# Non-quadratic Regularizers

Sewoong Oh

CSE446
University of Washington

# L1 Regularizer

- **sum absolute** or **L1 regularizer** uses

$$r(w) = |w_1| + |w_2| + \cdots + |w_d|$$

- this is the same as **L1 norm** of the weight vector
  (we write is as $w_{1:d}$ to emphasize that $w_0$ is the weight of the constant term that should not be regularized)

$$\|w_{1:d}\|_1 \triangleq |w_1| + |w_2| + \cdots + |w_d|$$

- we use empirical risk $\mathscr{L}(w) = \sum_{i=1}^{n} (w^T x_i - y_i)^2$
- with L1 regularizer, it is called **Lasso regression**

$$\text{minimize } \mathscr{L}(w) + \lambda \|w\|_1$$

- since it is a convex function, can be efficiently minimized using optimization (but unlike ridge regression, does not have a closed-form solution)

- it has interesting properties, making it attractive in practice (**sparsification**)

# Sparse coefficient vector

- suppose w is sparse, i.e. many of its entries are zero

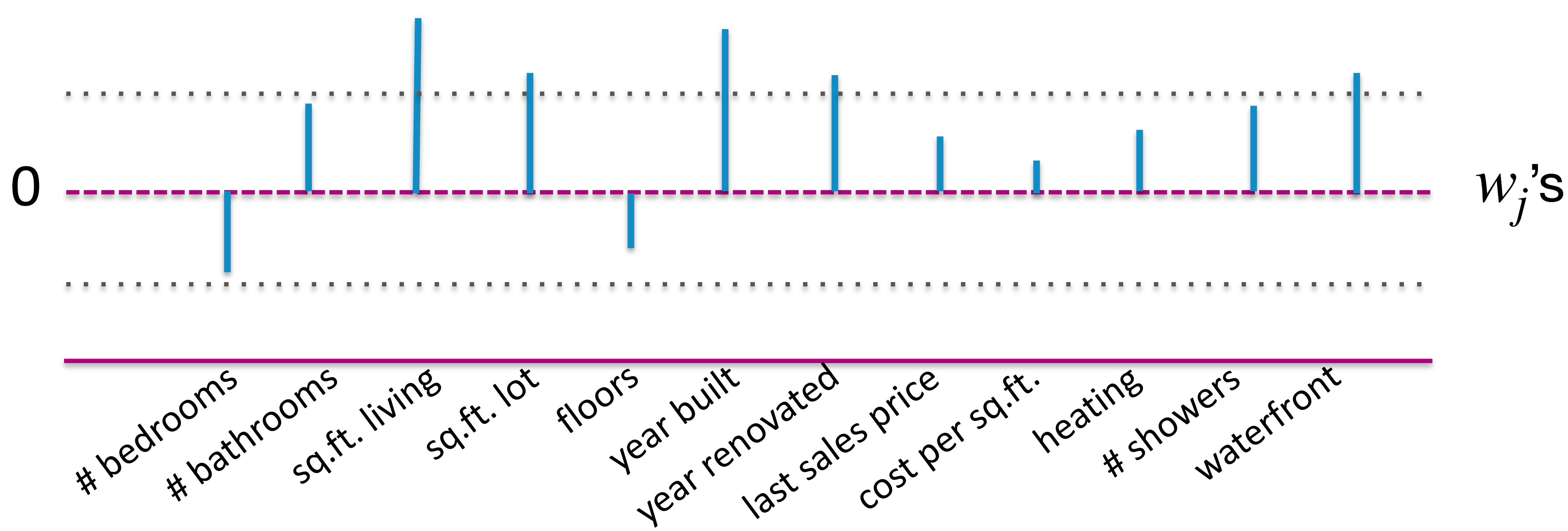
- prediction $\hat{y} = w^T x$ does not depend on features of $x = (x[1], \ldots, x[d])$ for which $w_j = 0$

- this means we select **some** features to use (i.e. those with $w_j \neq 0$)

- (potential) practical benefits of **sparse** w
  - true model might be sparse in real applications
    - e.g. polynomial fit
  - sparsity (i.e. the number of features used in prediction) is the simplest measure of complexity of a model
    - sparse models are natural choice of simple models
  - makes prediction model **simpler to interpret**
    - e.g. medical diagnosis
  - makes prediction faster (less computation)
  - but, manually engineering correct sparse set of features is extremely challenging

## Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

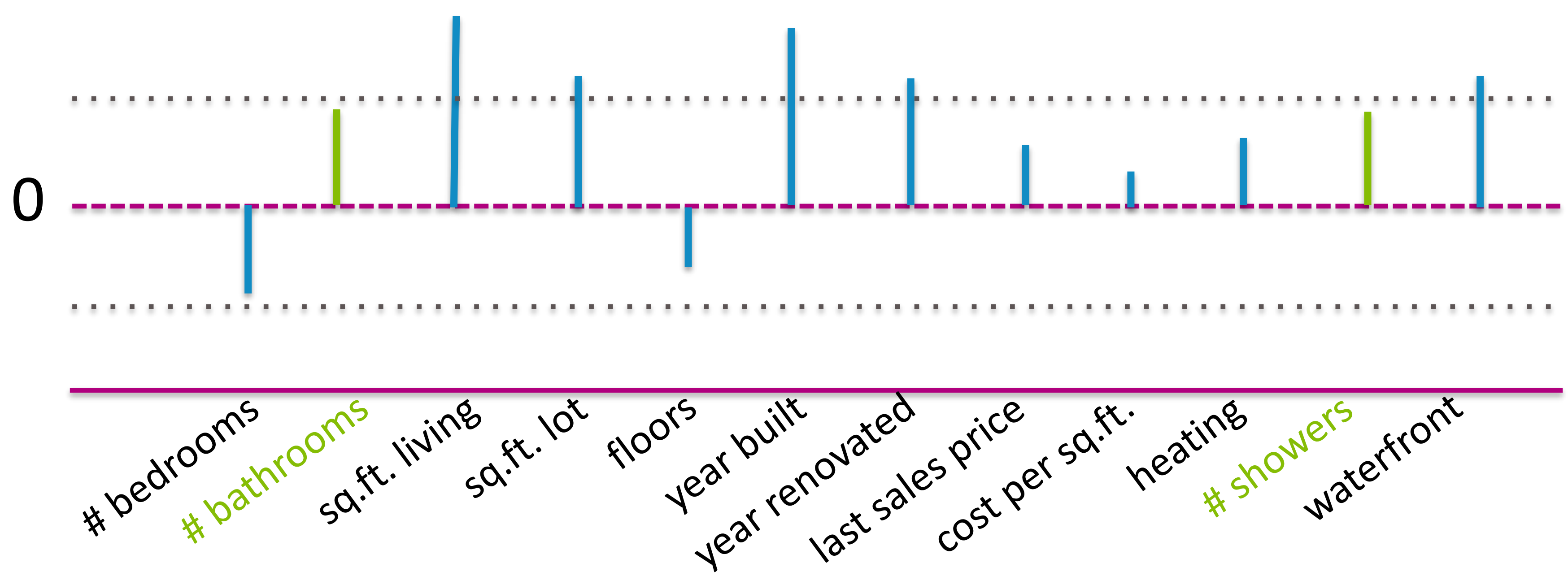
- sometimes sparse features are desired in practice
- consider running the following sparse feature selection method
  - run Ridge regression, with optimal lambda
  - Set to zero (shrink) those parameters that are smaller than a threshold




- Set threshold in order to keep the top 5, for example, parameters
- What is wrong with this approach?

## Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

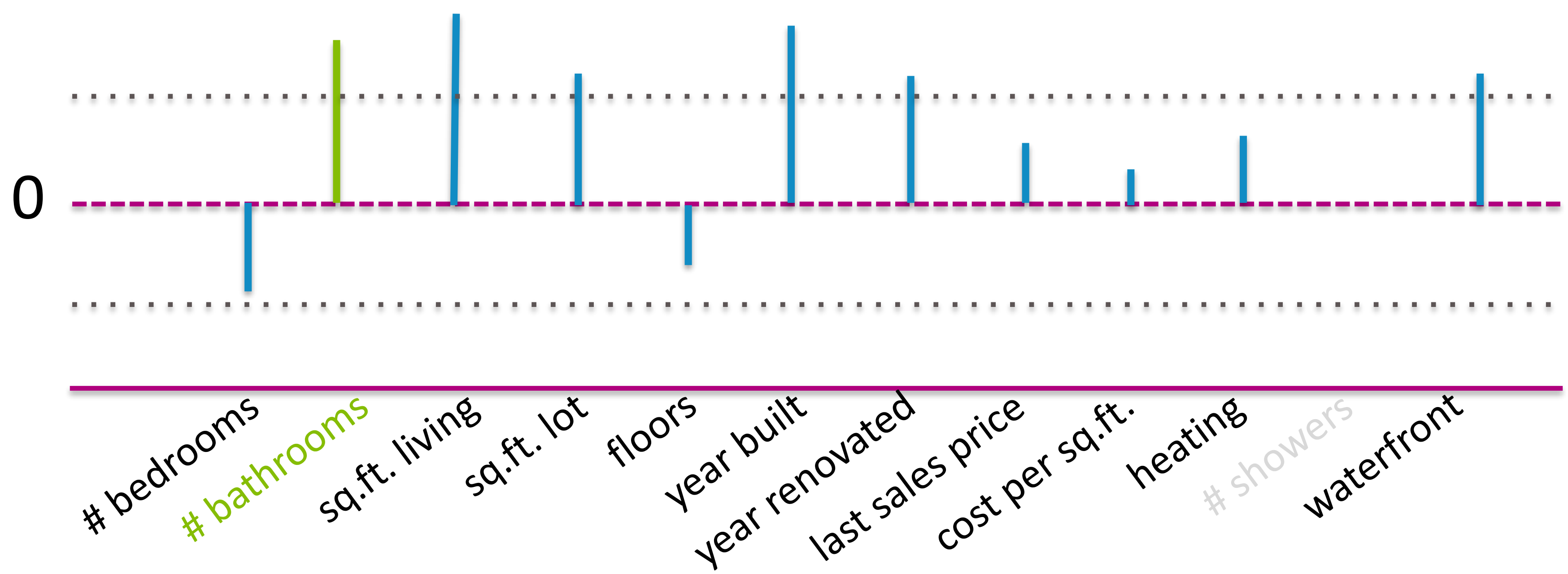
- sometimes sparse features are desired in practice
- consider running the following sparse feature selection method
  - run Ridge regression, with optimal lambda
  - shrink parameters that are smaller than a threshold




  - nothing measuring bathrooms is included!!

## Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

- If only one of the features were included when running Ridge regression, it would have survived



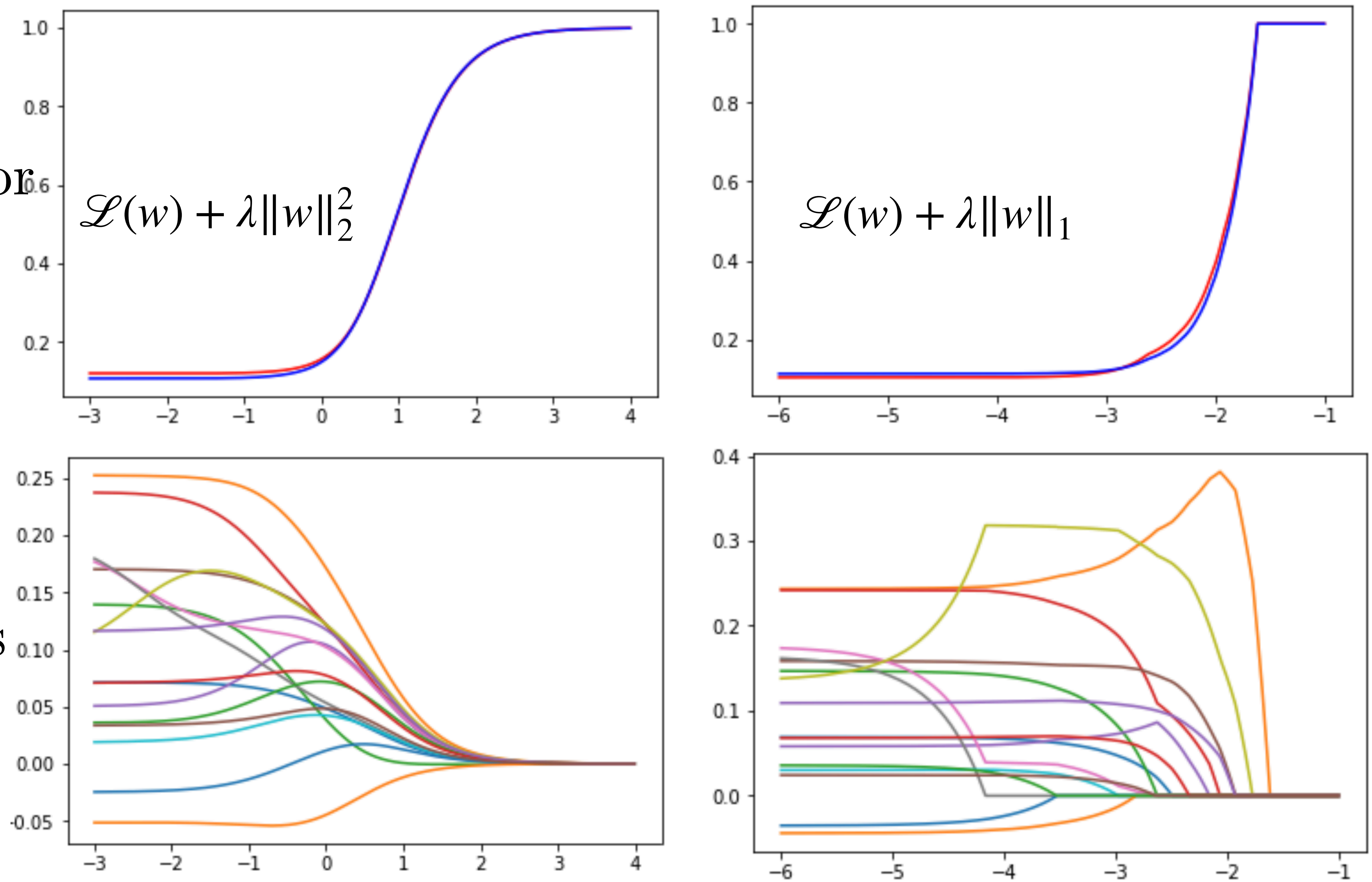

- thresholding Ridge regression parameters unnecessarily penalizes multiple similar features
- Lasso is a more principled way of selecting sparse features

# Example: house price with 16 features

test error is red and train error is blue

error

$w_i$'s



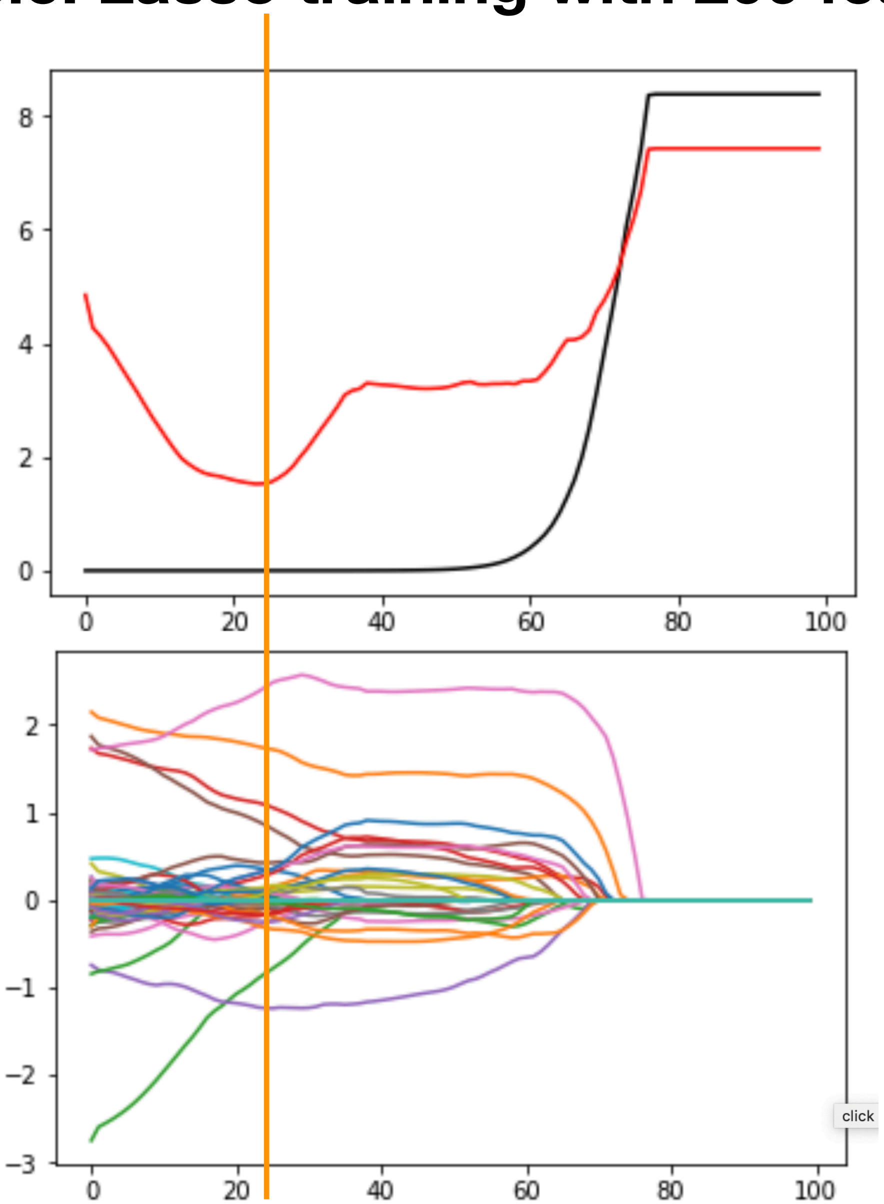

$\log_{10} \lambda$     $\log_{10} \lambda$

Ridge regression     Lasso regression

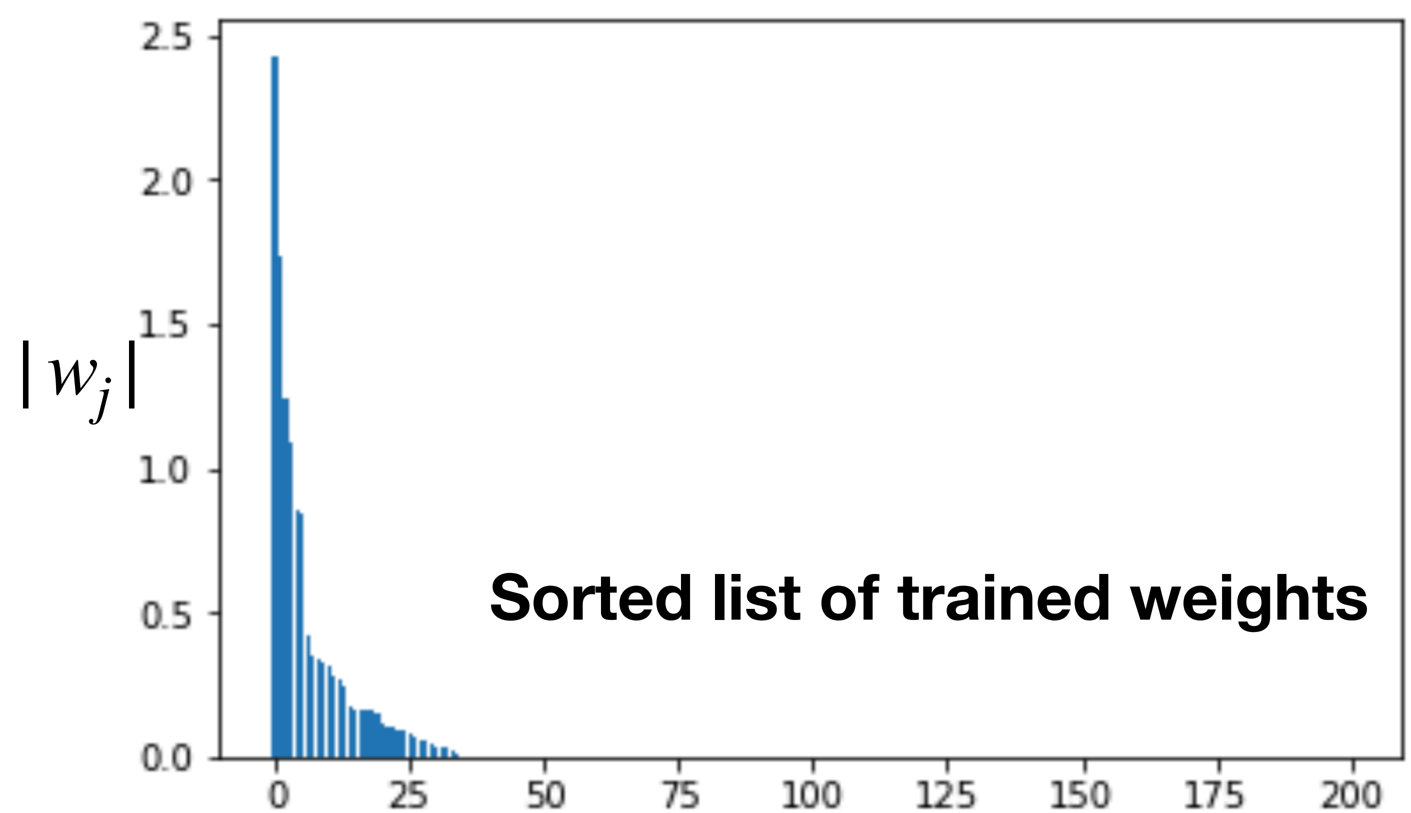
# Lasso regression naturally gives sparse features

- **feature selection** with Lasso regression
    1. choose which features to keep based on cross validation error
    2. keep only those features with non-zero parameters in $w$ at optimal $\lambda$
    3. **retrain** with the sparse model and $\lambda = 0$

