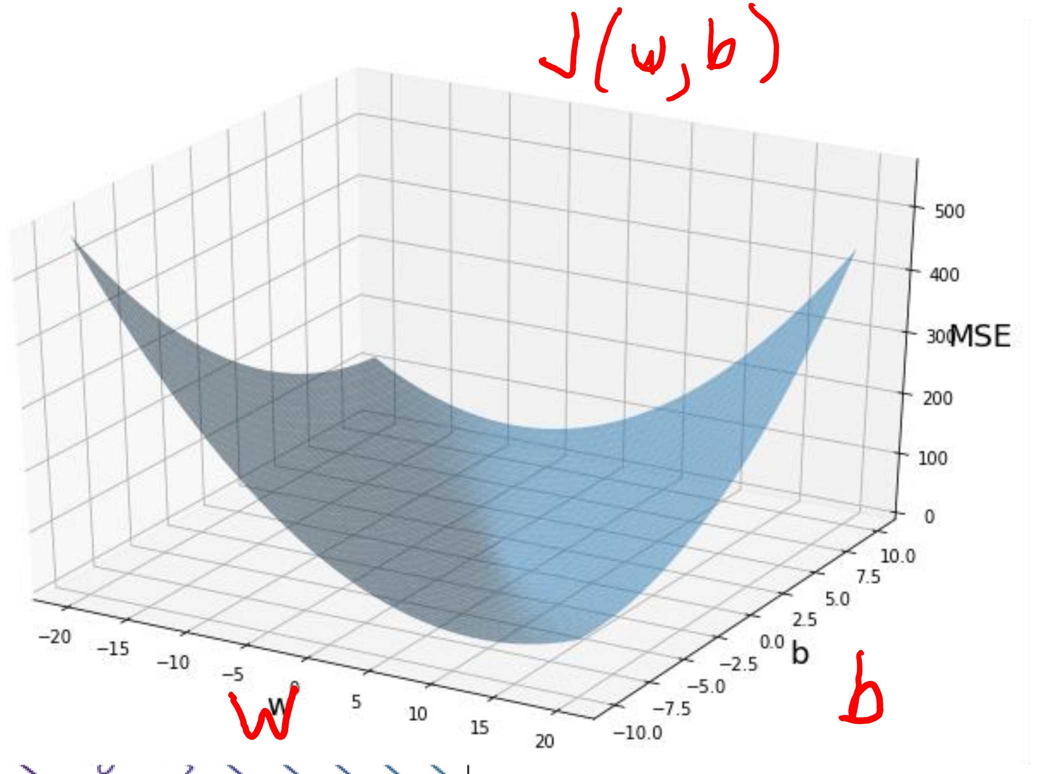


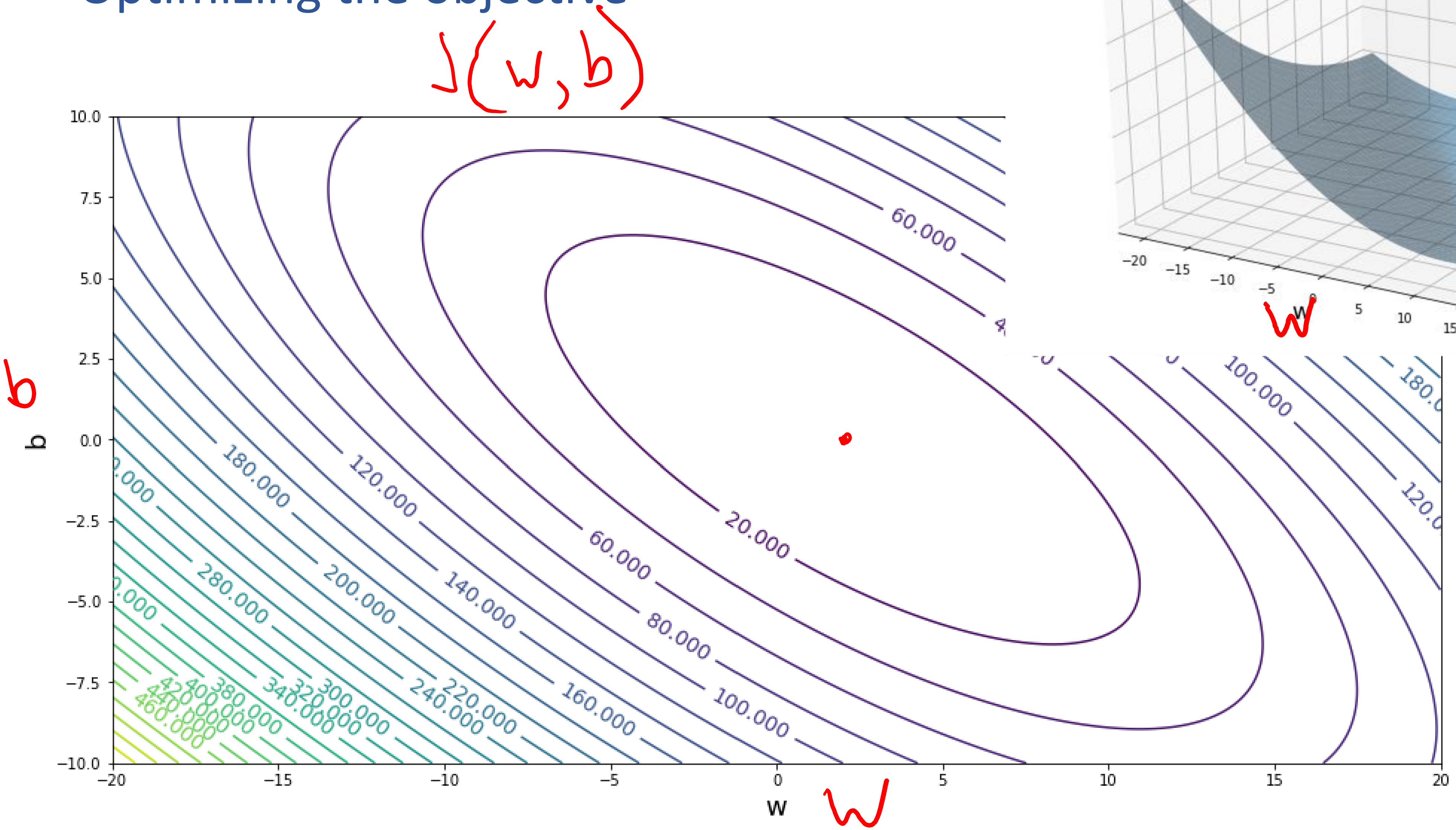




Linear Regression

Optimizing the objective



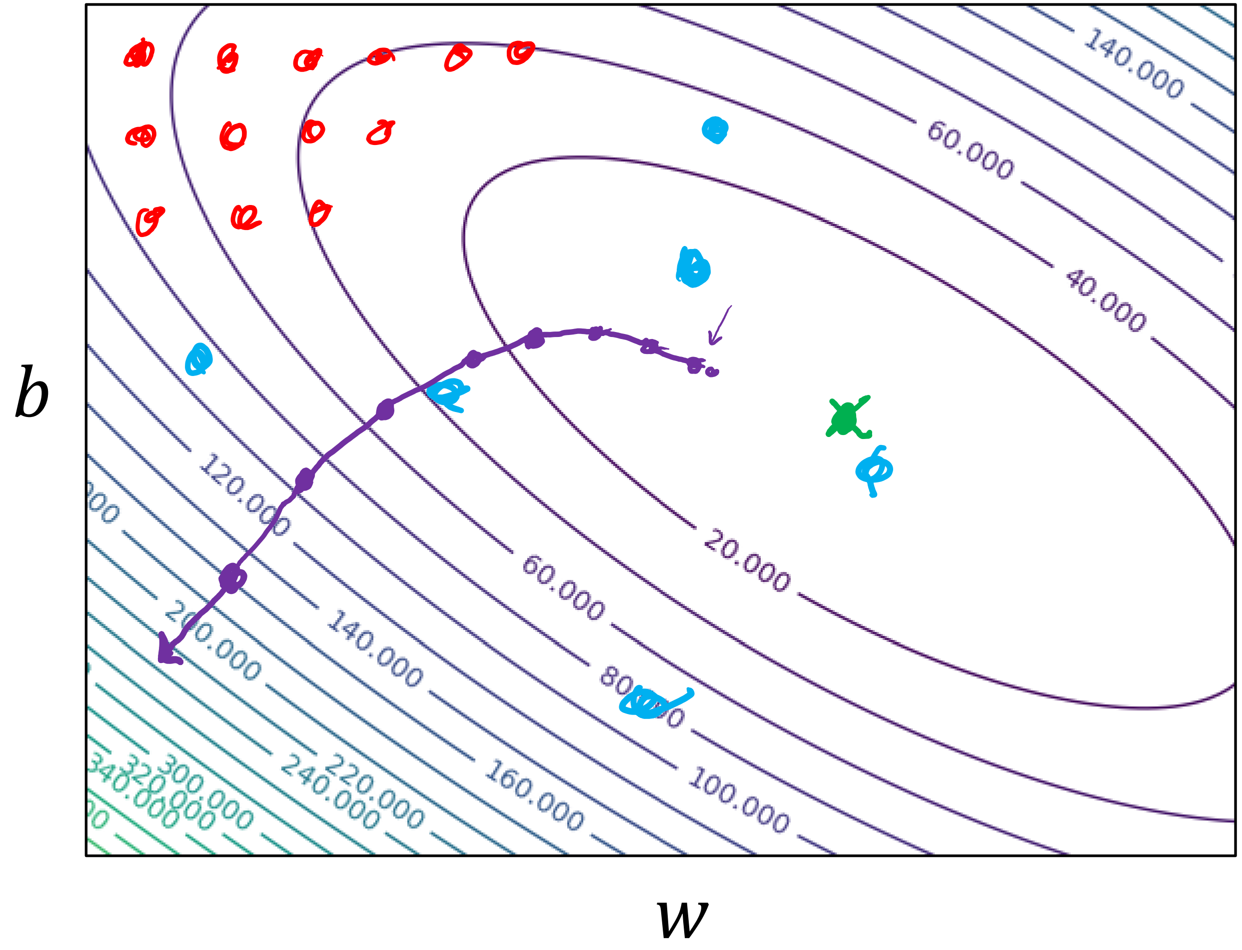


Linear Regression

Methods for optimizing the objective

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

$J(w, b)$



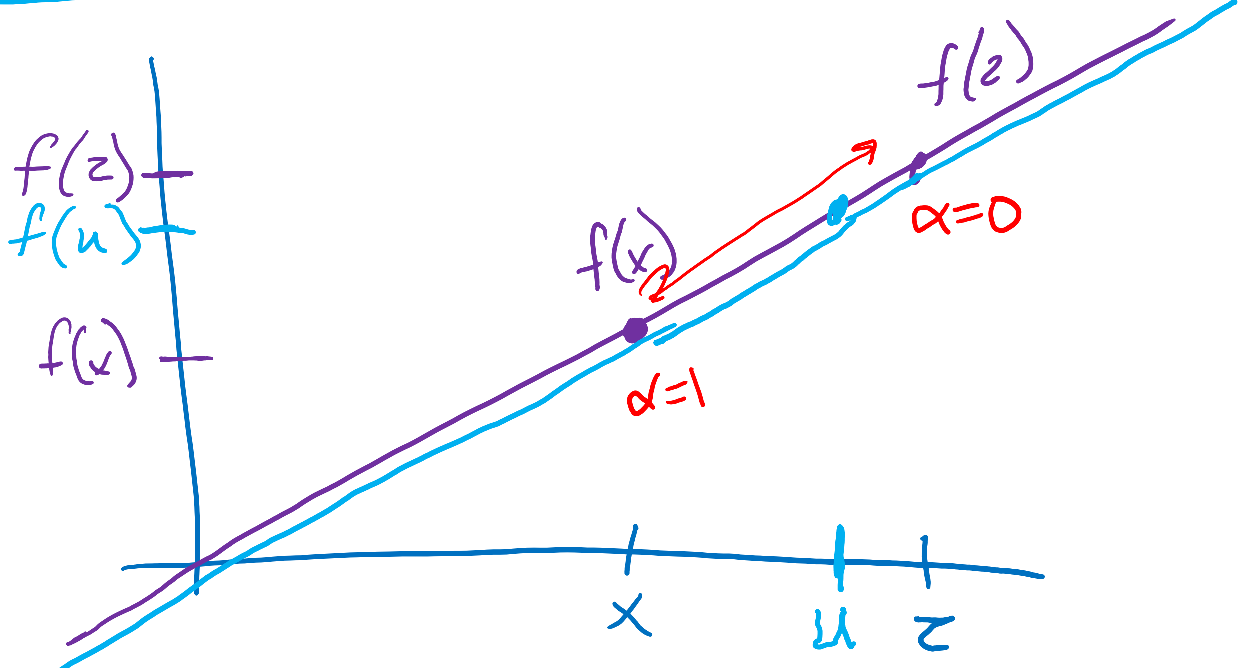
Optimization

Linear function

If $f(\boldsymbol{x})$ is linear, then:

