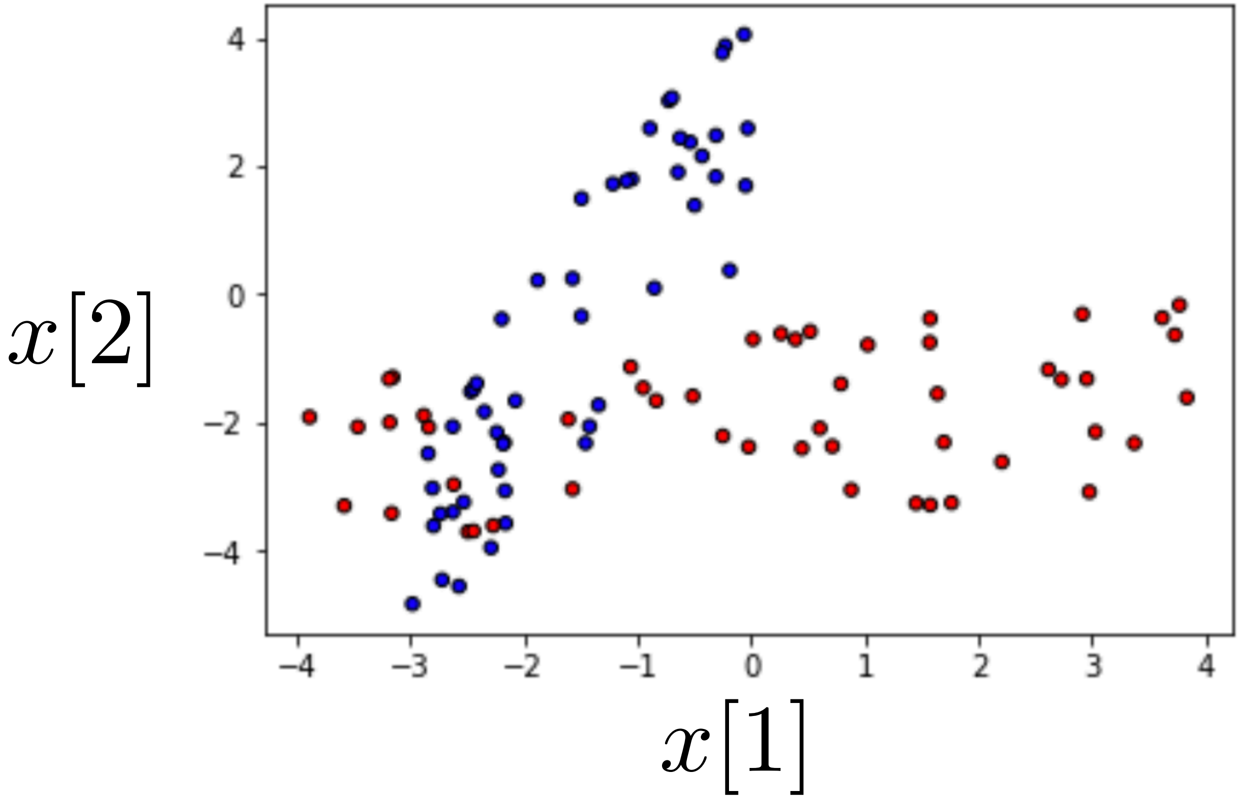
Logistics:
- New Office Hours now on course website. Some on zoom some in-person.

Lecture 12: Classification with logistic regression (continued)

- Regression: label is continuous valued
- Classification: label is discrete valued, e.g., {0,1}

- Note that logistic regression is
  a classification algorithm not a regression algorithm

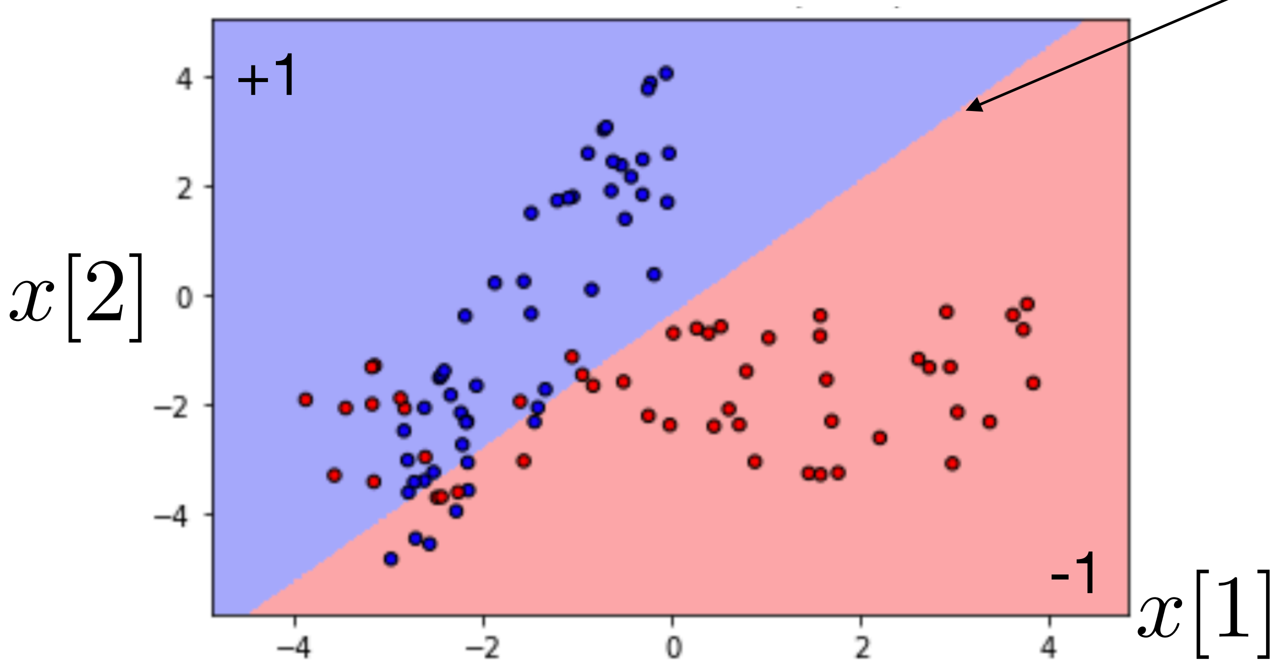
Training data for a binary classification problem



- in this example, each input is $x_i \in \mathbb{R}^2$
- Red points have label y_i=-1, blue points have label y_i=1
- We want a predictor that maps any $x \in \mathbb{R}^2$ to a prediction $\hat{y} \in \{-1, +1\}$

Example: linear classifier trained on 100 samples

simple decision boundary at $w^T x + b = 0$



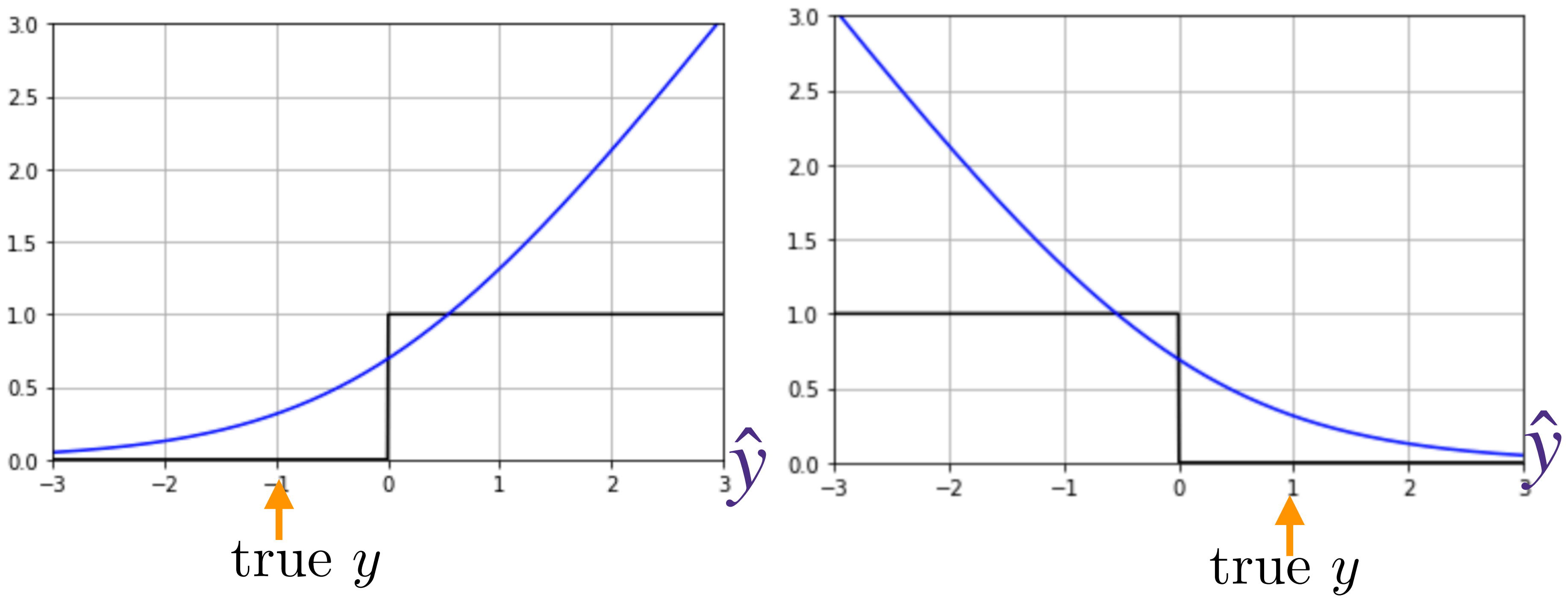
- We fit a linear model: $w_0 + w_1 x[1] + w_2 x[2] = 0.8 - 1.1x[1] + 0.9x[2]$
- predict using $\hat{y} = \mathrm{sign}(0.8 - 1.1x[1] + 0.9x[2])$
- decision boundary is the line (or hyperplane in higher dimensions) defined by

$$0.8 - 1.1x[1] + 0.9x[2] = 0$$

- note that a model $2w^T x + 2b$ has the same predictions as $w^T x + b$
- How do we find such a good linear classifier that fits the data?

Logistic loss $\ell(\hat{y}, y) = \log(1 + e^{-y\hat{y}})$

$$\ell(\hat{y}, -1) = \log(1 + e^{\hat{y}}) \qquad \ell(\hat{y}, +1) = \log(1 + e^{-\hat{y}})$$



- differentiable and convex in $\hat{y}$
- how do we show $\ell(\,\cdot\,, y)$ is convex?
- approximation of 0-1
- Most popular choice of a loss function for classification problems

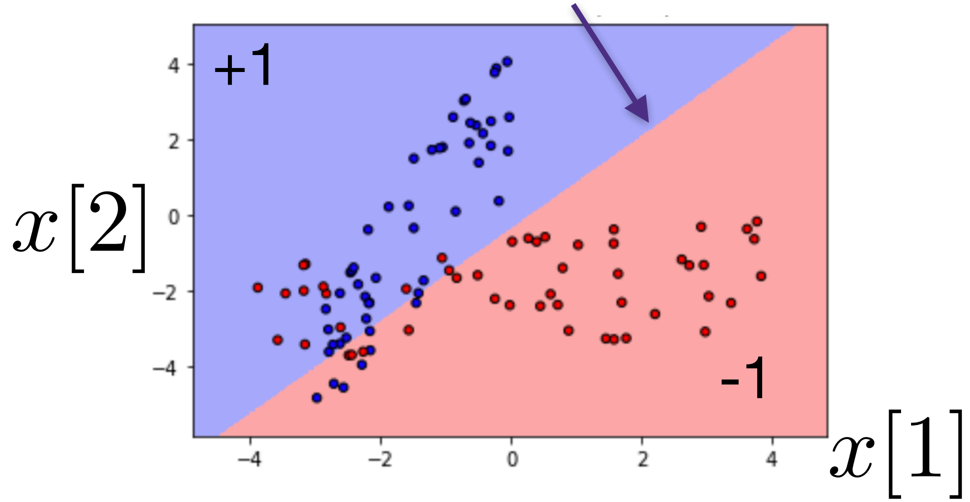
Logistic regression for binary classification

- Data $\mathscr{D} = \{(x_i \in \mathbb{R}^d, y_i \in \{-1, +1\})\}_{i=1}^n$
- Model: $\hat{y} = x^T w + b$
- Loss function: logistic loss $\ell(\hat{y}, y) = \log(1 + e^{-y\hat{y}})$
- Optimization: solve for

$$(\widehat{b}, \widehat{w}) = \arg\min_{b,w} \sum_{i=1}^{n} \log(1 + e^{-y_i(b + x_i^T w)})$$

- As this is a **smooth convex** optimization, it can be solved efficiently using gradient descent
- Prediction: $\text{sign}(b + x^T w)$

decision boundary at $w^T x + b = 0$



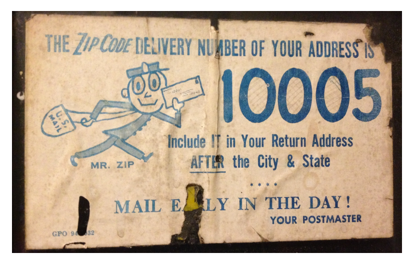
Multi-class regression

How do we encode general categorical data *y*?

- so far, we considered Binary case where there are two categories
- encoding y is simple: {+1,-1}
- We output a scalar prediction $\hat{y} = b + w^T x$, and take the sign

- multi-class classification predicts categorial y
- taking values in $y \in C = \{c_1, \ldots, c_k\}$
- c_j's are called **classes** or **labels**
- examples:

Country of birth
(Argentina, Brazil, USA,...)



Zipcode
(10005, 98195,...)

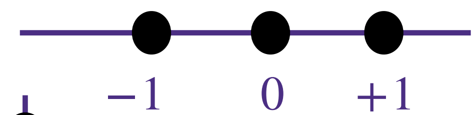
All English words

- a **k-class classifier** predicts y given x, but we need to represent categorical y with numerical values

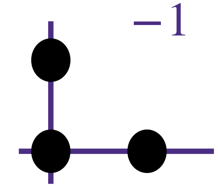
Embedding c_j's in real values

- Since we rely on optimization, which are numerical solvers, we need to **embed** raw categorical c_j's into real valued vectors
- there are many ways to embed categorial data
  - True->1, False->-1, can we use it for all binary classification?

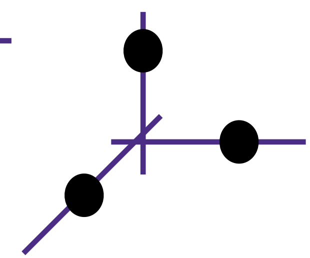
  - Yes->1, Maybe->0, No->-1 can we use it for all ternary classification?



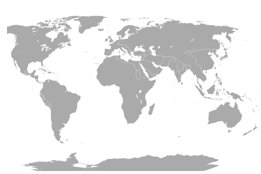
  - Yes->(1,0), Maybe->(0,0), No->(0,1)



  - Apple->(1,0,0), Orange->(0,1,0), Banana->(0,0,1)



- we use **one-hot embedding** (a.k.a. **one-hot encoding**), because it is neutral representation with no domain knowledge
  - each class is a standard basis vector in $k-$dimension



Country of birth
(Argentina, Brazil, USA,...)