**Example: Lasso training with 200 features**




- Lasso has only 35 non-zero components

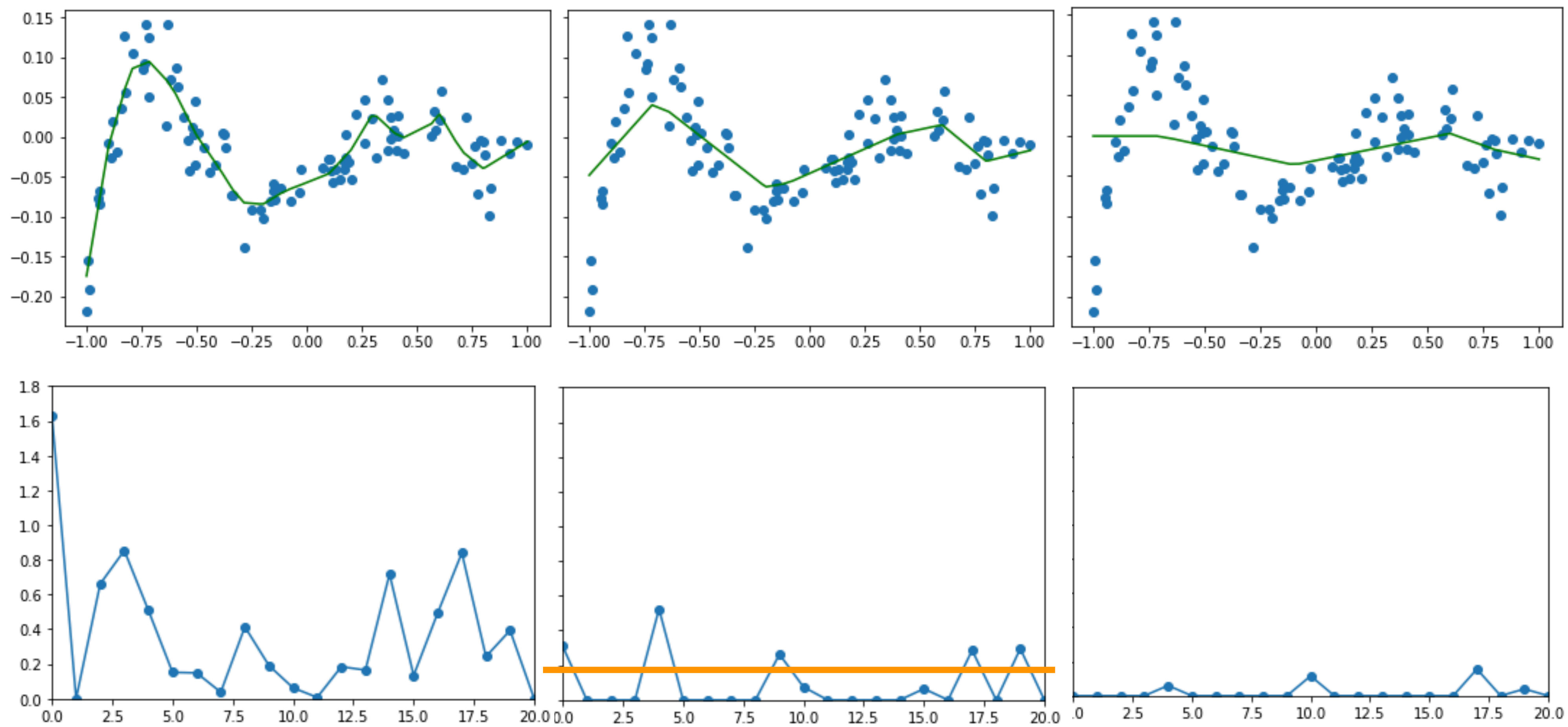


# Example: piecewise-linear fit

- We use Lasso on the piece-wise linear example

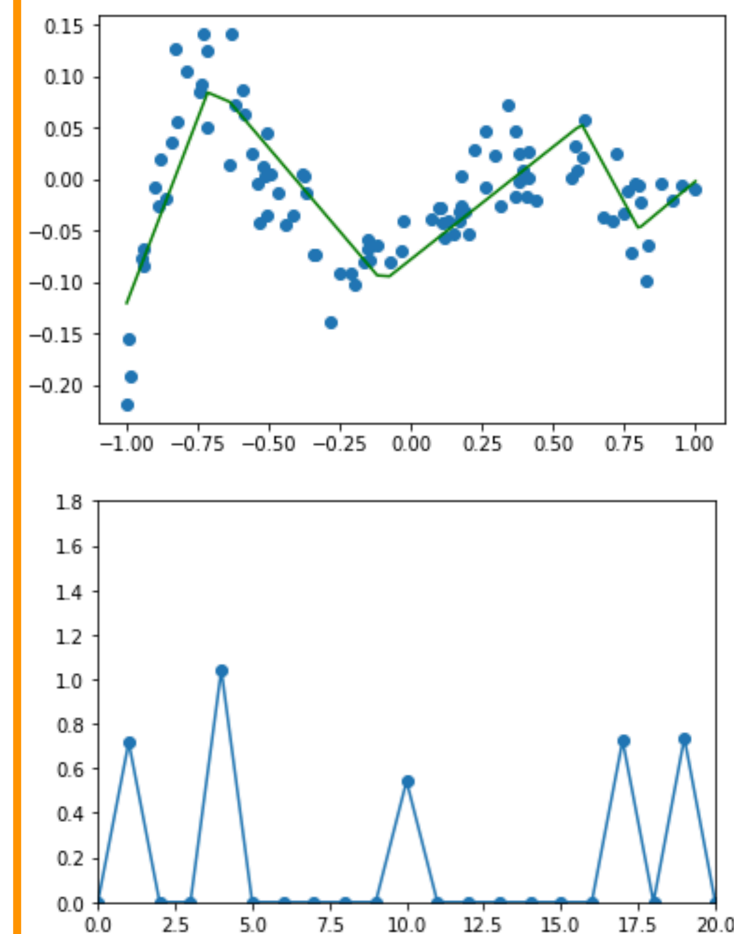
$$
\begin{aligned}
h_0(x) &= 1 \\
h_i(x) &= [x + 1.1 - 0.1i]^+
\end{aligned}
$$

$$\text{minimize}_w \ \mathscr{L}(w) + \lambda \|w\|_1$$




$\lambda = 10^{-8}$ $\lambda = 10^{-4}$ $\lambda = 2 \times 10^{-4}$

$$\text{minimize}_w \ \mathscr{L}(w)$$




$\lambda = 0$

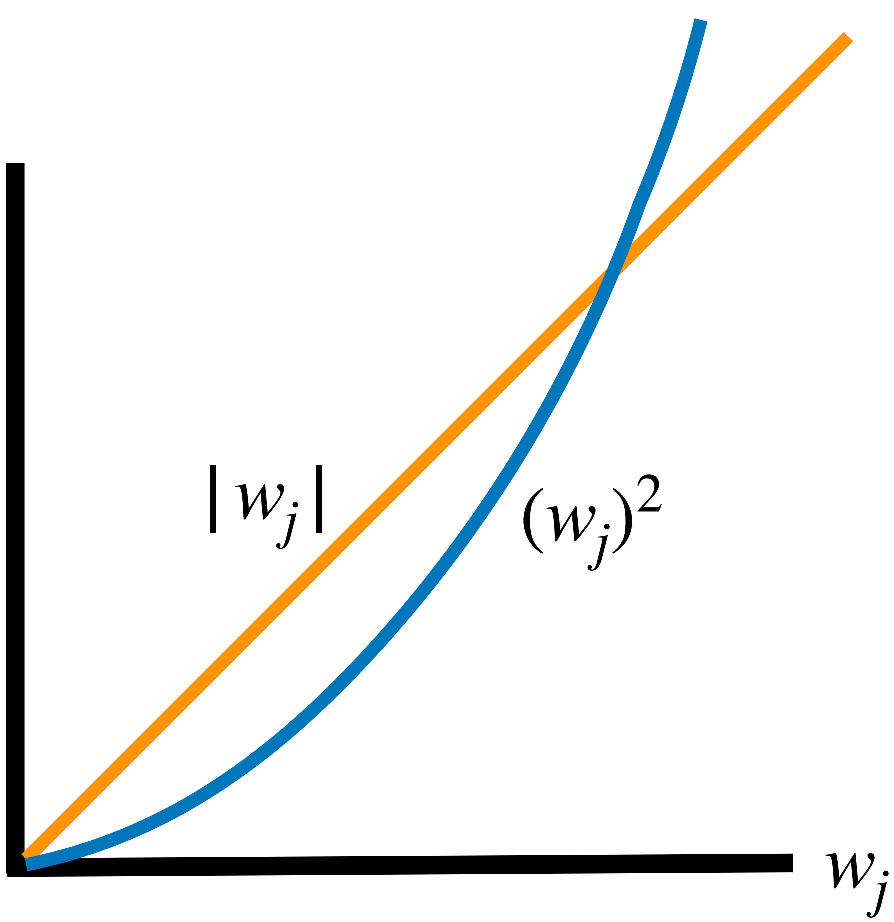
but only use selected features

- de-biasing (via re-training) is critical!

# Why does Lasso give sparse solutions?

- $$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2 \ + \ \lambda \|w\|_1$$

- comparing L1 with L2:
  - for L2 regularizer, once $w_j$ is small, $(w_j)^2$ is very small
  - so not much incentive to make coefficients go all the way to zero
  - for L1 regularizer, incentive to make $w_j$ smaller keeps up all the way until it is zero

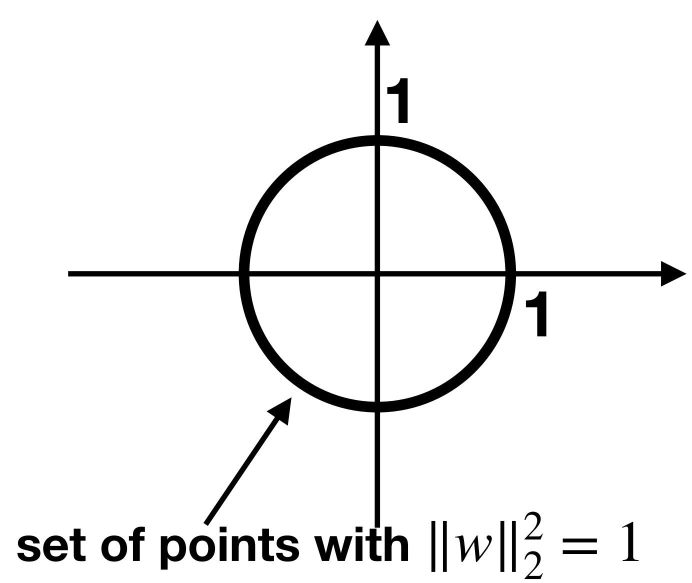


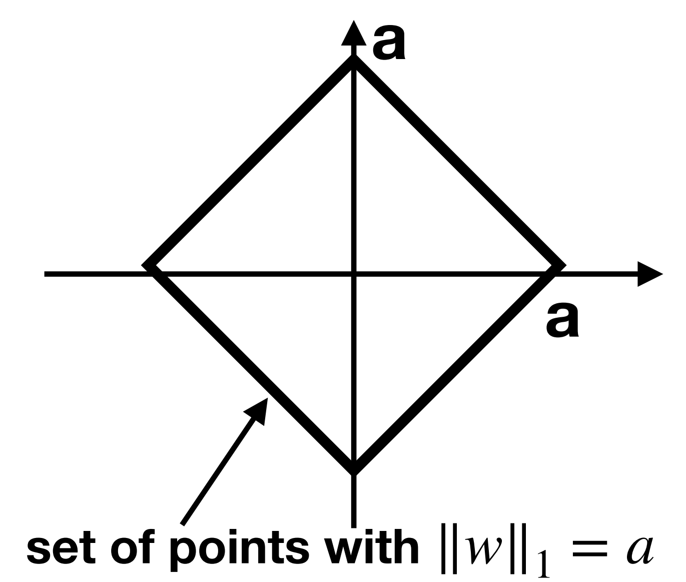

**Q. among all 2-dimensional vectors with**

$$\|w\|_2^2 = w_1^2 + w_2^2 = 1$$

**Which one has the smallest L1-norm,**

$$\|w\|_1 = |w_1| + |w_2|, ?$$



# Why does Lasso give sparse solutions?

- consider the optimal solution of a problem:

$$\hat{w}_\lambda \;=\; \arg\min_w \;\; \sum_{i=1}^{n} (w^T x_i - y_i)^2 \;+\; \lambda \|w\|_1$$

- for each given $\lambda$, there exists a $\mu$ such that the following problem has the exactly same solution

$$\hat{w}_\mu \;=\; \arg\min_w \;\; \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$\text{subject to} \;\; \|w\|_1 \;\leq\; \mu$$

- that is for any $\lambda$ there exists a $\mu$ such that
$$\hat{w}_\lambda \;=\; \hat{w}_\mu$$
- just as $\hat{w}_\lambda$ becomes sparse with increasing $\lambda$,
$\hat{w}_\mu$ becomes sparse with decreasing $\mu$
- hence, we study sparsity of the optimal solution of the second problem

# Why does Lasso give sparse solutions?

$$\text{minimize}_w \quad \underbrace{\sum_{i=1}^{n} (w^T x_i - y_i)^2}_{\mathscr{L}(w)} + \lambda \|w\|_1$$