- $f(\boldsymbol{x}+\boldsymbol{z}) = f(\boldsymbol{x}) + f(\boldsymbol{z})$
- $f(\alpha\boldsymbol{x}) = \alpha f(\boldsymbol{x}) \quad \forall\alpha$
- $f(\alpha\boldsymbol{x}+(1-\alpha)\boldsymbol{z}) = \alpha f(\boldsymbol{x}) + (1-\alpha)f(\boldsymbol{z}) \quad \forall\alpha$

$\alpha = 0.25$



Optimization

Convex function

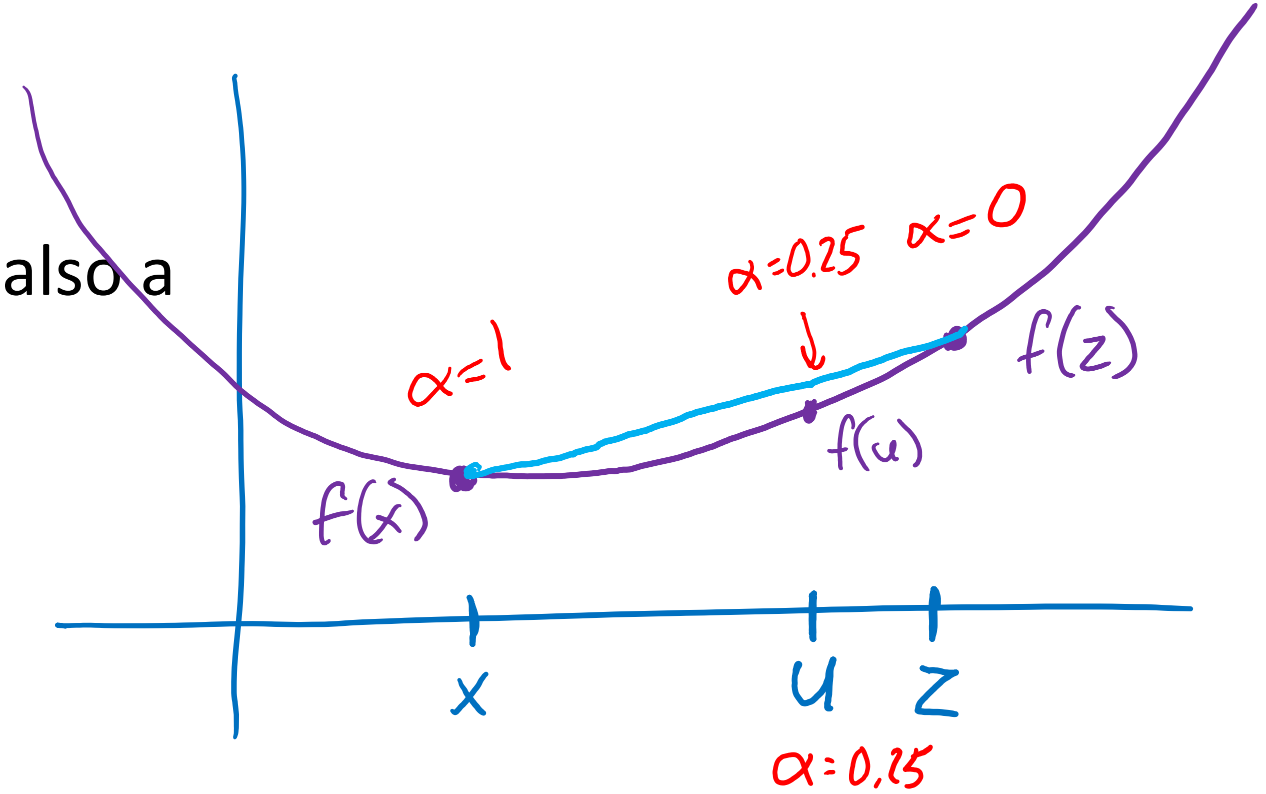
If $f(\boldsymbol{x})$ is convex, then:

- $f(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{z}) \leq \alpha f(\boldsymbol{x}) + (1-\alpha)f(\boldsymbol{z}) \quad \forall\ 0 \leq \alpha \leq 1$

Convex optimization

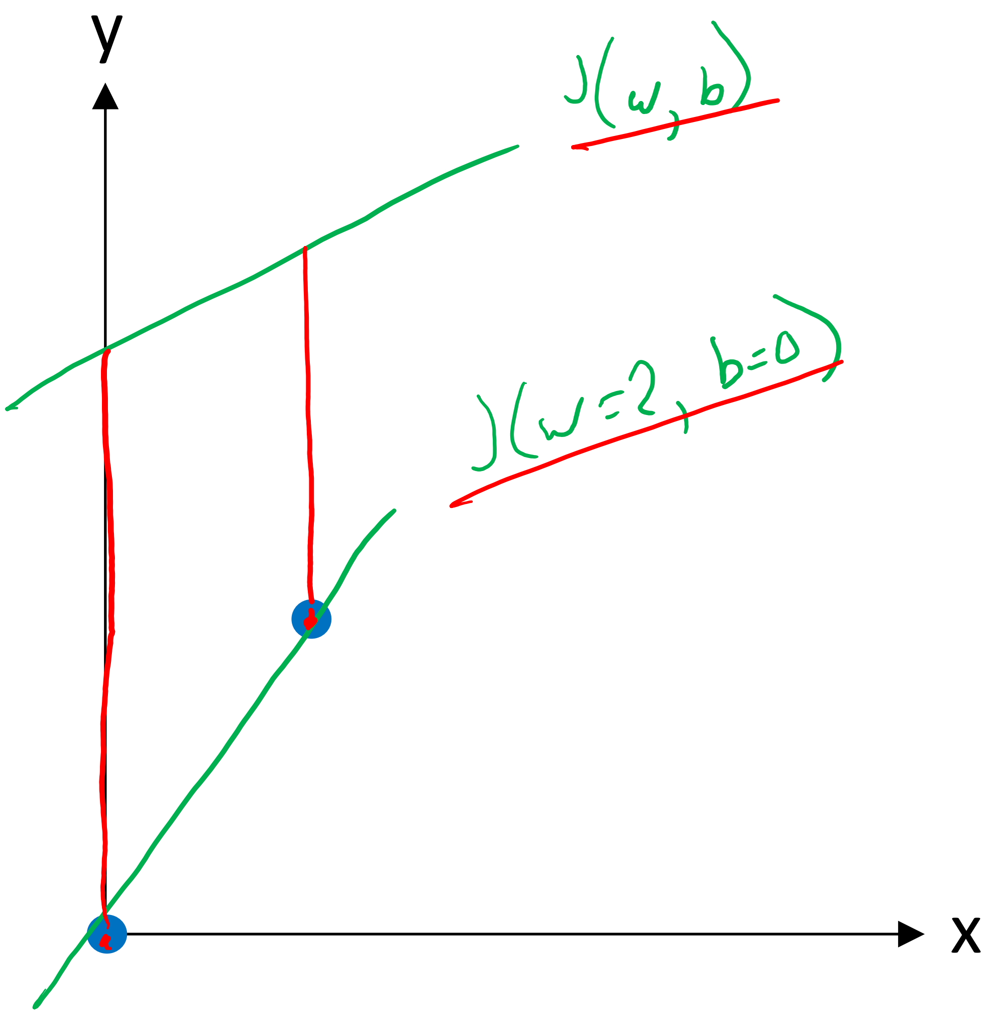
If $f(\boldsymbol{x})$ is convex, then:

- Every local minimum is also a global minimum ☺



Linear Regression

Optimizing the objective

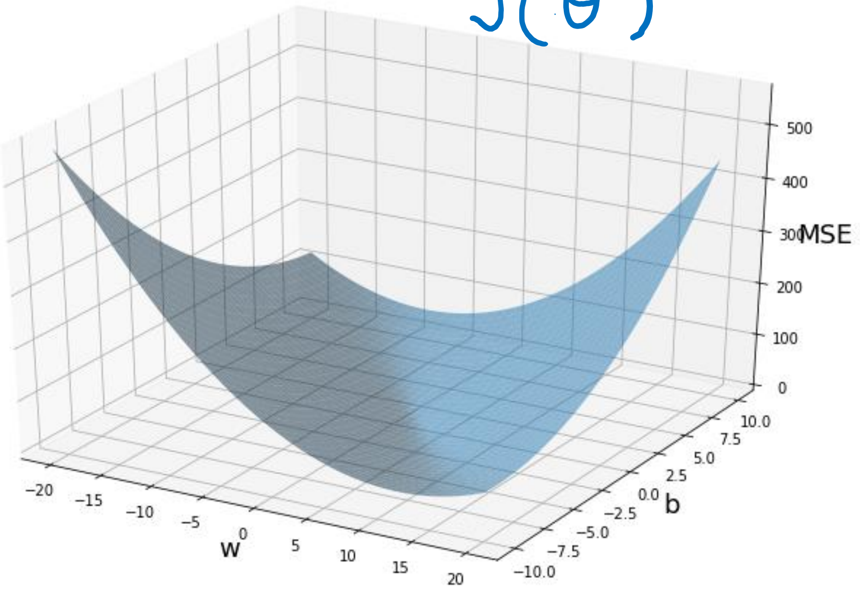


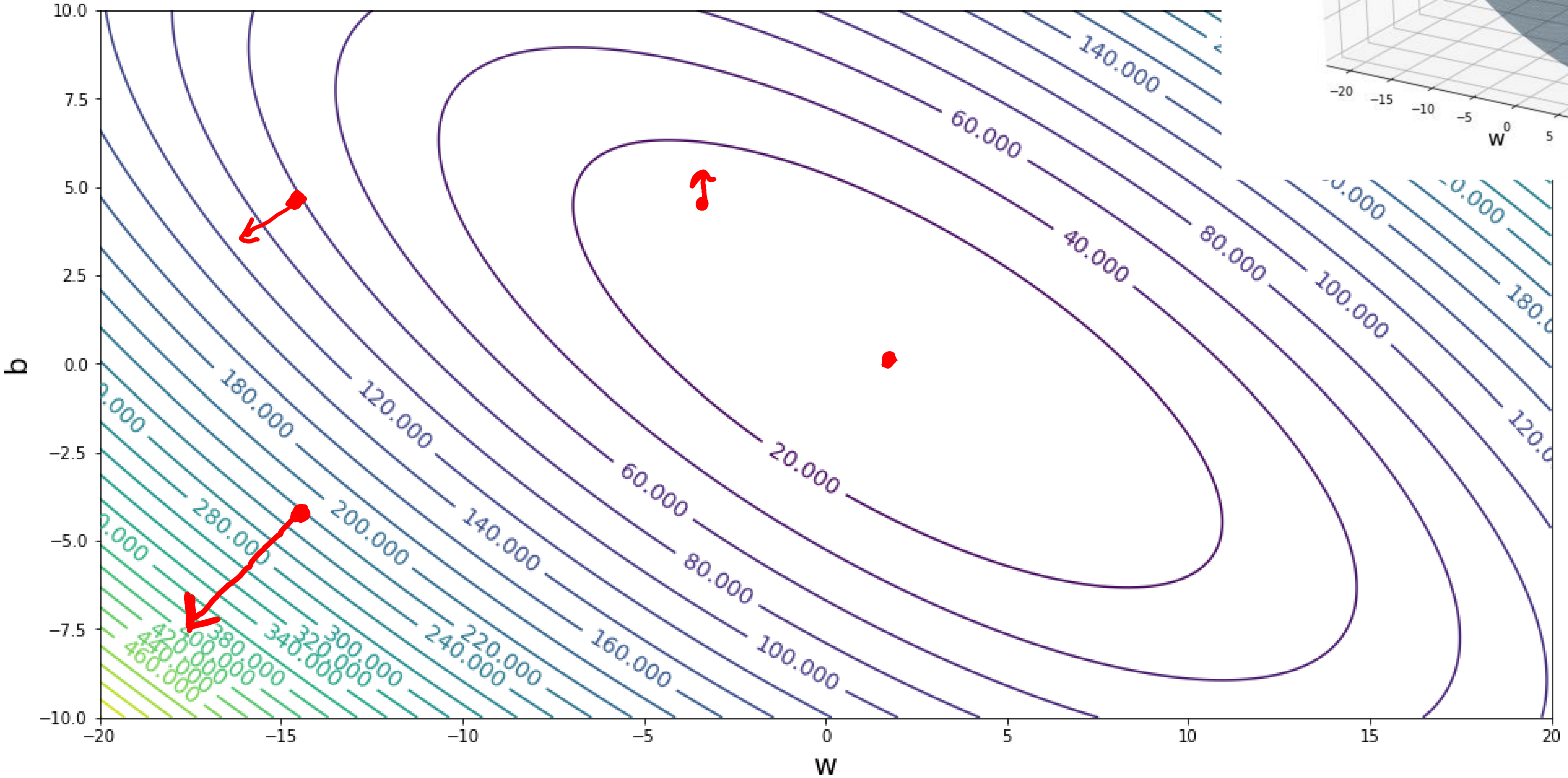
Optimization

Gradients

$$\vec{\theta} = \begin{bmatrix} b \\ w \end{bmatrix}$$

$J(\vec{\theta})$





$$\nabla_{\vec{\theta}} J(\vec{\theta}) = \begin{bmatrix} \frac{\partial J}{\partial b} \\ \frac{\partial J}{\partial w} \end{bmatrix}$$

Optimization

Gradients

function $f: \mathbb{R}^M \to \mathbb{R}$

gradient $\nabla f: \mathbb{R}^M \to \mathbb{R}^M$

Input $\vec{z} \in \mathbb{R}^M$

$$z = \begin{bmatrix} z_1 \\ \vdots \\ z_M \end{bmatrix}$$

$f(\vec{z})$

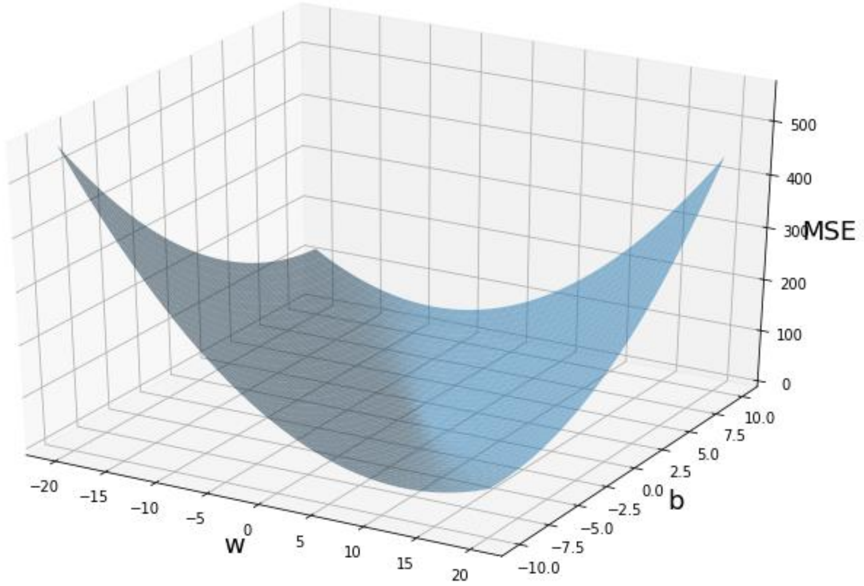
$\nabla f(\vec{z})$

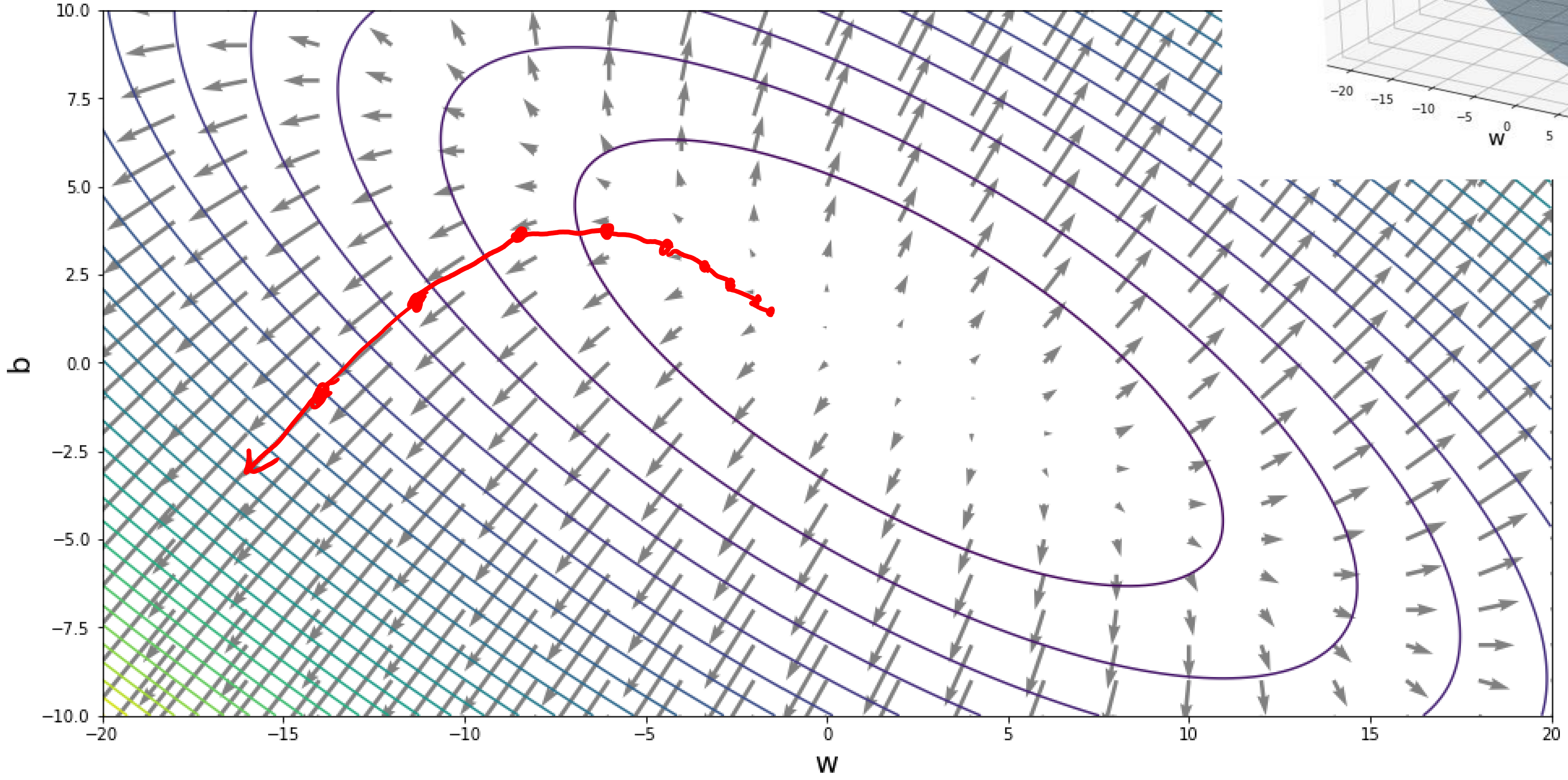
$$\nabla_z f(z) = \begin{bmatrix} \frac{\partial f}{\partial z_1} \\ \frac{\partial f}{\partial z_2} \\ \vdots \\ \frac{\partial f}{\partial z_M} \end{bmatrix}$$

$\nabla_{\vec{z}}\, g(\vec{z}, \vec{u})$

Optimization

Gradients



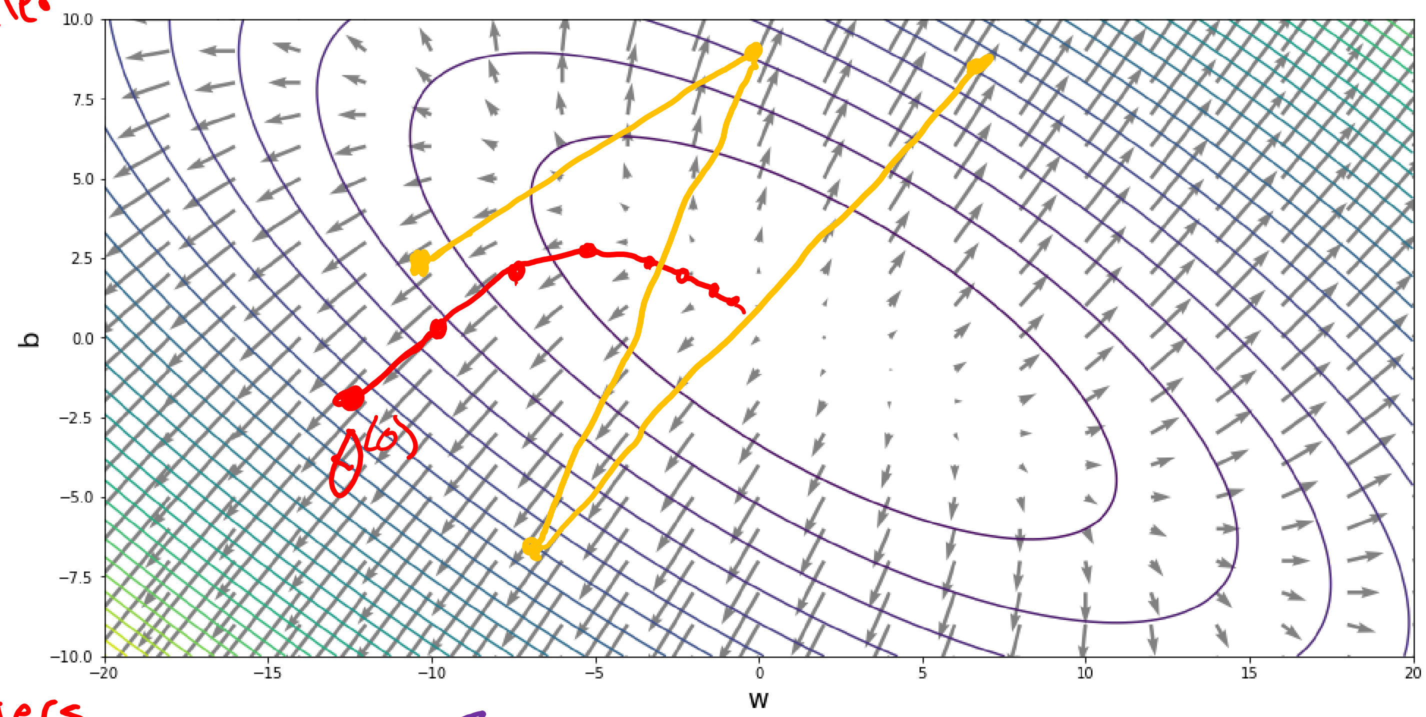


Optimization

hyperparameter

Gradient descent

Choose learning rate
$\alpha > 0$



Initial $\vec{\theta}^{(0)}$ $b^{(0)}$ $w^{(0)}$

parameters

Loop

$$\vec{\theta}^{(t+1)} \leftarrow \vec{\theta}^{(t)} - \alpha \nabla J(\theta^{(t)})$$

$$\begin{cases} b^{(t+1)} \leftarrow b^{(t)} - \alpha \dfrac{\partial J}{\partial b}(b^{(t)}, w^{(t)}) \\ w^{(t+1)} \leftarrow w^{(t)} - \alpha \dfrac{\partial J}{\partial w}(b^{(t)}, w^{(t)}) \end{cases}$$