$\underbrace{}_{\text{196 categories}}$

1-hot encoding

x	$h_1(x)$	$h_2(x)$	...	$h_{195}(x)$	$h_{196}(x)$
Brazil	0	1	0	0	0
Zimbabwe	0	0	0	0	1

$\underbrace{}_{\text{196 features}}$

Multi-class logistic regression

- **data**: categorical y in $\{c_1, \ldots, c_k\}$ with k categories

  we use **one-hot encoding**, s.t. $y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ implies that $y = c_1$

- **model**: linear vector-function makes a linear prediction $\hat{y} \in \mathbb{R}^k$

$$\hat{y}_i \;=\; f(x_i) \;=\; w^T x_i \quad \in \mathbb{R}^k$$

with model parameter matrix $w \in \mathbb{R}^{d \times k}$ and sample $x_i \in \mathbb{R}^d$

$$f(x_i) = \begin{bmatrix} f_1(x_i) \\ f_2(x_i) \\ \vdots \\ f_k(x_i) \end{bmatrix} = \underbrace{\begin{bmatrix} w_{1,0} & w_{1,1} & w_{1,2} & \cdots \\ w_{2,0} & w_{2,1} & w_{2,2} & \cdots \\ \vdots & & & \\ w_{k,0} & w_{k,1} & w_{k,2} & \cdots \end{bmatrix}}_{w^T} \underbrace{\begin{bmatrix} 1 \\ x_i[1] \\ \vdots \\ x_i[d] \end{bmatrix}}_{x_i} = \begin{bmatrix} w_{1,0} + w_{1,1}x_i[1] + w_{1,2}x_i[2] + \cdots \\ w_{2,0} + w_{2,1}x_i[1] + w_{2,2}x_i[2] + \cdots \\ \vdots \\ w_{k,0} + w_{k,1}x_i[1] + w_{k,2}x_i[2] + \cdots \end{bmatrix}$$

$$w = \begin{bmatrix} w[:,1] & w[:,2] & \cdots & w[:,k] \end{bmatrix}$$

Model parameter for category 1

2 equivalent approaches for binary Logistic regression

- What we learned so far:
  single model parameter $w \in \mathbb{R}^d$

$$\mathbb{P}(y_i = +1 \mid x_i) = \frac{1}{1 + e^{-w^T x_i}} = \frac{e^{w^T x_i}}{1 + e^{w^T x_i}}$$

$$\mathbb{P}(y_i = -1 \mid x_i) = 1 - \mathbb{P}(y_i = +1 \mid x_i)$$

- Another approach: two parameters $w_1, w_2 \in \mathbb{R}^d$
  - $w_1^T x$ represents how much category 1 is more likely at a data point x (relatively to $w_2^T x$)
  - $w_2^T x$ represents how much category 2 is more likely at a data point x

$$\mathbb{P}(y_i = 1 \mid x_i) = \frac{e^{w_1^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

$$\mathbb{P}(y_i = 2 \mid x_i) = \frac{e^{w_2^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

Maximum Likelihood Estimator

$$\text{maximize}_w \ \frac{1}{n}\sum_{i=1}^n \log(\mathbb{P}(y_i \mid x_i))$$

$$\text{maximize}_{w \in \mathbb{R}^d} \ \frac{1}{n}\sum_{i=1}^n \log\left(\frac{e^{y_i w^T x_i}}{1 + e^{y_i w^T x_i}}\right)$$

Representing the two categories as {-1,+1} helps simplify the notations, where we used the fact that $\frac{e^{-w^T x}}{1 + e^{-w^T x}} = 1 - \frac{e^{w^T x}}{1 + e^{w^T x}}$

$$\text{maximize}_{w_1, w_2 \in \mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \left\{ \boxed{\log\left(\frac{e^{w_{y_i}^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}\right)} \right\}$$

Representing the two categories as {1,2} is inline with the notation for the parameters w_1 and w_2, and we simplify the numerator using w_{y_i}

- Do these two approaches give different solutions?
- The second approach is more symmetric, and hence generalizes to larger classes

Logistic regression

2 classes

$$\mathbb{P}(y_i = 1 \,|\, x_i) = \frac{e^{w_1^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

$$\mathbb{P}(y_i = 2 \,|\, x_i) = \frac{e^{w_2^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

k classes

$$\mathbb{P}(y_i = 1 \,|\, x_i) = \frac{e^{w[:,1]^T x_i}}{e^{w[:,1]^T x_i} + \cdots + e^{w[:,k]^T x_i}}$$

$$\vdots$$

$$\mathbb{P}(y_i = k \,|\, x_i) = \frac{e^{w[:,k]^T x_i}}{e^{w[:,1]^T x_i} + \cdots + e^{w[:,k]^T x_i}}$$

Maximum Likelihood Estimator

$$\text{maximize}_w \;\; \frac{1}{n} \sum_{i=1}^{n} \log(\mathbb{P}(y_i \,|\, x_i))$$

$$\text{maximize}_{w_1, w_2 \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \boxed{\log\left(\frac{e^{w_{y_i}^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}} \right)} \right\}$$

$$\text{maximize}_{w \in \mathbb{R}^{d \times k}} \frac{1}{n} \sum_{i=1}^{n} \boxed{\log\left(\frac{e^{w[:,y_i]^T x_i}}{\sum_{j=1}^{k} e^{w[:,j]^T x_i}} \right)}$$

Gradient Descent

- how are we going to find the solution for

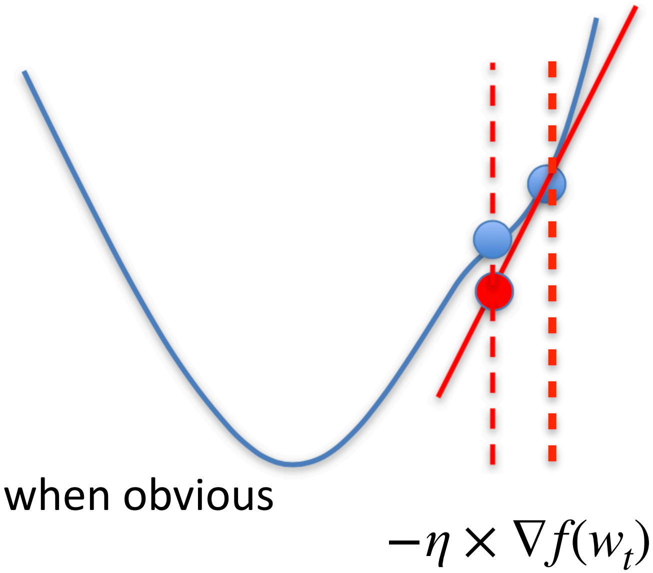
$$\arg\min_{b,w} \sum_{i=1}^{n} \ell(b + w^T x_i, y_i)$$

- e.g., Lasso, Logistic Regression do not have closed form solution for

$$\nabla_{b,w}\mathscr{L}(b, w) = 0$$

Gradient descent

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



- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

I will omit w in ∇_w when obvious

- Learning rate or step-size
- A hyper-parameter to be chosen by the analyst
- Can be fixed over the iterations,
  or can also change, in which case we use η_t
  to emphasize the fact that it changes over iteration t

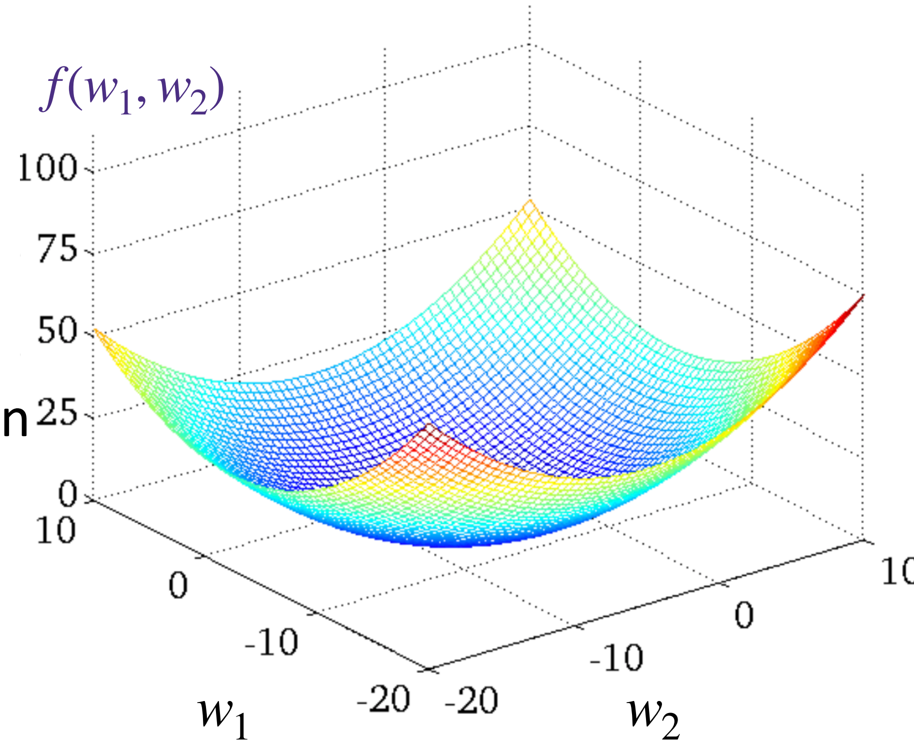
Running example: linear regression

- **Given data:**

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

- **Learning model parameters:**

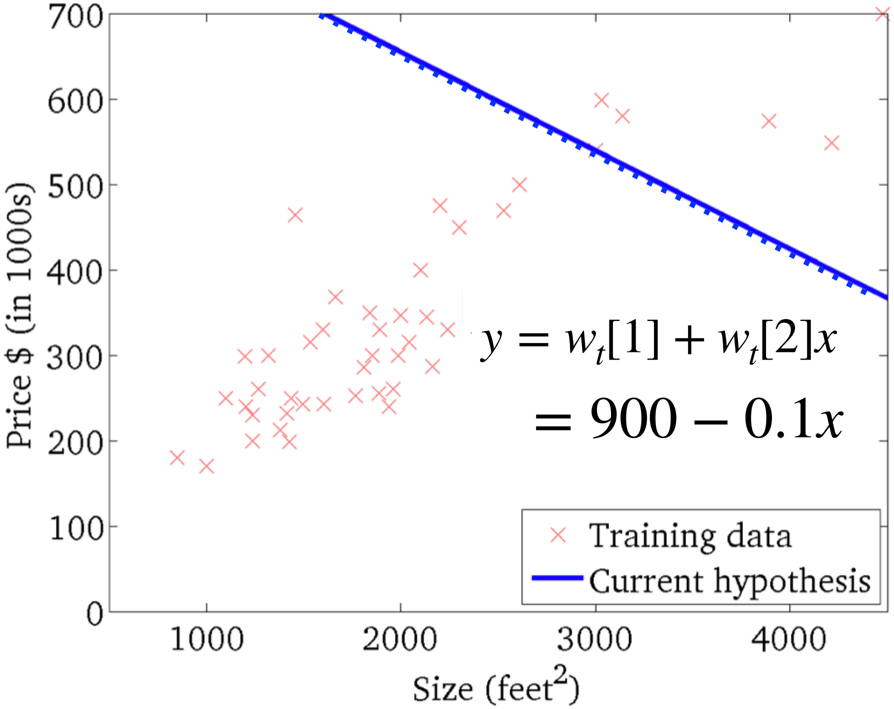
$$\hat{w}_{\text{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$



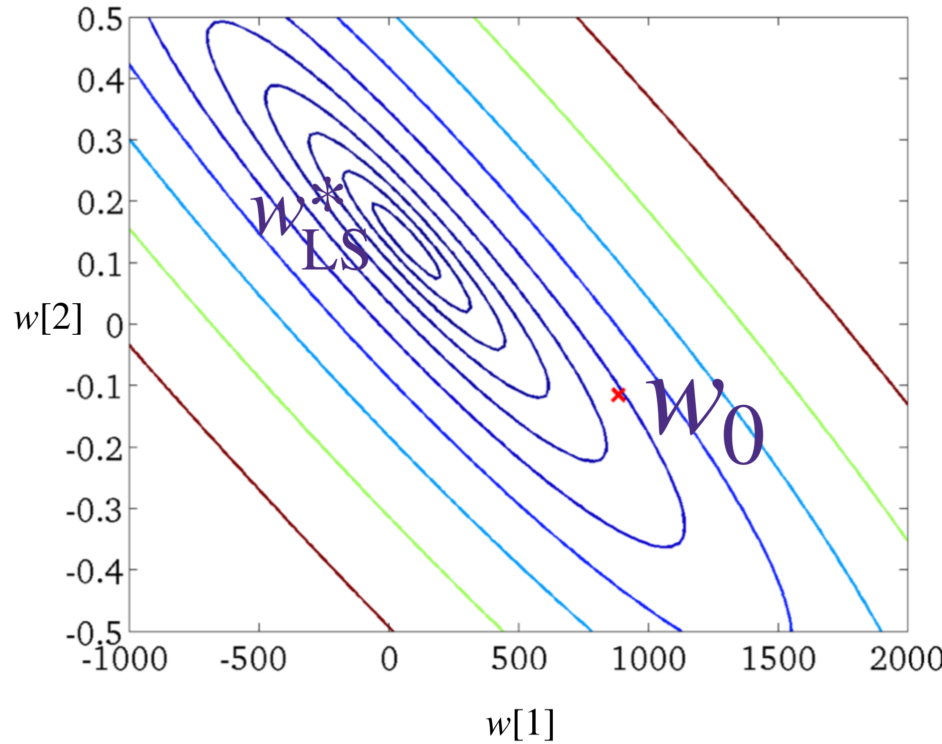
- Although we know the optimal solution in a closed form, we will use this as a running example to understand GD

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

$\{(x_i, y_i)\}_{i=1}^{n}$



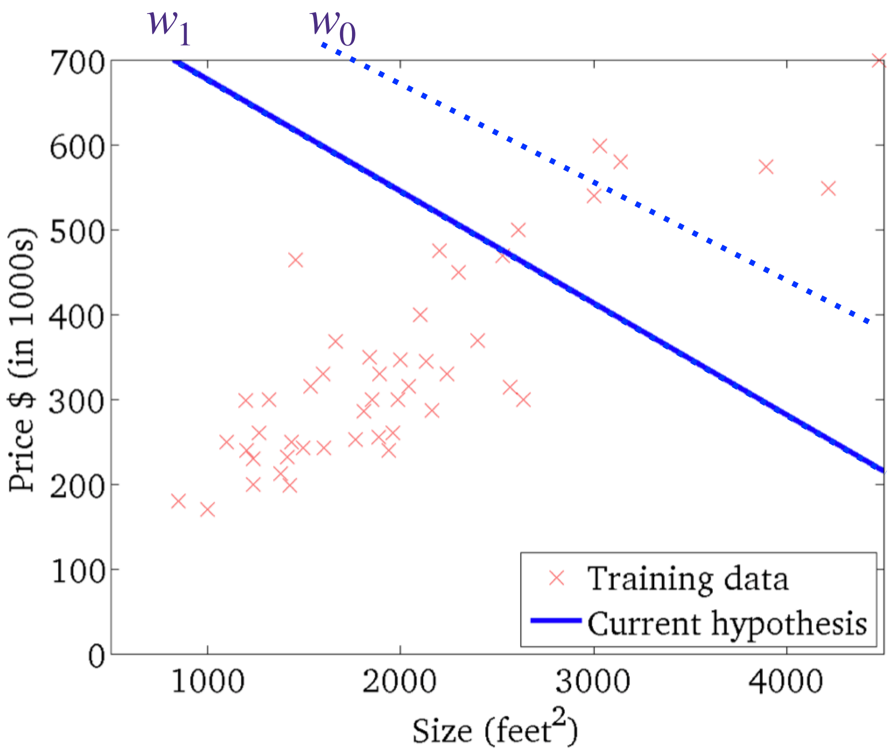
Evolution of the predictor



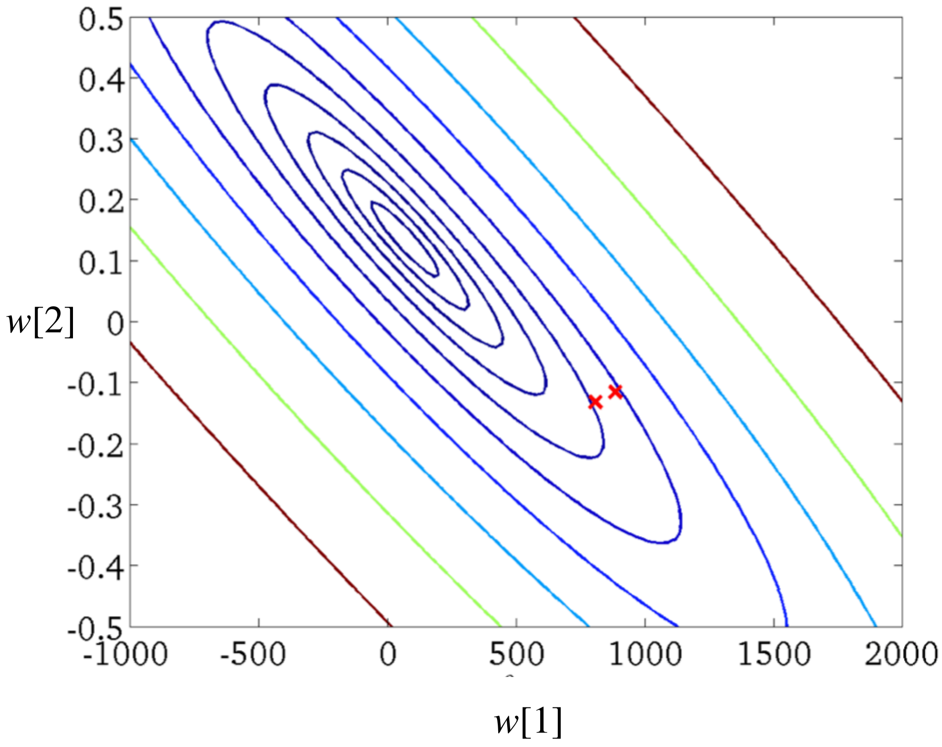
GD dynamics in the Parameter space

- Which direction will the GD move?

