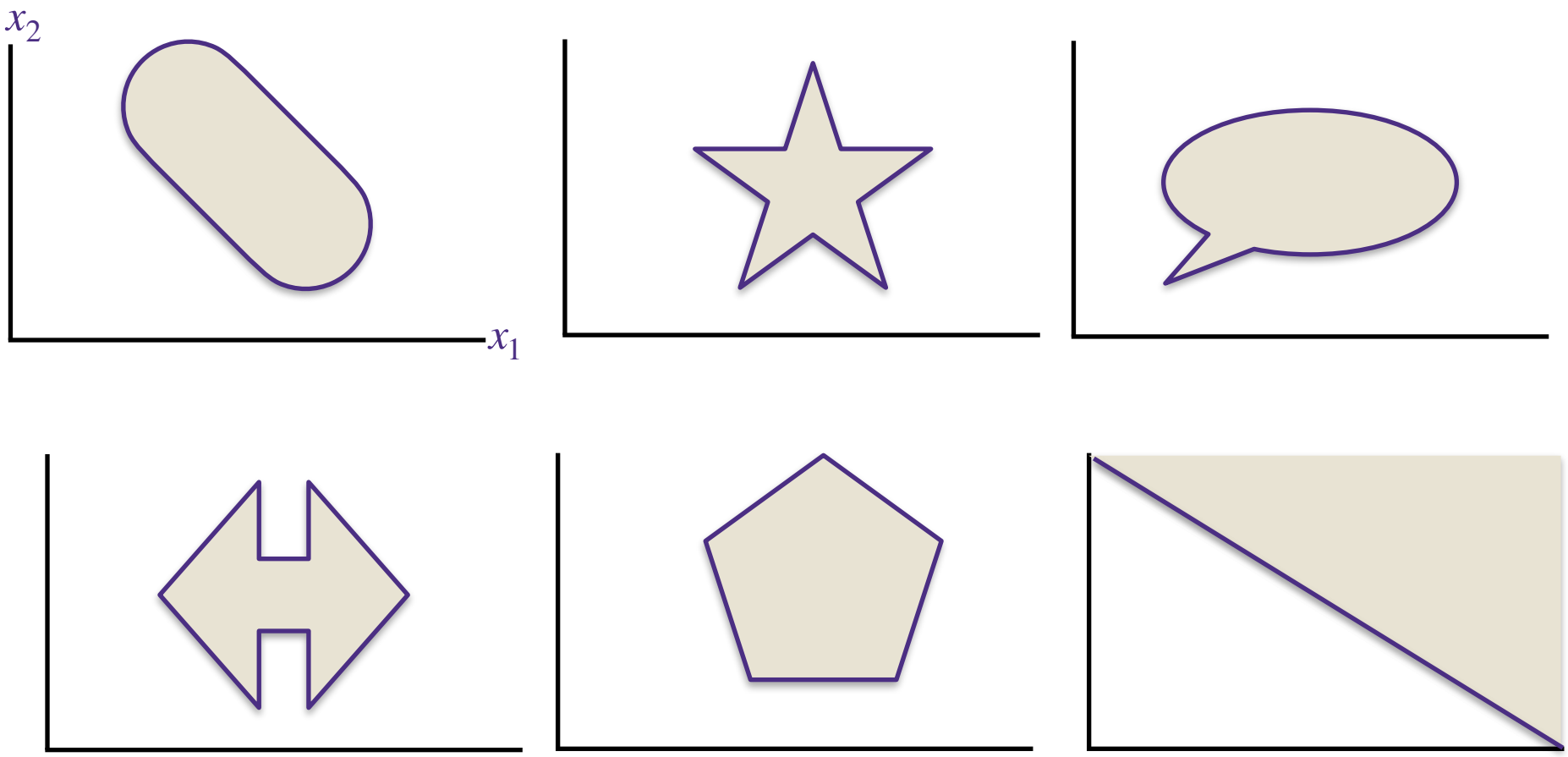
Convexity

What is a convex set?

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

What is a convex set?

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

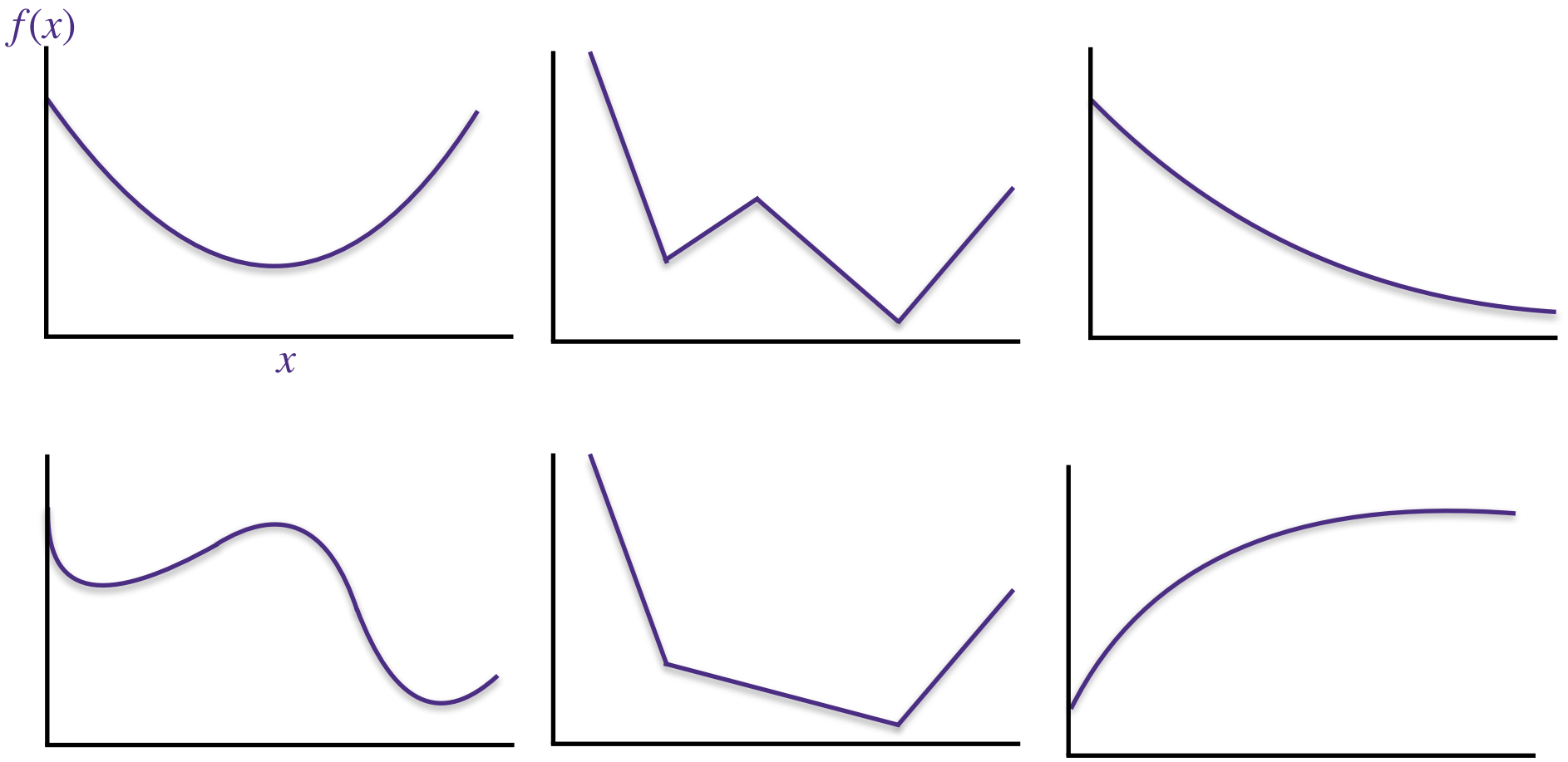


What is a convex function?

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$

What is a convex function?

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$



Convex functions and convex sets?

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

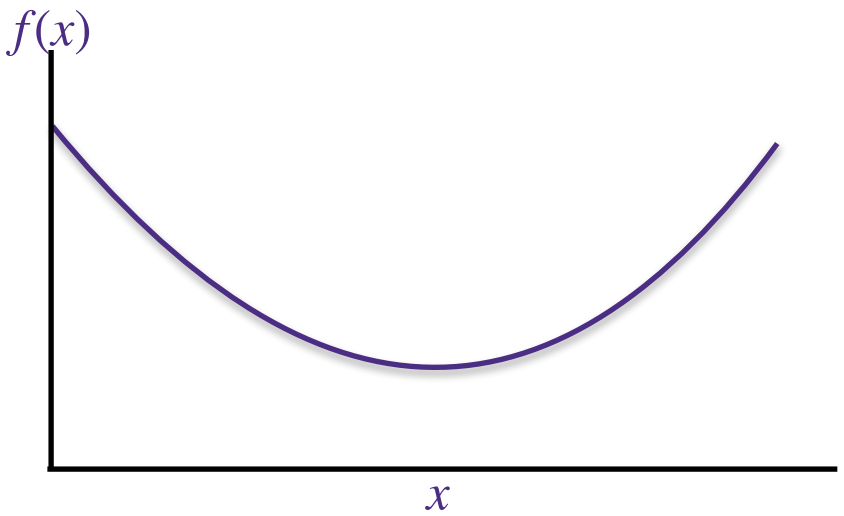
A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

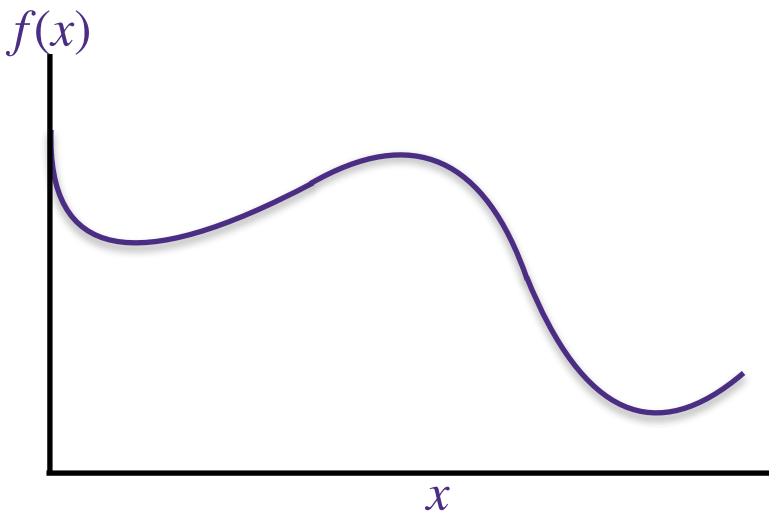
Convex functions and convex sets?

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex



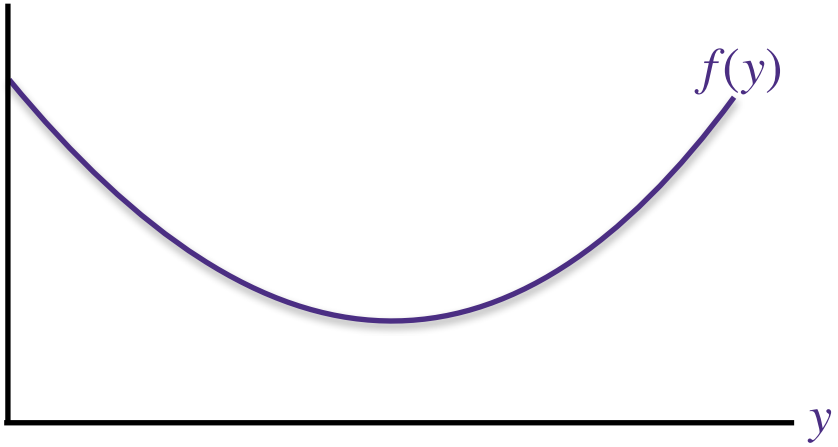


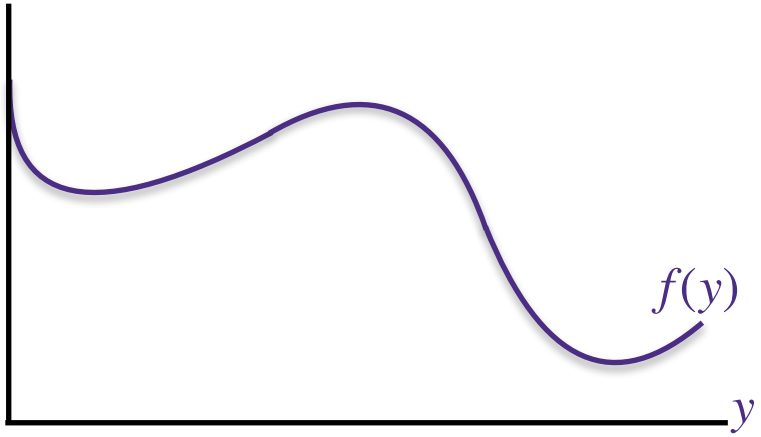
More definitions of convexity

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

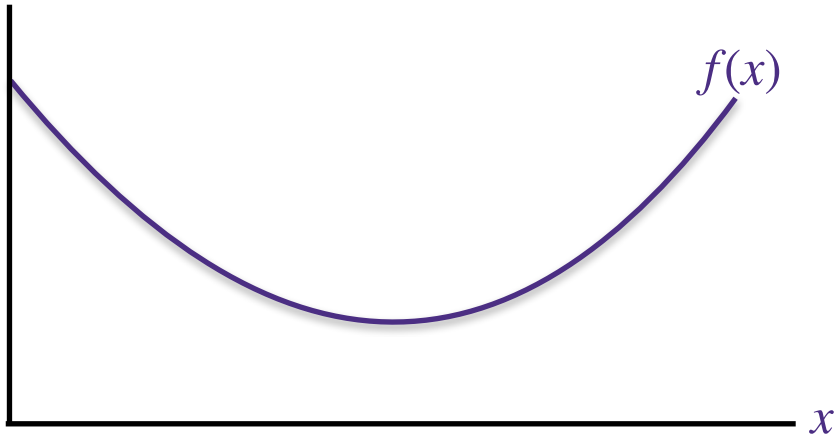
A function $f : \mathbb{R}^d \to \mathbb{R}$ that is differentiable everywhere is convex if $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in dom(f)$