Linear Regression

$\|\vec{z}\|_2^2 = \sum_{i=1}^{N} z_i^2 = \sum_{i=1}^{N} z_i z_i = \vec{z}^T \vec{z}$

Expanding objective before computing gradient

$$J(\boldsymbol{\theta}) = \frac{1}{N}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2$$
$$= \frac{1}{N}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})$$
$$= \frac{1}{N}\left(\boldsymbol{y}^T - \boldsymbol{\theta}^T\boldsymbol{X}^T\right)(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})$$
$$= \frac{1}{N}\left(\boldsymbol{y}^T\boldsymbol{y} - \underline{\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{y}} - \underline{\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{\theta}} + \boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$

these two are the same

$$= \frac{1}{N}\left(\boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$

Linear Regression

Gradient of objective with respect to parameters

$$J(\boldsymbol{\theta}) = \frac{1}{N}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2$$

$$= \frac{1}{N}\left(\boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$

$$\nabla J(\boldsymbol{\theta}) = \frac{1}{N}\left(0 - \underline{2\boldsymbol{X}^T\boldsymbol{y}} + \underline{\cancel{2\boldsymbol{\theta}^T\boldsymbol{X}^T\boldsymbol{X}}}\right)$$

$$= \frac{1}{N}\left(0 - 2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$

$$= \frac{2}{N}\left(-\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$

Dimension mismatch

$$\frac{\partial \boldsymbol{z}^T\boldsymbol{u}}{\partial \boldsymbol{z}} = \boldsymbol{u}$$

-- or --

$$\frac{\partial \boldsymbol{z}^T\boldsymbol{u}}{\partial \boldsymbol{z}} = \boldsymbol{u}^T$$

$$\frac{\partial \boldsymbol{z}^T\boldsymbol{A}\boldsymbol{z}}{\partial \boldsymbol{z}} = (\boldsymbol{A} + \boldsymbol{A}^T)\boldsymbol{z}$$

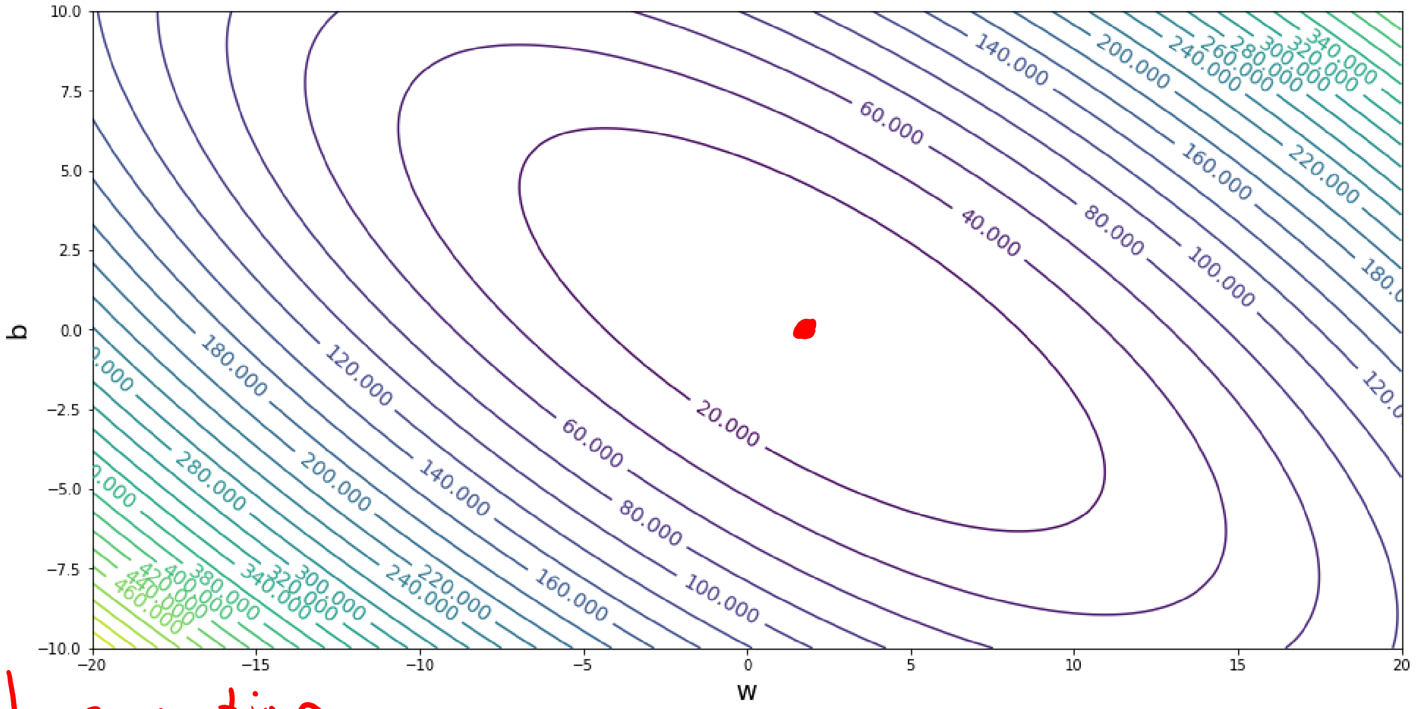
-- or --

$$\frac{\partial \boldsymbol{z}^T\boldsymbol{A}\boldsymbol{z}}{\partial \boldsymbol{z}} = \boldsymbol{z}^T(\boldsymbol{A} + \boldsymbol{A}^T)$$

Linear Regression

Closed-form solution

$$\nabla J(\boldsymbol{\theta}) = \frac{2}{N}\left(-\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}\right)$$



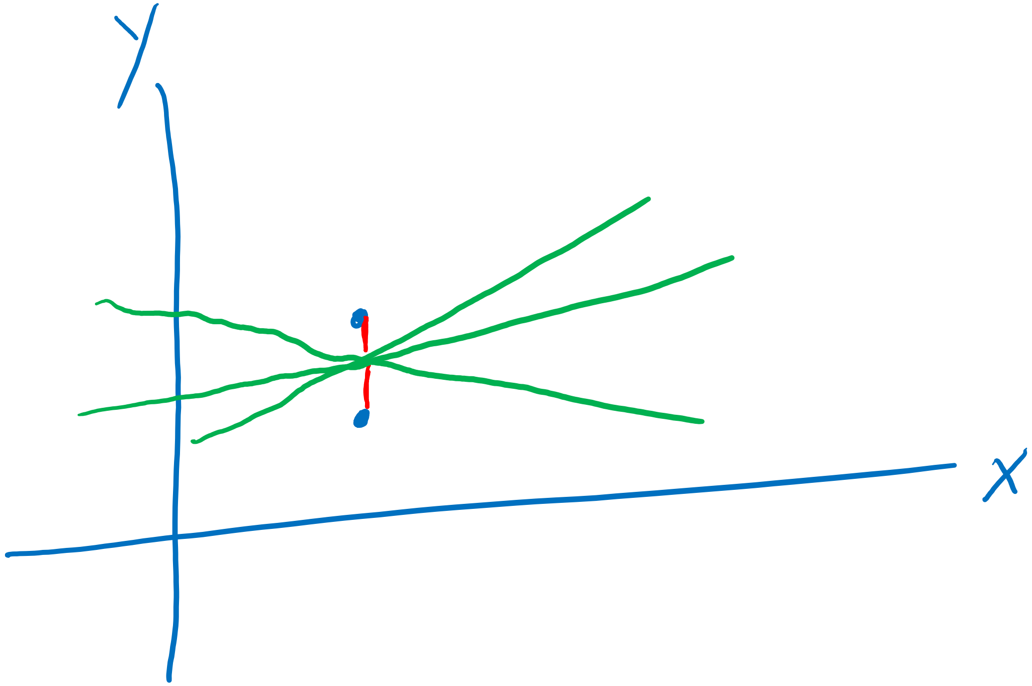
$$\nabla J(\theta) = 0$$

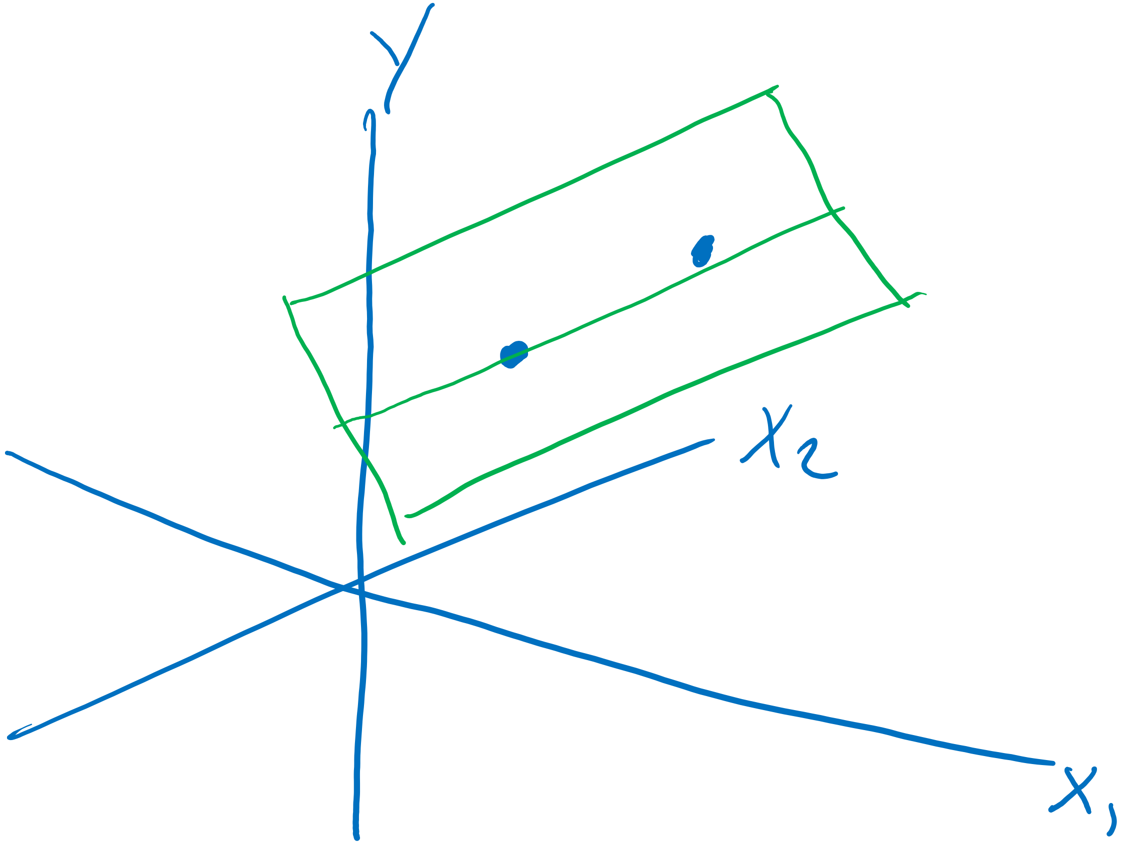
$$X^T X \theta = X^T y \quad \leftarrow \text{Normal equation}$$