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

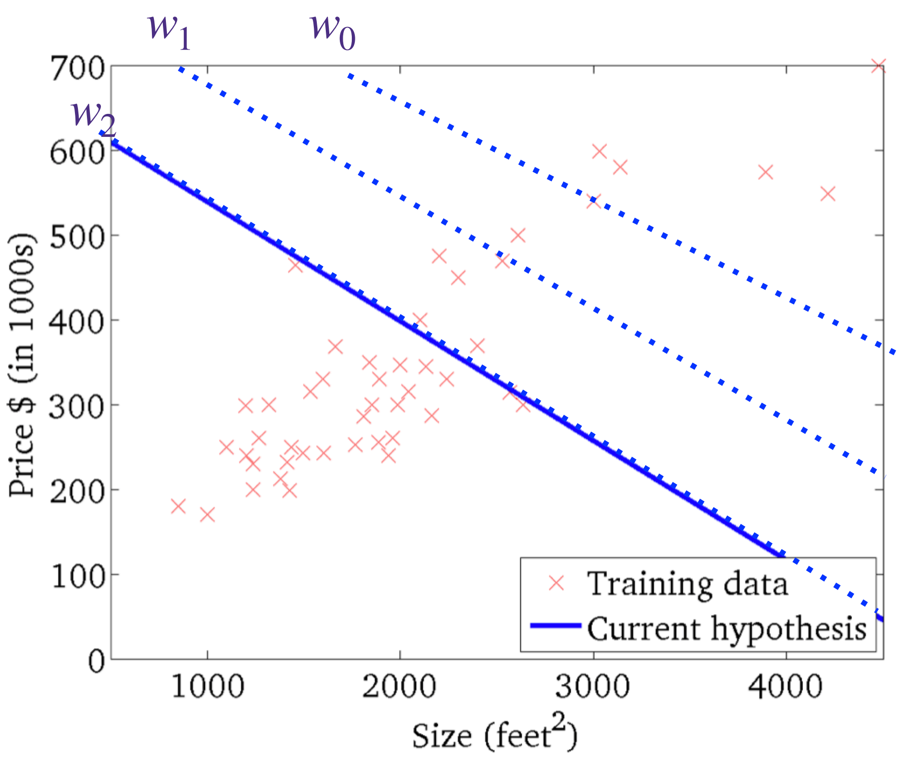


Evolution of the predictor

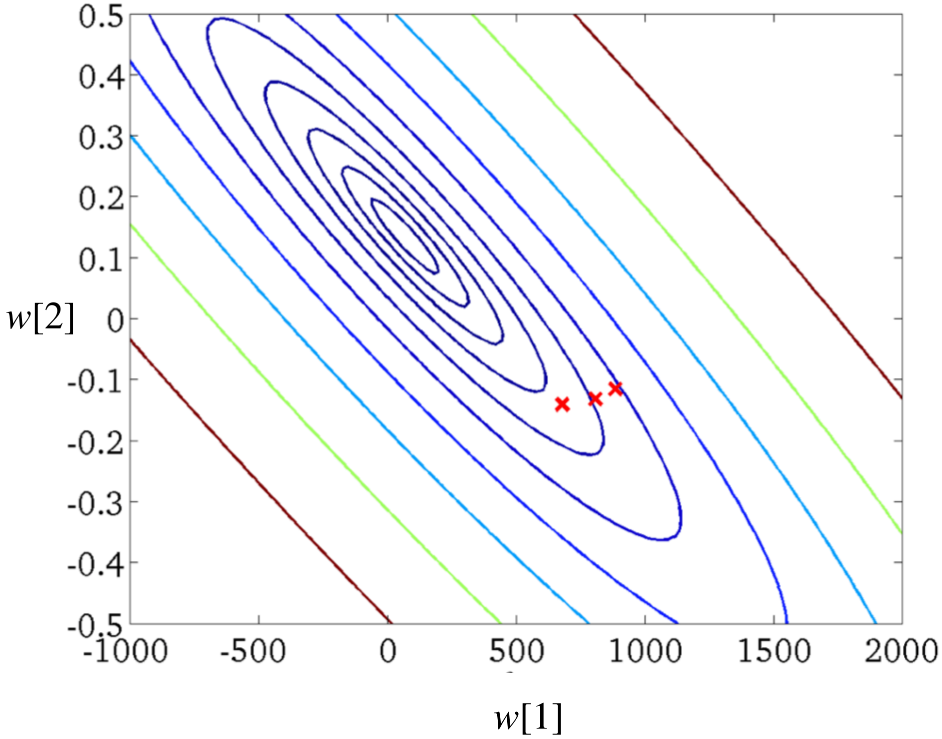


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

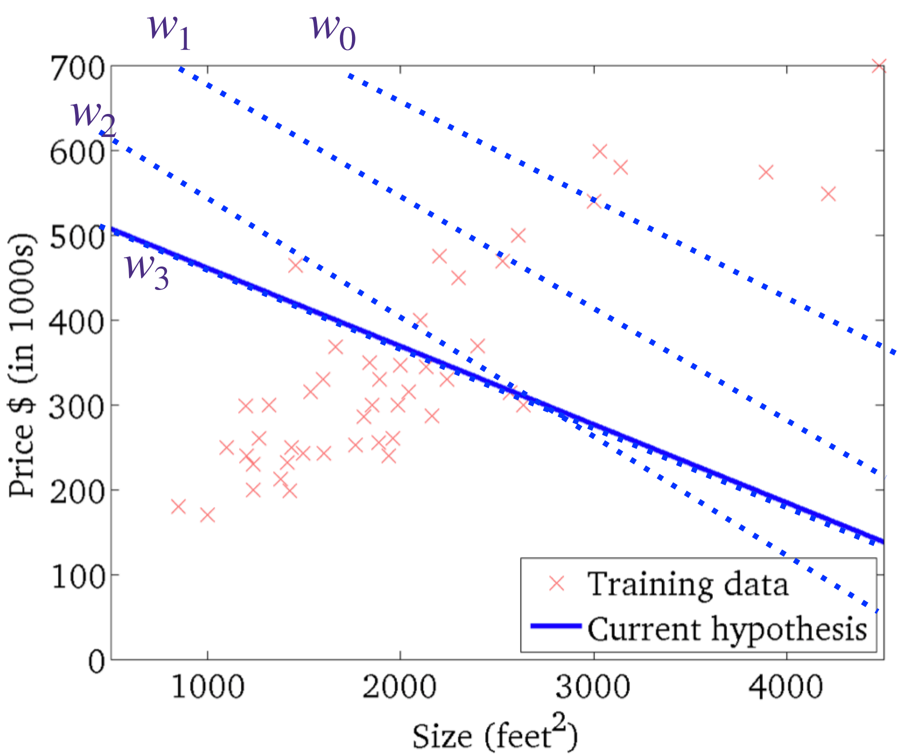


Evolution of the predictor

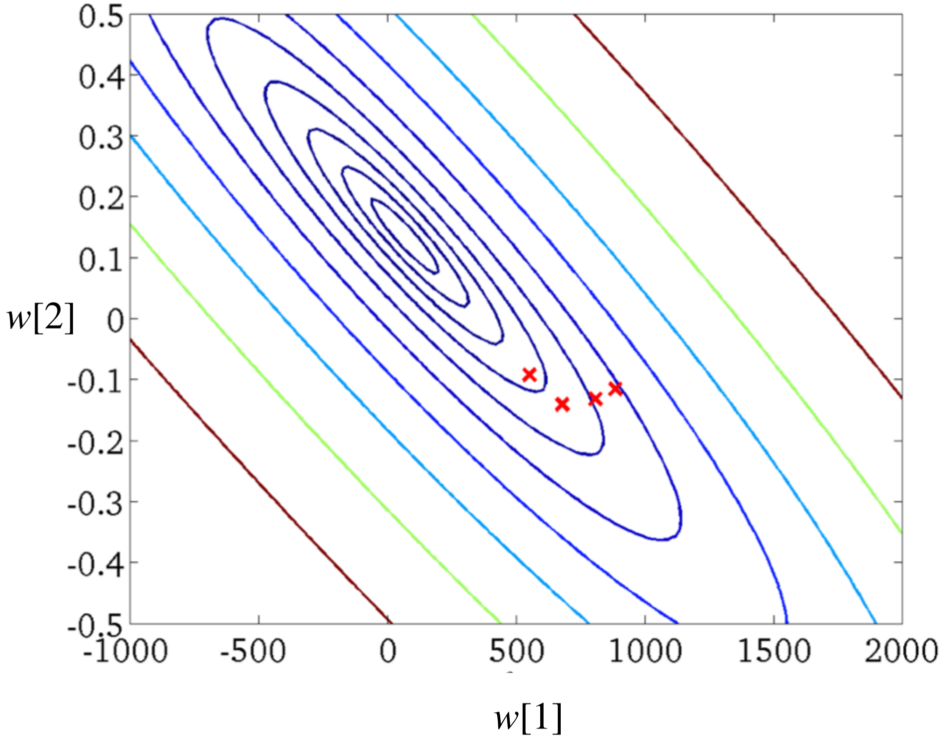


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

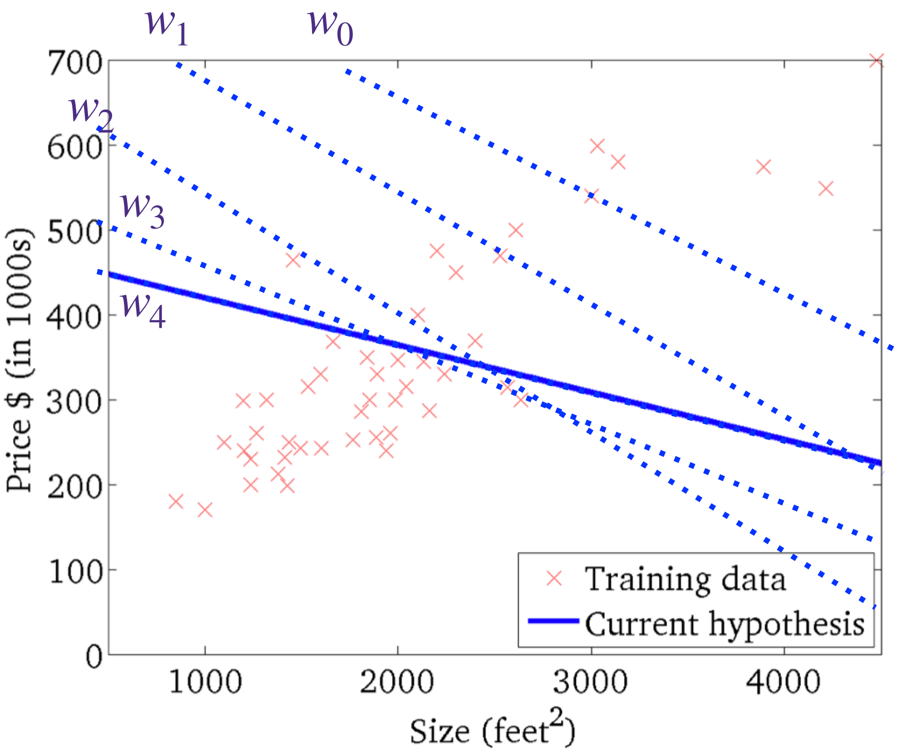


Evolution of the predictor

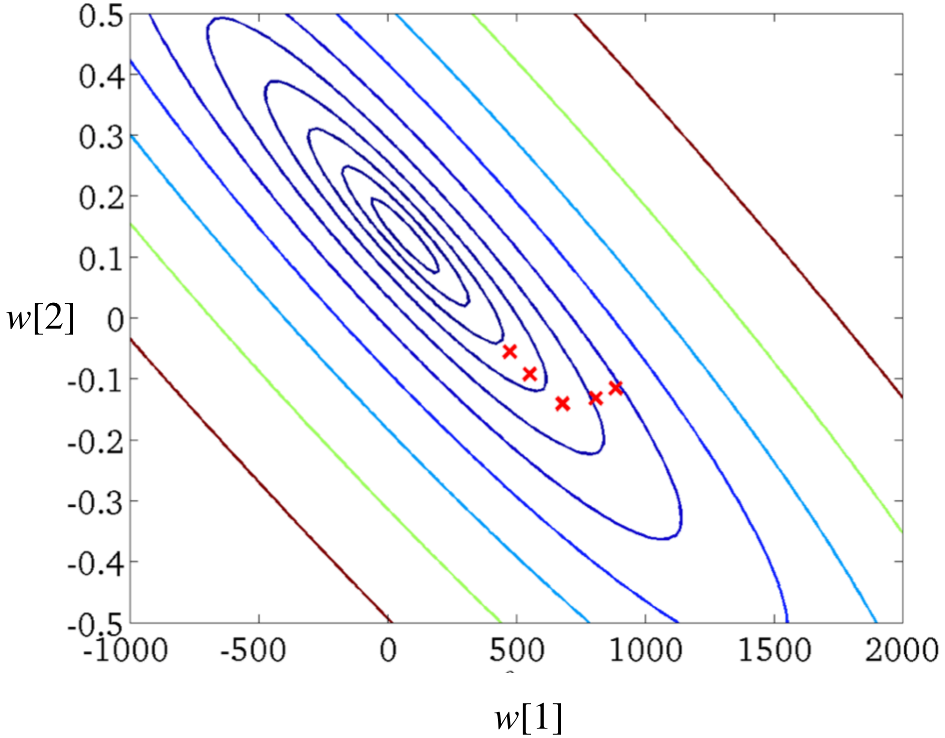


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

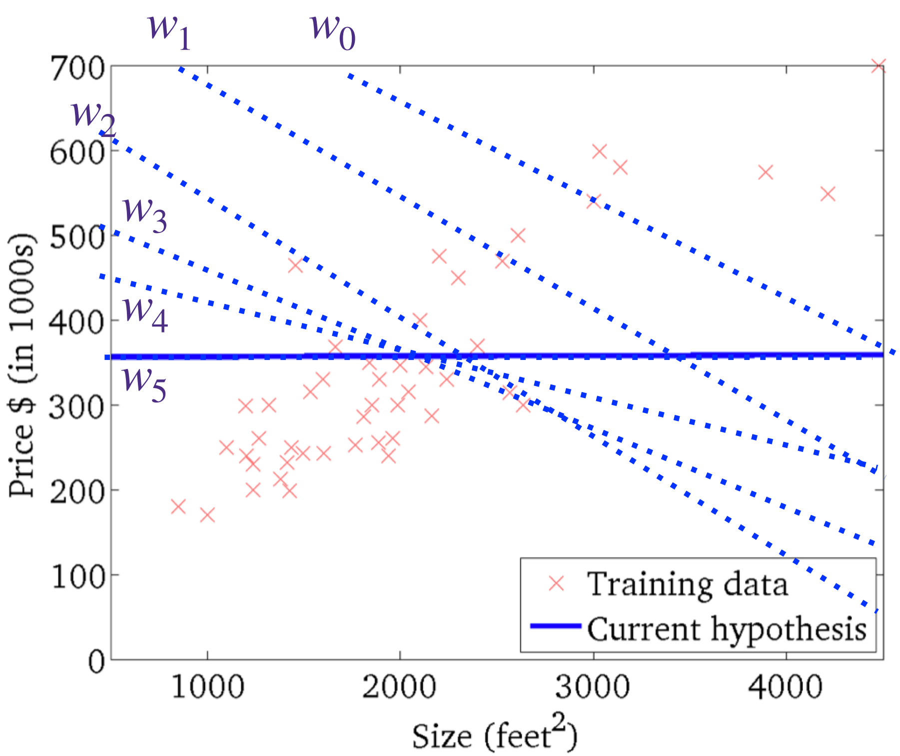


Evolution of the predictor

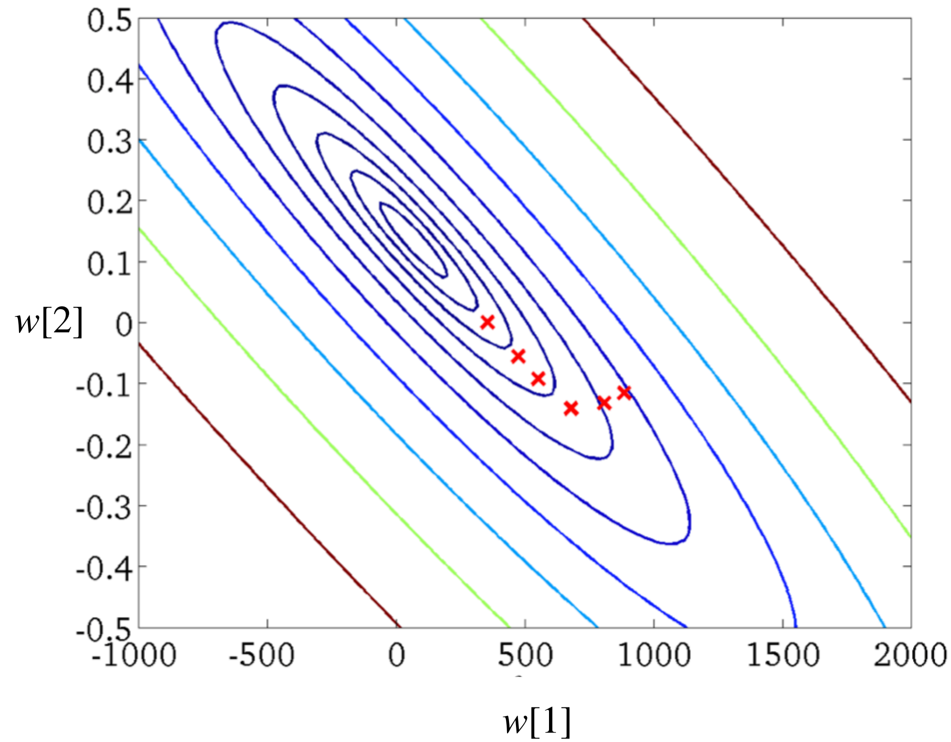


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