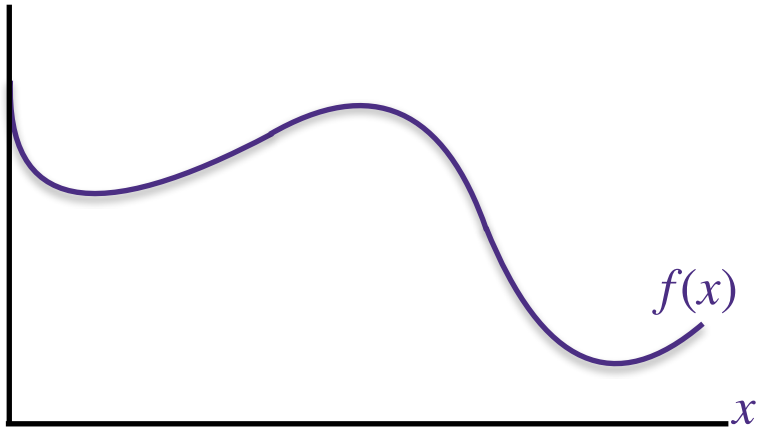


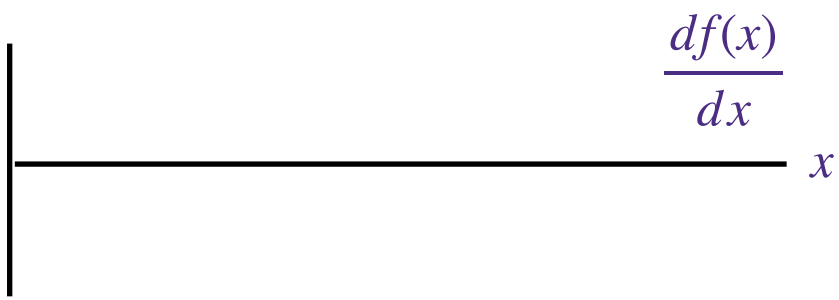


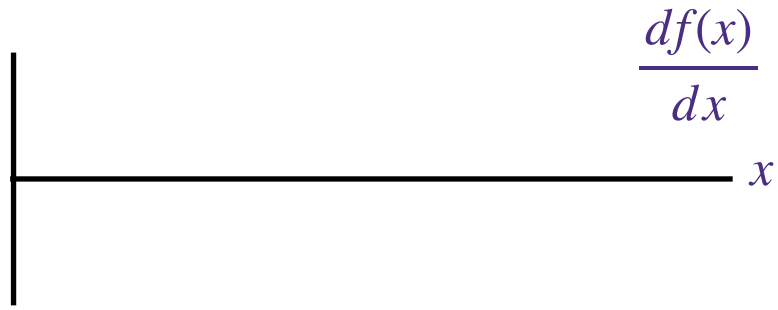
More definitions of convexity

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is twice-differentiable everywhere is convex if $\nabla^2 f(x) \succeq 0$ for all $x \in dom(f)$









More definitions of convexity

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is differentiable everywhere is convex if $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in dom(f)$

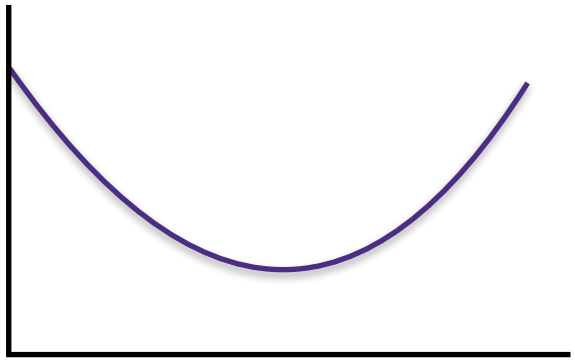
A function $f : \mathbb{R}^d \to \mathbb{R}$ that is twice-differentiable everywhere is convex if $\nabla^2 f(x) \succeq 0$ for all $x \in dom(f)$

Why do we care about convexity?

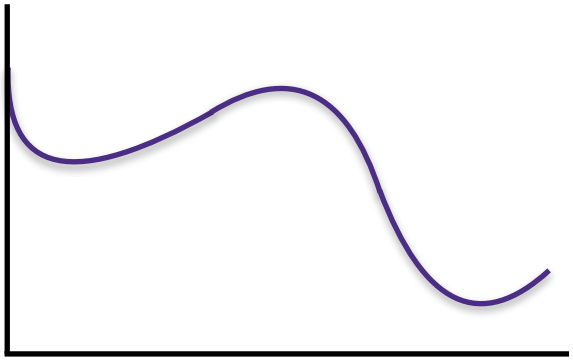
Convex functions

- All local minima are global minima
- Efficient to optimize (e.g., gradient descent)

Convex Function



Non-convex Function



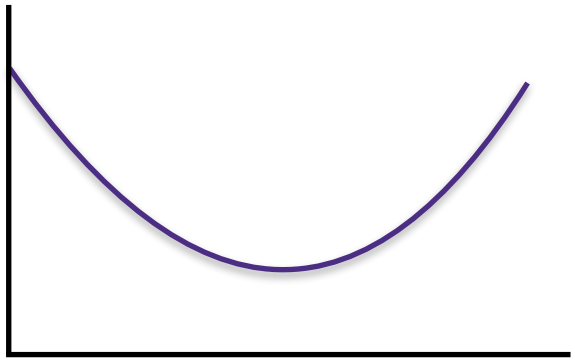
Gradient Descent on $\min_w f(w)$

Initialize: $w_0 = 0$

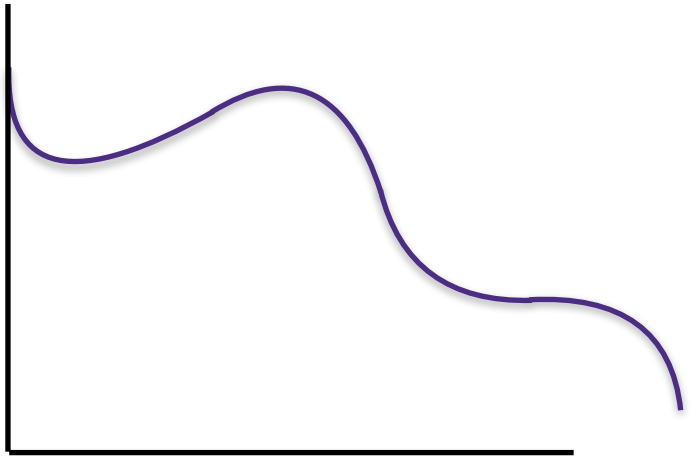
for $t = 1, 2, \ldots$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

Convex Function



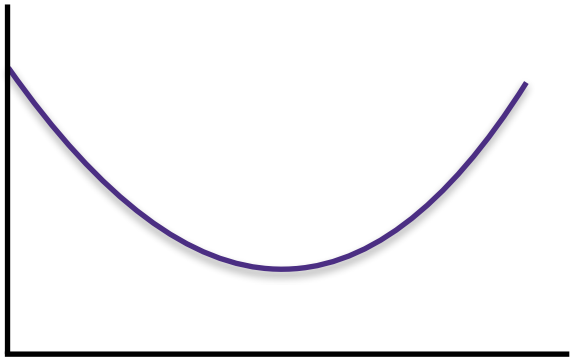
Non-convex Function



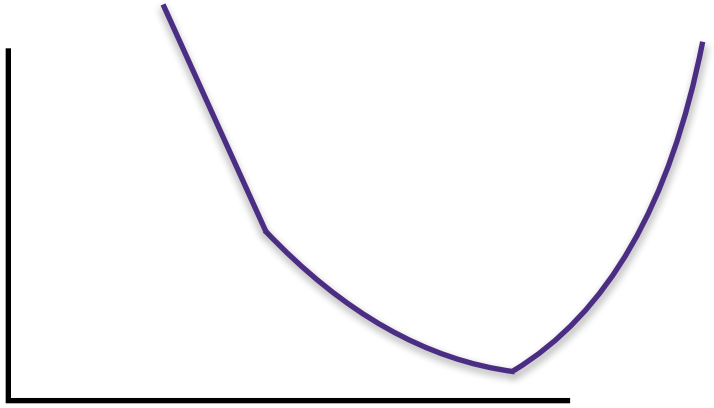
Sub-Gradient

Definition: a function is **non-smooth** if it is not differentiable everywhere

Smooth Convex Function



Non-smooth Convex Function



Definition: a vector $g \in \mathbb{R}^d$ is a **sub-gradient** at x if it satisfies

$$f(y) \geq f(x) + g^T(y - x) \text{ for all } y \in \mathbb{R}^d$$

for smooth convex functions, the minimum is achieved at points where gradient is zero

for non-smooth convex functions, the minimum is achieved at points where sub-gradient set includes the zero vector

Sub-Gradient Descent

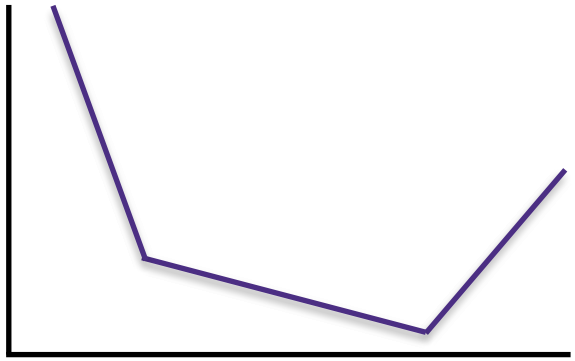
Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

Find any g_t such that $f(y) \geq f(w_t) + g_t^\top (y - w_t)$

$w_{t+1} = w_t - \eta g_t$

Convex Function



Non-convex Function

Coordinate descent

Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

Let $i_t = t \ \% \ d$

$$w_{t+1}^{(i_t)} = w_t^{(i_t)} - \eta_t \frac{\partial f(w)}{\partial w^{(i_t)}}\bigg|_{w=w_t}$$

Machine Learning Problems

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^{n} \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- **Learning a model's parameters:** $\sum_{i=1}^{n} \ell_i(w)$

$$\text{Logistic Loss: } \ell_i(w) = \log(1 + \exp(-y_i\, x_i^T w))$$

$$\text{Squared error Loss: } \ell_i(w) = (y_i - x_i^T w)^2$$

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$$

Optimization summary

- You can always run gradient descent whether f is convex or not. But you only have guarantees if f is convex
- Many bells and whistles can be added onto gradient descent such as momentum and dimension-specific step-sizes (Nesterov, Adagrad, ADAM, etc.)

Gradient Descent

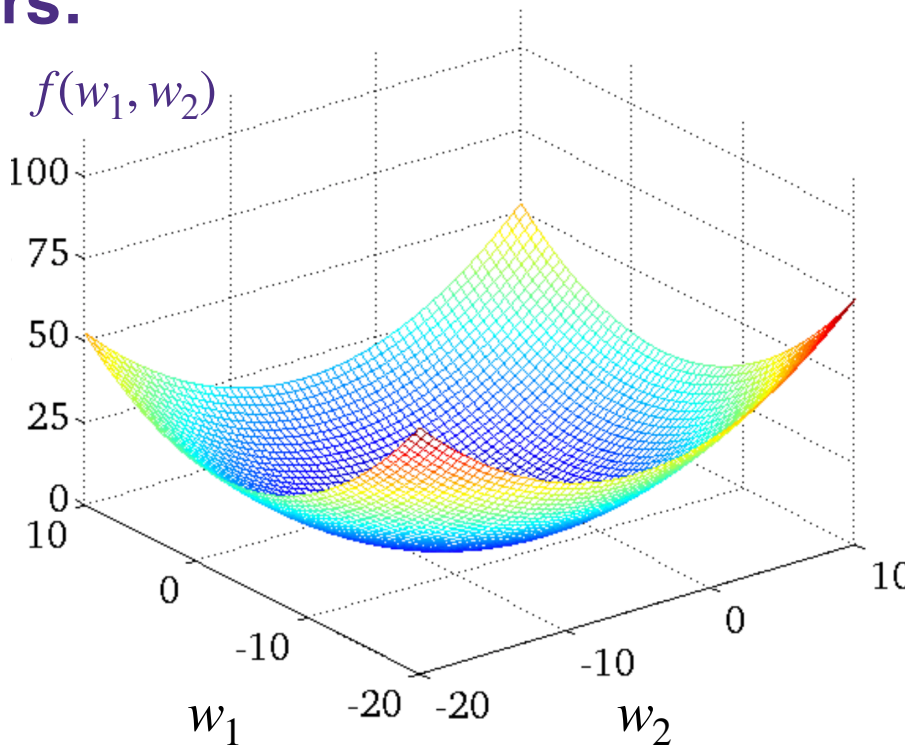
Running example: linear regression

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^{n} \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

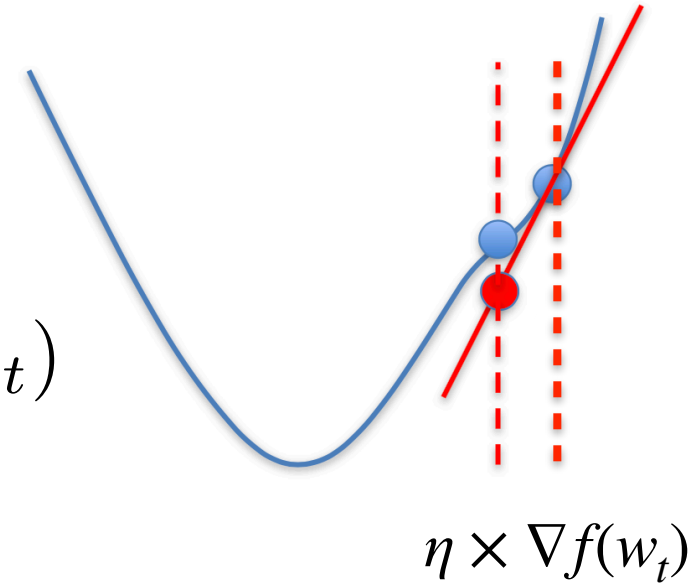
- **Learning a model's parameters:**

$$\hat{w}_{\text{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$



Gradient descent

Example of a general non-convex $f(w)$



Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

Learning rate or step-size,
a hyper-parameter to be chosen by the analyst

Gradient descent for linear regression

Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

For linear regression, we have

$$\hat{w}_{\mathrm{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$

Gradient descent for linear regression

Initialize: $w_0 = 0$

for $t = 1, 2, \ldots$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

For linear regression, we have

$$\hat{w}_{\text{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$

$$\nabla f(w_t) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w_t)$$

$$w_{t+1} = w_t + \eta 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w_t) = (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})w_t + 2\eta\mathbf{X}^T\mathbf{y}$$

Let the least-squares solution be $w^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

$$\begin{aligned} w_{t+1} - w^* &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})w_t + 2\eta\mathbf{X}^T\mathbf{y} - w^* \\ &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})(w_t - w^*) + 2\eta\mathbf{X}^T\mathbf{y} - 2\eta\mathbf{X}^T\mathbf{X}w^* \\ &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})(w_t - w^*) \end{aligned}$$

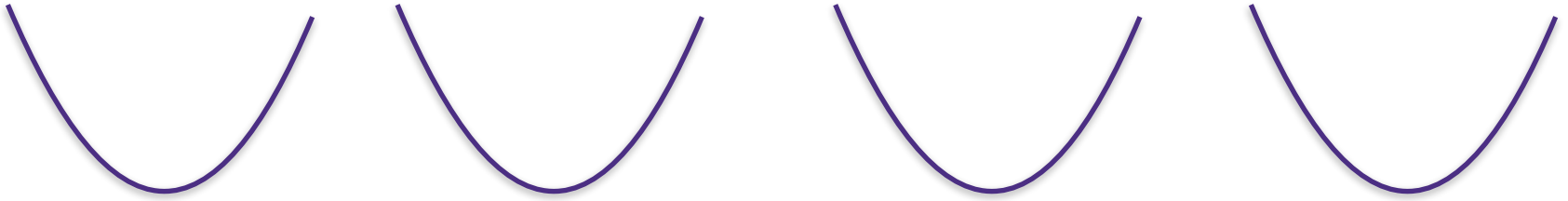
Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \; = \; (\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X})(w_t - w^*)$$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \implies \begin{aligned} w_{t+1} - w^* &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})(w_t - w^*) \\ &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^2(w_{t-1} - w^*) \\ &\vdots \\ &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^{t+1}(w_0 - w^*) \end{aligned}$$

In one dimension, $2\mathbf{X}^T\mathbf{X} = a$ is a scalar



$0 < \eta < 1/a$ $\quad$ $\eta = 1/a$ $\quad$ $1/a < \eta < 2/a$ $\quad$ $\eta > 2/a$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \; = \; (\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X})^{t+1}(w_0 - w^*)$$