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

Linear Regression

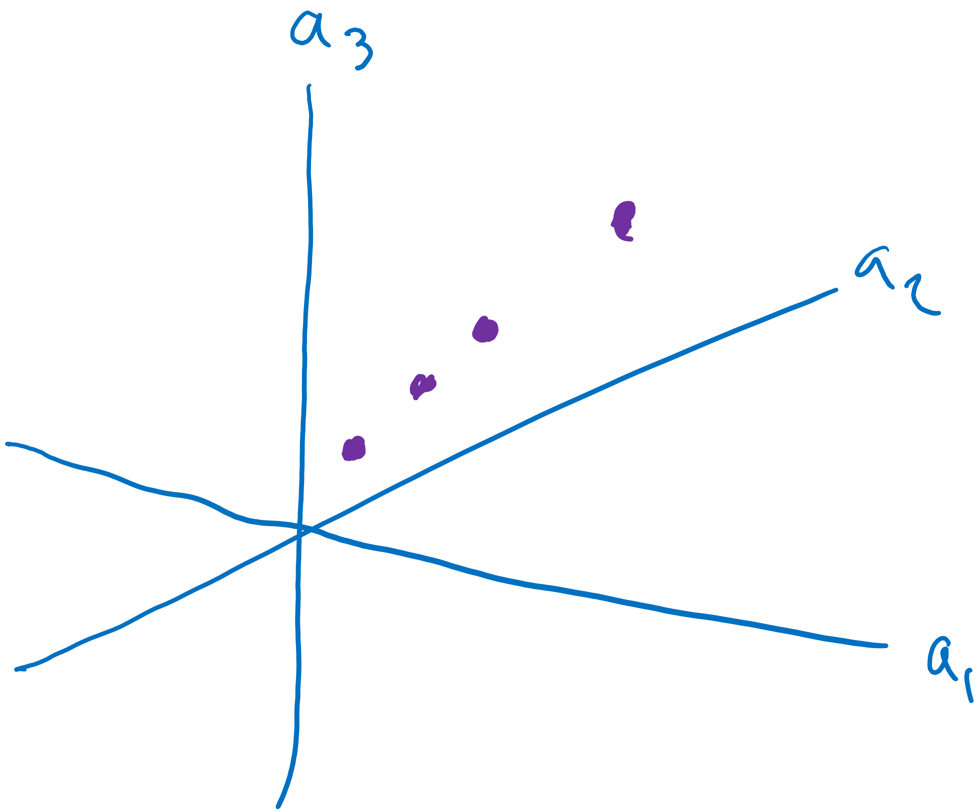
Number of solutions





A Note on Matrix Rank

Underlying dimensionality of the data



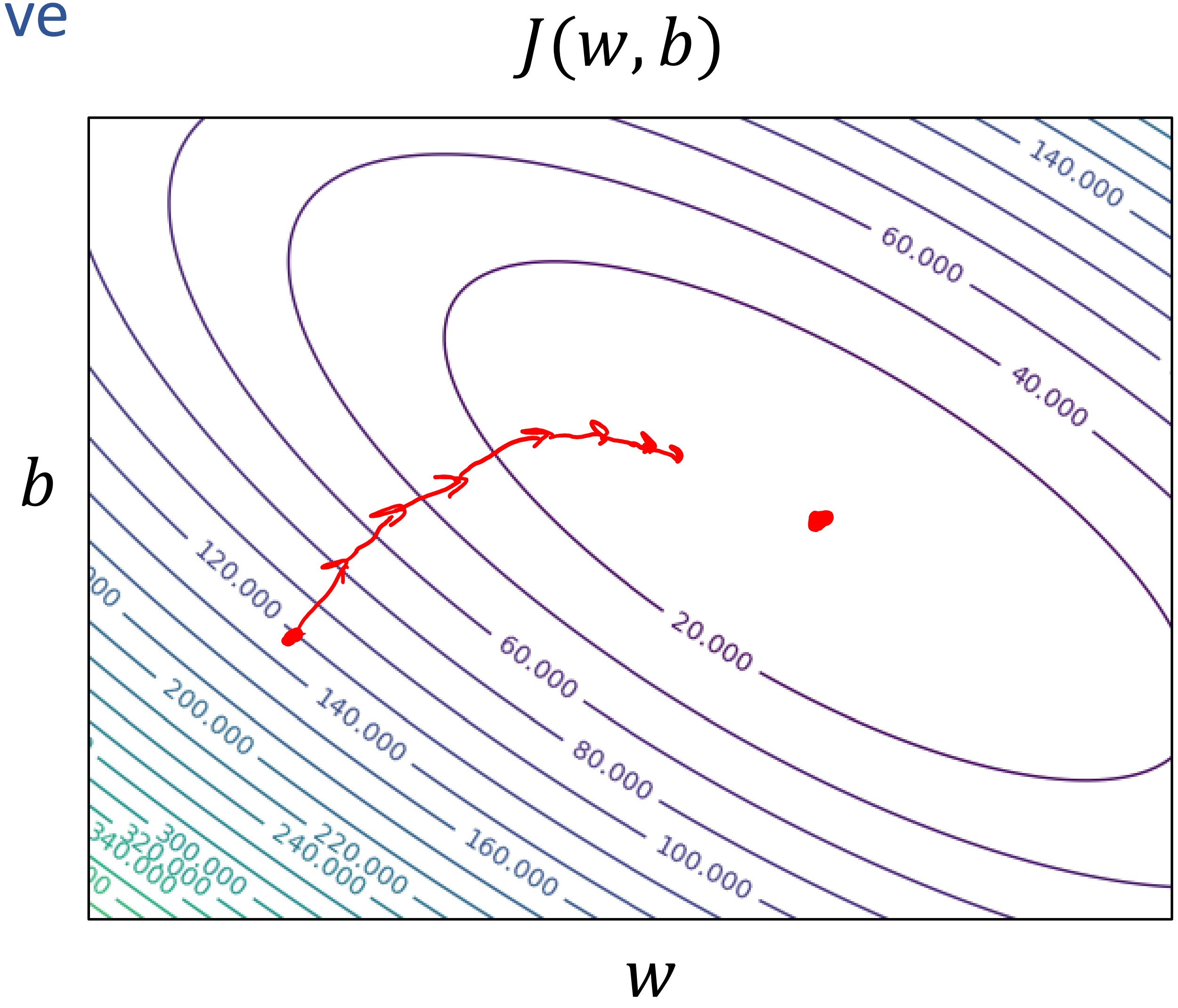
$$A = \begin{array}{ccc} a_1 & a_2 & a_3 \\ \end{array}$$
$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 5 & 5 & 5 \\ 3 & 3 & 3 \end{bmatrix}$$

$\text{Rank}(A) = 1$

Linear Regression

Methods for optimizing the objective

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

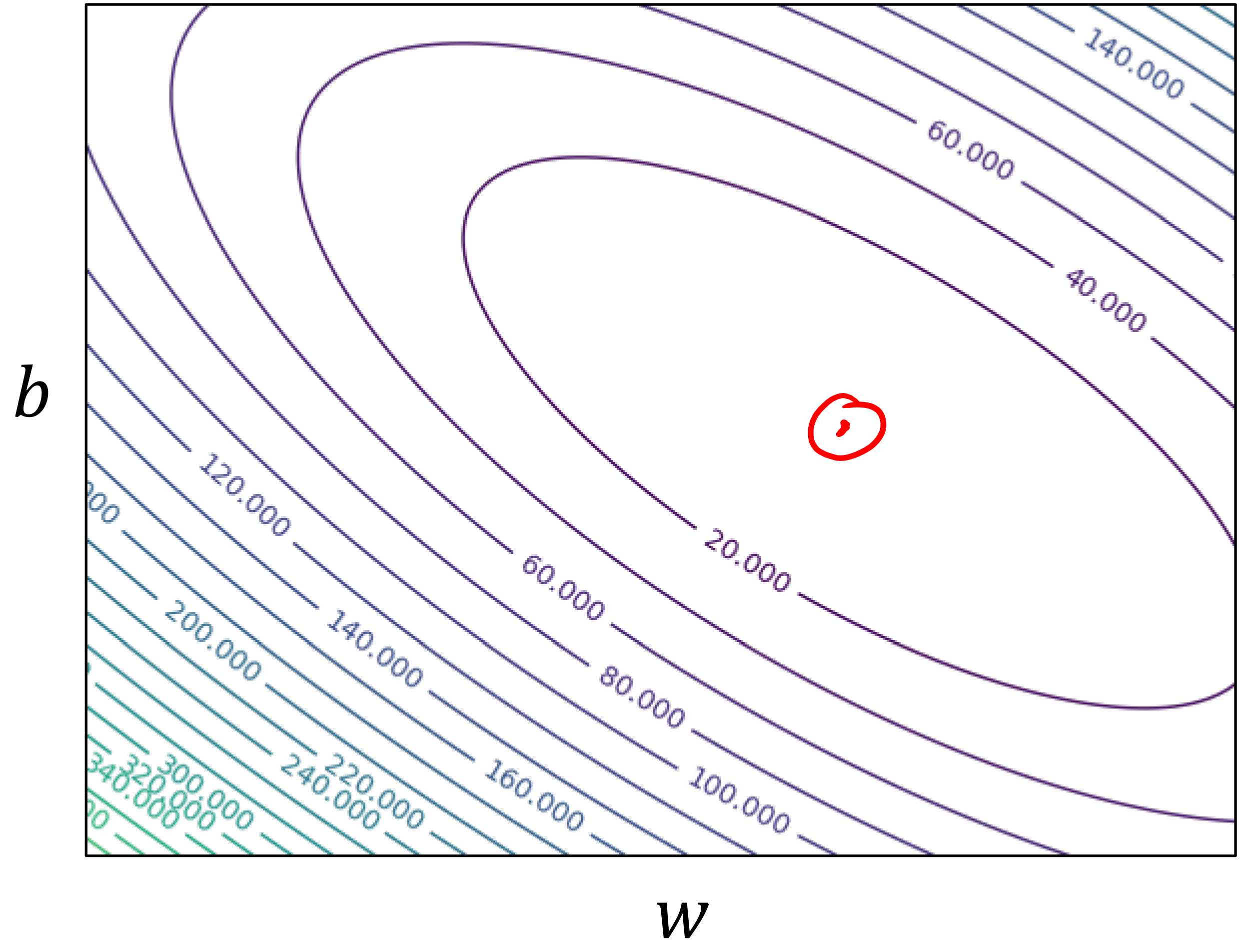


Linear Regression

Methods for optimizing the objective

- Grid search
- Random search
- Closed-form solution
- (Batch) Gradient descent
- **Stochastic gradient descent**

$J(w,b)$



Linear Regression Gradient Descent

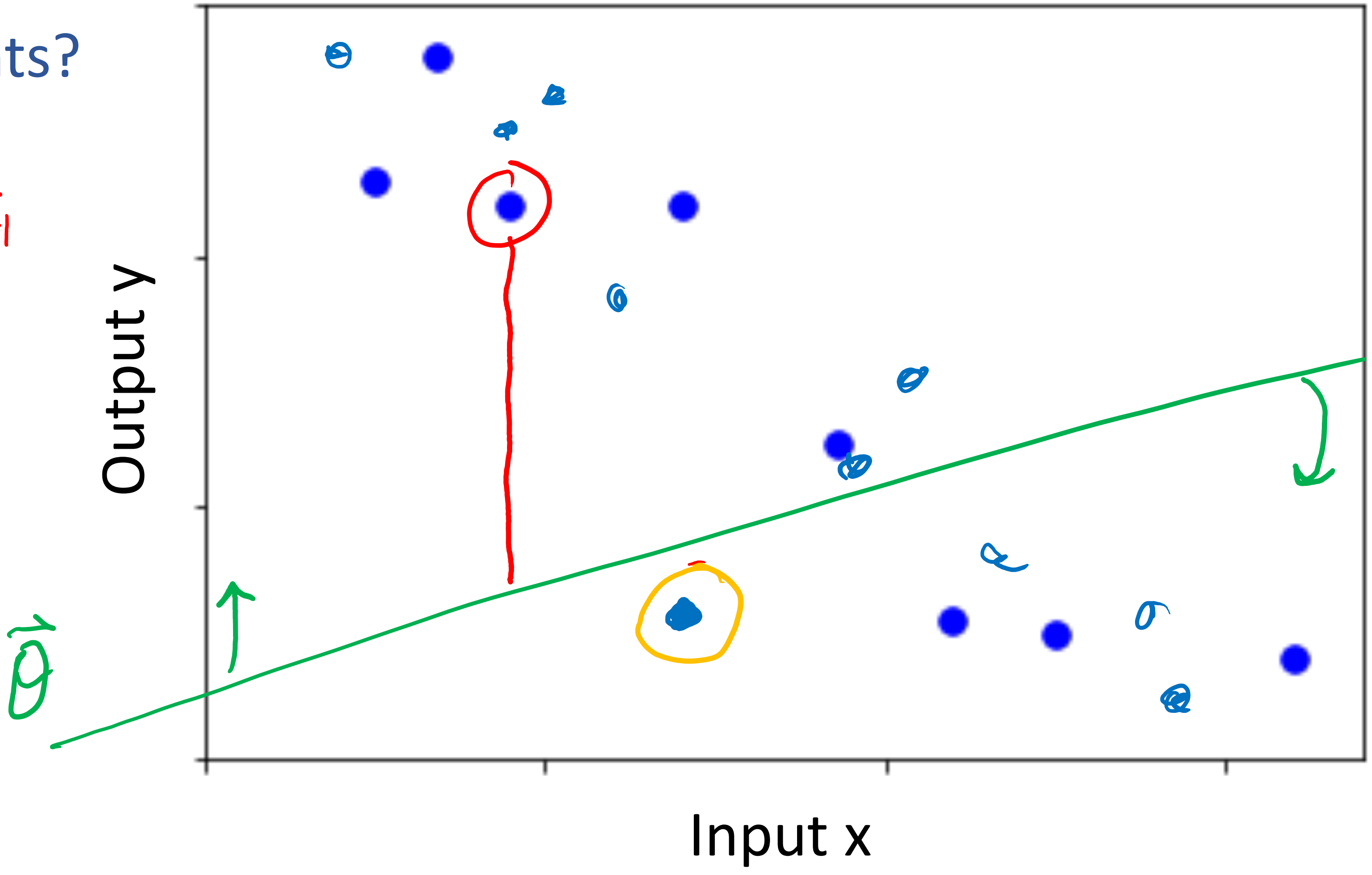
$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \theta^T x^{(i)}\right)^2$$

$$\nabla_\theta J$$

What happens in gradient descent when we have N=1,000,000 training points?