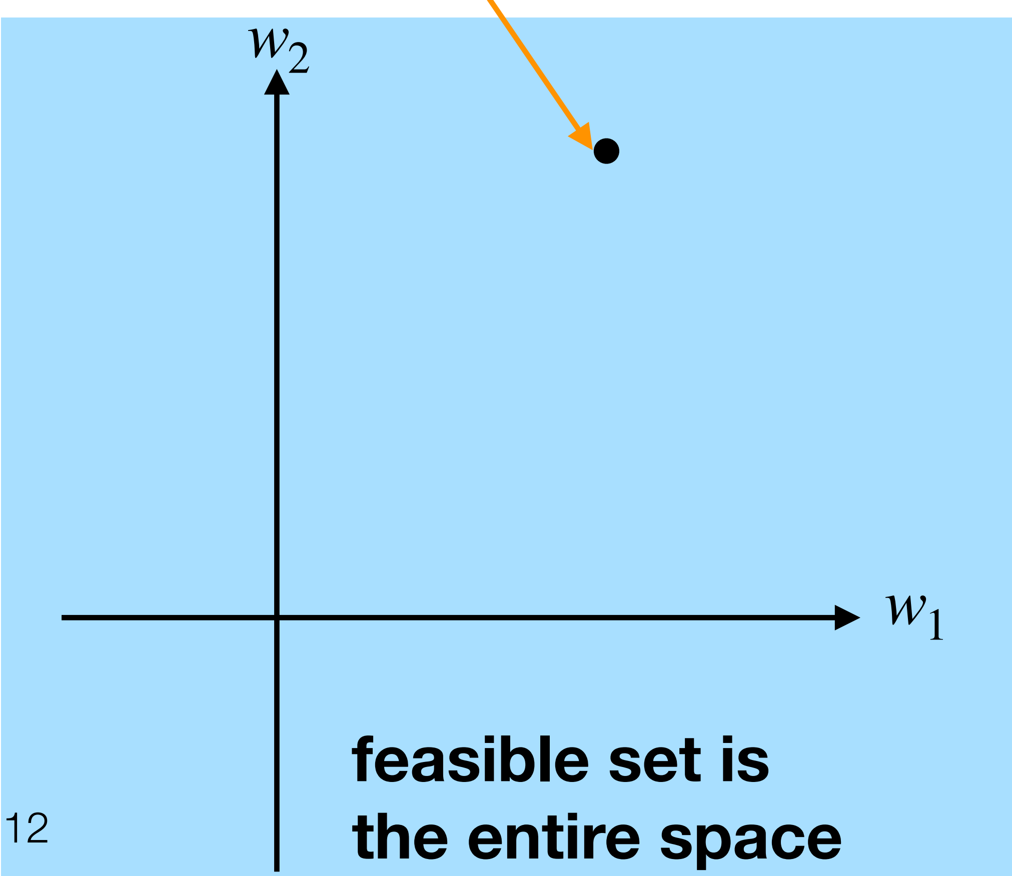
$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

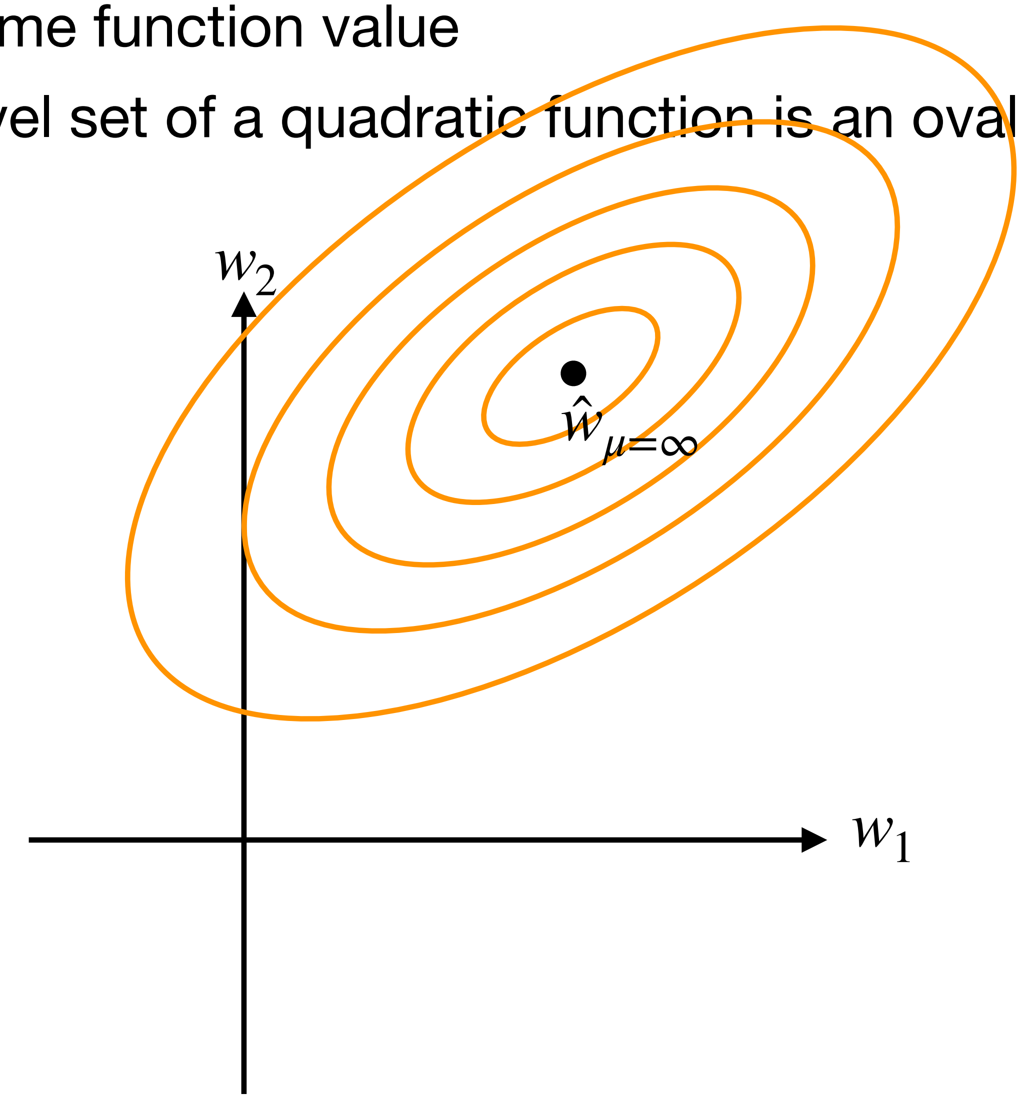
$$\text{subject to} \quad \|w\|_1 \leq \mu$$

- the **level set** of a function $\mathscr{L}(w_1, w_2)$ is defined as the set of points $(w_1, w_2)$ that have the same function value
- the level set of a quadratic function is an oval

**Optimal solution $\hat{w}_{\mu=\infty}$**

**when $\lambda = 0$ (equivalent to $\mu = \infty$)**



# Why does Lasso give sparse solutions?

$$\text{minimize}_w \quad \underbrace{\sum_{i=1}^{n} (w^T x_i - y_i)^2}_{\mathscr{L}(w)} + \lambda \|w\|_1$$

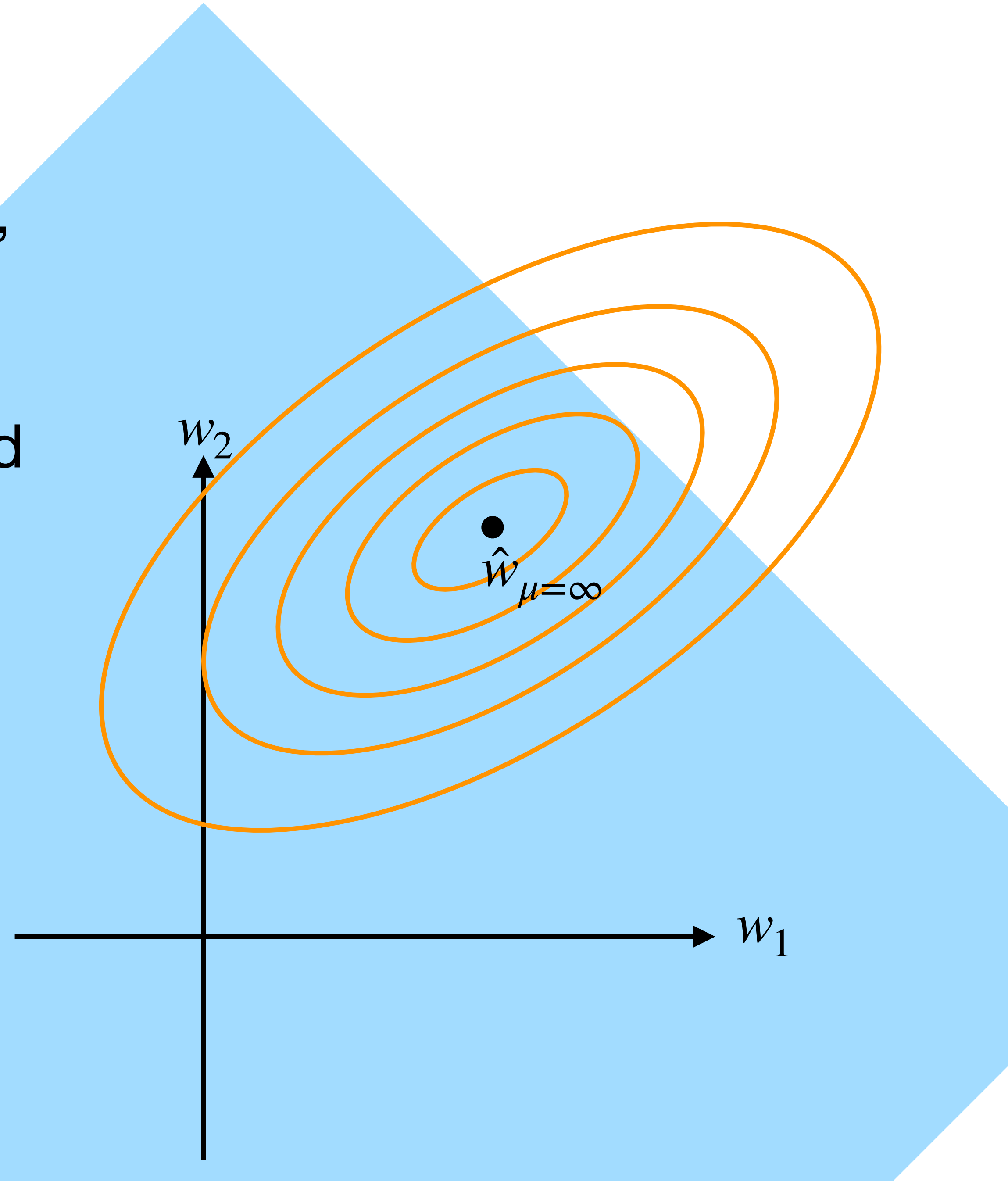
$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$\text{subject to} \quad \|w\|_1 \leq \mu$$

- as we decrease $\mu$ from infinity (which is the same as increasing regularization parameter $\lambda$), the feasible set becomes smaller
- the shape of the **feasible set** is what is known as $L_1$ ball, which is a high dimensional diamond
- In 2-dimensions, it is a diamond

$$\left\{ (w_1, w_2) \,\middle|\, |w_1| + |w_2| \leq \mu \right\}$$

- when $\mu$ is large enough such that $\mu > \|\hat{w}_{\mu=\infty}\|_1$, then the optimal solution does not change as the feasible set includes the un-regularized optimal solution