In multi dimensions, **eigenvalues** of $\mathbf{X}^T\mathbf{X}$ are important

(you will see why I say $\mathbf{X}^T\mathbf{X}$ instead of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ in couple of slides)

Let the eigenvalue decomposition of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ be $Q^{-1}DQ$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \;=\; (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^{t+1}(w_0 - w^*)$$

In multi dimensions, **eigenvalues** of $\mathbf{X}^T\mathbf{X}$ are important

(you will see why I say $\mathbf{X}^T\mathbf{X}$ instead of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ in couple of slides)

Let the eigenvalue decomposition of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ be $Q^{-1}DQ$

Then, $w_{t+1} - w^* \;=\; (Q^{-1}DQ)^{t+1}(w_0 - w^*)$

$$= \underbrace{Q^{-1}DQQ^{-1}DQ\cdots Q^{-1}DQ}_{t+1 \text{ times}}(w_0 - w^*)$$

$$= Q^{-1}D^{t+1}Q\,(w_0 - w^*)$$

$$Q(w_{t+1} - w^*) \;=\; D^{t+1}Q\,(w_0 - w^*)$$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* = (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^{t+1}(w_0 - w^*)$$

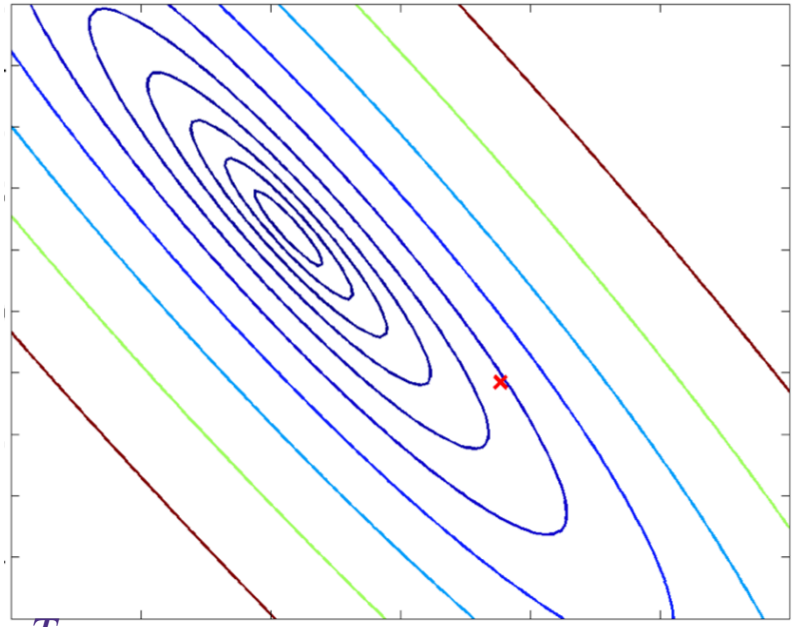
$$Q(w_{t+1} - w^*) = D^{t+1} Q\,(w_0 - w^*)$$

In direction q_1

$$0 < \lambda_1(\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})$$

In direction q_2

$$-1 < \lambda_2(\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X}) < 0$$



Gradient descent for linear regression

Recall that in each eigen direction

$$q_i^T(w_{t+1} - w^*) = \left(\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})\right)^{t+1} q_i^T(w_0 - w^*)$$

We want the error to decay fast in all directions, whose bottleneck is the largest and the smallest eigen values:

$$\lambda_{\min}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \text{ and } \lambda_{\max}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$$

We want to choose the learning rate η such that

$$-1 \ll \lambda_d(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \leq \cdots \leq \lambda_1(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \ll 1$$

Gradient descent for linear regression

Recall that in each eigen direction

$$q_i^T(w_{t+1} - w^*) = \lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})^{t+1} q_i^T(w_0 - w^*)$$

I will not prove the facts that $\|q_i\|_2 = 1$ and $q_i^T q_j = 0$

Claim: $\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) = 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})$

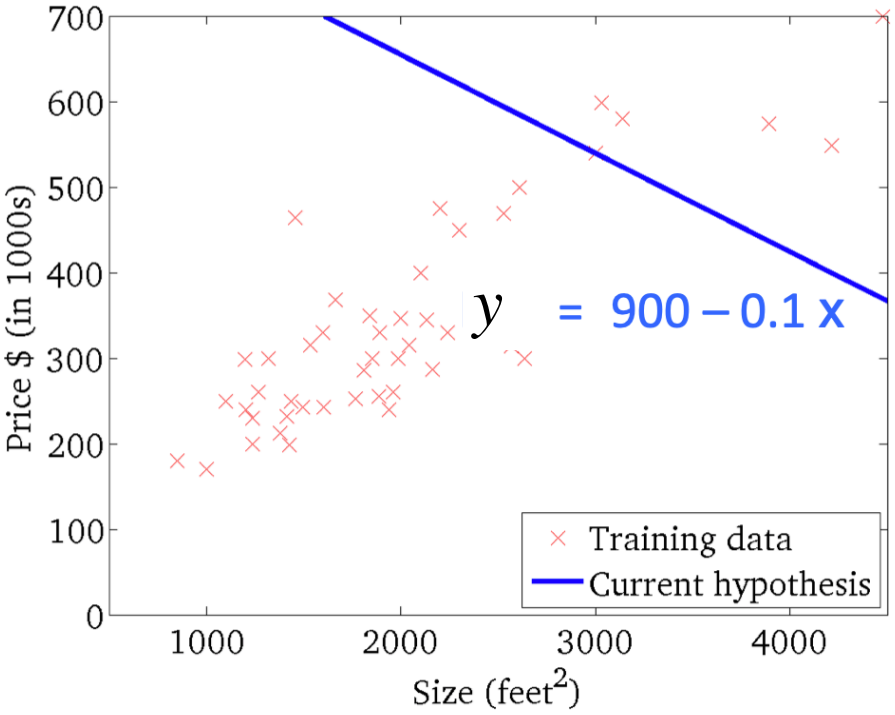
Claim: $\lambda_i(\mathbf{X}^T\mathbf{X}) \geq 0$

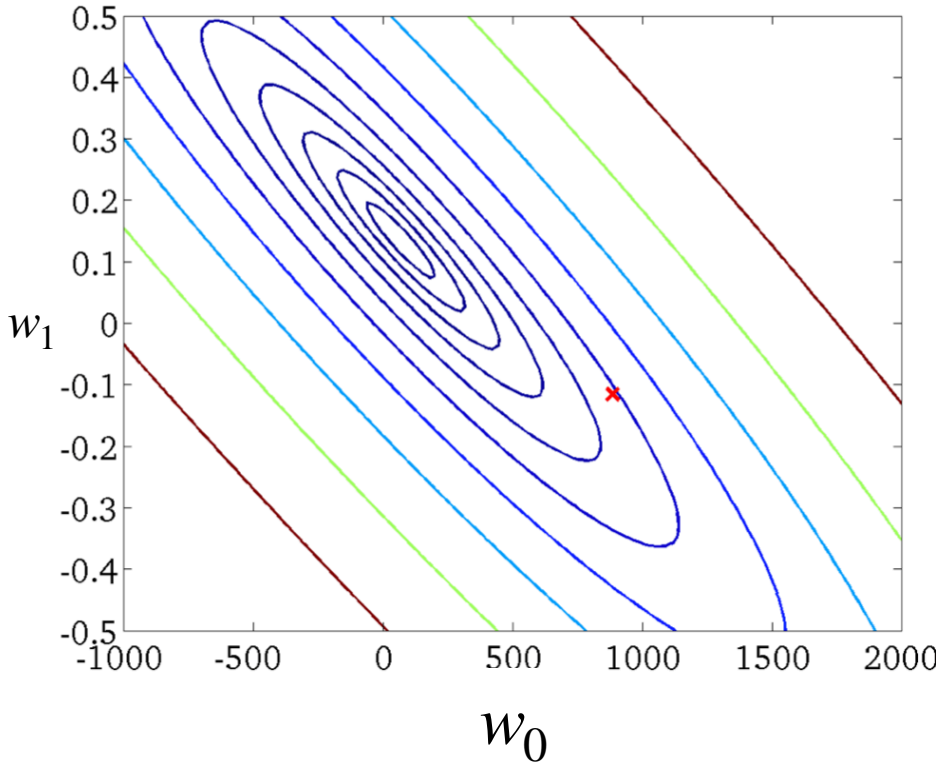
$\eta = 0$

$1 - 2\eta\lambda_{\max}(\mathbf{X}^T\mathbf{X}) = -1$

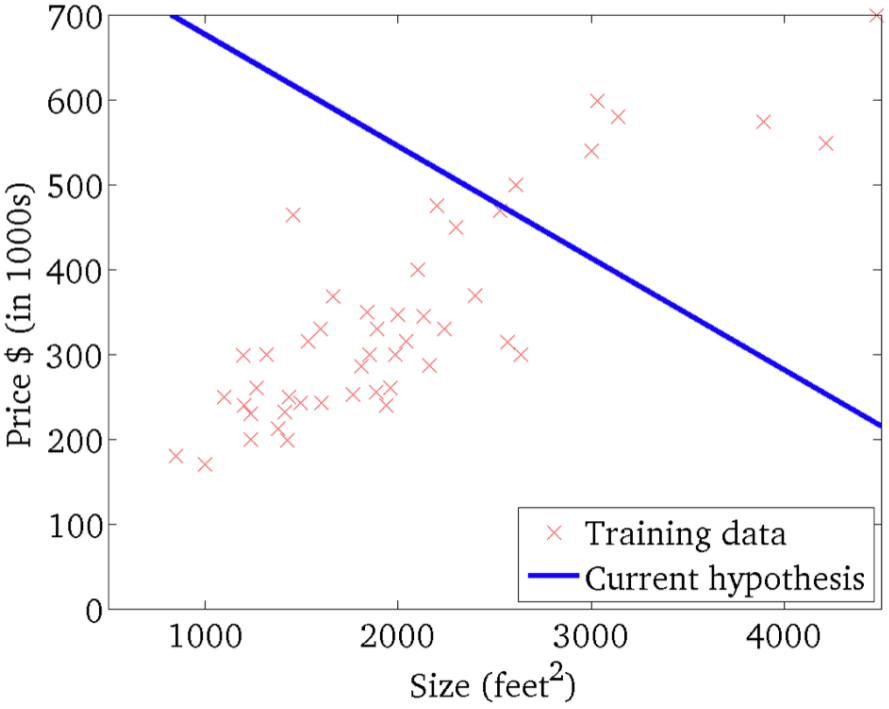
$$\eta = \frac{1}{\lambda_{\max}(\mathbf{X}^T\mathbf{X})}$$

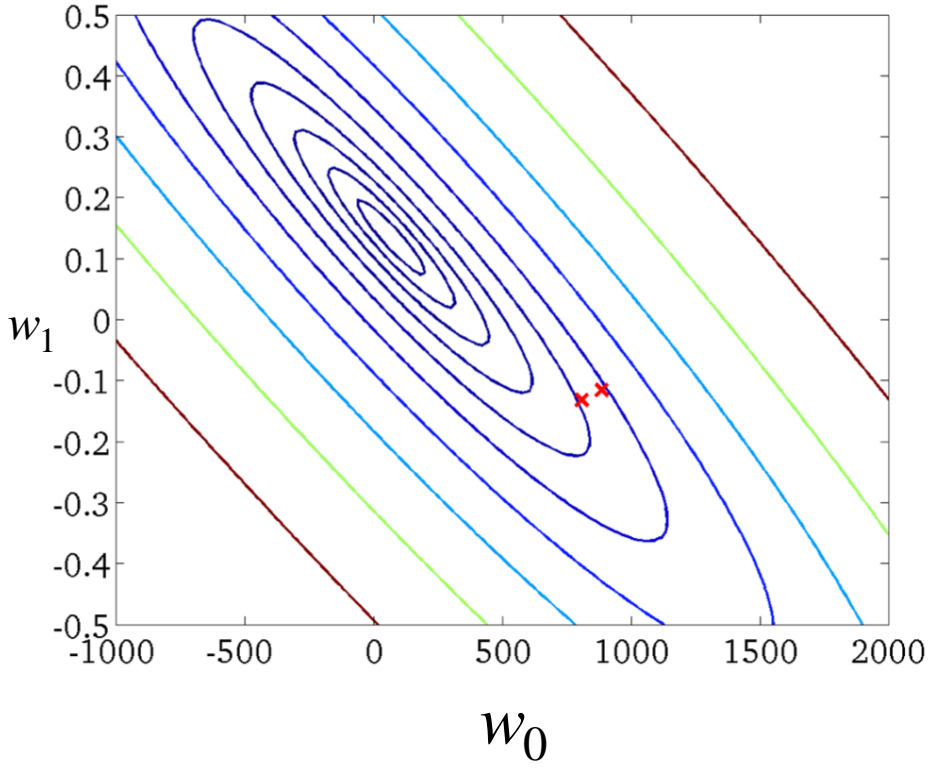
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
$y = 900 - 0.1\,x$
Training data
Current hypothesis

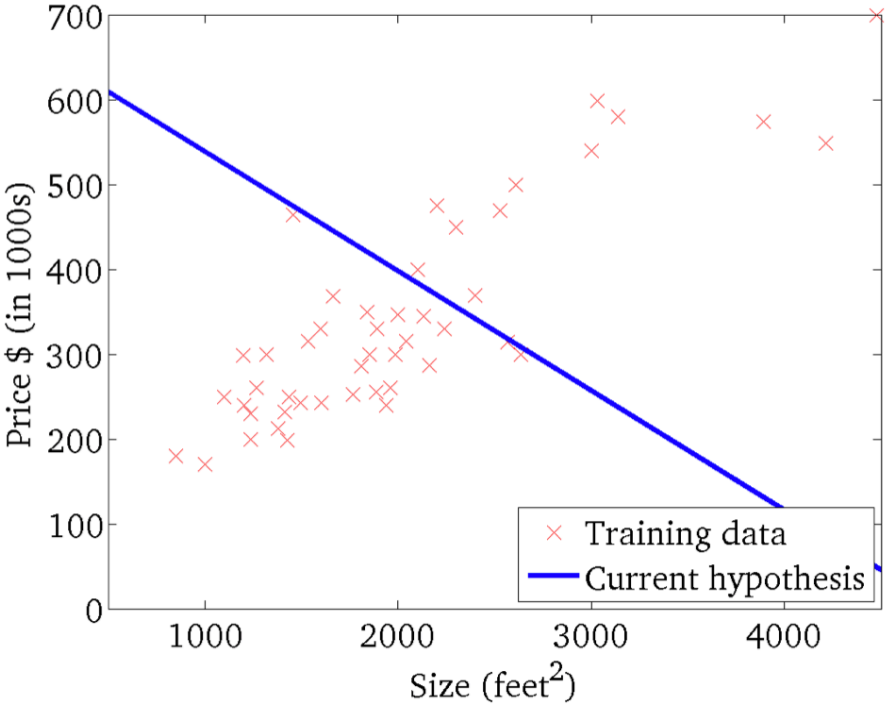

w_1
w_0

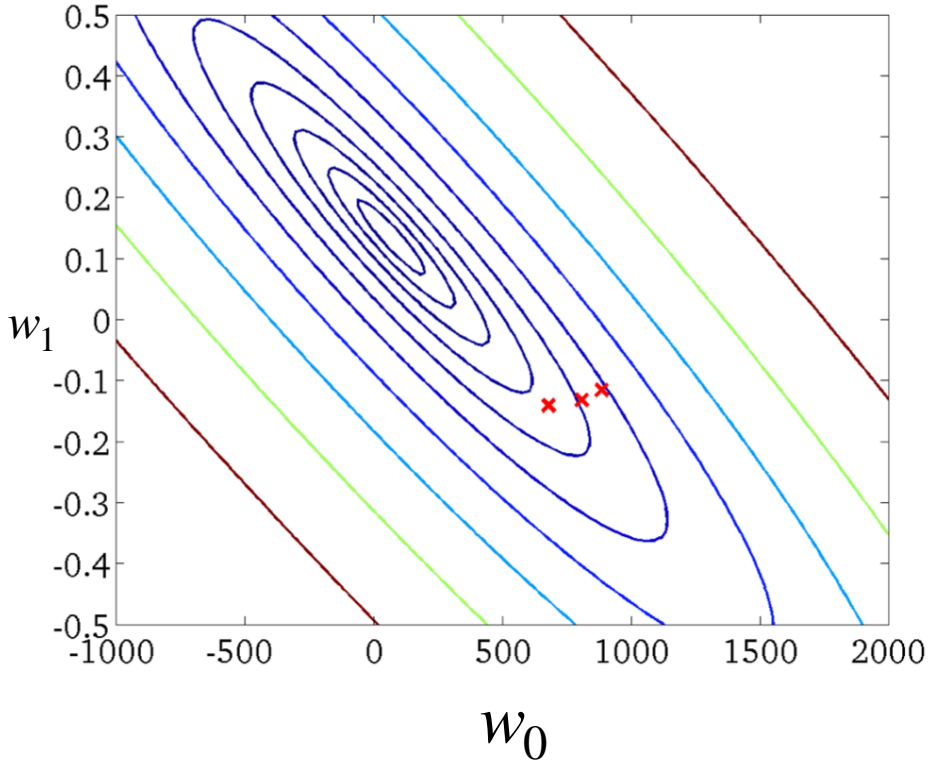
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000