$$\nabla_\theta J = \begin{bmatrix} \frac{\partial J}{\partial b} \\ \\ \frac{\partial J}{\partial w} \end{bmatrix}$$

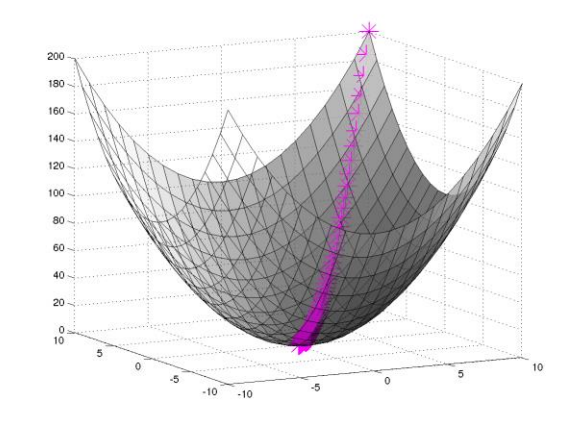
$$\sum_{i=1}^{N}$$



(Batch) Gradient Descent

Algorithm 1 Gradient Descent

procedure GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
  while not converged **do**
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \underbrace{\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})}$
  return $\boldsymbol{\theta}$

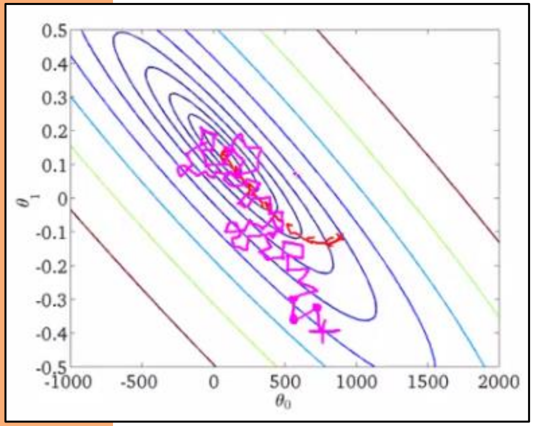


M

Stochastic Gradient Descent (SGD)

Algorithm 2 Stochastic Gradient Descent (SGD)

procedure SGD($\mathcal{D}, \boldsymbol{\theta}^{(0)}$)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
    while not converged **do**
        $i \sim$ Uniform($\{1, 2, \ldots, N\}$)
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
    return θ

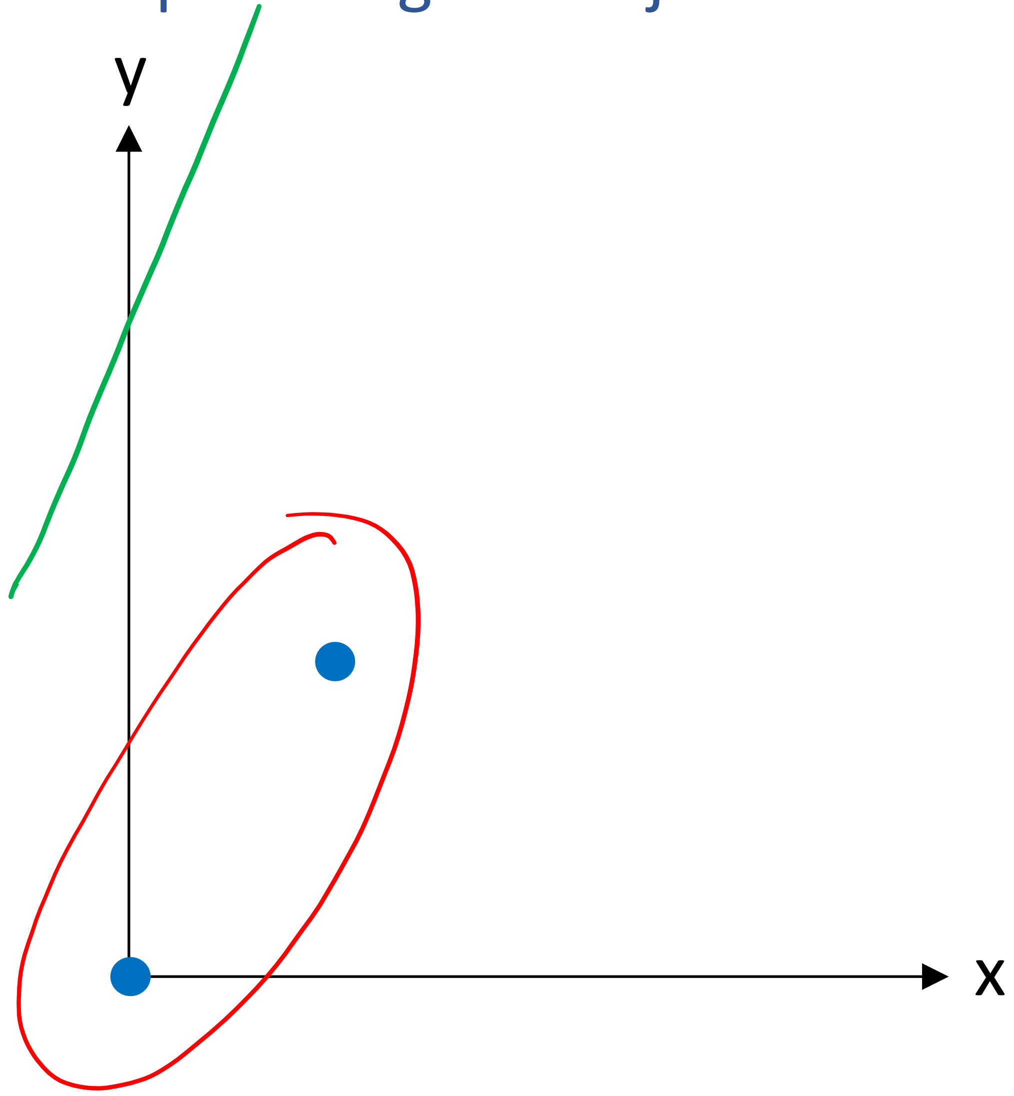


We need a per-example objective:

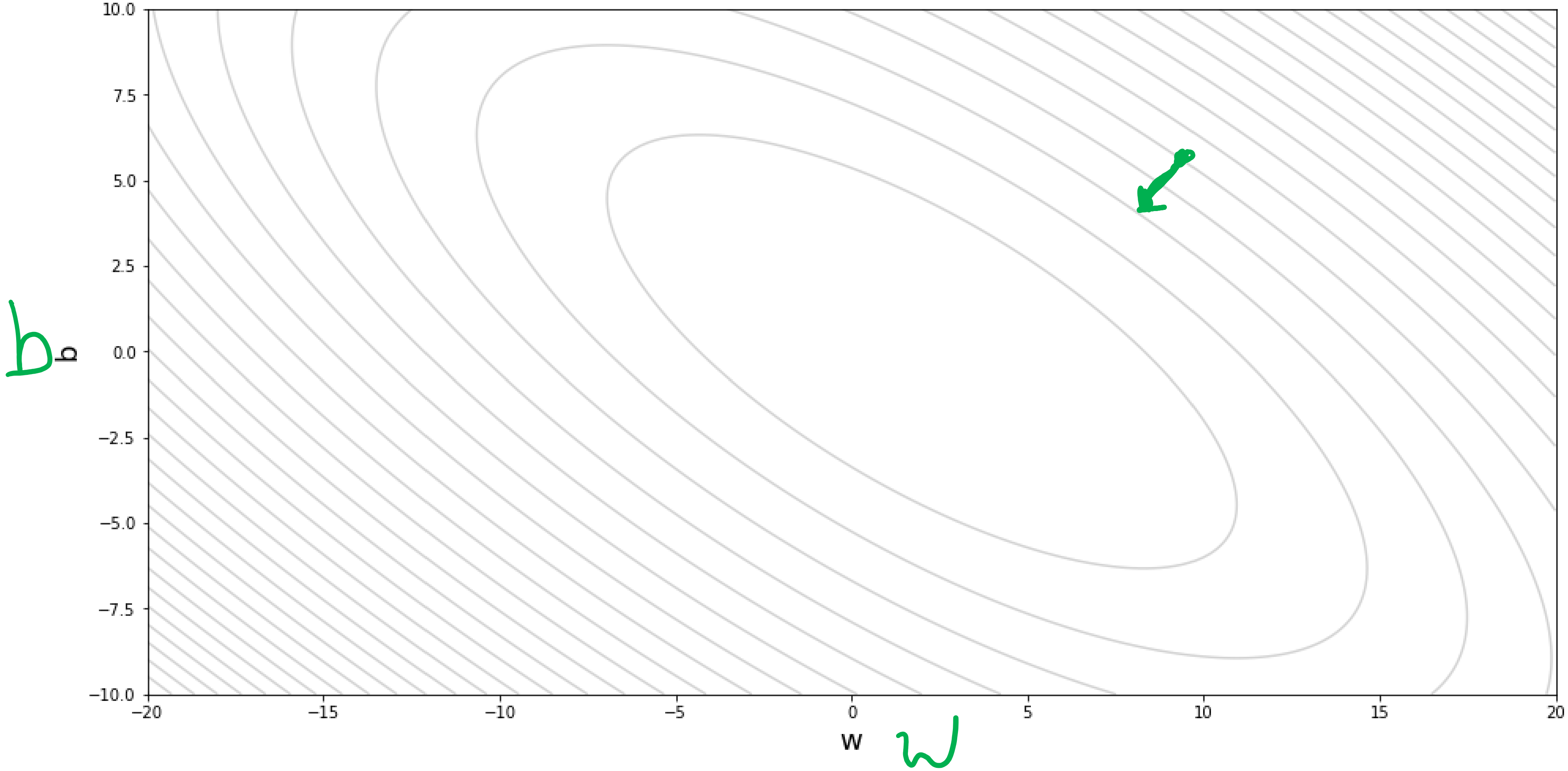
$$\text{Let } J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$