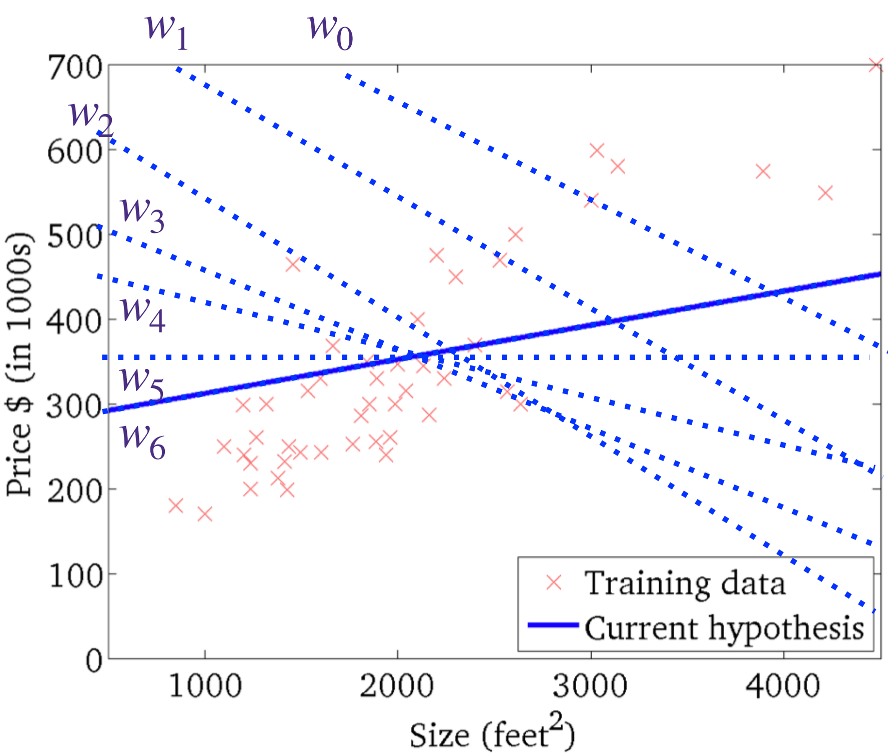


Evolution of the predictor

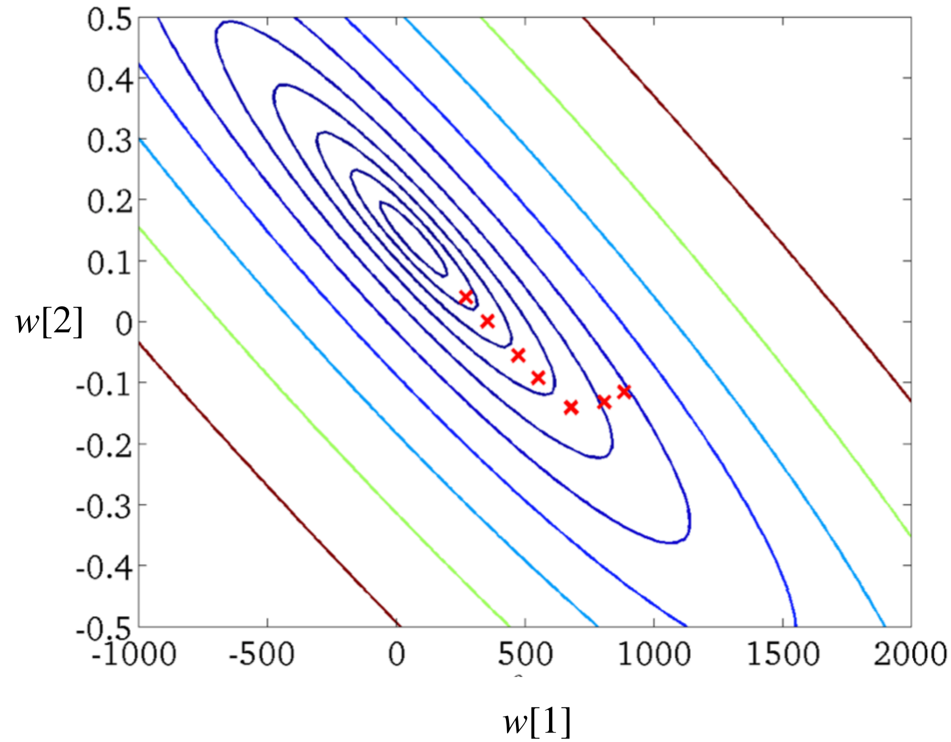


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,...
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

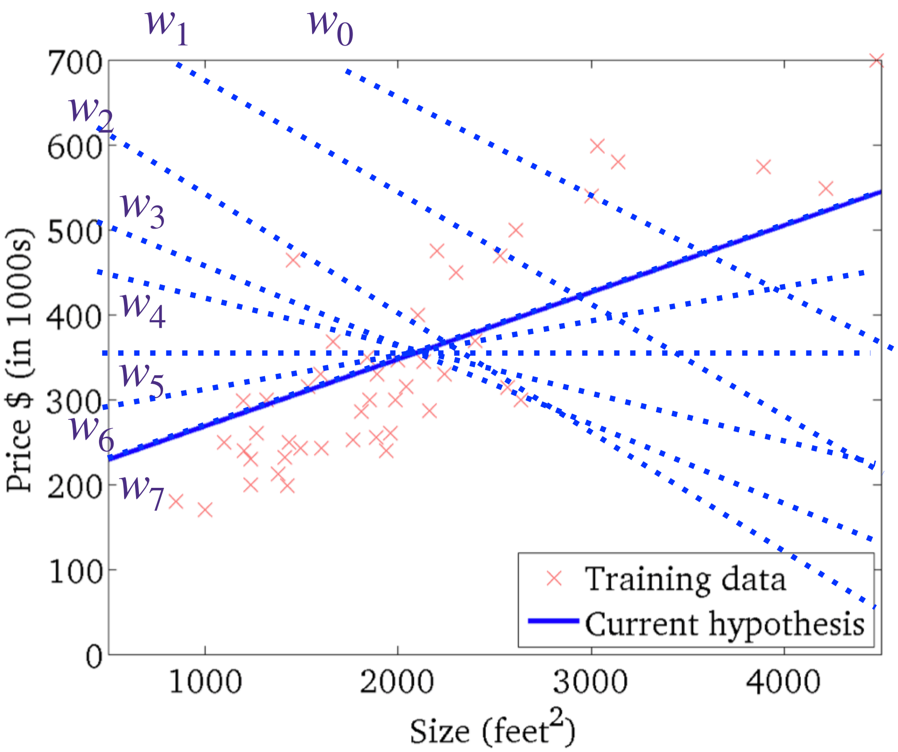


Evolution of the predictor

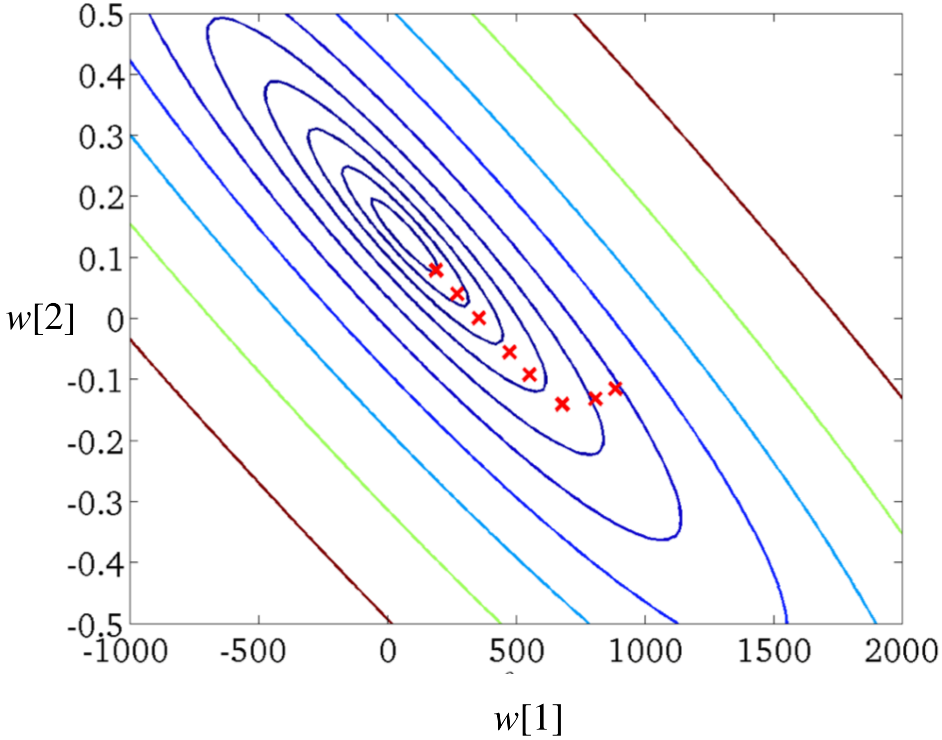


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
    - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

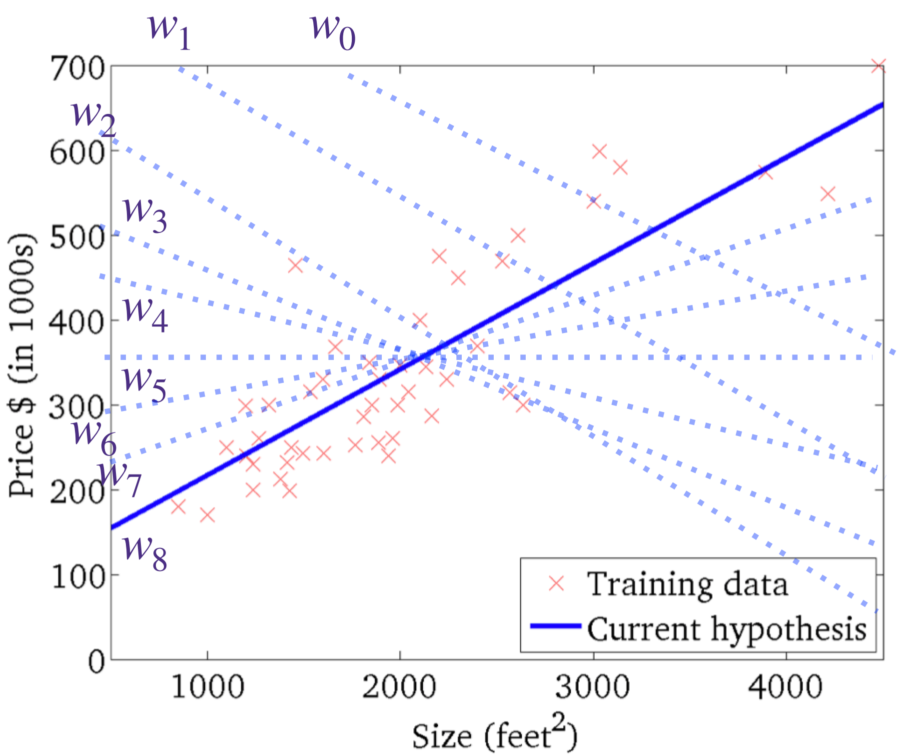


Evolution of the predictor

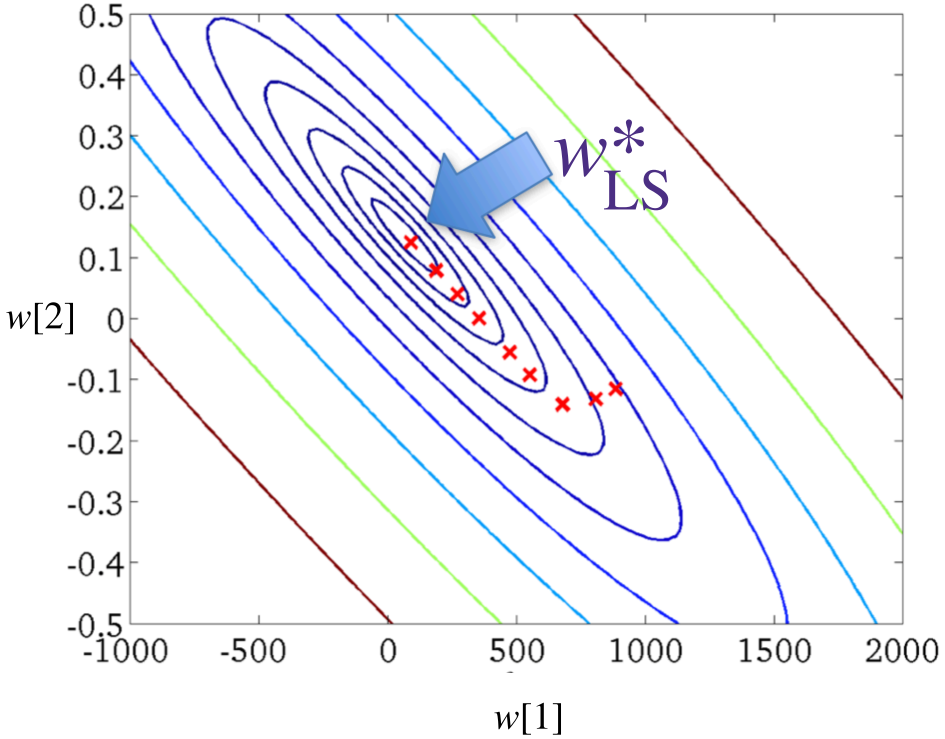


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,...
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$