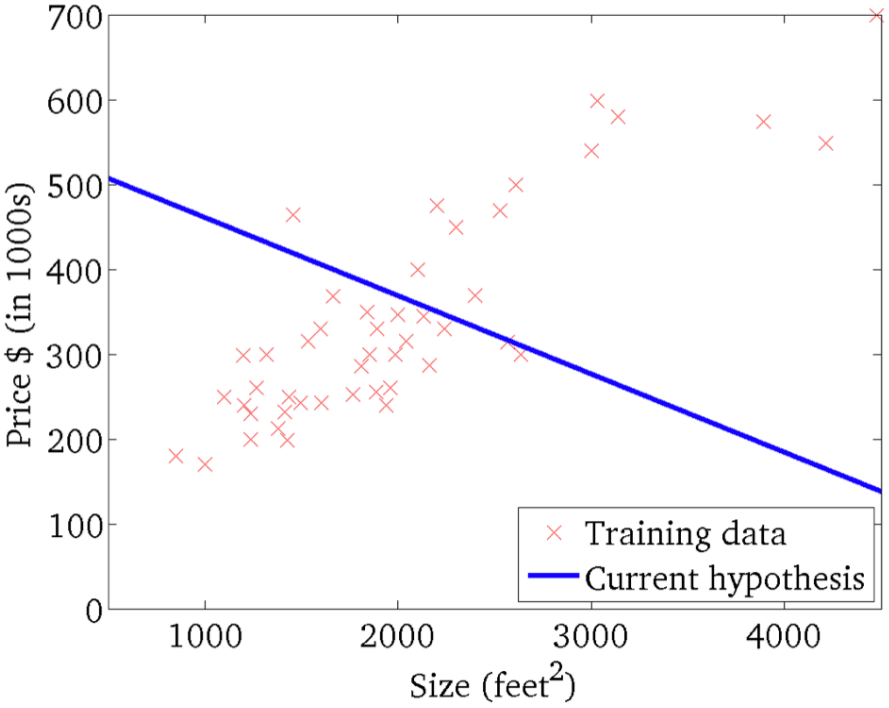

w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

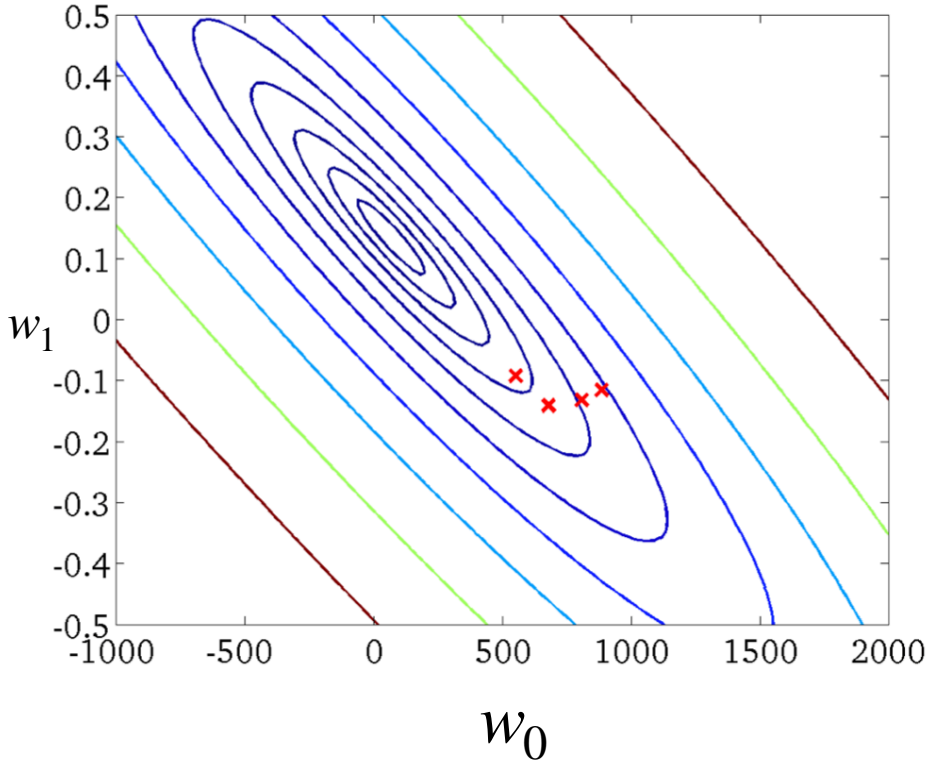
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000

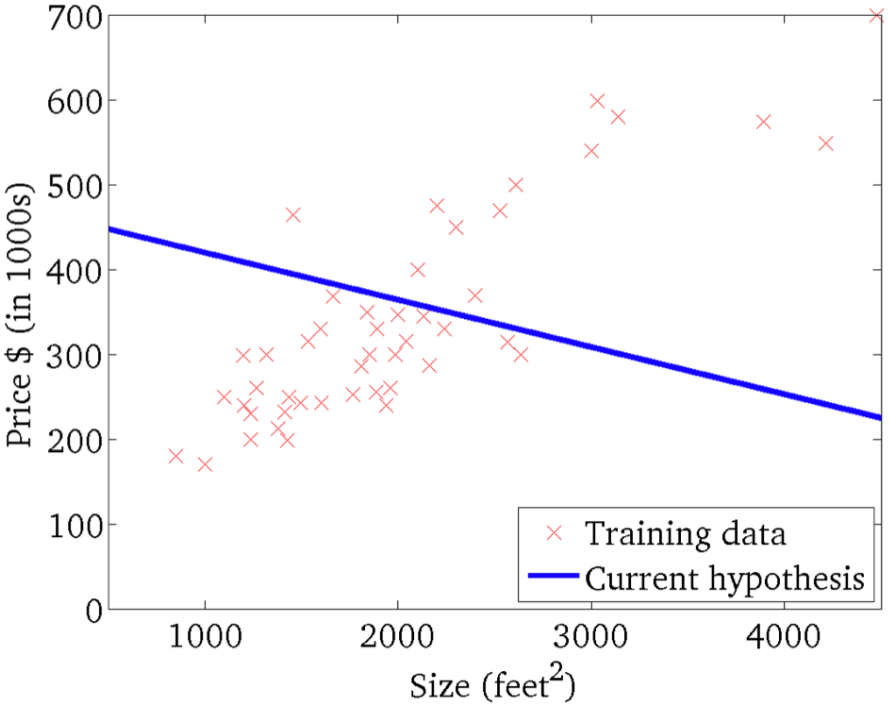

w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

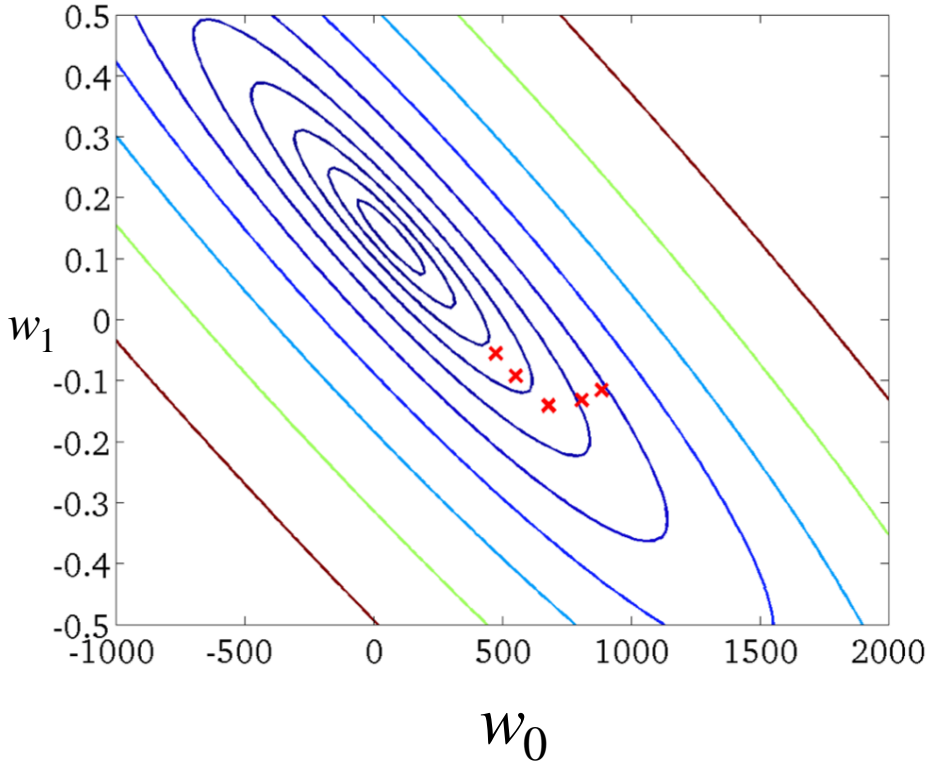
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000

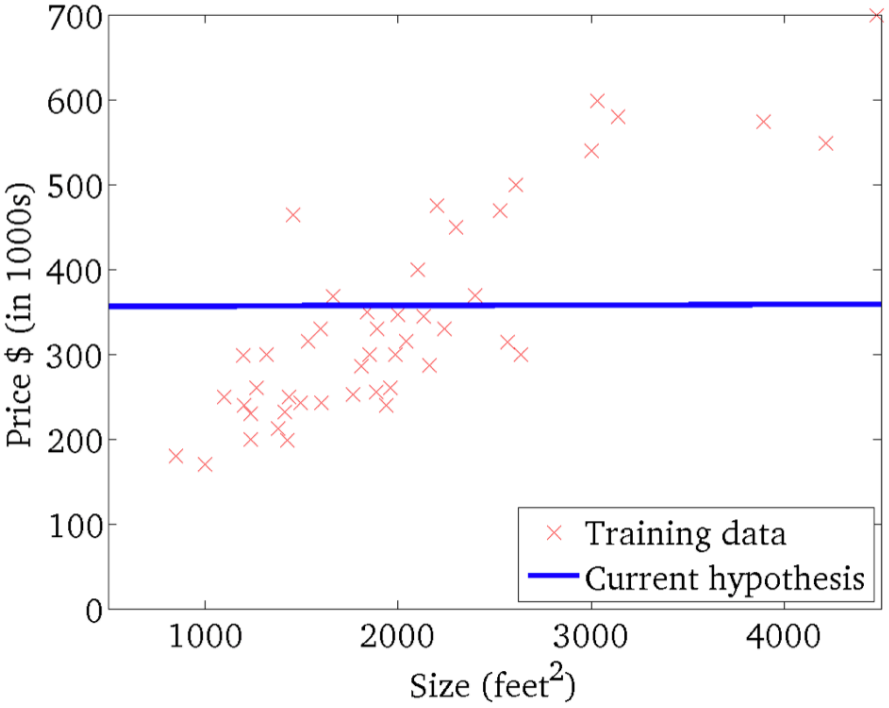

w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

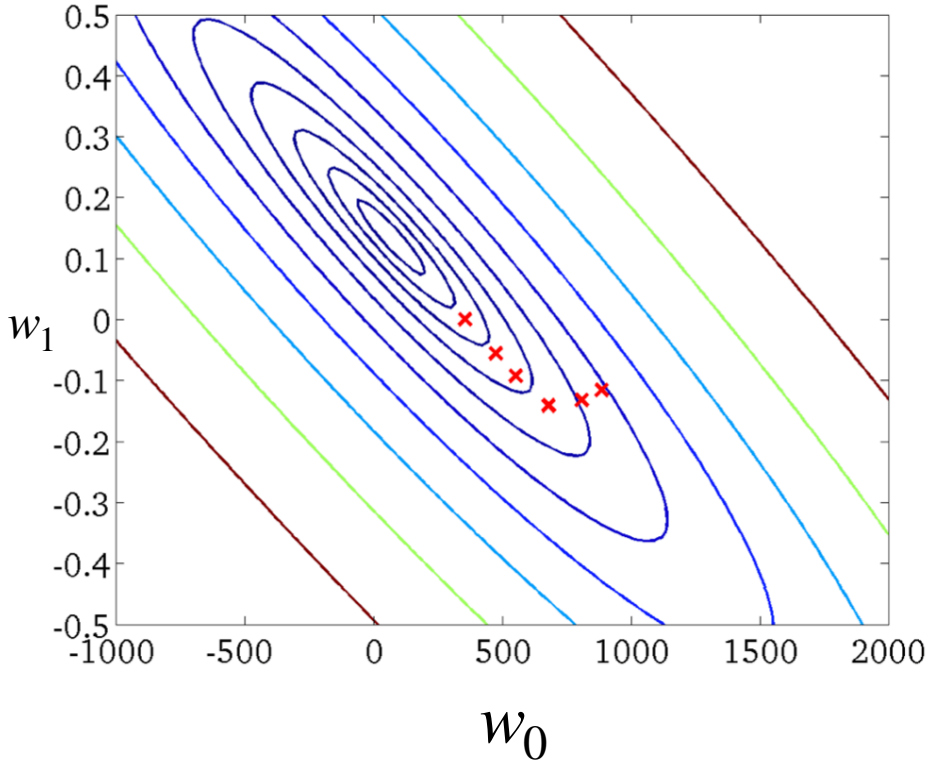
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000