Linear Regression

Optimizing the objective

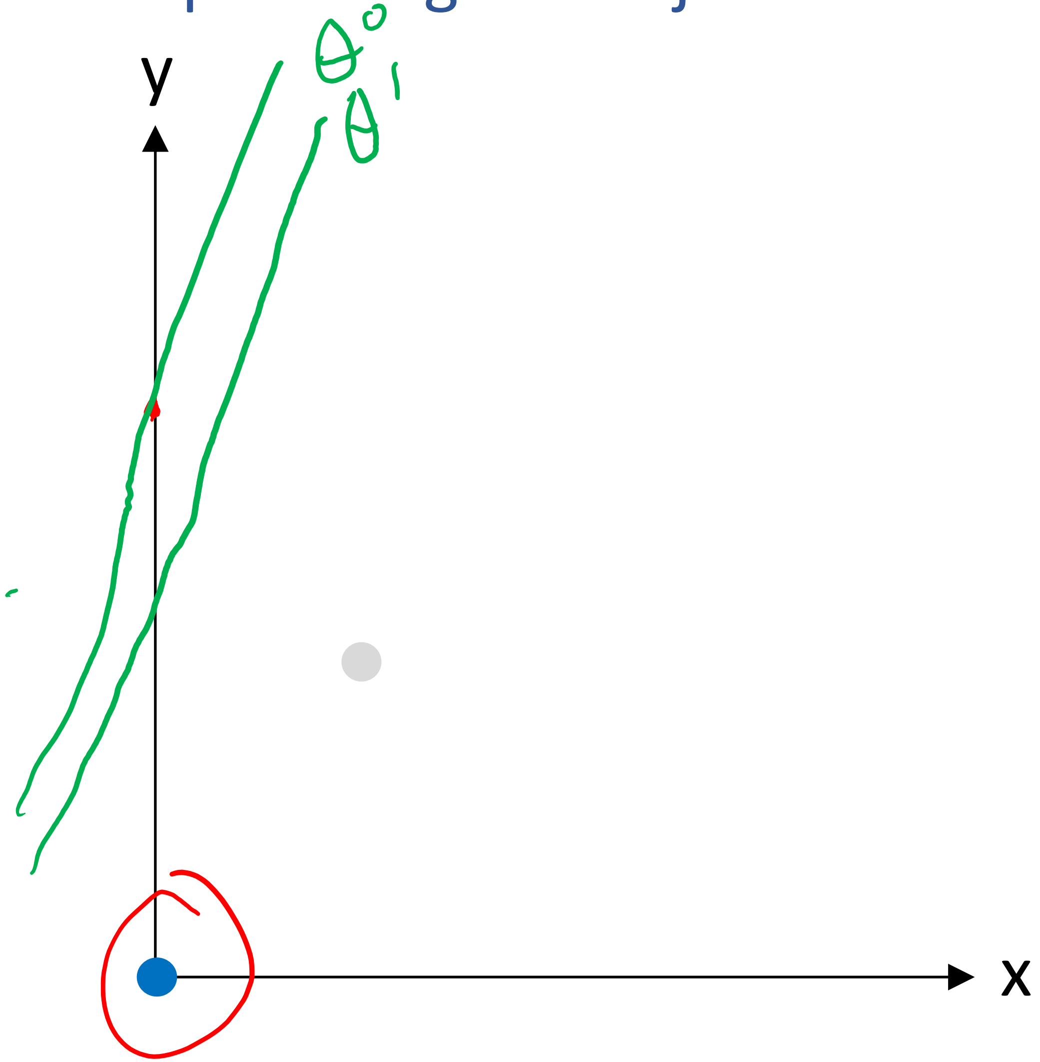


Stochastic Gradient Descent

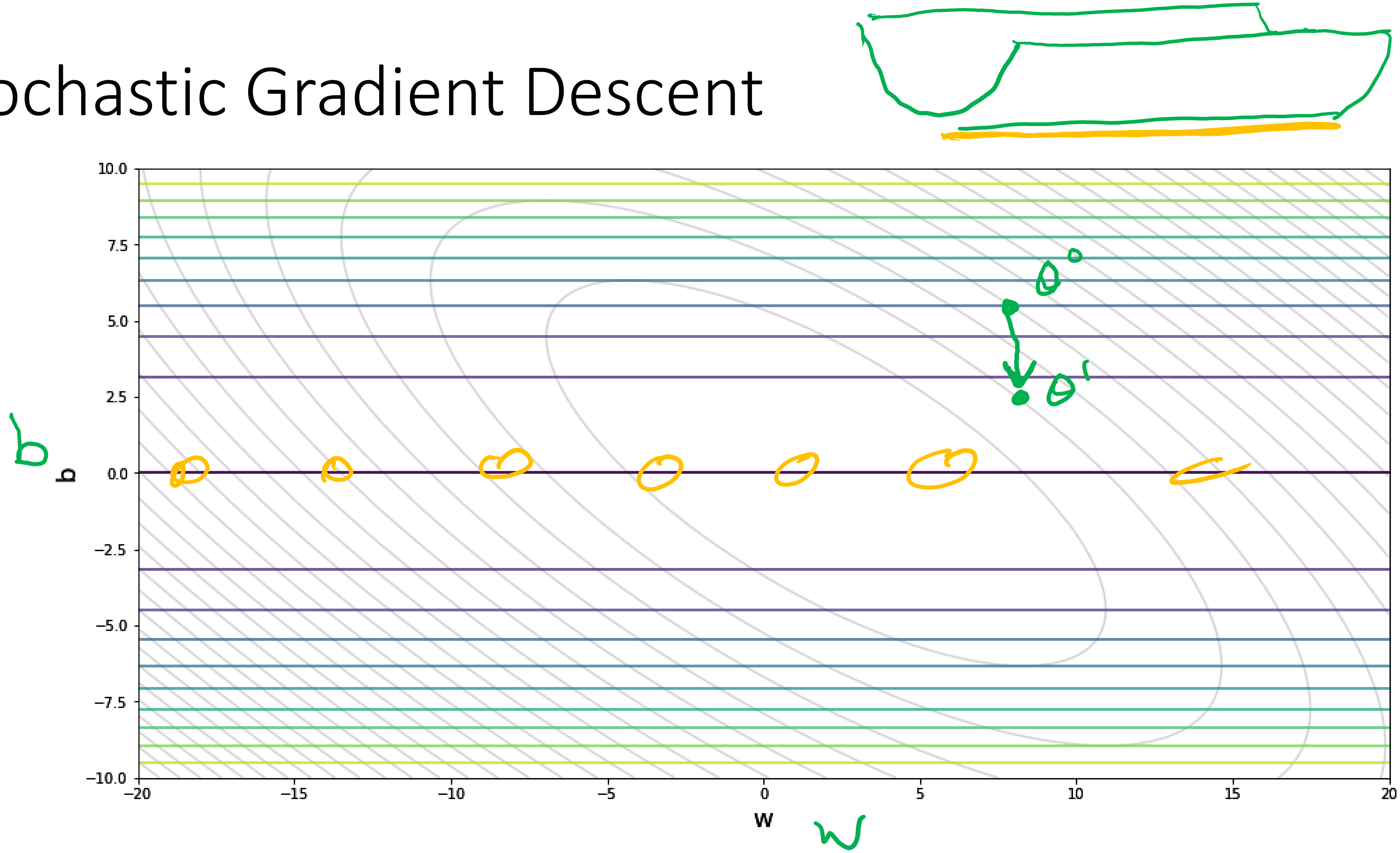


Linear Regression

Optimizing the objective

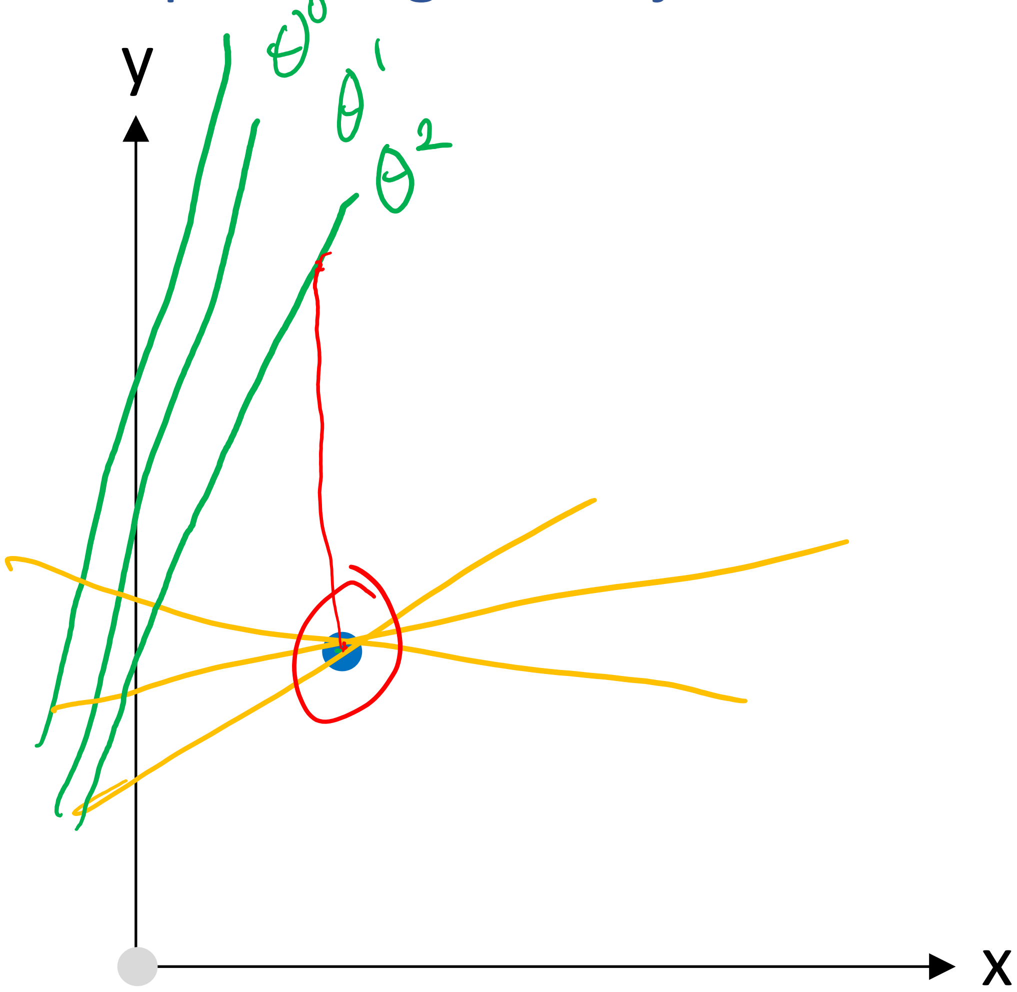


Stochastic Gradient Descent

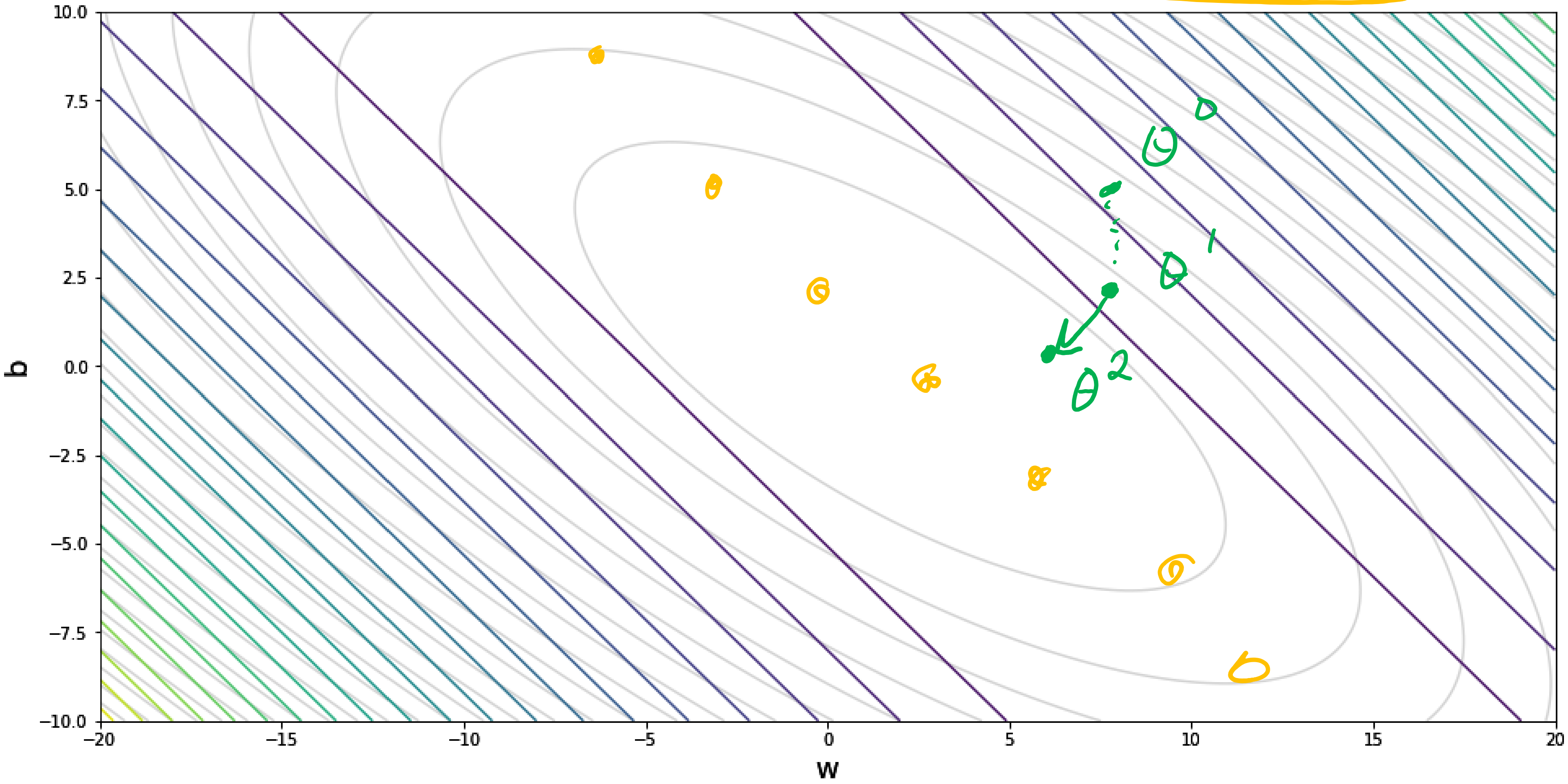


Linear Regression

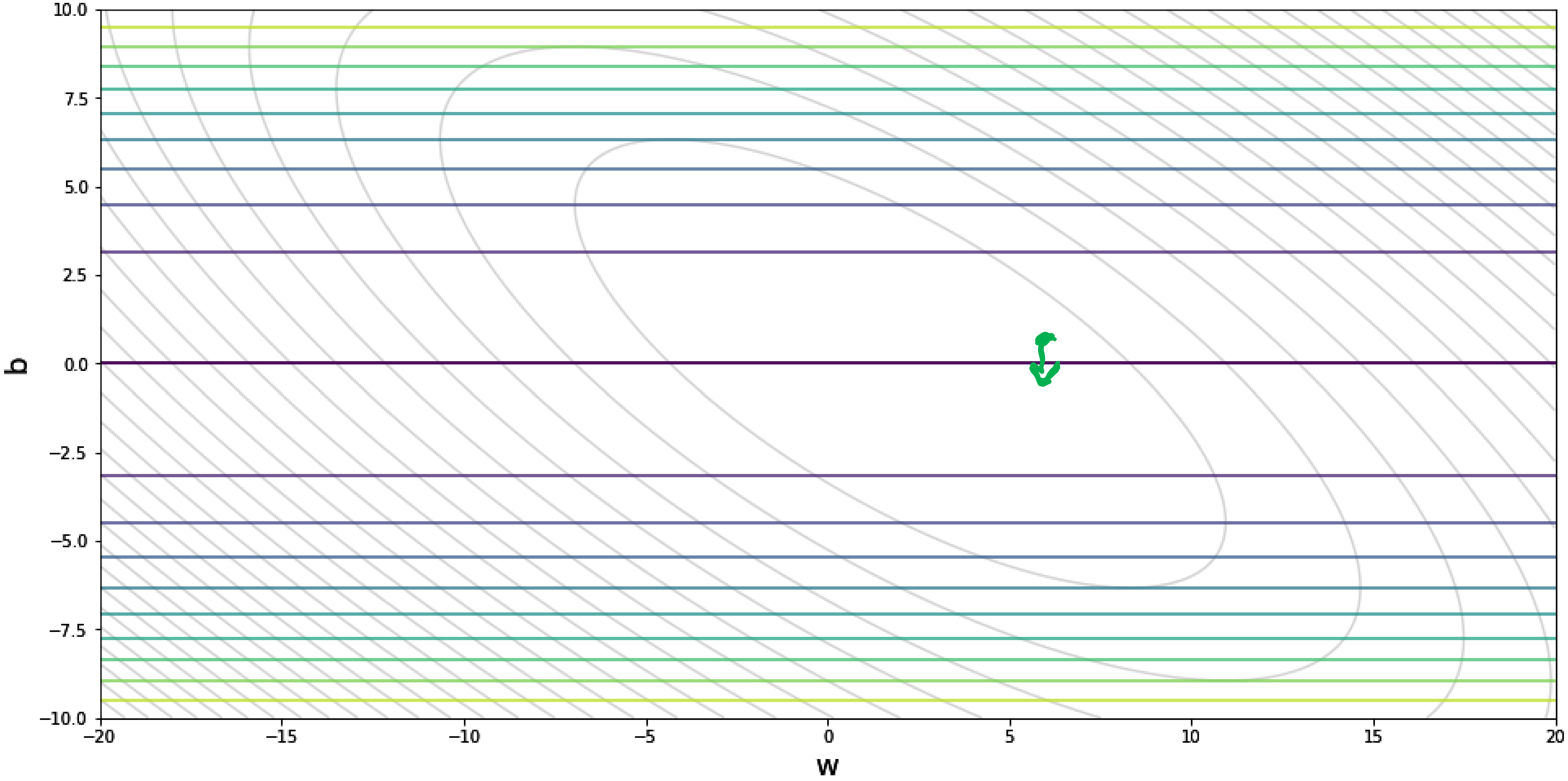
Optimizing the objective



Stochastic Gradient Descent



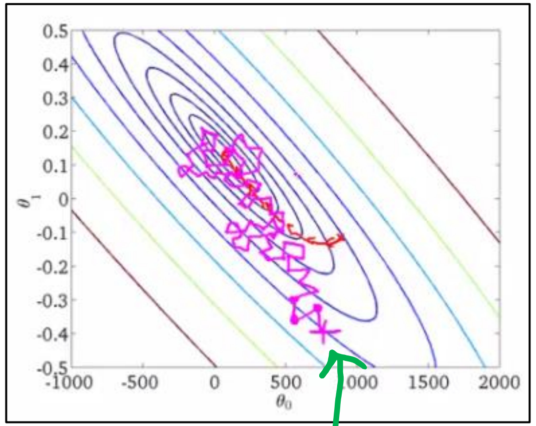
Stochastic Gradient Descent



Stochastic Gradient Descent (SGD)

Algorithm 2 Stochastic Gradient Descent (SGD)

procedure SGD($\mathcal{D}, \boldsymbol{\theta}^{(0)}$)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
    while not converged **do**
        $i \sim$ Uniform($\{1, 2, \ldots, N\}$)
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
    return θ



We need a per-example objective:

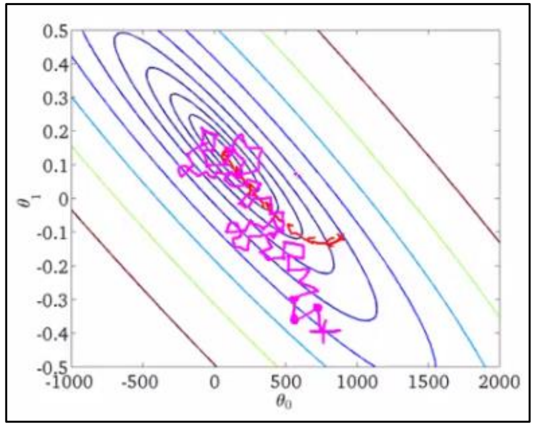
$$\text{Let } J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$

Slide credit: CMU MLD Matt Gormley

Stochastic Gradient Descent (SGD)