Evolution of the predictor



GD dynamics in the Parameter space

Gradient descent for linear regression

- In this example of linear regression, we can derive exactly the gradient descent trajectory
- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

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

- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

For linear regression, we have

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

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

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

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 (GD) for Linear Regression (LR)

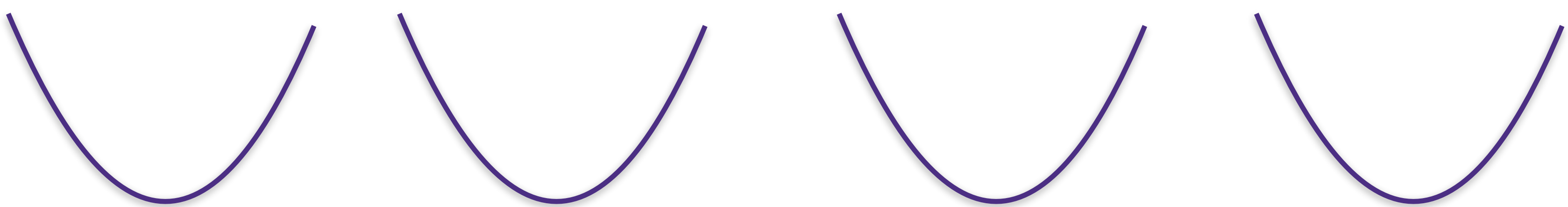
- We use this analytical derivation of GD for LR to understand how the choice of step size impacts the algorithm

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

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, and $w_{t+1} - w^* = (1 - \eta a)^{t+1}(w_0 - w^*)$



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

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 consider eigenvalues of $\mathbf{X}^T\mathbf{X}$, 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 consider eigenvalues of $\mathbf{X}^T\mathbf{X}$,
  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

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

- Where eigenvalue decomposition of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ is $Q^{-1}DQ$
- Let $Q = \begin{bmatrix} - & q_1^T & - \\ - & q_2^T & - \\ & \vdots & \end{bmatrix}$, then the above multi-dimensional dynamics of GD can be decomposed into multiple 1-d dynamics we saw before
- In direction q_1, the error decreases multiplicatively according to

$$q_1^T(w_{t+1} - w^*) = D_{11}^{t+1}\,q_1^T(w_0 - w^*)$$

$$q_2^T(w_{t+1} - w^*) = D_{22}^{t+1}\,q_2^T(w_0 - w^*)$$

$$\vdots$$

Gradient descent for linear regression

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

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

$$q_1^T (w_{t+1} - w^*) = D_{11}^{t+1}\, q_1^T (w_0 - w^*)$$

$$q_2^T (w_{t+1} - w^*) = D_{22}^{t+1}\, q_2^T (w_0 - w^*)$$

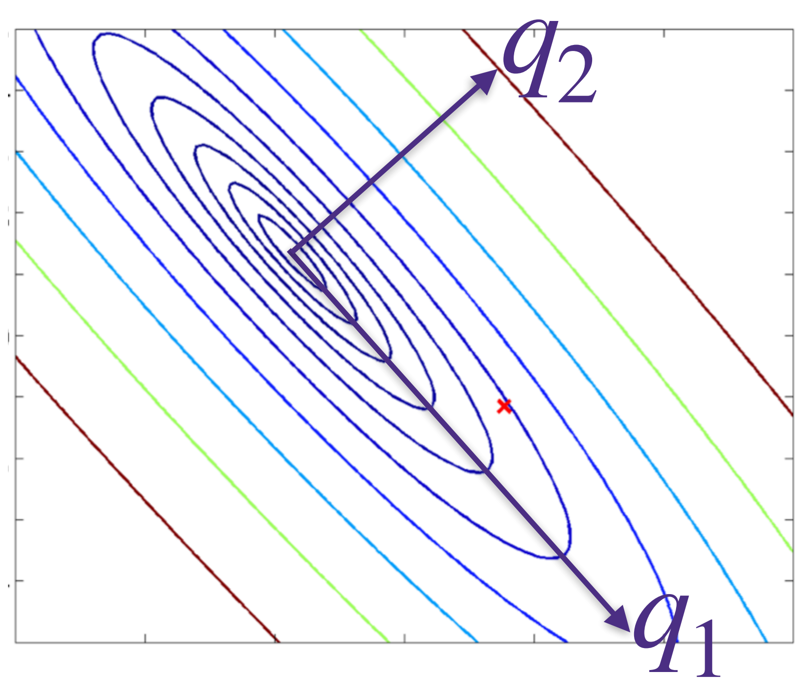
- For example suppose, the step size η is chosen such that

In direction q_1

$0 < D_{11} < 1$

In direction q_2

$-1 < D_{22} < 0$

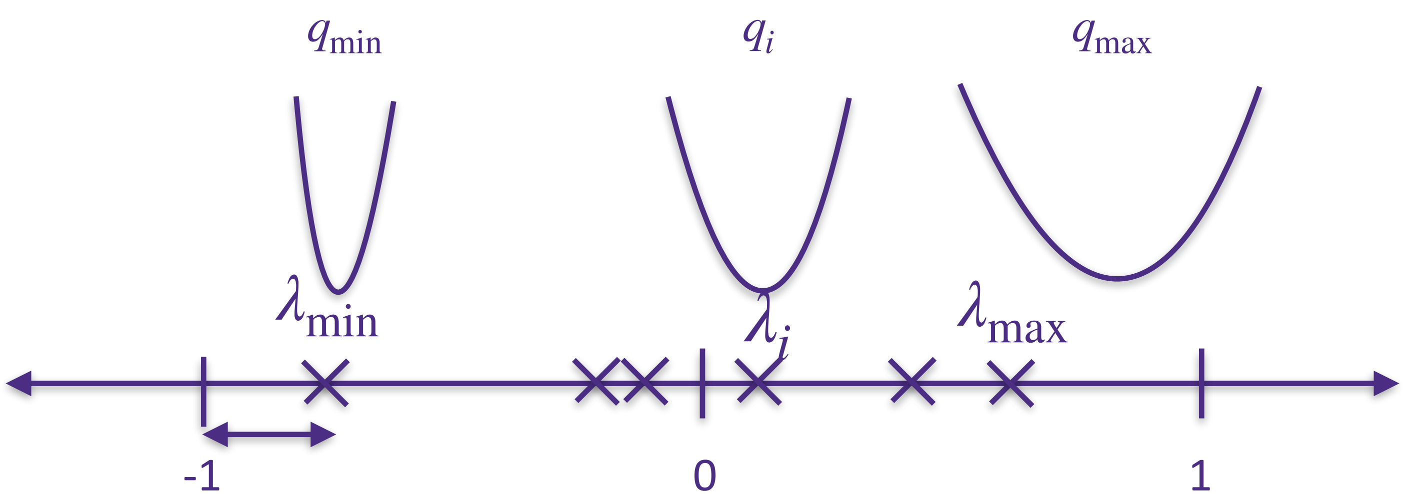


Gradient descent for linear regression

- Note that $D_{ii} := \lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$ is defined as the i-th Eigen value of the matrix $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$
- Recall that in each eigen direction error evolves as

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

- We want the error to decay fast in all directions, whose bottleneck are the largest and the smallest eigen values: $\lambda_{\min}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$ and $\lambda_{\max}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$



We want to choose the learning rate η such that

$$-1 \ll \lambda_{\min}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \le \cdots \le \lambda_{\max}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \ll 1$$

Gradient descent for linear regression

- We use linear algebra to answer the question:
  how does the rate of error decay change with step size η
- Recall that in each eigen direction error decays as

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

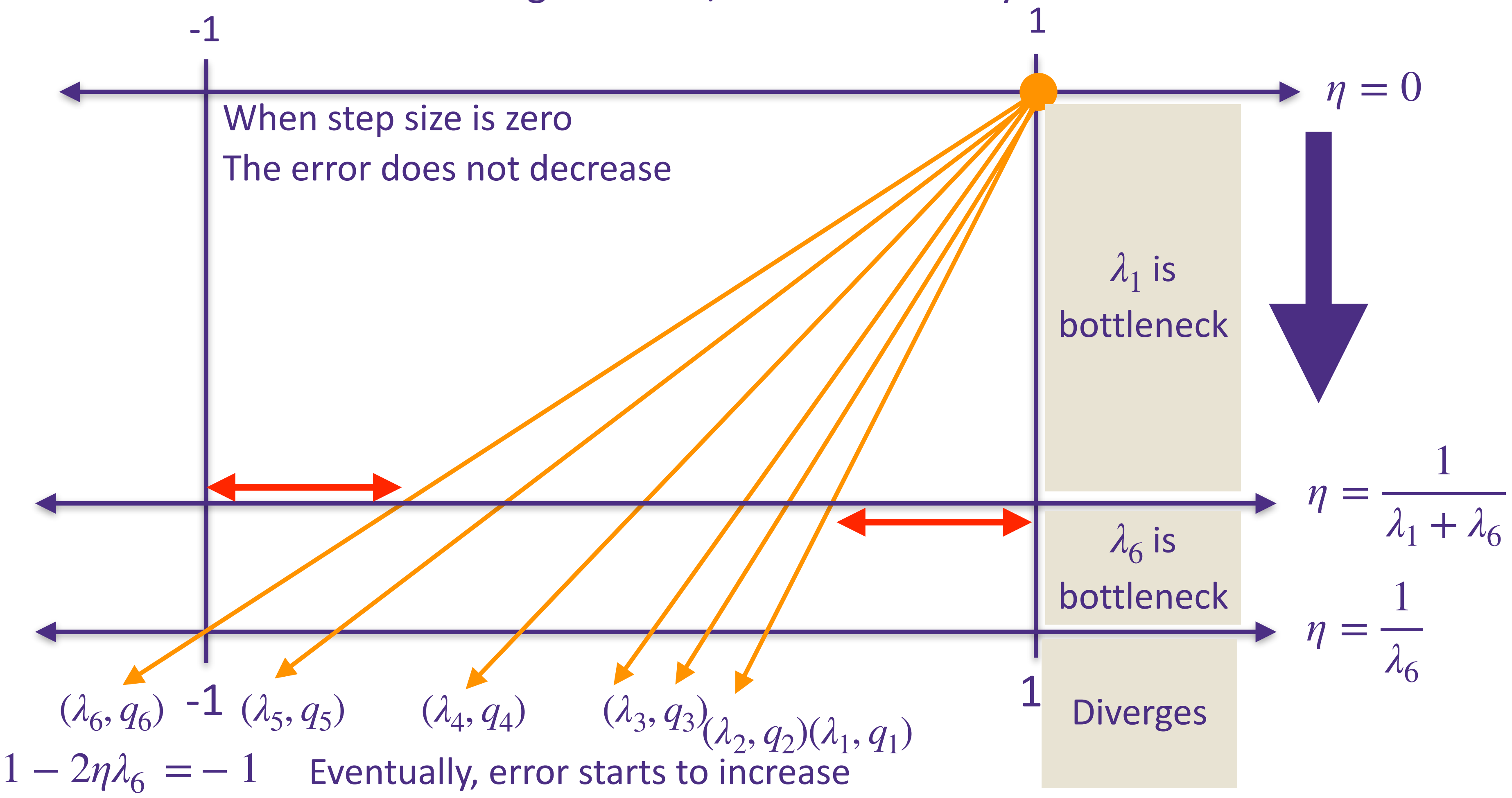
- I will not prove the facts that $\|q_i\|_2 = 1$ and $q_i^T q_j = 0$

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

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

Gradient descent for linear regression

- Claim: $\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) = 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})$, and $\lambda_i(\mathbf{X}^T\mathbf{X}) \geq 0$
- We plot $\{\ 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})\ \}$ for increasing step sizes, the largest $\left|1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})\right|$ is the bottleneck determining how fast/slow error decays



Gradient descent for logistic regression

- Now we know how to find the global minimum of a logistic regression problem, numerically

Loss function: Conditional Likelihood

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

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

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

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

What is known for Gradient descent

- $f(\cdot)$ is L-smooth if $\|\nabla f(w) - \nabla f(v)\|_2 \leq L\|w - v\|_2$ for all $w, v \in \mathbb{R}^d$
- $f(\cdot)$ is μ-strongly convex if $f(w) \geq f(v) + \nabla f(v)^T(w - v) + \frac{\mu}{2}\|w - v\|_2^2$
- 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

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

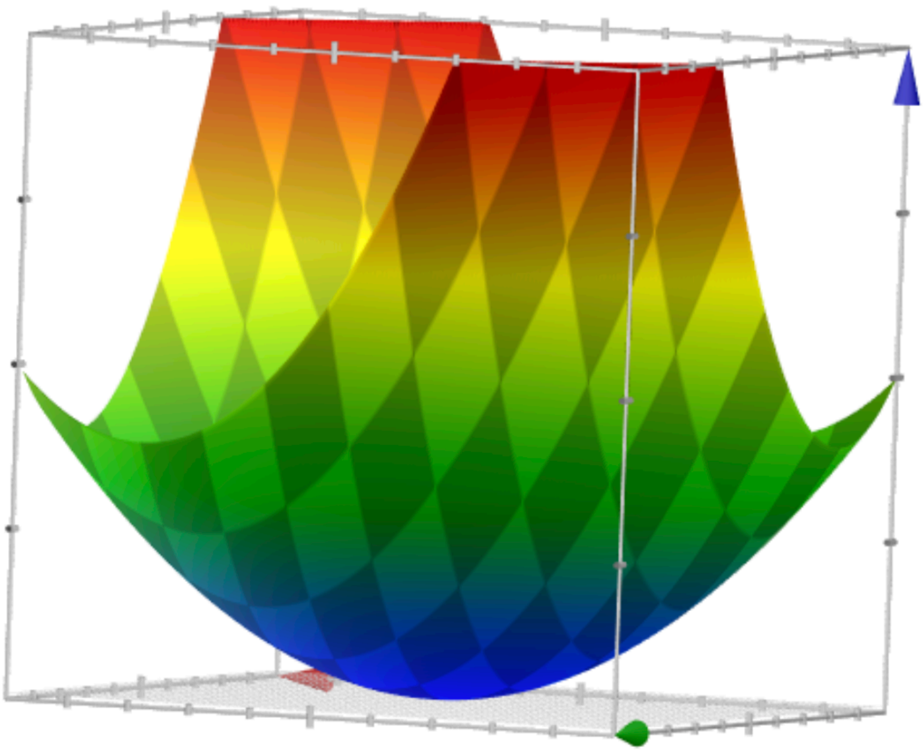
Questions?