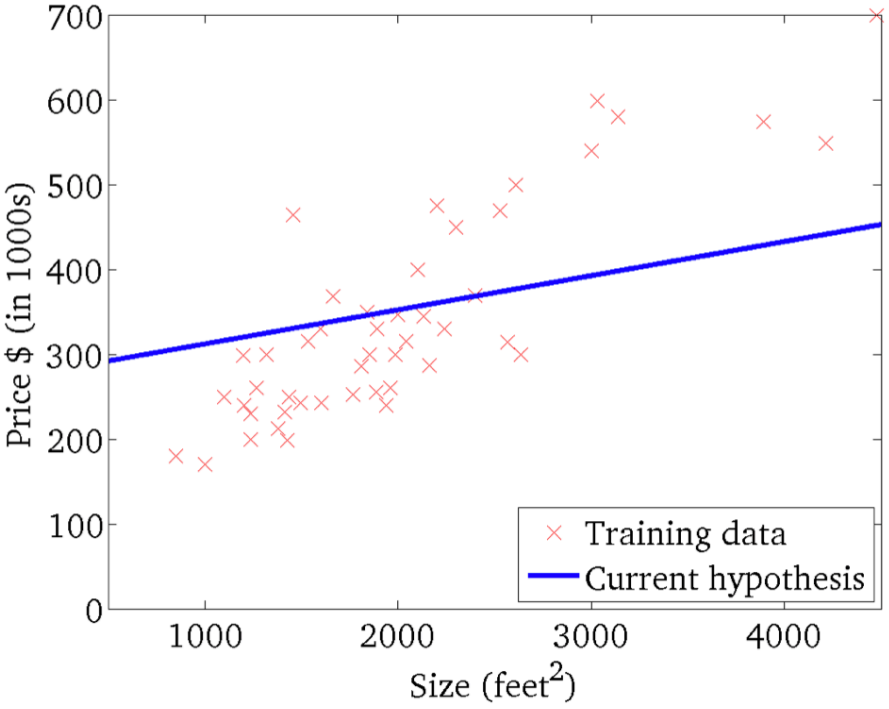

w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

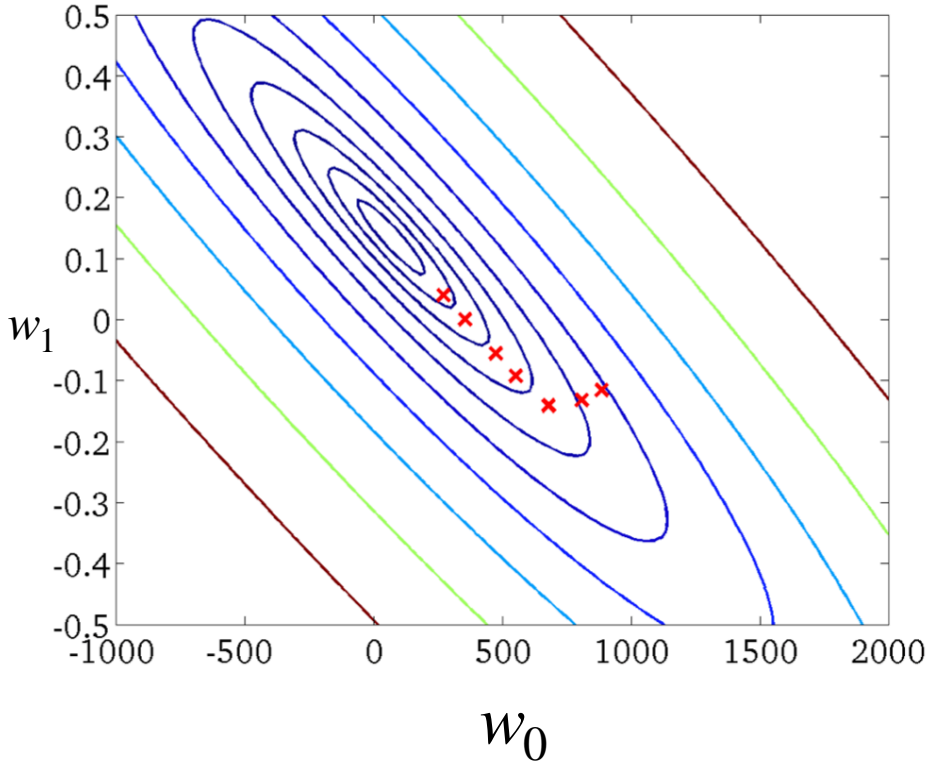
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis

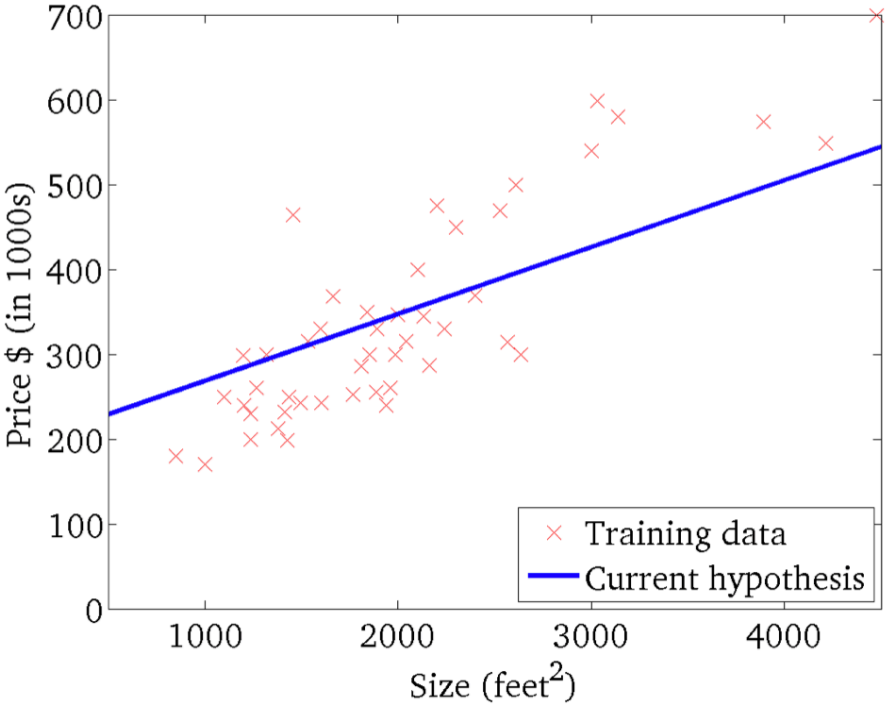

w_1
w_0

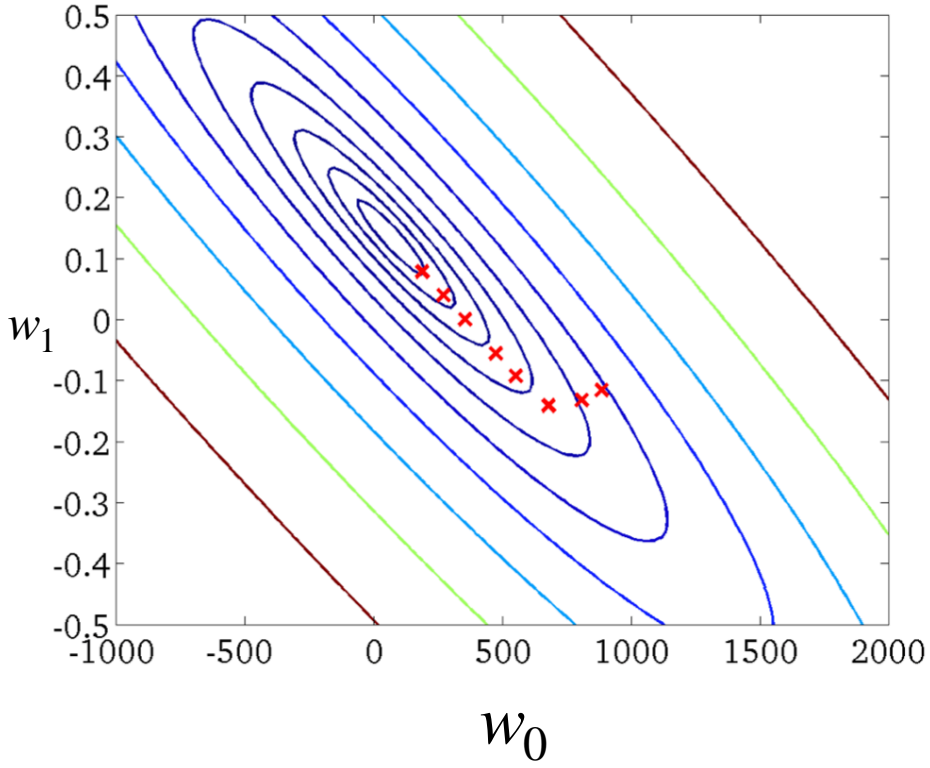
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000

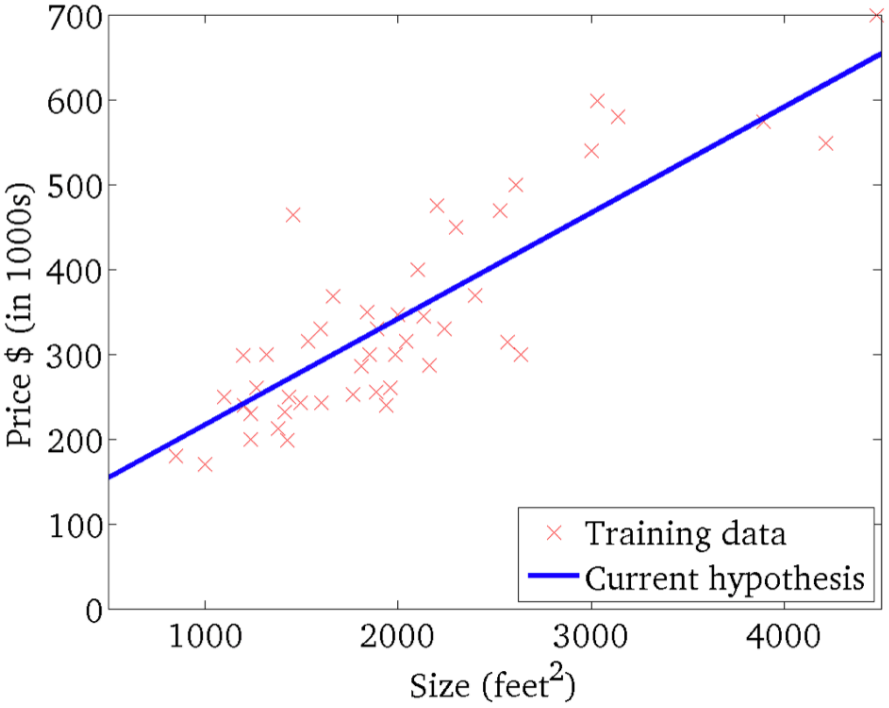

w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

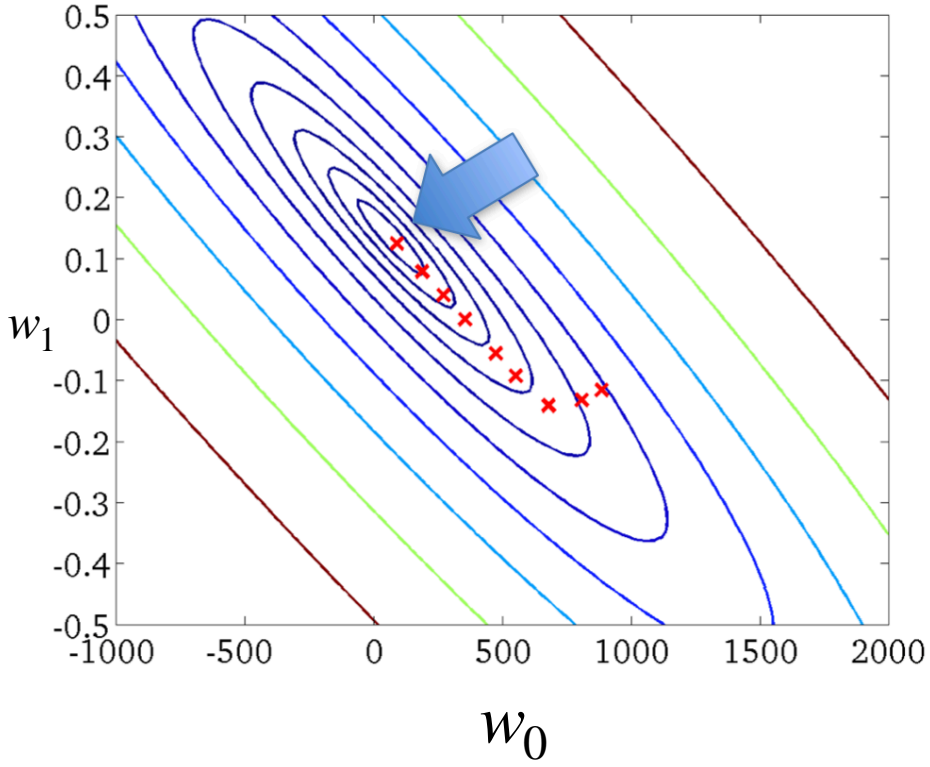
$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000


w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

$$y = w_0 + w_1 x$$


Price $ (in 1000s)
Size (feet^2)
Training data
Current hypothesis
0
100
200
300
400
500
600
700
1000
2000
3000
4000


w_1
w_0
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000

Gradient descent for logistic regression

Loss function: Conditional Likelihood

$$\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

$$\nabla f(w) = \sum_{i=1}^n \frac{1}{1 + \exp(-y_i x_i^T w)} \exp(-y_i x_i^T w)(-y_i x_i)$$

Gradient descent

- For L-smooth functions, with a fixed step size $\eta < 1/L$
  - if $f(w)$ is convex,

$$f(w_t) - f(w^*) \leq \frac{\|w_0 - w^*\|_2^2}{2\eta t}$$

  - if $f(w)$ is μ-strongly convex,

$$f(w_t) - f(w^*) \leq \left(1 - \eta\mu\right)^t \left(f(w_0) - f(w^*)\right)$$

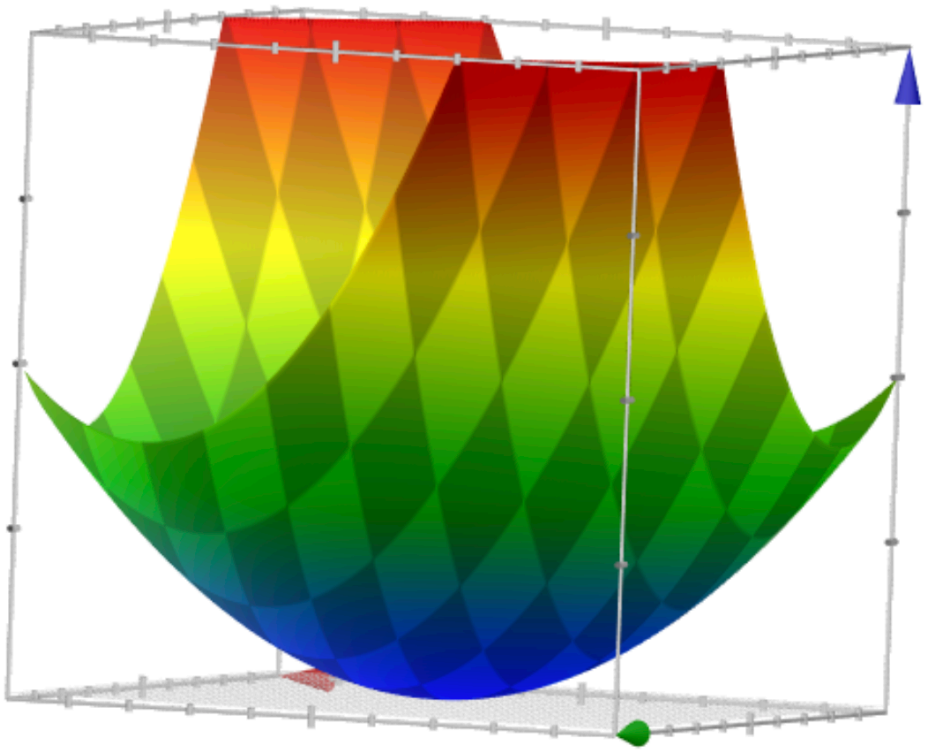
- What can we do for non-smooth function $f(w)$?
  - for example, LASSO