# Why does Lasso give sparse solutions?

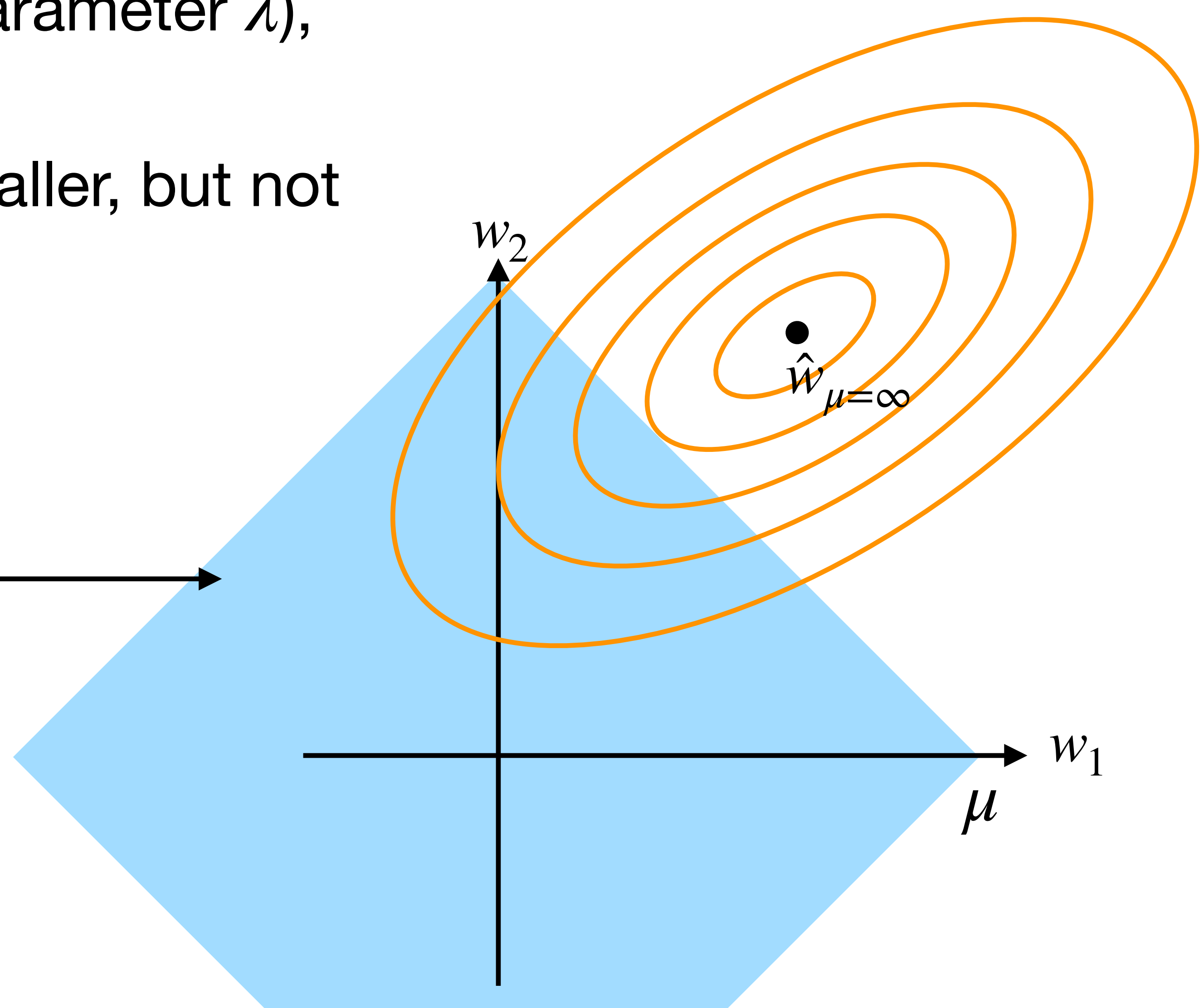
$$\text{minimize}_w \quad \underbrace{\sum_{i=1}^{n} (w^T x_i - y_i)^2}_{\mathscr{L}(w)} + \lambda \|w\|_1$$

$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$\text{subject to} \quad \|w\|_1 \leq \mu$$

- as we decrease $\mu$ from infinity, (which is the same as increasing regularization parameter $\lambda$), the **feasible set** becomes smaller
- initially, both $w_1$ and $w_2$ become smaller, but not zero

**feasible set:** $\{w \in \mathbb{R}^2 \mid \|w\|_1 \leq \mu\}$



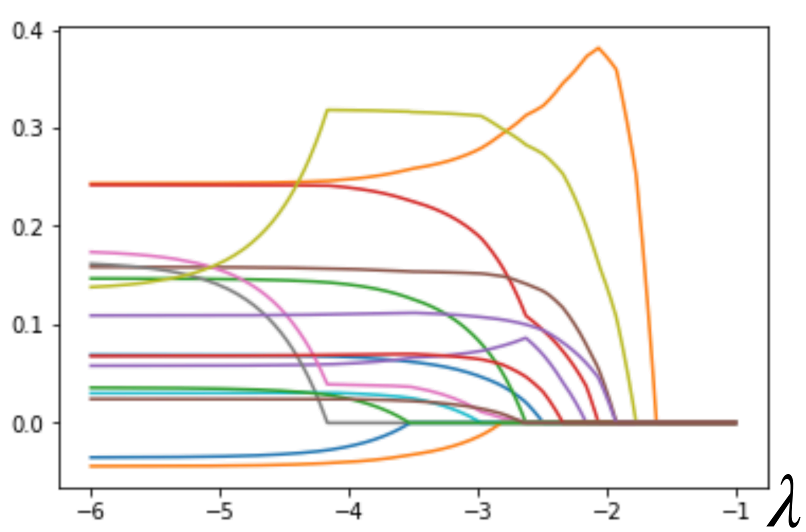
# Why does Lasso give sparse solutions?

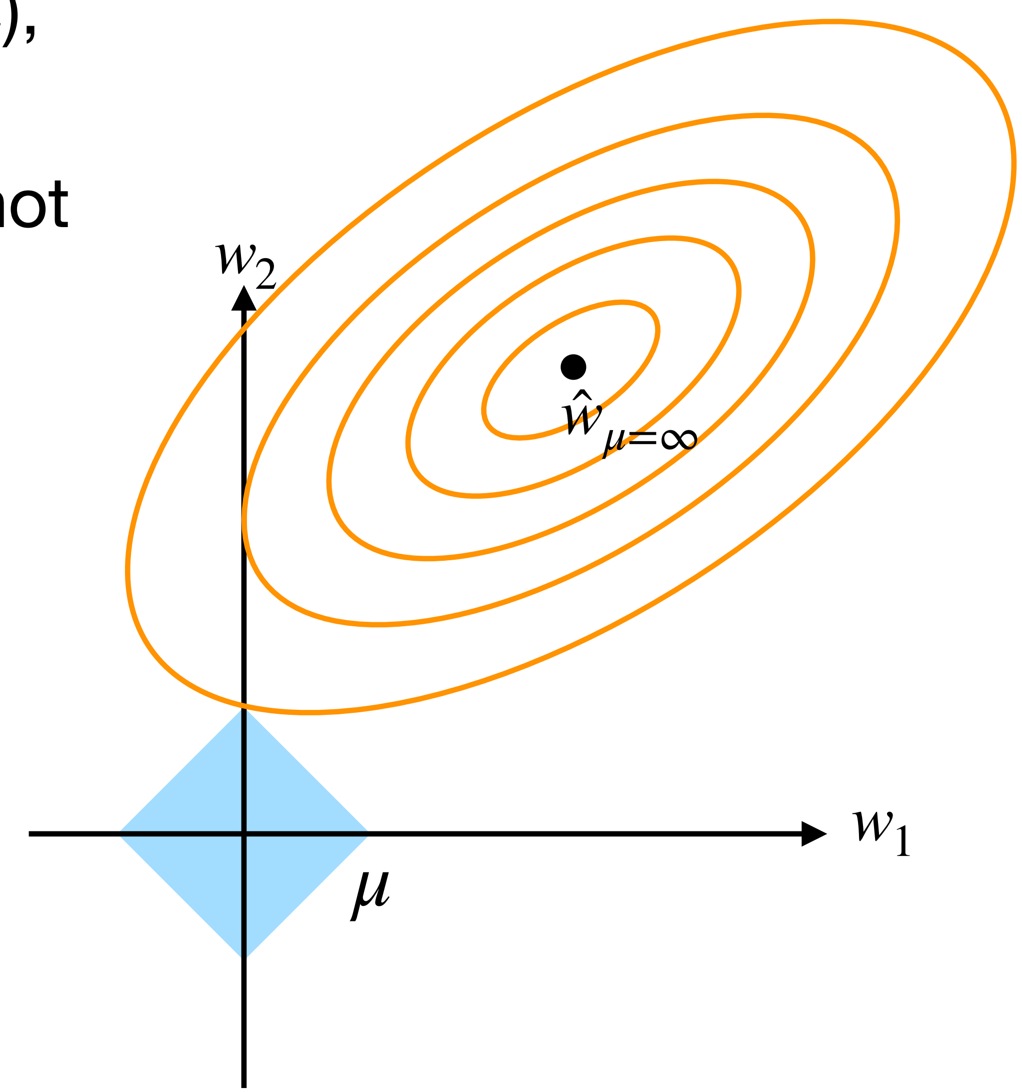
$$\text{minimize}_w \quad \underbrace{\sum_{i=1}^{n} (w^T x_i - y_i)^2}_{\mathcal{L}(w)} + \lambda \|w\|_1$$

$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$\text{subject to} \quad \|w\|_1 \leq \mu$$

- as we decrease $\mu$ from infinity, (which is the same as increasing regularization parameter $\lambda$), the feasible set becomes smaller
- initially, both $w_1$ and $w_2$ become smaller, but not zero
- eventually, $w_j$'s become zero one by one
- this explains the regularization path of **Lasso**



# In the case of Ridge regression

$$\text{minimize}_w \quad \underbrace{\sum_{i=1}^{n} (w^T x_i - y_i)^2}_{\mathscr{L}(w)} + \ \lambda \|w\|_2^2$$

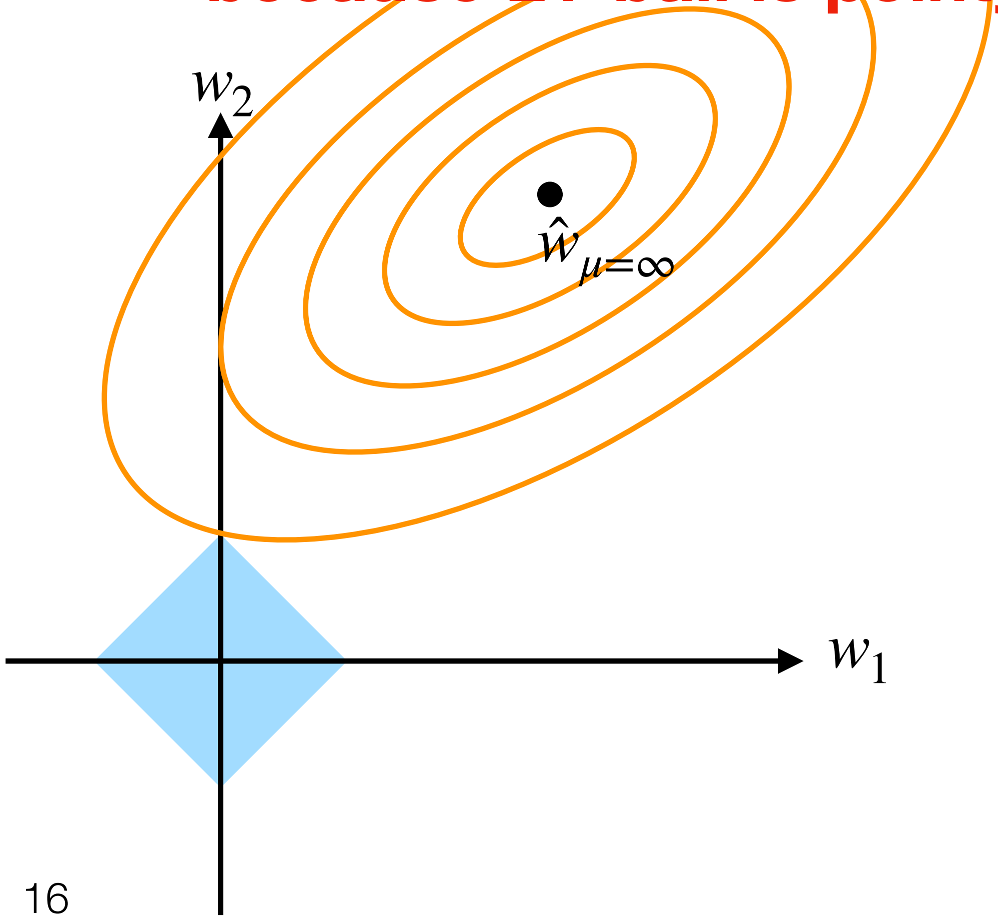
$$\text{minimize}_w \quad \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