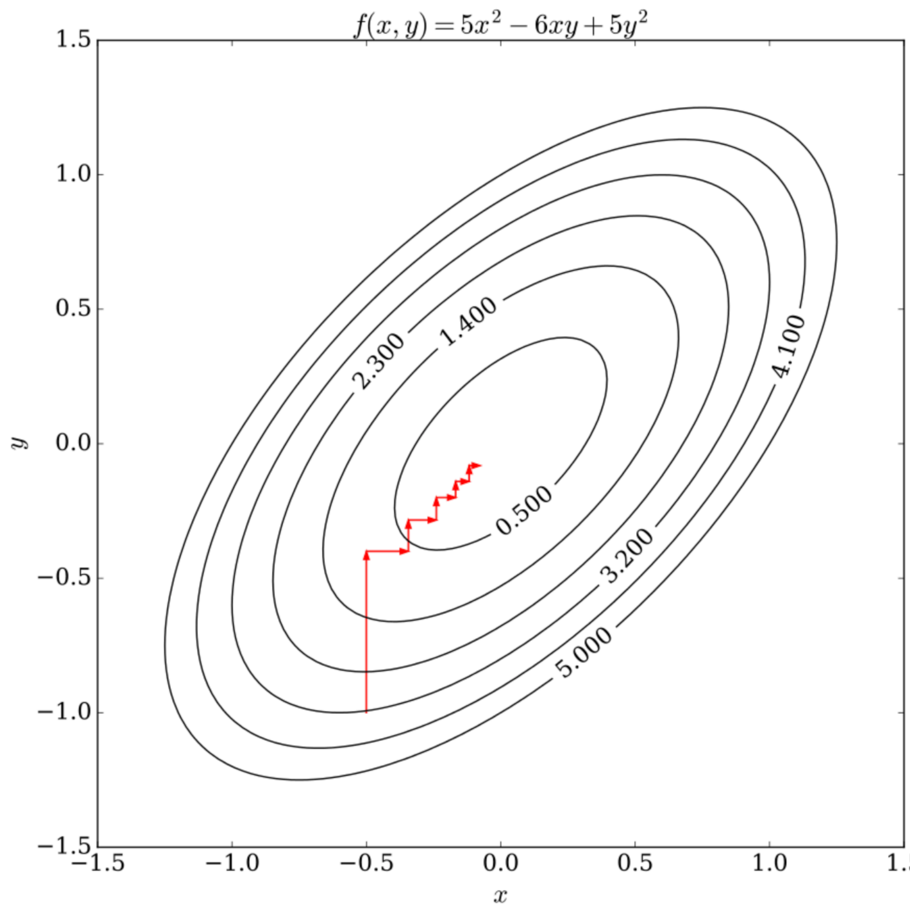
Lecture 13: Coordinate Descent

- How to solve Lasso

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

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

- 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

$j=1$ for example

$$\min_{w_1} \| Xw - y \|_2^2$$

$$= \left\| \begin{bmatrix} x_1 & X_{2:d} \end{bmatrix} \begin{bmatrix} w_1 \\ \hline w_{2:d} \end{bmatrix} - \begin{bmatrix} y \end{bmatrix} \right\|_2^2 \longrightarrow (aw_1 - b)^2 + \text{Const}$$

$$= \| x_1 \cdot \underset{\mathbb{R}}{\underset{\uparrow}{w_1}} - (y - X_{2:d} \cdot w_{2:d}) \|_2^2$$

$$= x_1^T x_1 \cdot w_1^2 - 2 x_1^T (y - X_{2:d} \cdot w_{2:d}) \cdot w_1 + \text{Const}$$

Define notation (a, b)

$$= (aw_1 - b)^2 + \text{Const}$$

$$a \triangleq \sqrt{x_1^T x_1}, \quad b \triangleq \frac{x_1^T (y - X_{2:d} w_{2:d})}{\sqrt{x_1^T x_1}}$$

$$w_1^* = \underset{w_1 \in \mathbb{R}}{\arg\min} \ (aw_1 - b)^2 + \text{Const} = \frac{b}{a} = \frac{x_1^T (y - X_{2:d} \cdot w_{2:d}^{(t)})}{x_1^T x_1} = w_1^{(t+1)}$$

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

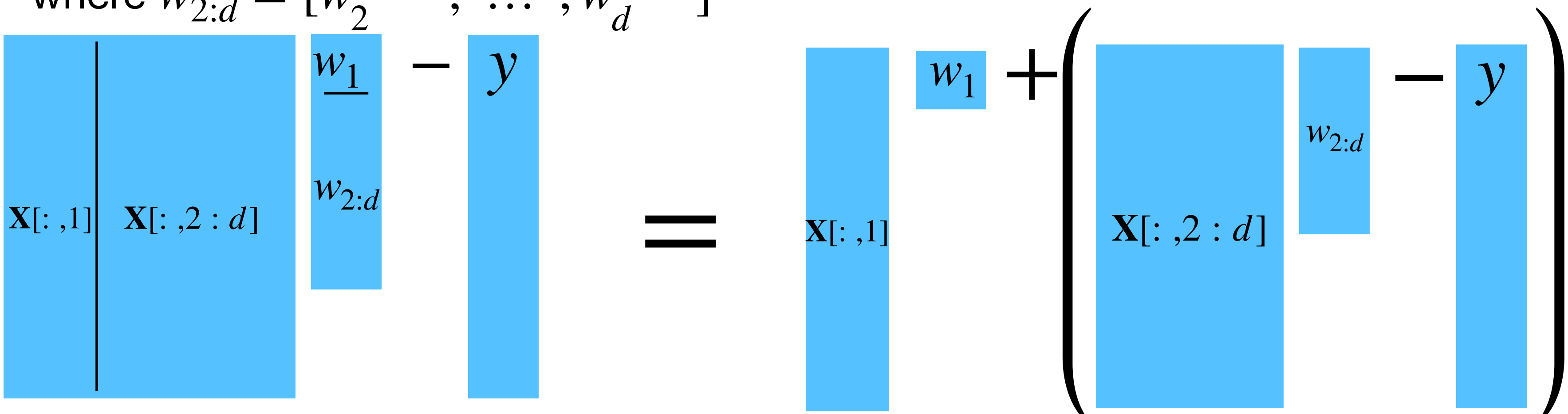
$$w_1^{(t)} \leftarrow \arg\min_{w_1} \|\mathbf{X}w - \mathbf{y}\|_2^2$$

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

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

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

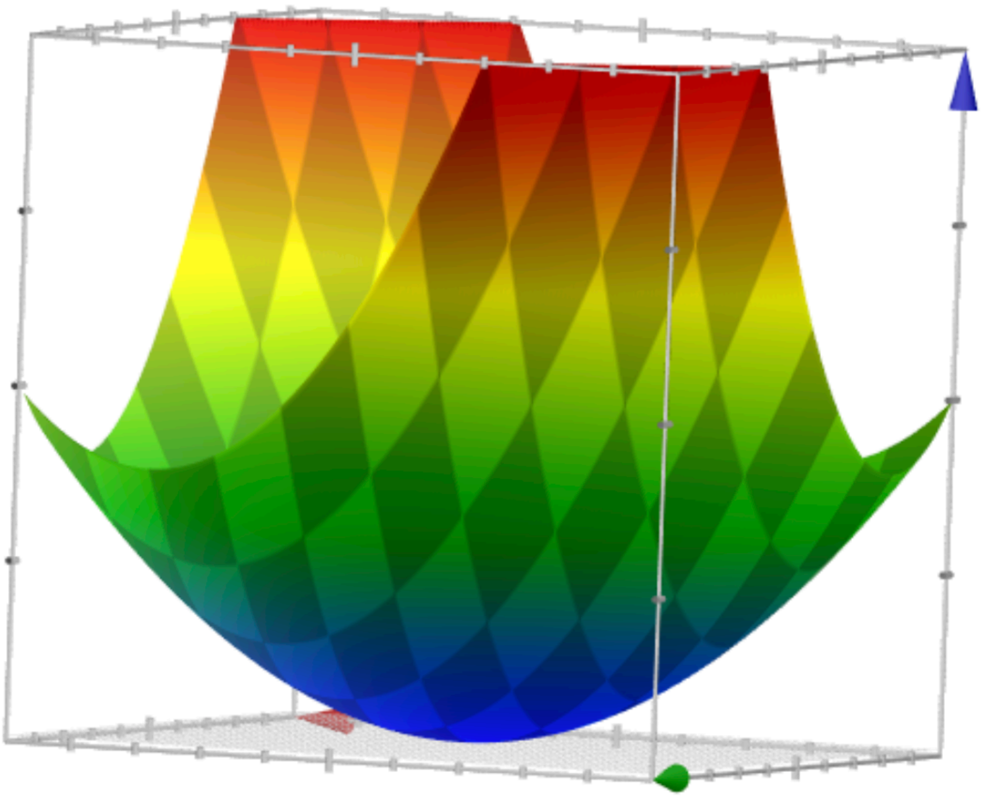
where $w_{2:d} = [w_2^{(t-1)},\ \dots\ , w_d^{(t-1)}]^T$

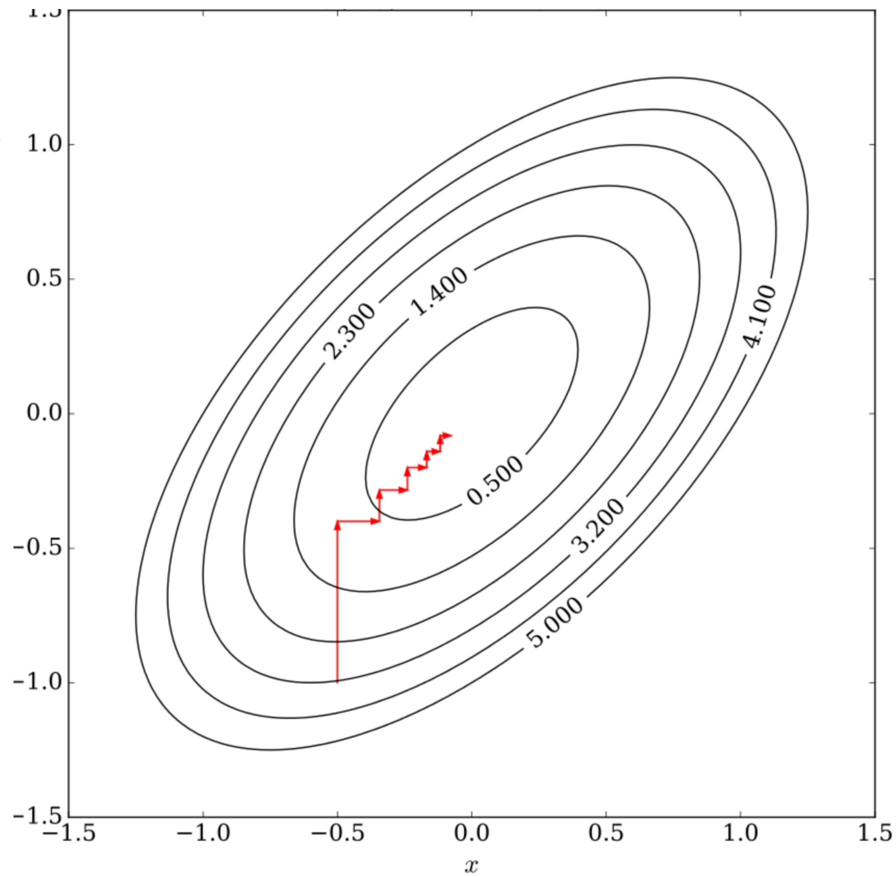


- we know from linear least squares that the minimizer is

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

- 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

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

  - hence, the coordinate descent update boils down to

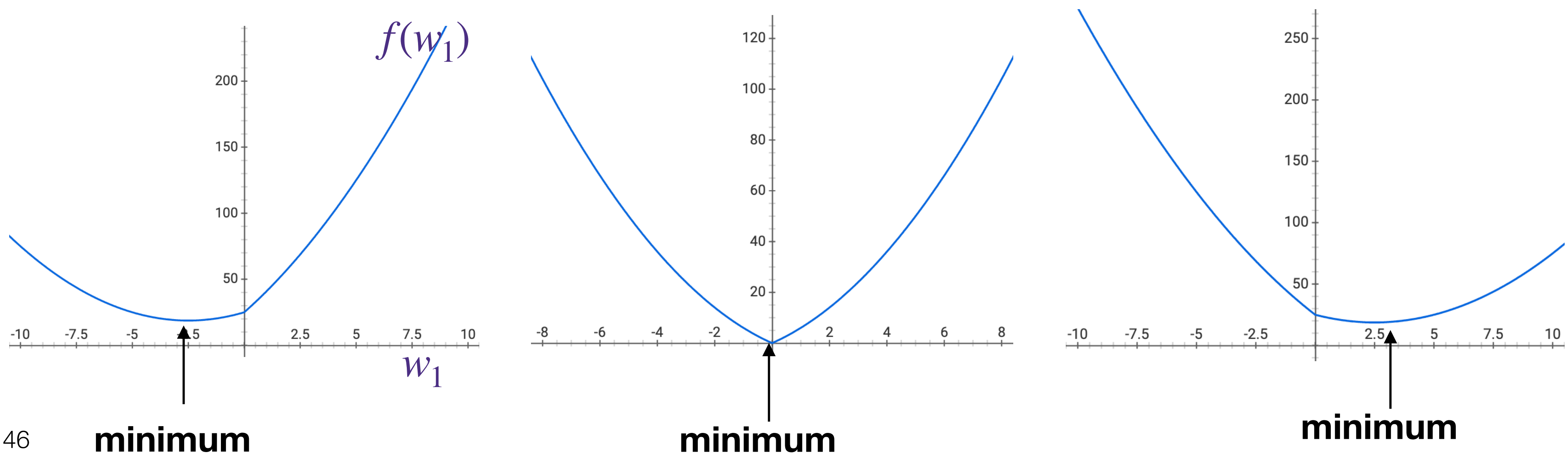
$$w_1^{(t)} \leftarrow \arg\min_{w_1} \underbrace{\left\| \mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-2:d}\right) \right\|_2^2 + \lambda |w_1|}_{f(w_1)}$$

Convexity

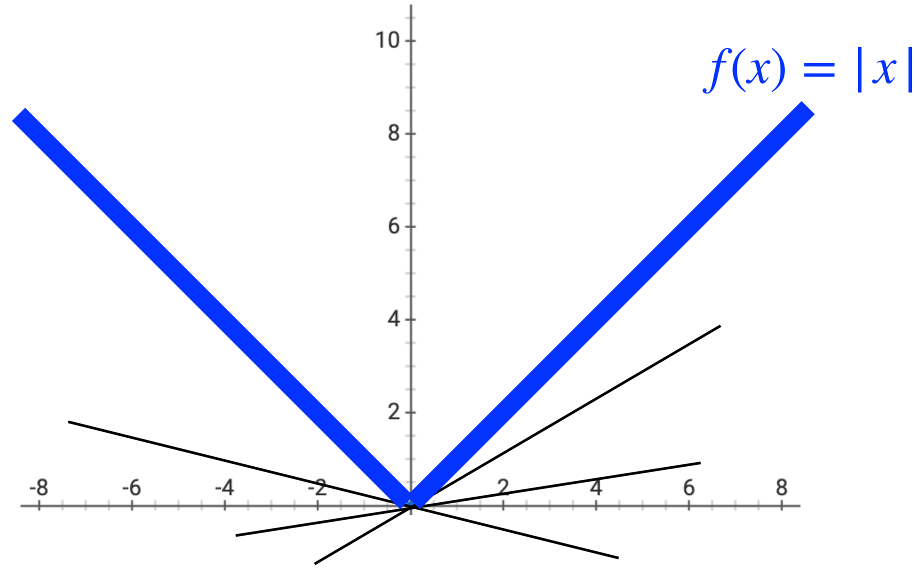
- to find the minimizer of $f(w_1)$, let's study some properties
- for simplicity, we represent the objective function as

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

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



Convexity



- 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

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

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

Computing the sub-gradient

$$a = \sqrt{X_1^T X_1}$$

$$b = \frac{X_1^T (y - X_{2:d} w_{2:d}^{(t)})}{\sqrt{X_1^T X_1}}$$

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

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

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

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

$$[-2ab - \lambda,\ -2ab + \lambda]$$

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

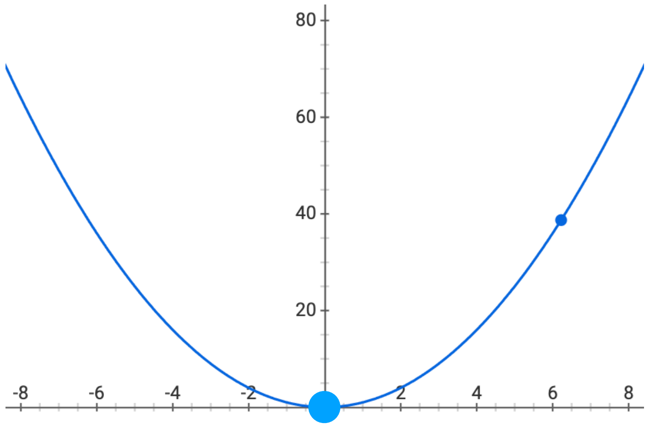
- $f(w_1)$ is non-differentiable, and its sub-gradient is