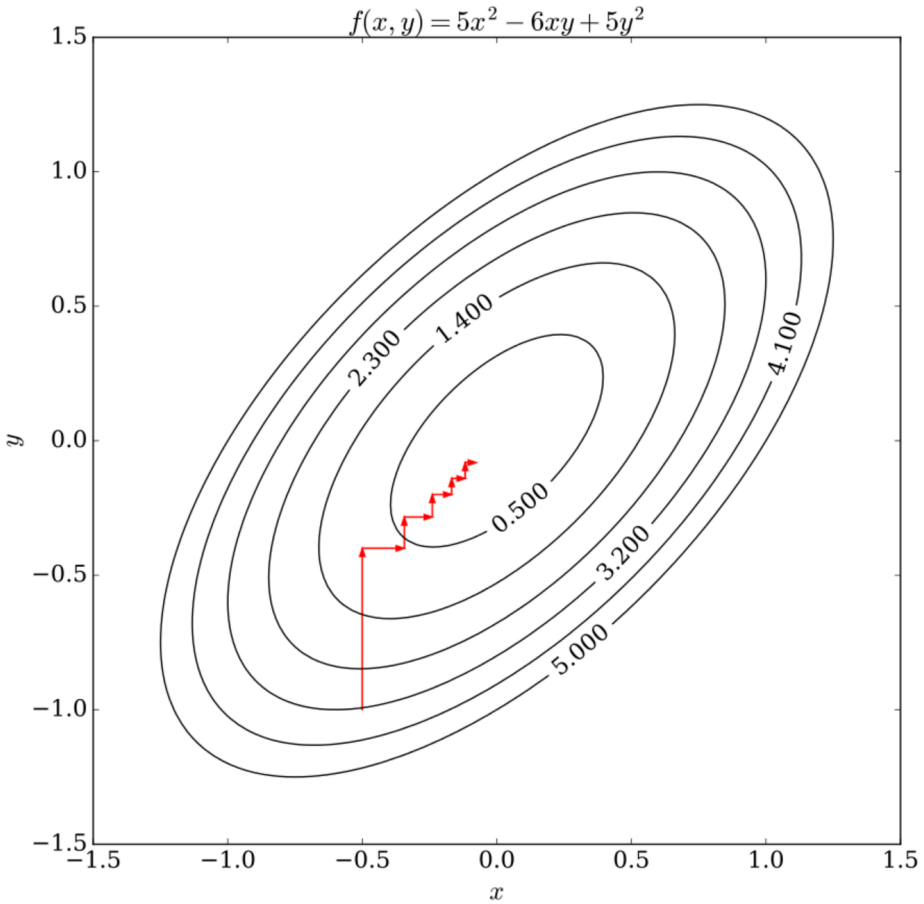
$$\hat{w}_{\text{Lasso}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2 + \lambda\|w\|_1}_{f(w)}$$

Coordinate Descent

Optimization: how do we solve Lasso?

- among many methods to find the solution, we will learn **coordinate descent method**
- as an illustrating example, we show coordinate descent updates on finding the minimum of $f(x, y) = 5x^2 - 6xy + 5y^2$





How do we solve Lasso: $\min_{w} \mathscr{L}(w) + \lambda \|w\|_1$?

- **Coordinate descent**
  - **input: training data S_{train}, max # of iterations T**
  - **initialize: $w^{(0)} = \mathbf{0} \in \mathbb{R}^d$**
  - **for $t = 1, \ldots, T$**
    - **for $j = 1, \ldots, d$**
      - **fix $w_1^{(t)}, \ldots, w_{j-1}^{(t)}$ and $w_{j+1}^{(t-1)}, \ldots, w_d^{(t-1)}$, and**

$$w_j^{(t)} \leftarrow \arg\min_{w_j \in \mathbb{R}} \mathscr{L}\left(\begin{bmatrix} w_1^{(t)} \\ \vdots \\ w_{j-1}^{(t)} \\ w_j \\ w_{j+1}^{(t-1)} \\ \vdots \\ w_d^{(t-1)} \end{bmatrix}\right) + \lambda \left\| \begin{bmatrix} w_1^{(t)} \\ \vdots \\ w_{j-1}^{(t)} \\ w_j \\ w_{j+1}^{(t-1)} \\ \vdots \\ w_d^{(t-1)} \end{bmatrix} \right\|_1$$

this is a one-dimensional optimization, which is much easier to solve

Coordinate descent for (un-regularized) linear regression

- let us understand what coordinate descent does on a simpler problem of linear least squares, which minimizes

$$\text{minimize}_w \mathscr{L}(w) = \|\mathbf{X}w - \mathbf{y}\|_2^2$$

- note that we know that the optimal solution is

$$\hat{w}_{\text{LS}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{y}$$

so we do not need to run any optimization algorithm

- we are solving this problem with **coordinate descent** as a starting example

- the main challenge we address is, how do we update $w_j^{(t)}$?

- let us derive an **analytical rule** for updating $w_j^{(t)}$

Coordinate descent for (un-regularized) linear regression

Coordinate descent for (un-regularized) linear regression

- we will study the case $j = 1$, for now (other cases are almost identical)
- when updating $w_1^{(t)}$, recall that

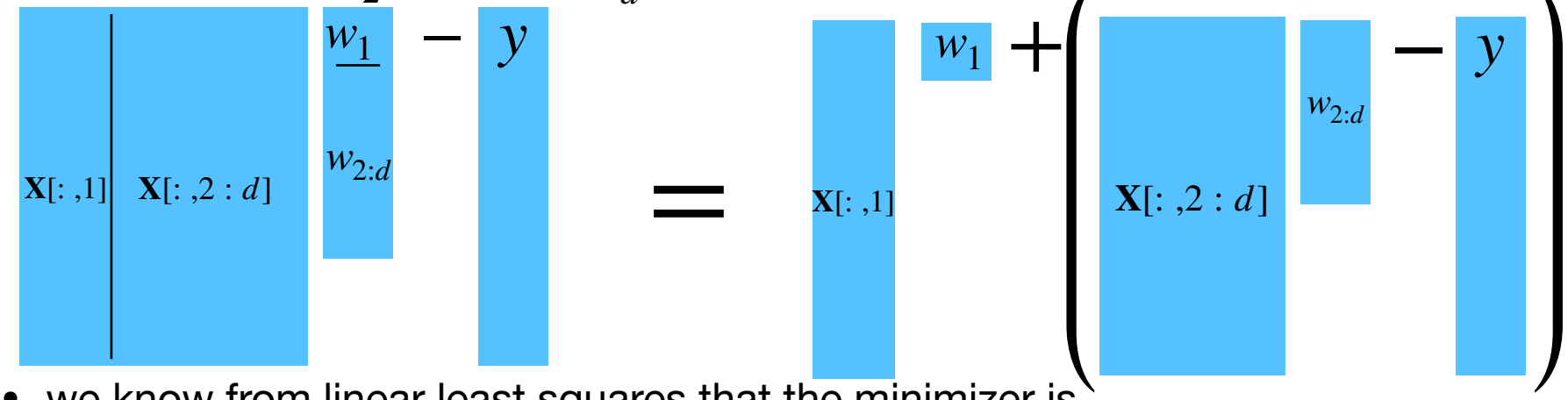
$$w_1^{(t)} \leftarrow \arg\min_{w_1} \|\mathbf{X}w - \mathbf{y}\|_2^2$$

where $w = [w_1, \; w_2^{(t-1)}, \; \ldots \; , w_d^{(t-1)}]^T$

- first step is to write the objective function in terms of the variable we are optimizing over, that is w_1:

$$\mathscr{L}(w) = \left\| \mathbf{X}[:,1]w_1 + \mathbf{X}[:,2:d]w_{2:d} - \mathbf{y} \right\|_2^2$$

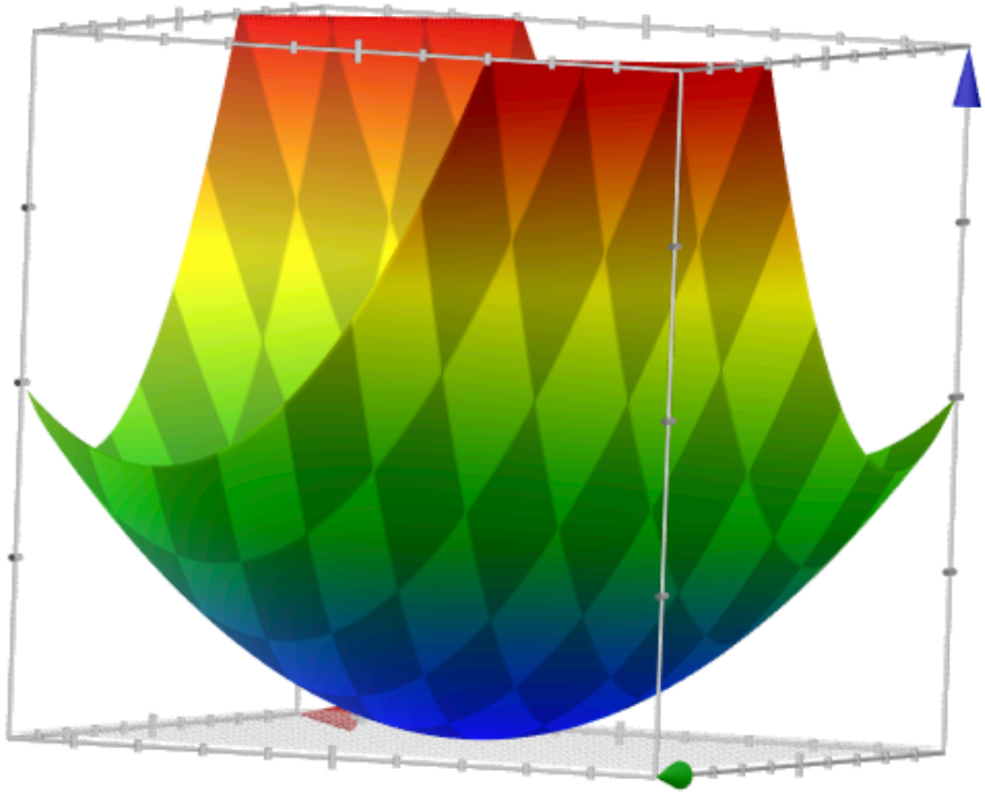
where $w_{2:d} = [w_2^{(t-1)}, \; \ldots \; , w_d^{(t-1)}]^T$

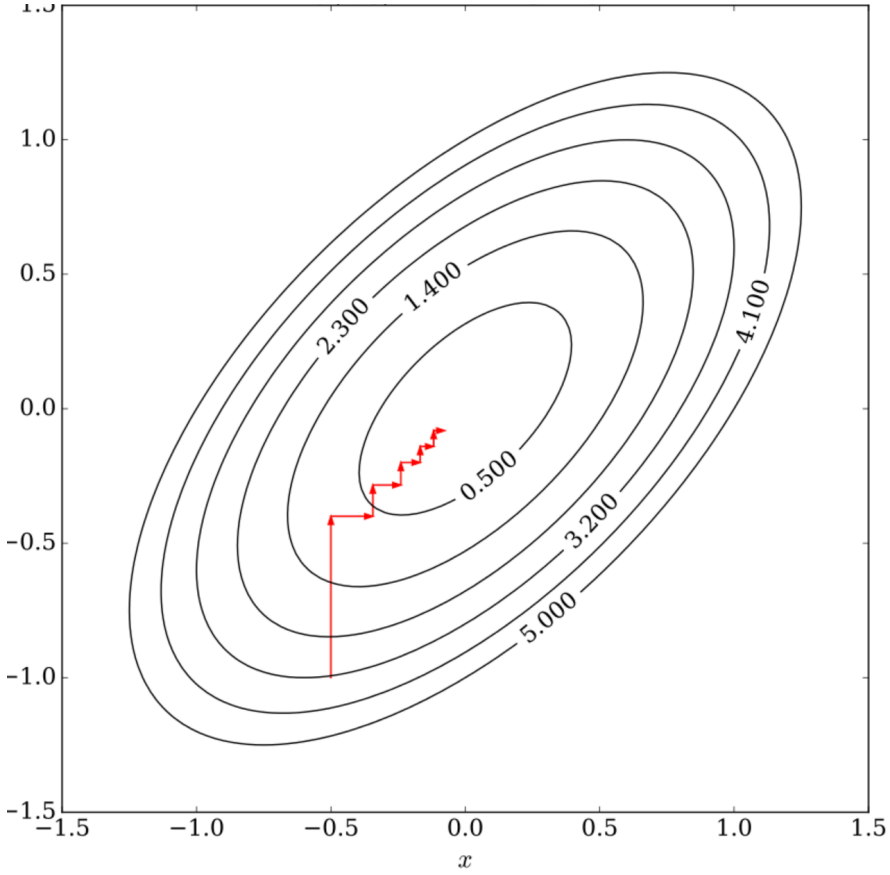


- we know from linear least squares that the minimizer is

$$w_1^{(t)} \leftarrow (\mathbf{X}[:,1]^T\mathbf{X}[:,1])^{-1}\,\mathbf{X}[:,1]^T\left(\mathbf{y} - \mathbf{X}[:,2:d]w_{2:d}\right)$$

- Coordinate descent applied to a quadratic loss





Coordinate descent for Lasso

- let us apply coordinate descent on Lasso, which minimizes

$$\text{minimize}_w \mathscr{L}(w) + \lambda \|w\|_1 = \|\mathbf{X}w - \mathbf{y}\|_2^2 + \lambda \|w\|_1$$

- the goal is to derive an **analytical rule** for updating $w_j^{(t)}$'s

- let us first write the update rule explicitly for $w_1^{(t)}$
  - first step is to write the loss in terms of w_1

$$\left\|\mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{2:d}\right)\right\|_2^2 + \lambda\Big(|w_1| + \underbrace{\|w_{2:d}\|_1}_{\text{constant}}\Big)$$

  - hence, the coordinate descent update boils down to

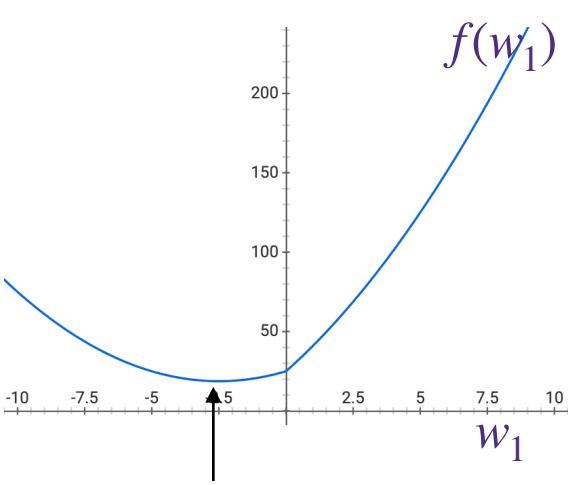
$$w_1^{(t)} \leftarrow \arg\min_{w_1} \underbrace{\left\|\mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-2:d}\right)\right\|_2^2 + \lambda |w_1|}_{f(w_1)}$$