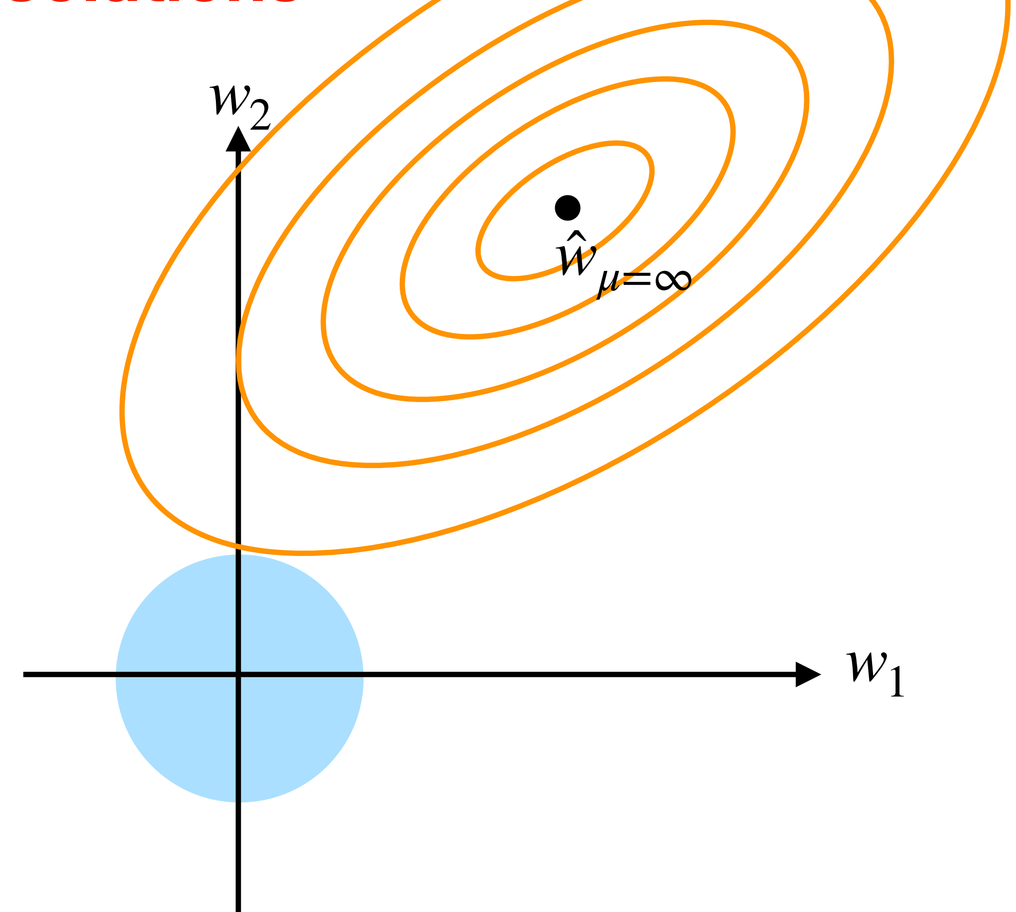
$$\text{subject to} \quad \|w\|_2^2 \ \leq \ \mu$$

- for ridge regression, the feasible set is an $L_2$-norm ball, which is actually a **ball**

$$\{(w_1, w_2) \mid w_1^2 + w_2^2 \leq \mu\}$$

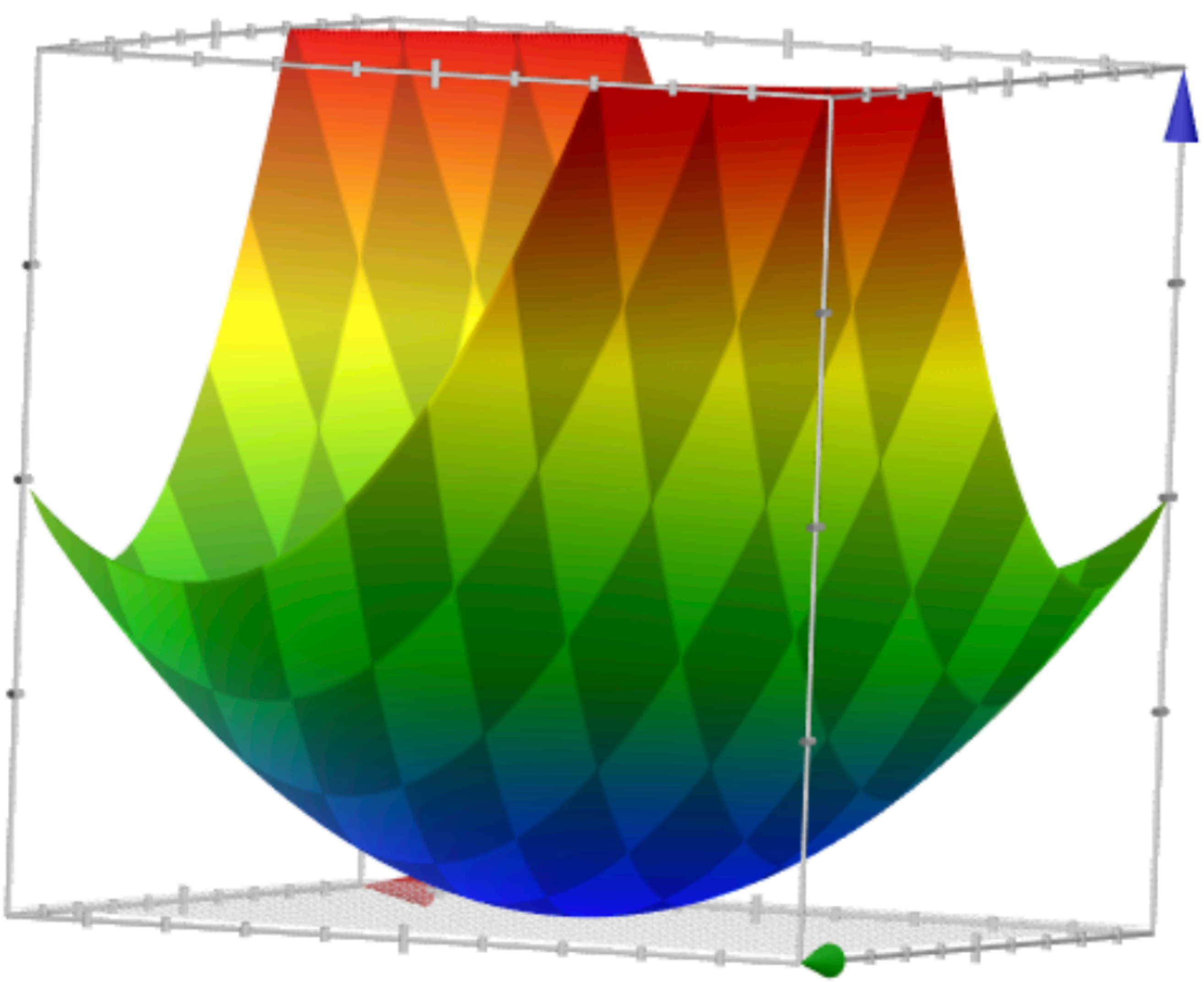
- hence, the solution is not sparse
- **because L1-ball is pointy, we get sparse solutions**

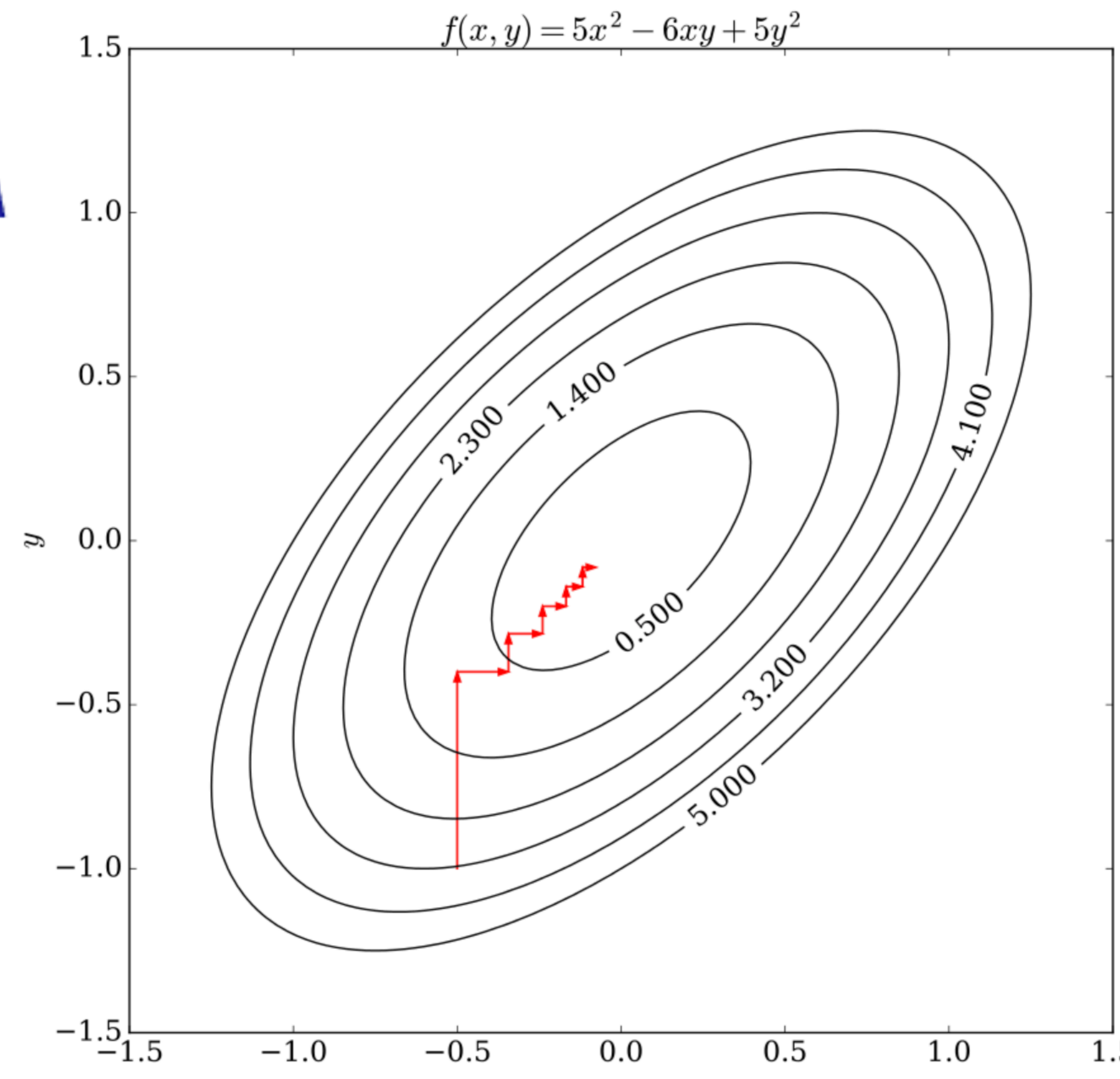


# Optimization: how do we solve Lasso?

- 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 $f(x, y) = 5x^2 - 6xy + 5y^2$