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

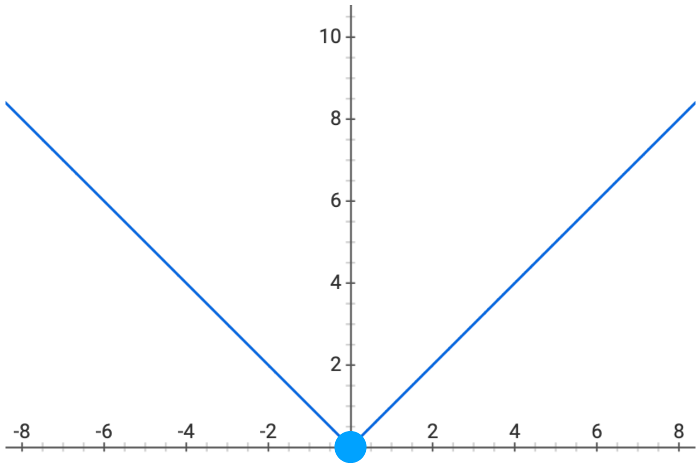
$$= \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

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 for $(aw_1 - b)^2 + \lambda |w_1|$

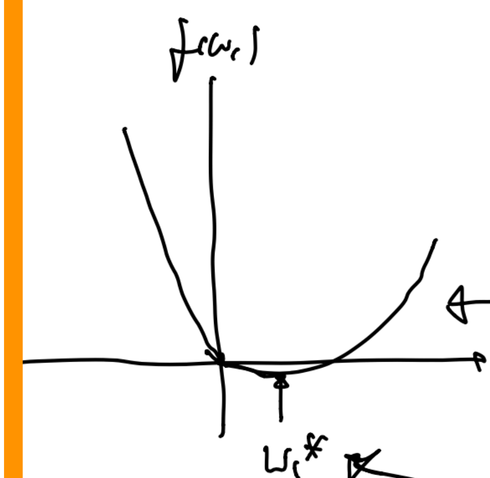
- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

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

Case 1. $w_1^* > 0. \longrightarrow 2a(aw_1^* - b) + \lambda = 0$

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

$\therefore$ If $2ab > \lambda$, then $w_1^* = \frac{2ab - \lambda}{2a^2}$



Computing the sub-gradient for $(aw_1 - b)^2 + \lambda |w_1|$

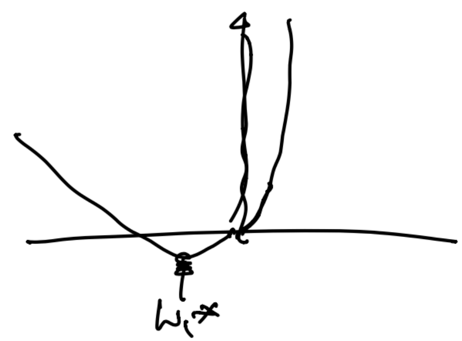
- the minimizer $w_1^{(t)}$ is when zero is included in the sub-gradient

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

Case 2: $w_1^* < 0$, $2a(aw_1^* - b) - \lambda = 0$

$$w_1^* = \frac{2ab+\lambda}{2a^2} < 0$$

$\therefore$ If $2ab < -\lambda$, $w_1^* = \frac{2ab+\lambda}{2a^2}$



Computing the sub-gradient for $(aw_1 - b)^2 + \lambda |w_1|$

- 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 3: $w_1^* = 0$,

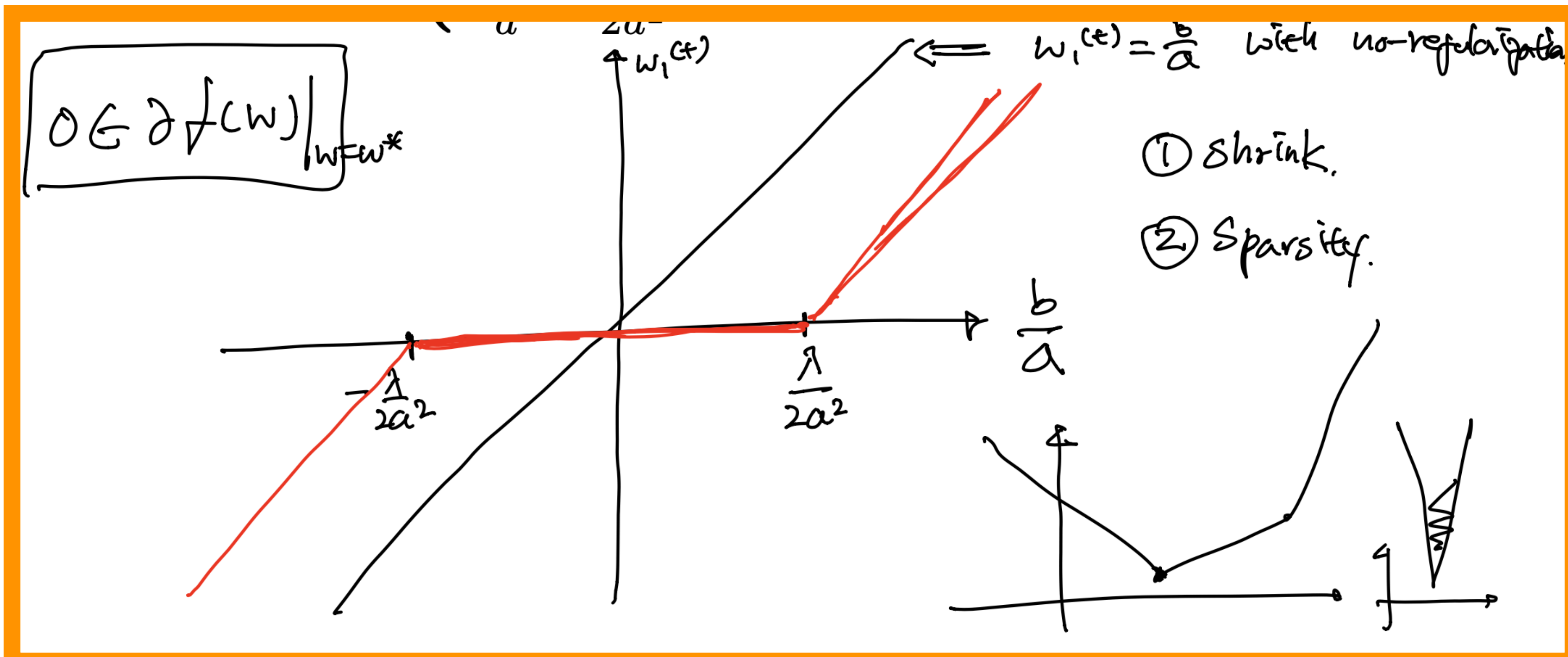
$-2ab - \lambda \leq 0 \leq -2ab + \lambda$

$-\lambda \leq 2ab \leq \lambda. \longrightarrow w_1^* = 0.$

Computing the sub-gradient for $(aw_1 - b)^2 + \lambda|w_1|$

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

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



How do we find the minimizer?

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

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

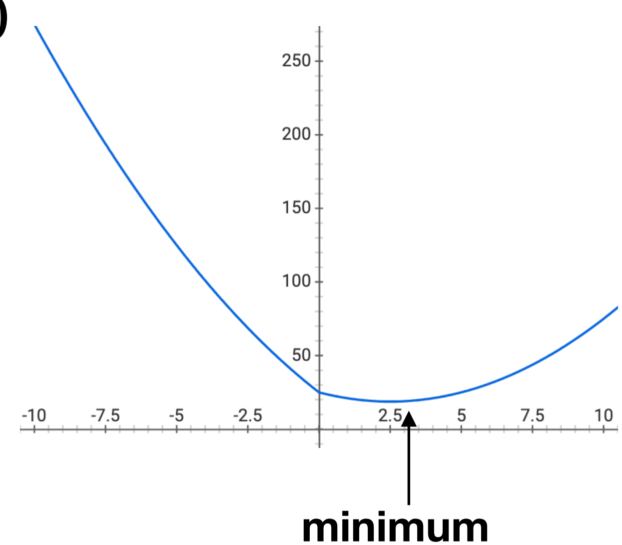
- case 1:
  - $2a(aw_1 - b) + \lambda = 0$ for some $w_1 > 0$
  - this happens when

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

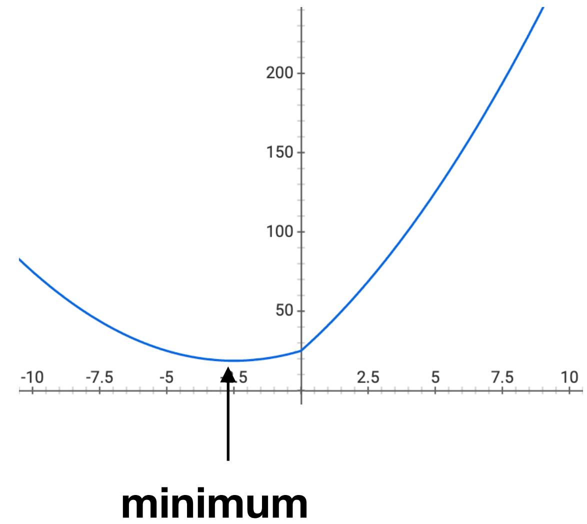
  - hence,

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

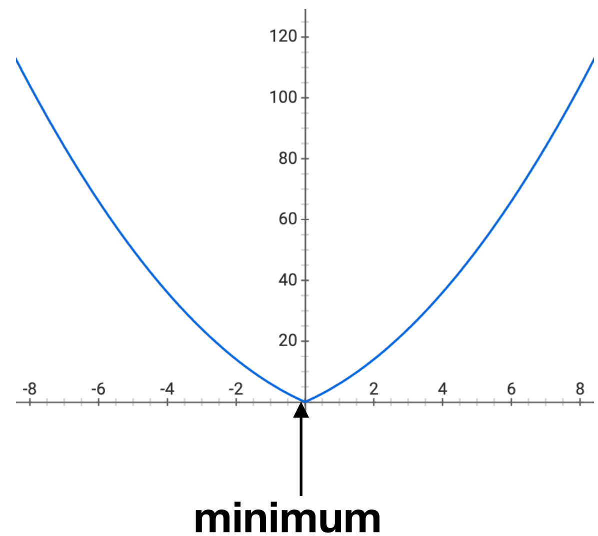
if $\lambda < 2ab$



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

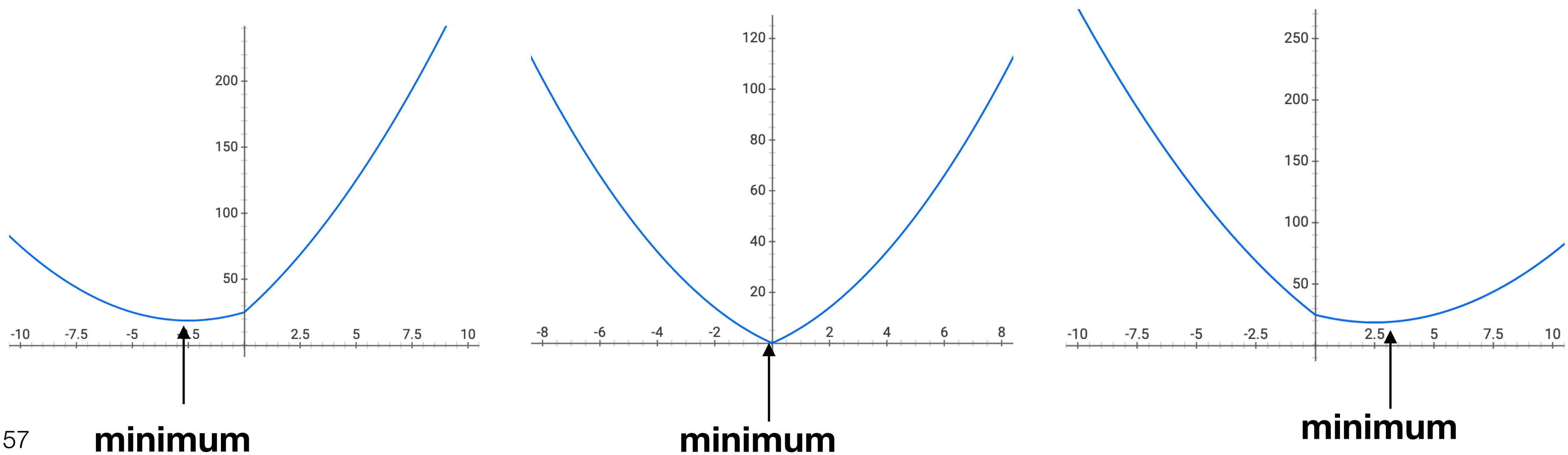


Coordinate descent on Lasso

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

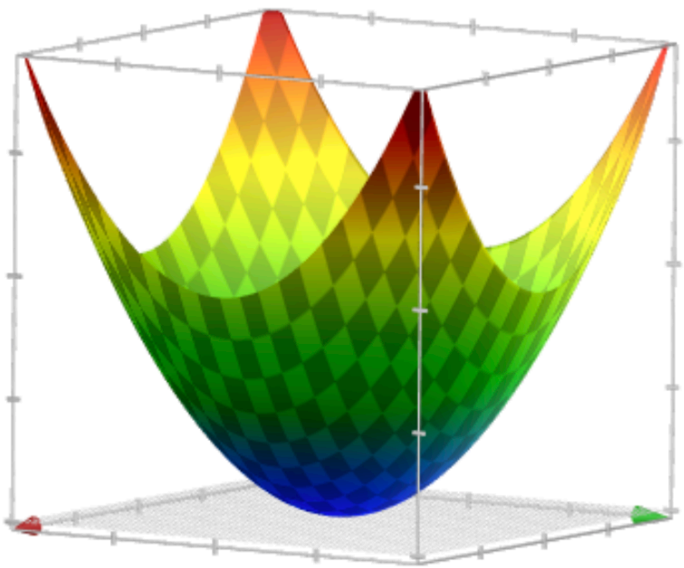
$$w_1^{(t)} \leftarrow \begin{cases} \frac{b}{a} - \frac{\lambda}{2a^2} & \text{for } 2ab > \lambda \\ 0 & \text{for } -\lambda \leq 2ab \leq \lambda \\ \frac{b}{a} + \frac{\lambda}{2a^2} & \text{for } \lambda < -2ab \end{cases}$$

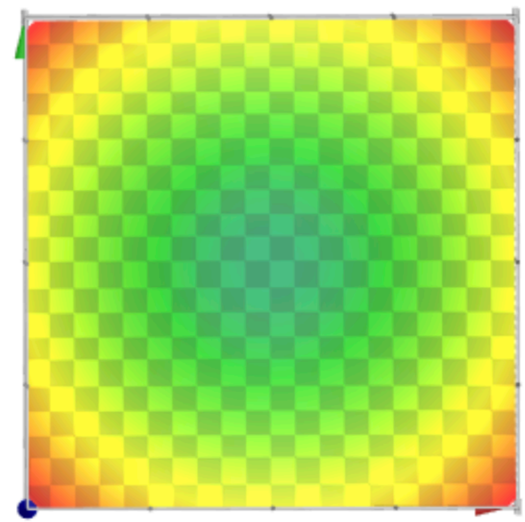
- where $a = \sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}$, and $b = \dfrac{\mathbf{X}[:,1]^T \big(\mathbf{y} - \mathbf{X}[:,2:d] w_{-1}\big)}{\sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}}$



When does coordinate descent work?