Convexity

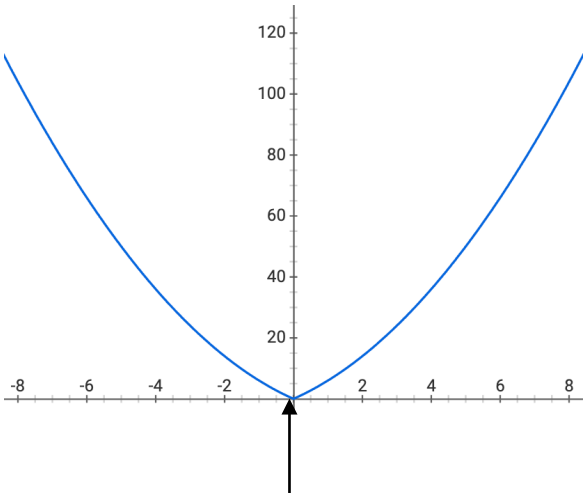
- to find the minimizer of $f(w_1)$, let's study some properties
- for simplicity, we represent the objective function as

$$f(w_1) = (aw_1 - b)^2 + \lambda \mid w_1 \mid$$

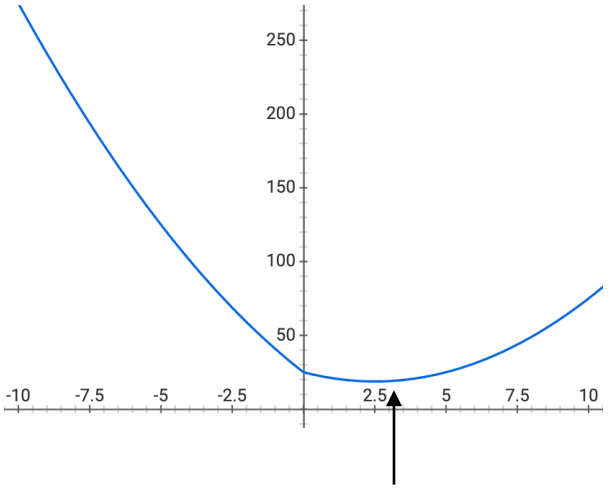
- this function is
  - **convex**, and
  - **non-differentiable**
- depending on the values of a and b, the function looks like one of the three below



minimum



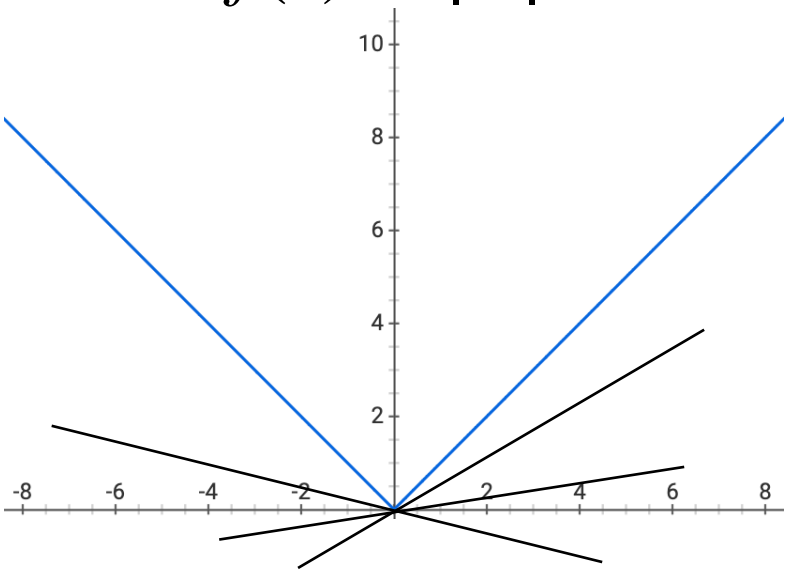
minimum



minimum

Convexity

$f(x) = |x|$



- for a **non-differentiable** function, gradient is not defined at some points, for example at $x = 0$ for $f(x) = |x|$
- at such points, **sub-gradient** plays the role of gradient
  - sub-gradient at a differentiable point is the same as the gradient
  - sub-gradient at a non-differentiable point is a set of vector satisfying

$$\partial f(x) = \{ g \in \mathbb{R}^d \,|\, f(y) \geq f(x) + g^T(y - x), \text{ for all } y \in \mathbb{R}^d \}$$

- for example, $\partial |x| = \begin{cases} +1 & \text{for } x > 0 \\ [-1, 1] & \text{for } x = 0 \\ -1 & \text{for } x < 0 \end{cases}$

Computing the sub-gradient

$$w_1^{(t)} = \arg\min_{w_1} \underbrace{\left\| \mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-1}\right) \right\|_2^2 + \lambda |w_1|}_{f(w_1)}$$

Computing the sub-gradient

$$w_1^{(t)} = \arg\min_{w_1} \underbrace{\left\| \mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-1}\right) \right\|_2^2 + \lambda\,|\,w_1\,|}_{f(w_1)}$$

- this is $f(w_1) = (aw_1 - b)^2 + \lambda\,|\,w_1\,|$ + constants, with
  - $a = \sqrt{\mathbf{X}[:,1]^T\mathbf{X}[:,1]}$, and
  - $b = \dfrac{\mathbf{X}[:,1]^T\left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-1}\right)}{\sqrt{\mathbf{X}[:,1]^T\mathbf{X}[:,1]}}$

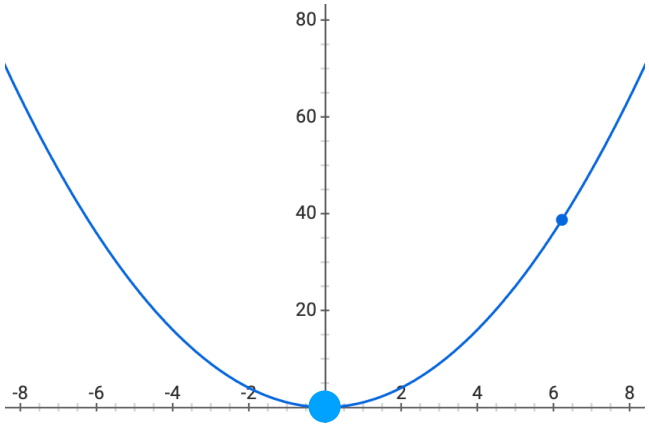
- $f(w_1)$ is non-differentiable, and its sub-gradient is

$$\partial f(w_1) = (2a(aw_1 - b) + \lambda\,\partial\,|\,w_1\,|$$

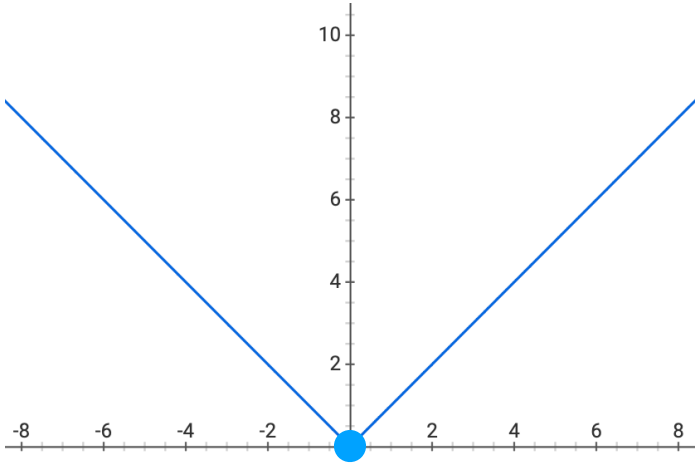
$$= \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

Convexity

- for convex differentiable functions, the minimum is achieved at points where gradient is zero



- for convex non-differentiable functions, the minimum is achieved at points where sub-gradient includes zero



Computing the sub-gradient

- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

$$\partial f(w_1) = \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

Computing the sub-gradient

- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

$$\partial f(w_1) = \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

Computing the sub-gradient

- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

$$\partial f(w_1) = \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

Computing the sub-gradient

- considering all three cases, we get the following update rule by setting the sub-gradient to zero

$$w_1^{(t)} \leftarrow \begin{cases} \frac{b}{a} - \frac{\lambda}{2a^2} & \text{for } 2ab > \lambda \\ 0 & \text{for } -\lambda \leq 2ab \leq \lambda \\ \frac{b}{a} + \frac{\lambda}{2a^2} & \text{for } \lambda < -2ab \end{cases}$$

How do we find the minimizer?

- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

$$\partial f(w_1) = \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

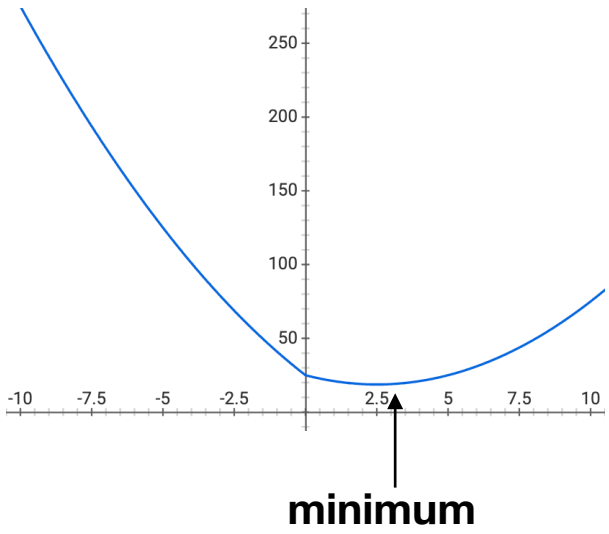
- case 1:
  - $2a(aw_1 - b) + \lambda = 0$ for some $w_1 > 0$
  - this happens when

$$w_1 = \frac{-\lambda + 2ab}{2a^2} > 0$$

  - hence,

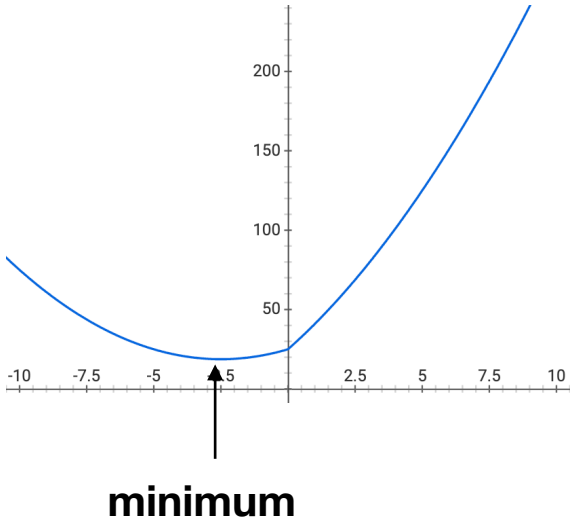
$$w_1^{(t)} \leftarrow \frac{b}{a} - \frac{\lambda}{2a^2},$$

  if $\lambda < 2ab$



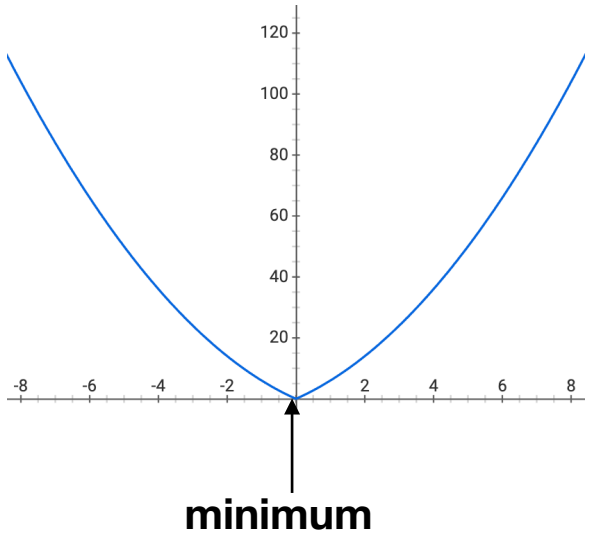
- case 2:
  - $2a(aw_1 - b) - \lambda = 0$ for some $w_1 < 0$
  - this happens when
$$w_1 = \frac{\lambda + 2ab}{2a^2} < 0$$
  - hence,
$$w_1^{(t)} \leftarrow \frac{b}{a} + \frac{\lambda}{2a^2},$$

    if $\lambda < -2ab$



- case 3:
  - $0 \in [-2ab - \lambda, -2ab + \lambda]$
  - and $w_1 = 0$
  - hence,
$$w_1^{(t)} \leftarrow 0,$$

    if $-\lambda \leq 2ab \leq \lambda$

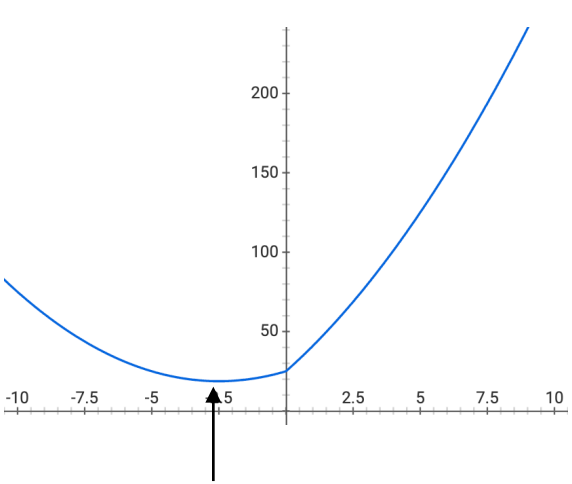


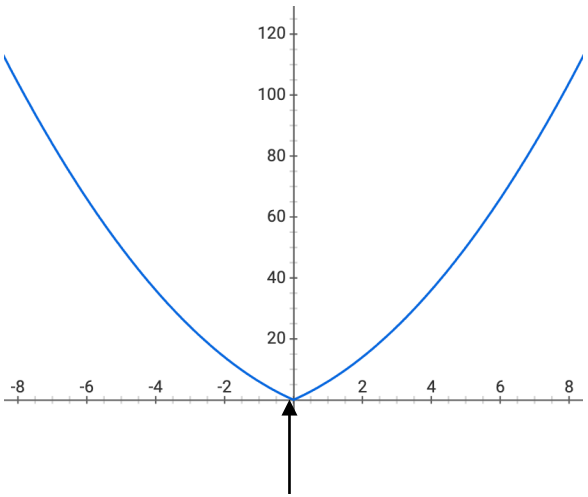
Coordinate descent on Lasso

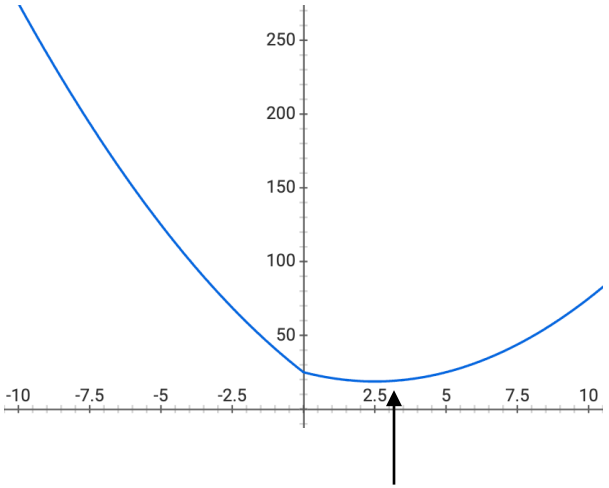
- considering all three cases, we get the following update rule by setting the sub-gradient to zero

$$w_1^{(t)} \leftarrow \begin{cases} \frac{b}{a} - \frac{\lambda}{2a^2} & \text{for } 2ab > \lambda \\ 0 & \text{for } -\lambda \leq 2ab \leq \lambda \\ \frac{b}{a} + \frac{\lambda}{2a^2} & \text{for } \lambda < -2ab \end{cases}$$