Algorithm 2 Stochastic Gradient Descent (SGD)

procedure SGD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
    while not converged **do**
        for $i \in$ shuffle($\{1, 2, \ldots, N\}$) **do**
            $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
    return $\boldsymbol{\theta}$



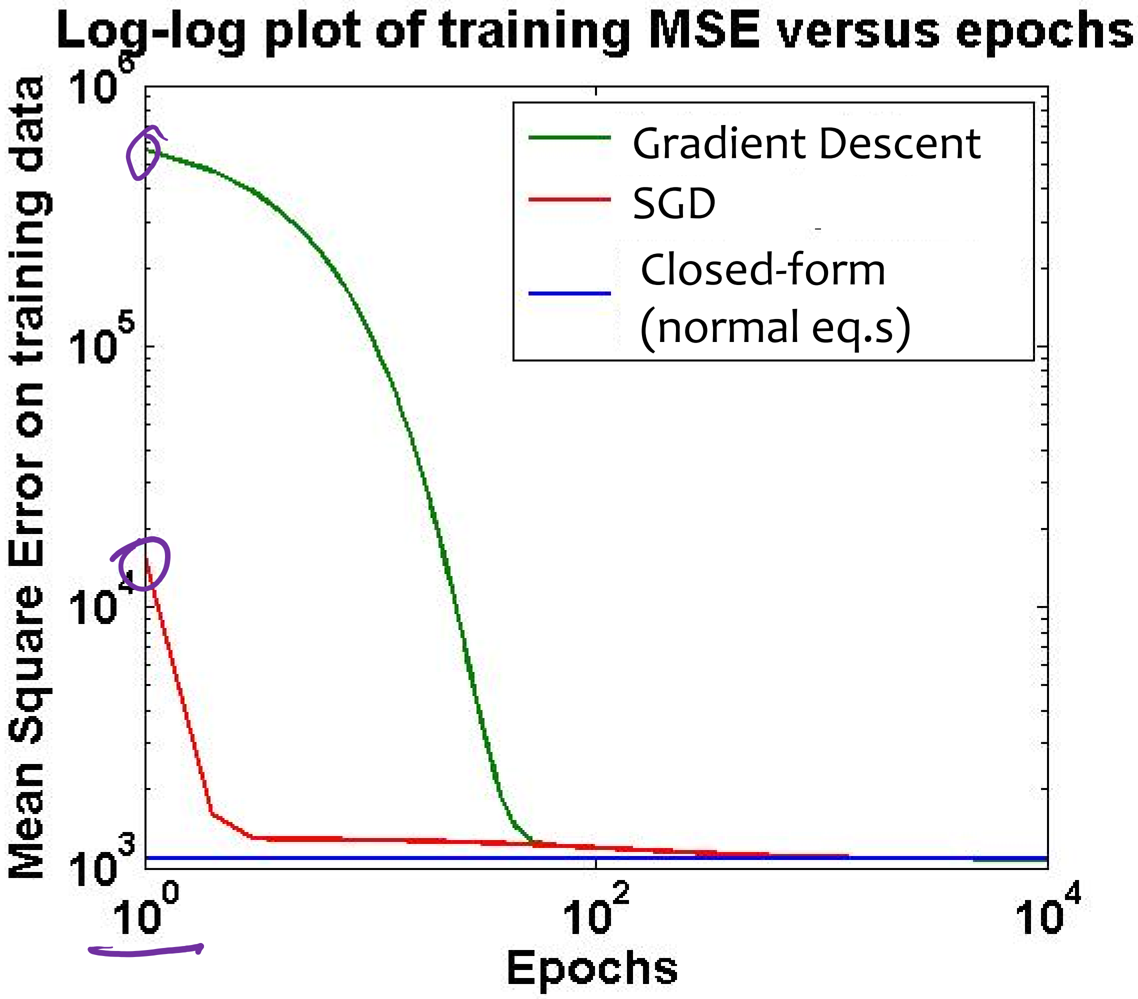
In practice, it is common to implement SGD using sampling **without** replacement (i.e. shuffle({1,2,… N}), even though most of the theory is for sampling **with** replacement (i.e. Uniform({1,2,… N}).

We need a per-example objective:

$$\text{Let } J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$

Slide credit: CMU MLD Matt Gormley

Convergence Curves



- *Def*: an **epoch** is a single pass through the training data

1. For GD, only **one update** per epoch
2. For SGD, ***N* updates** per epoch
   N = (# train examples)

- SGD reduces MSE much more rapidly than GD
- For GD / SGD, training MSE is initially large due to uninformed initialization

Slide credit: CMU MLD Matt Gormley and Eric P. Xing