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

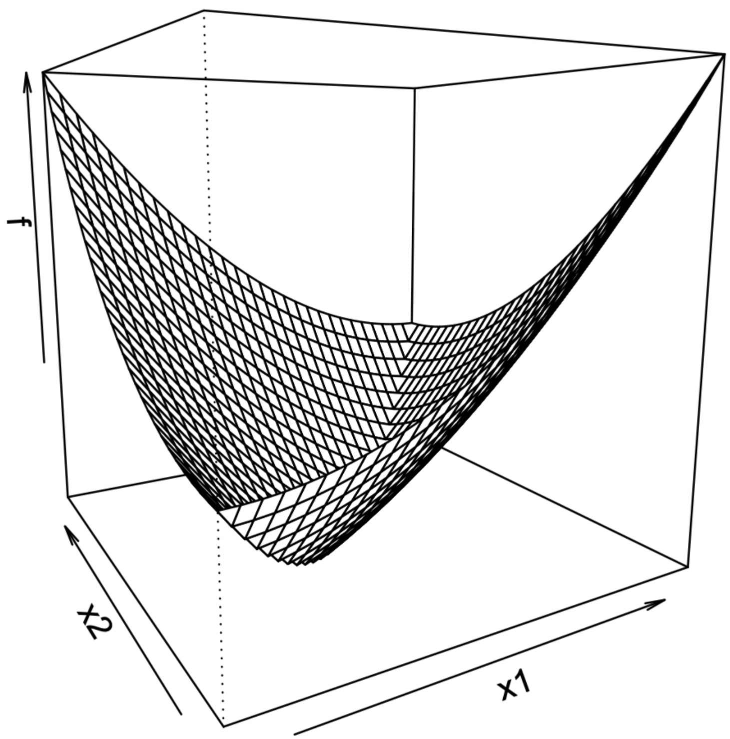


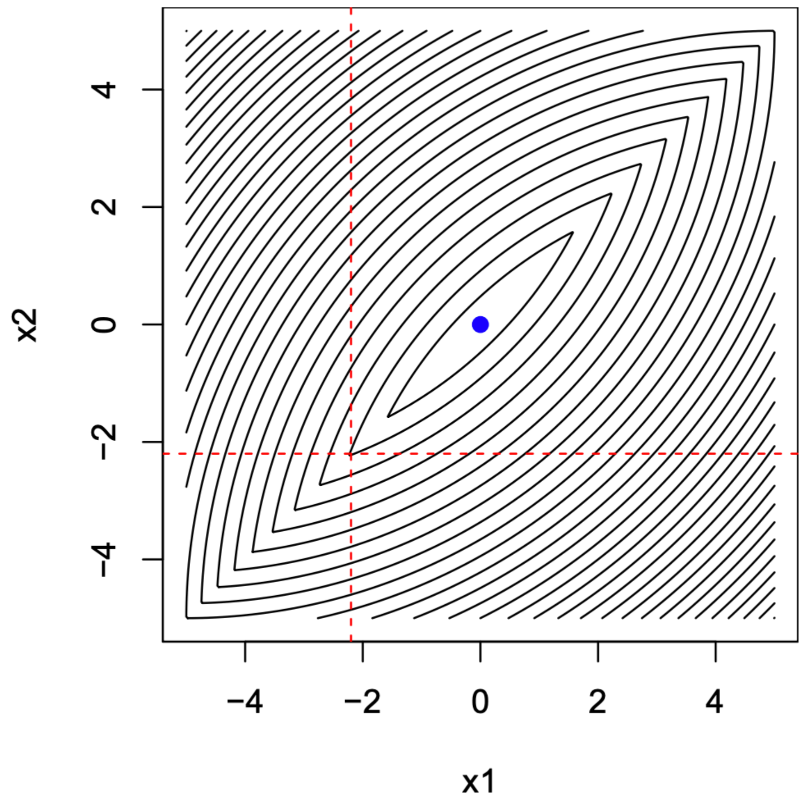


- when coordinate descent has stopped, that means $\frac{\partial f(x)}{\partial x_j} = 0$ for all $j \in \{1, \ldots, d\}$
- this implies that the gradient $\nabla_x f(x) = 0$, which happens only at minimum

When does coordinate descent work?

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



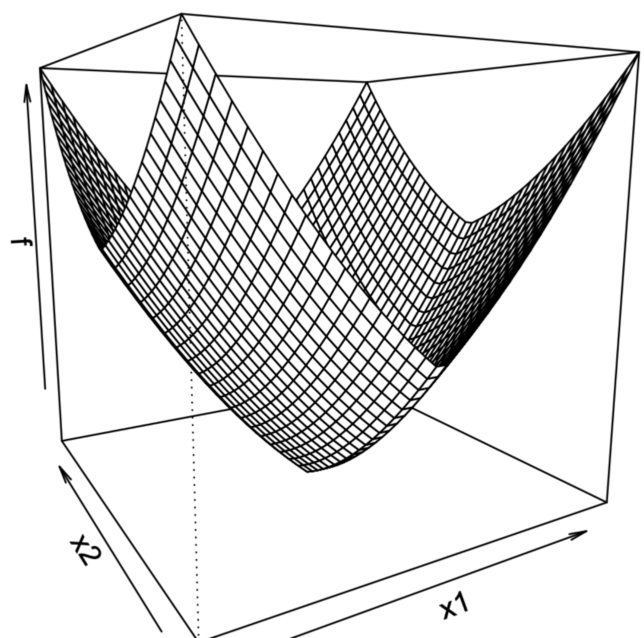


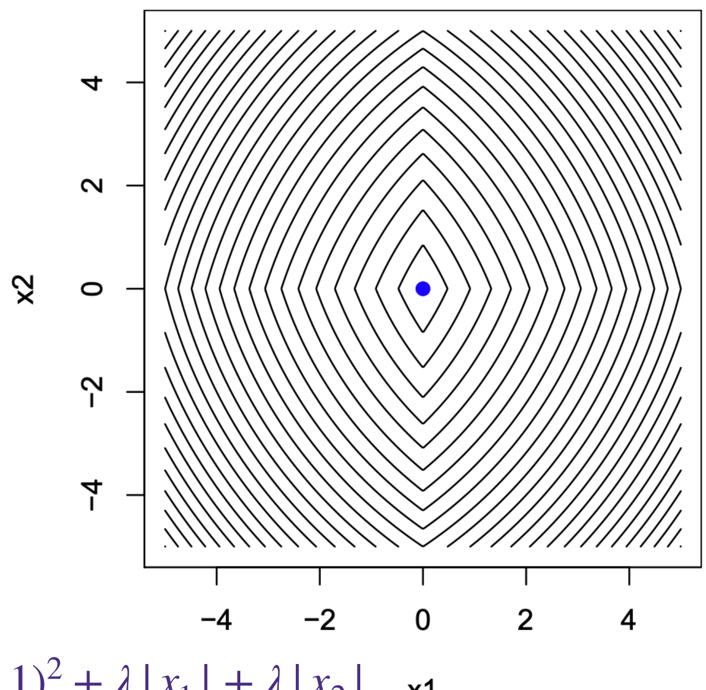
-

$$f(x_1, x_2) = (3x_1 + 4x_2 + 1)^2 + \lambda |x_1 - x_2|$$

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
-





$f(x_1, x_2) = (3x_1 + 4x_2 + 1)^2 + \lambda |x_1| + \lambda |x_2|$

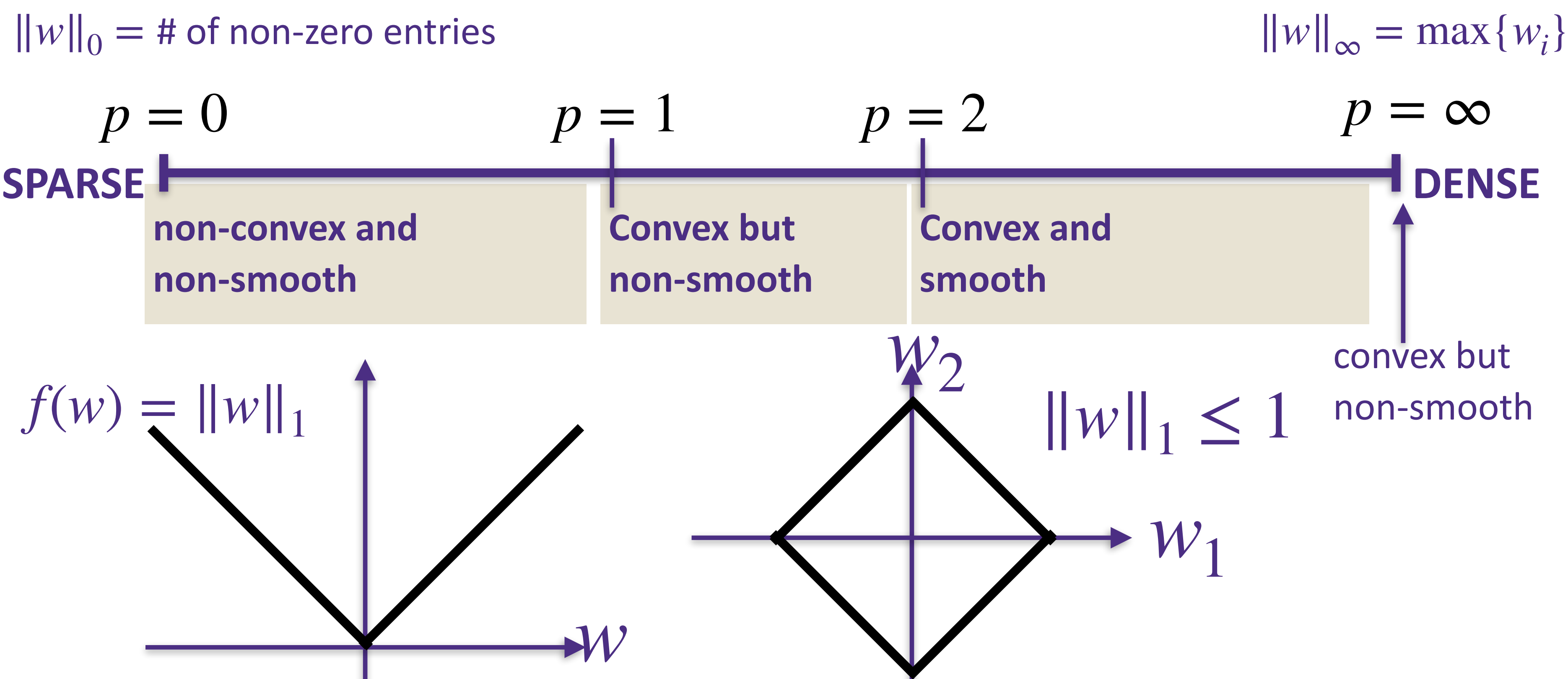
Questions?

Lecture 14:
Stochastic Gradient Descent

-What do we use in practice?

Sparsity/Complexity tradeoff

- ℓ_p-norm of a vector is defined as $\|w\|_p \triangleq \left(w_1^p + w_2^p + \cdots + w_d^p\right)^{1/p}$
- Consider regularized least squares problem of minimizing
  $$\mathscr{L}(w) = \sum_{i=1}^{n} (y_i - w^T x_i)^2 \; + \; \lambda \, \|w\|_p^p$$
- This is ridge regression for $p = 2$ and Lasso for $p = 1$



Machine Learning Problems

- Given data: $\{(x_i, y_i)\}_{i=1}^n$ $\quad x_i \in \mathbb{R}^d \quad y_i \in \mathbb{R}$
- Learning a model's parameters: $\frac{1}{n}\sum_{i=1}^n \ell_i(w) = \frac{1}{n}\sum_{i=1}^n \ell(f_w(x_i), y_i)$

- Gradient Descent (GD):

  one update takes cdn operations/time for some constant c>0

$$w_{t+1} \leftarrow w_t - \eta \frac{1}{n} \sum_{i=1}^n \nabla \ell_i(w_t)$$

- Stochastic Gradient Descent (SGD): one update takes cd operations/time

$$w_{t+1} \leftarrow w_t - \eta \nabla \ell_{I_t}(w_t)$$

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

- SGD is an unbiased estimate of the GD

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

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 $\{1, \ldots, n\}$ 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 $\|w_0 - w_*\|_2^2 \leq R$ and $\sup_w \max_i \|\nabla \ell_i(w)\|_2^2 \leq G$ then

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

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

(In practice use last iterate)

Convergence rate: $O\left(\frac{1}{\sqrt{T}}\right)$

(Fixed optimal step size)

$$-\frac{R}{2T\eta^2} + \frac{G}{2} = 0$$

We want to show that

$$\mathbb{E}\left[\ell\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - \ell(w_*)\right] \leq \mathbb{E}\left[\frac{1}{T}\sum_{i=1}^{T} \ell(w_t) - \ell(w_*)\right]$$

Follows from convexity of $\ell(\cdot)$ and Jensen's inequality (3 slides later)

$$\leq \frac{1}{T}\sum_{i=1}^{T} \mathbb{E}\left[\ell(w_t) - \ell(w_*)\right]$$

Follows from linearity of expectation

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

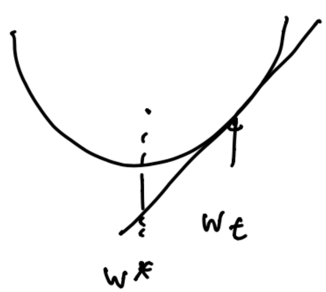
We are left to show this

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

$$= \underbrace{\mathbb{E}[||w_t - w_*||^2]} + \eta^2 \underbrace{\mathbb{E}[||\nabla \ell_{I_t}(w_t)||^2]}_{\leq G} \boxed{- 2\eta\, \mathbb{E}[(w_t - w_*)^T \nabla \ell_{I_t}(w_t)]}$$

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

$*$ Convexity $\ell_i(w)$: $\ell(w^*) \geq \ell(w_t) + \nabla \ell_i(w_t)^T (w_* - w_t)$



$$\rightarrow \ell(w^*) - \ell(w_t) \geq \boxed{-\nabla \ell_i(w_t)^T (w_t - w_*)}$$

Stochastic Gradient Descent

Proof

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

$$\leq \mathbb{E}[\|w_t - w_*\|^2] + \eta^2 G + 2\eta\, \mathbb{E}[\ell_{I_t}(w^*) - \ell_{I_t}(w_t))$$

$$\mathbb{E}[\ell_{I_t}(w_t) - \ell_{I_t}(w^*)] \leq \frac{\mathbb{E}[\|w_t - w_*\|^2] - \mathbb{E}[\|w_{t+1} - w_*\|^2] + \eta^2 G}{2\eta}$$

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\ell_{I_t}(w_t) - \ell_{I_t}(w^*)] \leq \frac{1}{2\eta T}\left\{\mathbb{E}[\|w_0 - w_*\|^2] - \mathbb{E}[\|w_T - w_*\|^2]\right\} + \cancel{T}\eta^2 G$$

Stochastic Gradient Descent

Proof

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

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

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

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

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

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

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

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

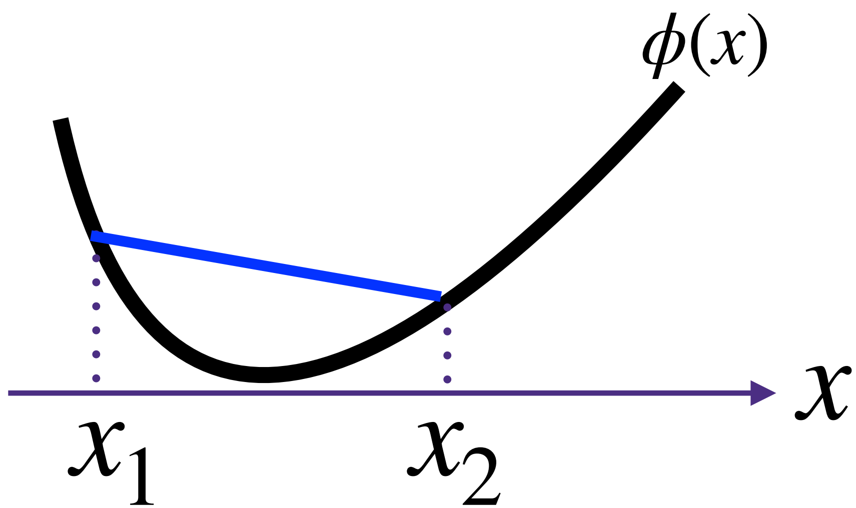
Stochastic Gradient Descent

Proof

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

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

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



Stochastic Gradient Descent

Proof

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

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

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

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

$$\eta = \sqrt{\frac{R}{GT}}$$

Mini-batch SGD

- Instead of one iterate, average B stochastic gradient together

- Advantages:
  - Smaller variance: the variance of the stochastic gradient is smaller by a factor of $1/\sqrt{B}$
  - Parallelization: each gradient in the mini-batch can be computed in parallel

- If you have regularizer, $\frac{1}{n}\sum_{i=1}^{n} \ell_i(w) + r(w)$, then update with the stochastic gradient of the loss and gradient of the regularizer

Questions?