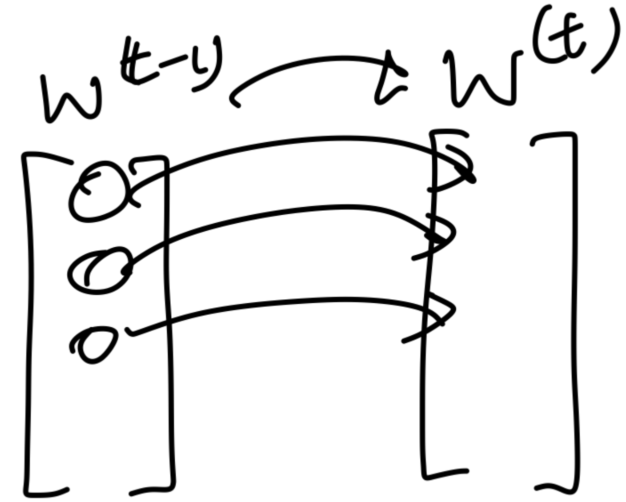
$$\min_{w_1,\dots,w_d} f(w_1,\dots,w_d) = \|X\cdot w - y\|_2^2 + \lambda\|w\|_1$$

Input: $S_{train}$, $T$

Initialize $w^{(0)} = 0$

For $t = 1, \dots, T$

For $j = 1, \dots, d$




$$w_j^{(t)} \leftarrow \arg\min_{w_j} f\left(\begin{bmatrix} w_1^{(t)} \\ \vdots \\ w_{j-1}^{(t)} \\ w_j \\ w_{j+1}^{(t-1)} \\ \vdots \\ w_d^{(t-1)} \end{bmatrix}\right)$$

# Optimization: how do we solve Lasso?

- $$\text{minimize}_w \quad \|\mathbf{X}w - \mathbf{y}\|_2^2 + \lambda \|w\|_1$$
- we will study one method (coordinate descent) to solve the problem and find the minimizer $\hat{w}_{\text{lasso}}$
- **Coordinate descent**
  - **input: training data $S_{\text{train}}$, 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 least squares

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

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

- 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 for illustration purpose

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

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

t-th iteration, $j = 1$

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

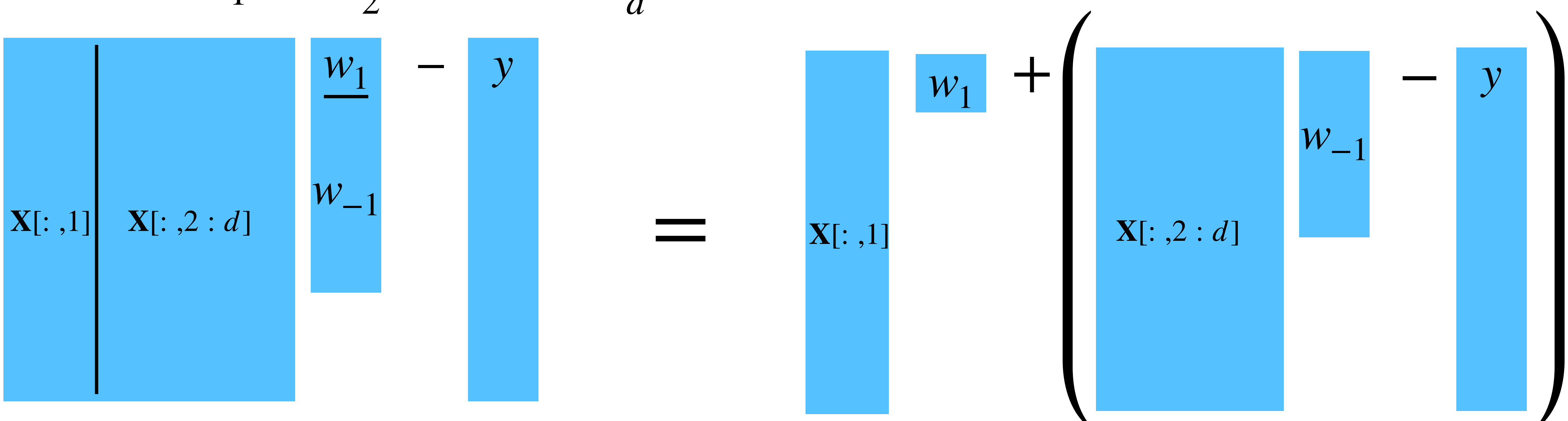
$$w_{-1} = \begin{bmatrix} w_2^{(t-1)} \\ \vdots \\ w_d^{(t-1)} \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} X_1 & X_{2:d} \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} w_1 \\ \hline w_{-1} \end{bmatrix}}_{w} - \underbrace{\begin{bmatrix} \ \\ \ \end{bmatrix}}_{y} = \|X_1 \cdot w_1 - \underbrace{(y - X_{2:d} \cdot w_{-1})}\|_2^2$$

$$\boxed{w_1^{(t)} \leftarrow (X_1^T X_1)^{-1} X_1^T \cdot (y - X_{2:d} w_{-1})}$$

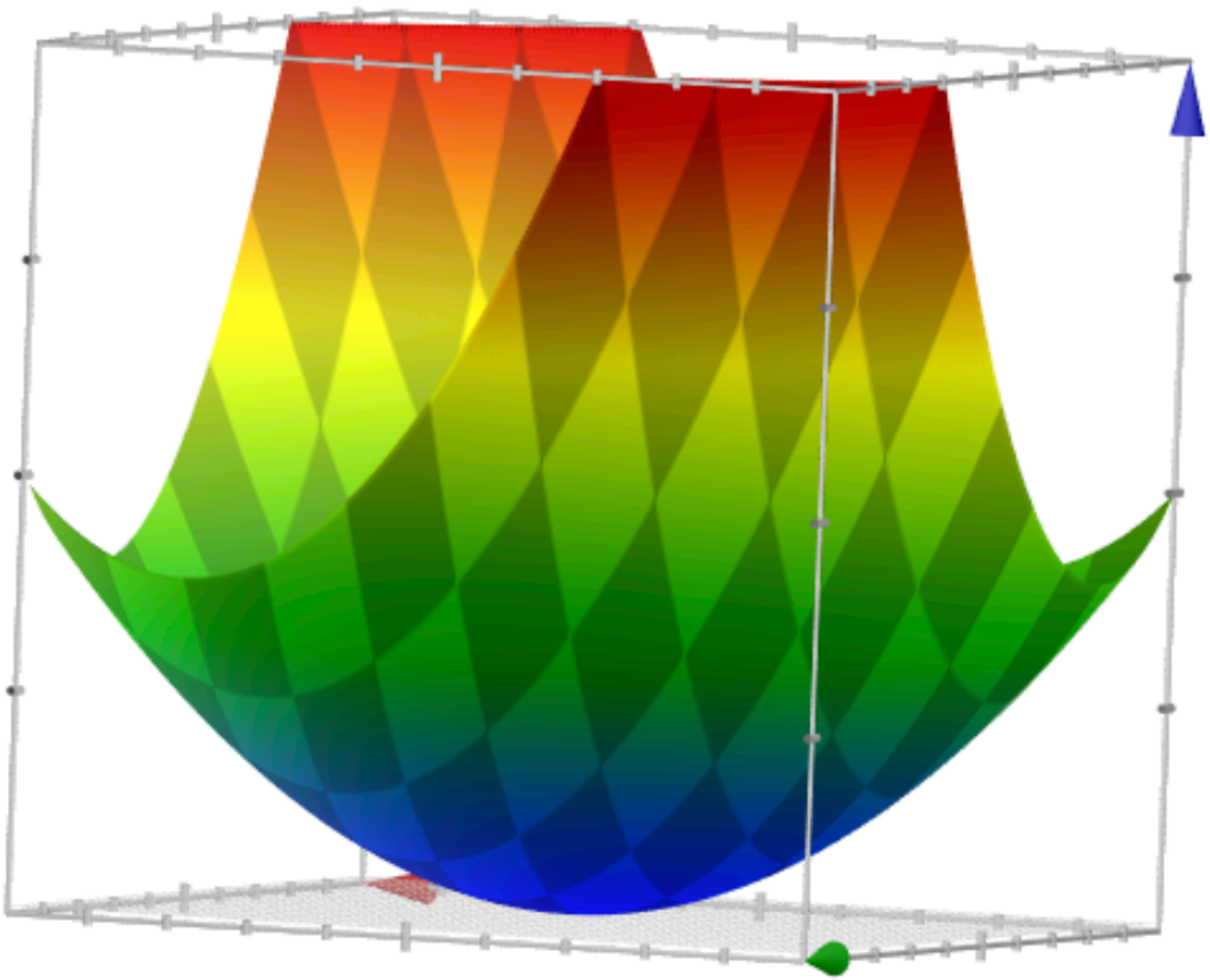
# Coordinate descent for (un-regularized) linear least squares

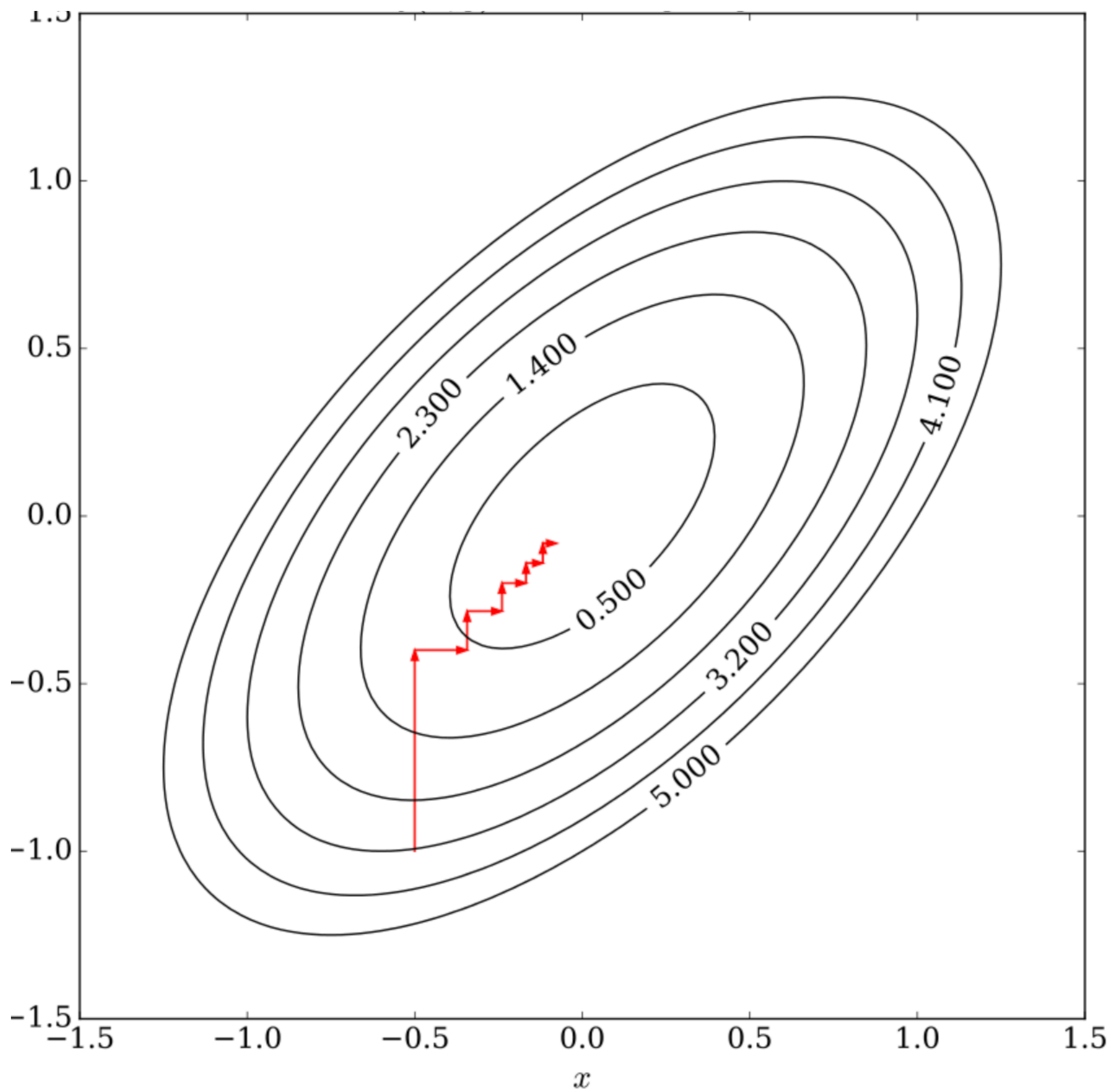
- we will study the case $j = 1$, for now (other cases are almost identical)
- when updating $w_1^{(t)}$, recall that
  $$w_1^{(t)} \leftarrow \arg\min_{w_1} \|\mathbf{X}w - \mathbf{y}\|_2^2$$
  where $w = [w_1, \; w_2^{(t-1)}, \; \dots \; , w_d^{(t-1)}]^T$
- first step is to write the objective function in terms of the variable we are optimizing over, that is $w_1$:
  $$\mathcal{L}(w) = \left\| \mathbf{X}[:,1]w_1 + \mathbf{X}[:,2:d]w_{-1} - \mathbf{y} \right\|_2^2$$
  where $w_{-1} = [w_2^{(t-1)}, \; \dots \; , w_d^{(t-1)}]^T$