- where $a = \sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}$, and $b = \frac{\mathbf{X}[:,1]^T \left(\mathbf{y} - \mathbf{X}[:,2:d] w_{-1}\right)}{\sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}}$

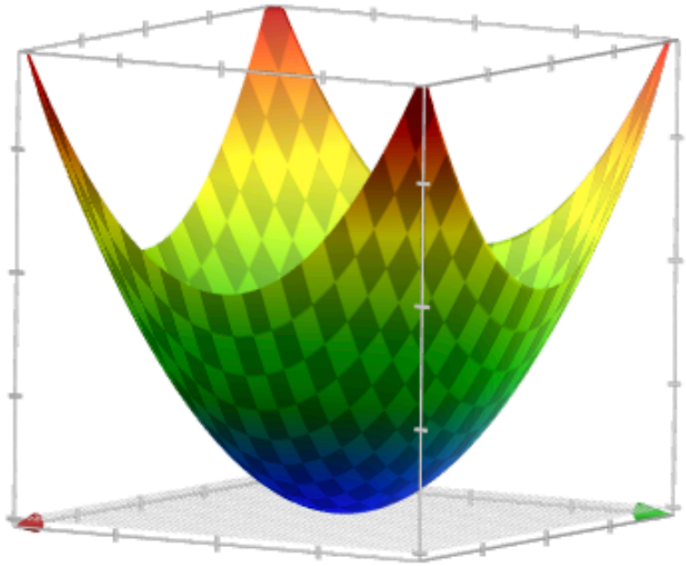


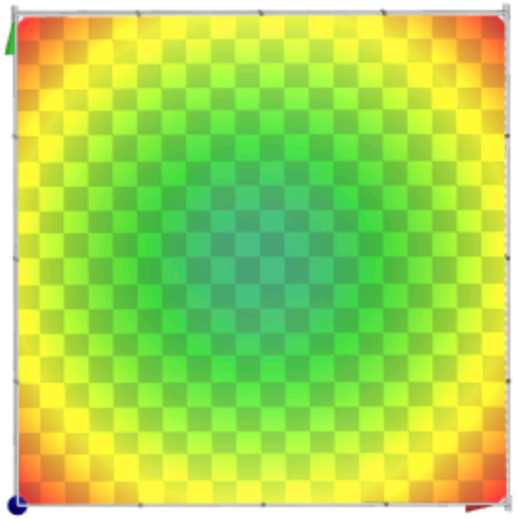




When does coordinate descent work?

- Consider minimizing a **differentiable convex** function $f(x)$, then coordinate descent converges to the global minima

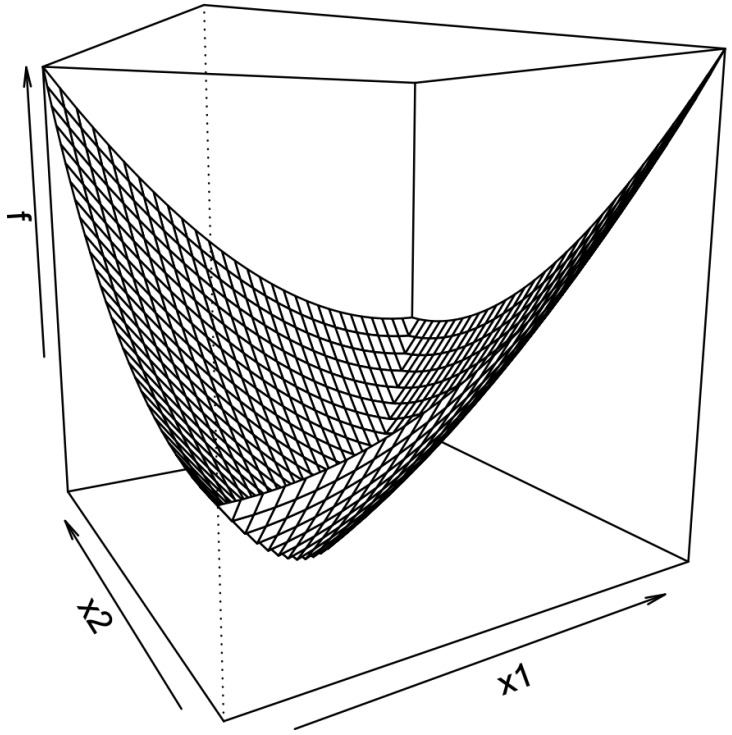


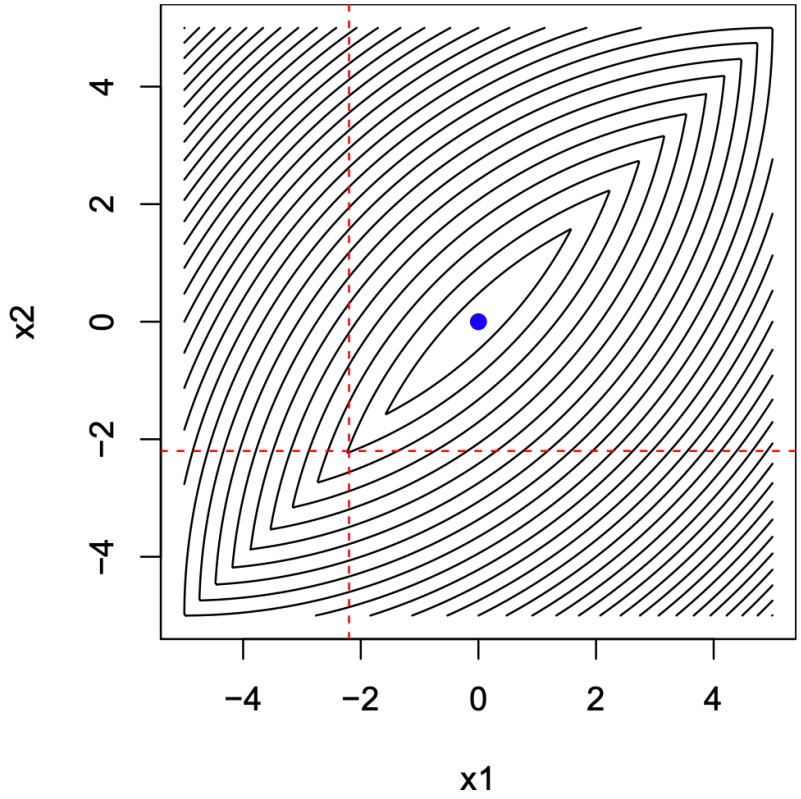


- when coordinate descent has stopped, that means $\frac{\partial f(x)}{\partial x_j} = 0$ for all $j \in \{1, \ldots, d\}$
- this implies that the gradient $\nabla_x f(x) = 0$, which happens only at minimum

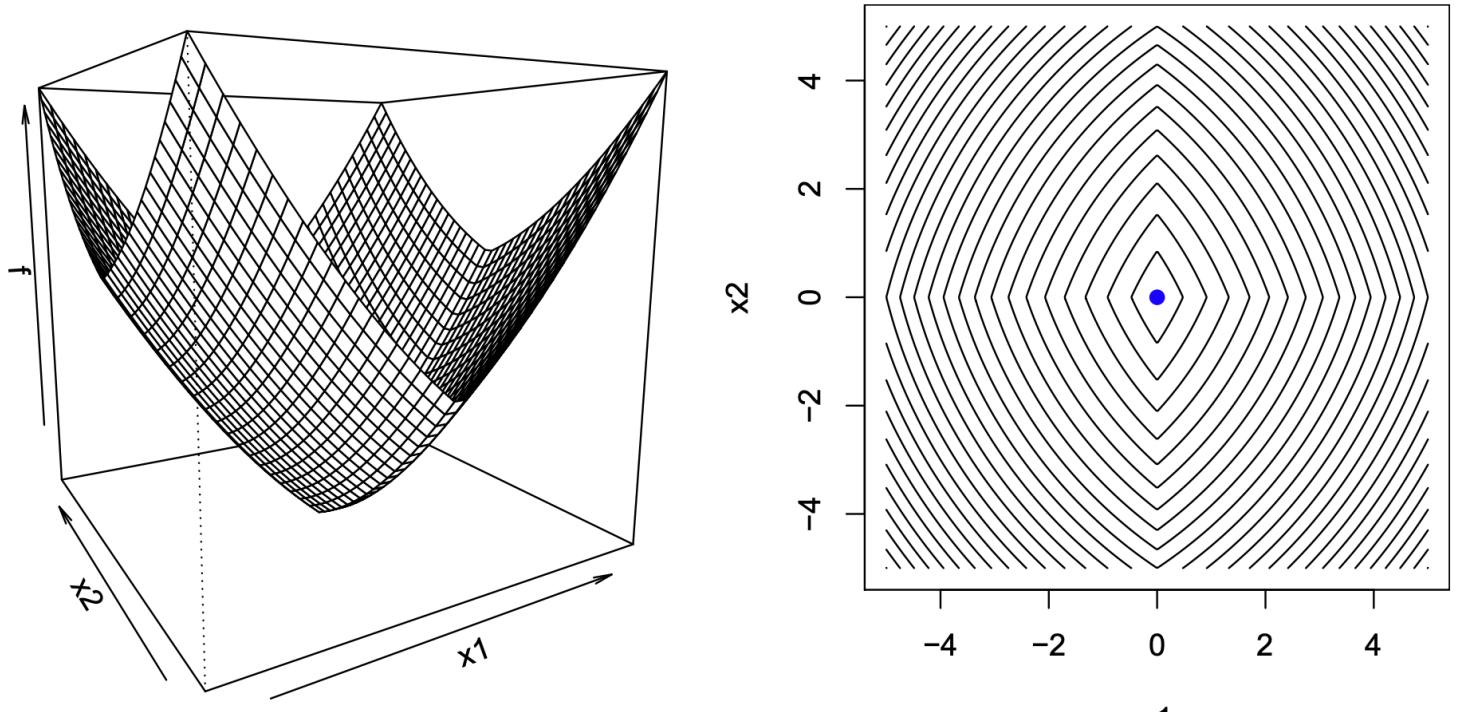
When does coordinate descent work?

- Consider minimizing a **non-differentiable convex** function $f(x)$, then coordinate descent can get stuck





When does coordinate descent work?

- then how can coordinate descent find optimal solution for Lasso?
- consider minimizing a **non-differentiable convex** function but has a structure of $f(x) = g(x) + \sum_{j=1}^{d} h_j(x_j)$, with differentiable convex function $g(x)$ and coordinate-wise non-differentiable convex functions $h_j(x_j)$'s, then coordinate descent converges to the global minima
- 

Stochastic Gradient Descent

Machine Learning Problems

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^{n} \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- **Learning a model's parameters:** $\frac{1}{n}\sum_{i=1}^{n} \ell_i(w)$

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$$

Machine Learning Problems

- **Given data:**

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- **Learning a model's parameters:** $\frac{1}{n}\sum_{i=1}^{n} \ell_i(w)$

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$$

Stochastic Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w) \Big|_{w=w_t}$$

I_t drawn uniform at random from $\{1, \ldots, n\}$

$$\mathbb{E}[\nabla \ell_{I_t}(w)] =$$

Stochastic Gradient Descent

Theorem

Let $w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w)\Big|_{w=w_t}$ I_t drawn uniform at random from $\{1, \dots, n\}$ so that

$$\mathbb{E}\big[\nabla \ell_{I_t}(w)\big] = \frac{1}{n}\sum_{i=1}^{n} \nabla \ell_i(w) =: \nabla \ell(w)$$

If $\|w_0 - w_*\|_2^2 \le R$ and $\sup_w \max_i \|\nabla \ell_i(w)\|_2^2 \le G$ then

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \le \frac{R}{2T\eta} + \frac{\eta G}{2} \le \sqrt{\frac{RG}{T}} \qquad \eta = \sqrt{\frac{R}{GT}}$$

$$\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$$

(In practice use last iterate)

Stochastic Gradient Descent

Proof

$$\mathbb{E}[||w_{t+1} - w_*||_2^2] = \mathbb{E}[||w_t - \eta \nabla \ell_{I_t}(w_t) - w_*||_2^2]$$

Stochastic Gradient Descent

Proof

$$\mathbb{E}[||w_{t+1} - w_*||_2^2] = \mathbb{E}[||w_t - \eta \nabla \ell_{I_t}(w_t) - w_*||_2^2]$$

Stochastic Gradient Descent

Proof

$$\mathbb{E}[||w_{t+1} - w_*||_2^2] = \mathbb{E}[||w_t - \eta \nabla \ell_{I_t}(w_t) - w_*||_2^2]$$

$$= \mathbb{E}[||w_t - w_*||_2^2] - 2\eta \mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*)] + \eta^2 \mathbb{E}[||\nabla \ell_{I_t}(w_t)||_2^2]$$

$$\leq \mathbb{E}[||w_t - w_*||_2^2] - 2\eta \mathbb{E}[\ell(w_t) - \ell(w_*)] + \eta^2 G$$

$$\mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*)] = \mathbb{E}\Big[\mathbb{E}[\nabla \ell_{I_t}(w_t)^T (w_t - w_*) | I_1, w_1, \ldots, I_{t-1}, w_{t-1}]\Big]$$

$$= \mathbb{E}\Big[\nabla \ell(w_t)^T (w_t - w_*)\Big]$$

$$\geq \mathbb{E}\Big[\ell(w_t) - \ell(w_*)\Big]$$

$$\sum_{t=1}^{T} \mathbb{E}[\ell(w_t) - \ell(w_*)] \leq \frac{1}{2\eta} \left(\mathbb{E}[||w_1 - w_*||_2^2] - \mathbb{E}[||w_{T+1} - w_*||_2^2] + T\eta^2 G\right)$$

$$\leq \frac{R}{2\eta} + \frac{T\eta G}{2}$$

Stochastic Gradient Descent

Proof

Jensen's inequality:
For any random $Z \in \mathbb{R}^d$ and convex function $\phi : \mathbb{R}^d \to \mathbb{R}$, $\phi(\mathbb{E}[Z]) \leq \mathbb{E}[\phi(Z)]$

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \leq \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\ell(w_t) - \ell(w_*)] \qquad \bar{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$$

Stochastic Gradient Descent

Proof

Jensen's inequality:
For any random $Z \in \mathbb{R}^d$ and convex function $\phi : \mathbb{R}^d \to \mathbb{R}$, $\phi(\mathbb{E}[Z]) \leq \mathbb{E}[\phi(Z)]$

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \leq \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\ell(w_t) - \ell(w_*)] \qquad \bar{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$$

$$\mathbb{E}[\ell(\bar{w}) - \ell(w_*)] \leq \frac{R}{2T\eta} + \frac{\eta G}{2} \leq \sqrt{\frac{RG}{T}} \qquad \eta = \sqrt{\frac{R}{GT}}$$

Mini-batch SGD

Instead of one iterate, average B stochastic gradient together

Advantages:

- Smaller variance
- Parallelization