- we know from linear least squares that the minimizer is
  $$w_1^{(t)} \leftarrow (\mathbf{X}[:,1]^T\mathbf{X}[:,1])^{-1}\mathbf{X}[:,1]^T \left( \mathbf{y} - \mathbf{X}[:,2:d]w_{-1} \right)$$

- Coordinate descent applied to a quadratic loss





# Coordinate descent for Lasso

- let us apply coordinate descent on Lasso, which minimizes

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

- 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 $w_1$

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

  - hence, the coordinate descent update boils down to

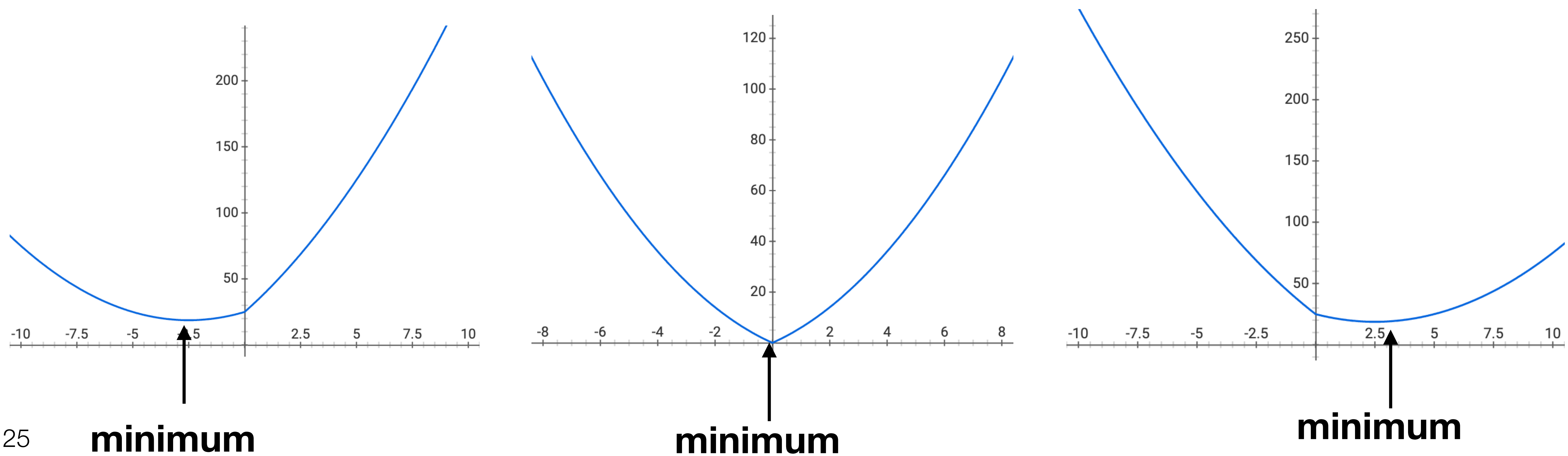
$$w_1^{(t)} \leftarrow \arg\min_{w_1} \underbrace{\left\| \mathbf{X}[:,1]w_1 - \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-1}\right) \right\|_2^2 + \lambda |w_1|}_{f(w_1)}$$

# Convexity

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

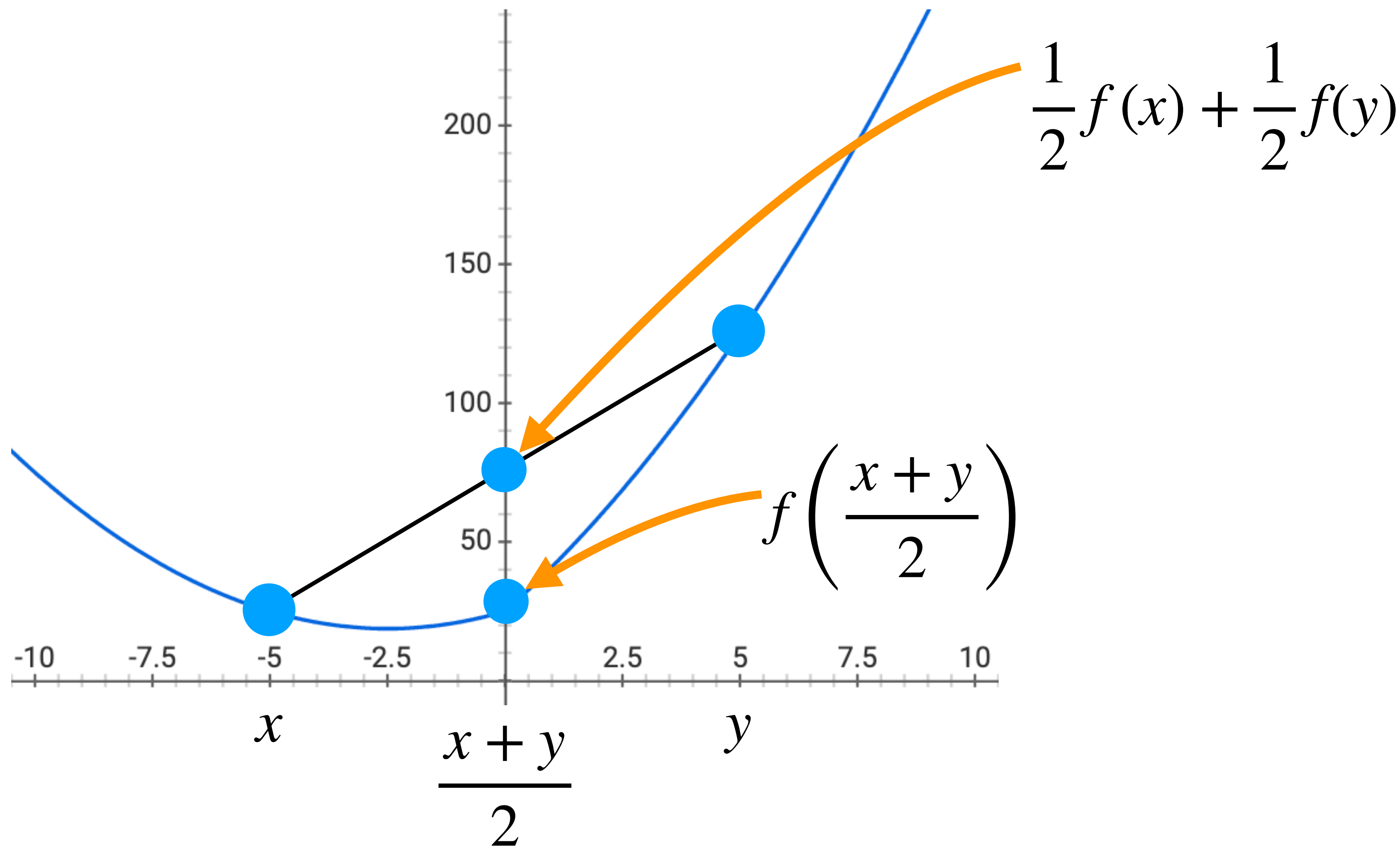
$$f(w_1) = (aw_1 - b)^2 + \lambda |w_1|$$

- 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

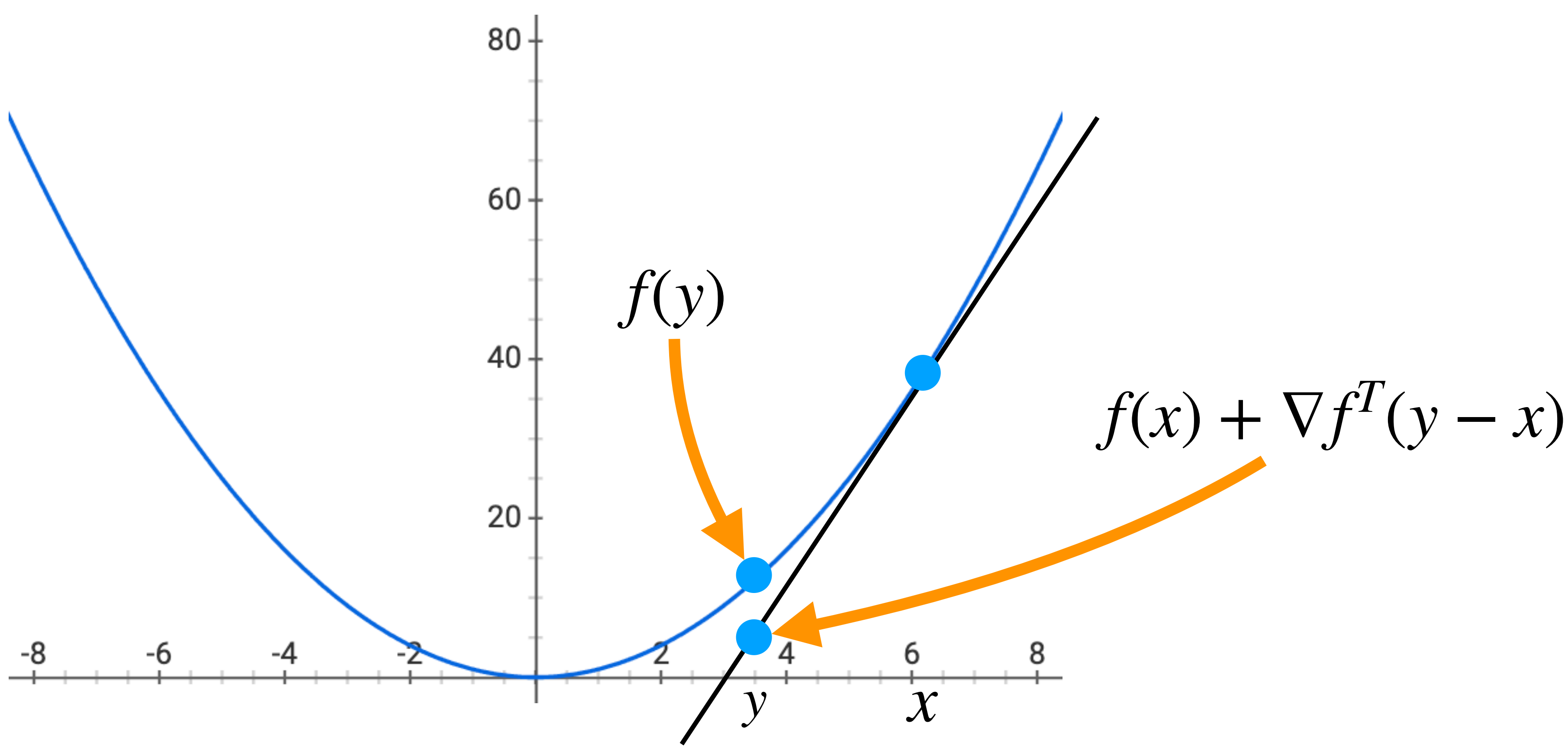
- A function $f(x)$ is **convex** if and only if
  - $f(ax + (1-a)y) \quad \leq \quad a f(x) + (1-a) f(y)$
    for all $a \in [0,1]$ and all $x, y$



# Convexity

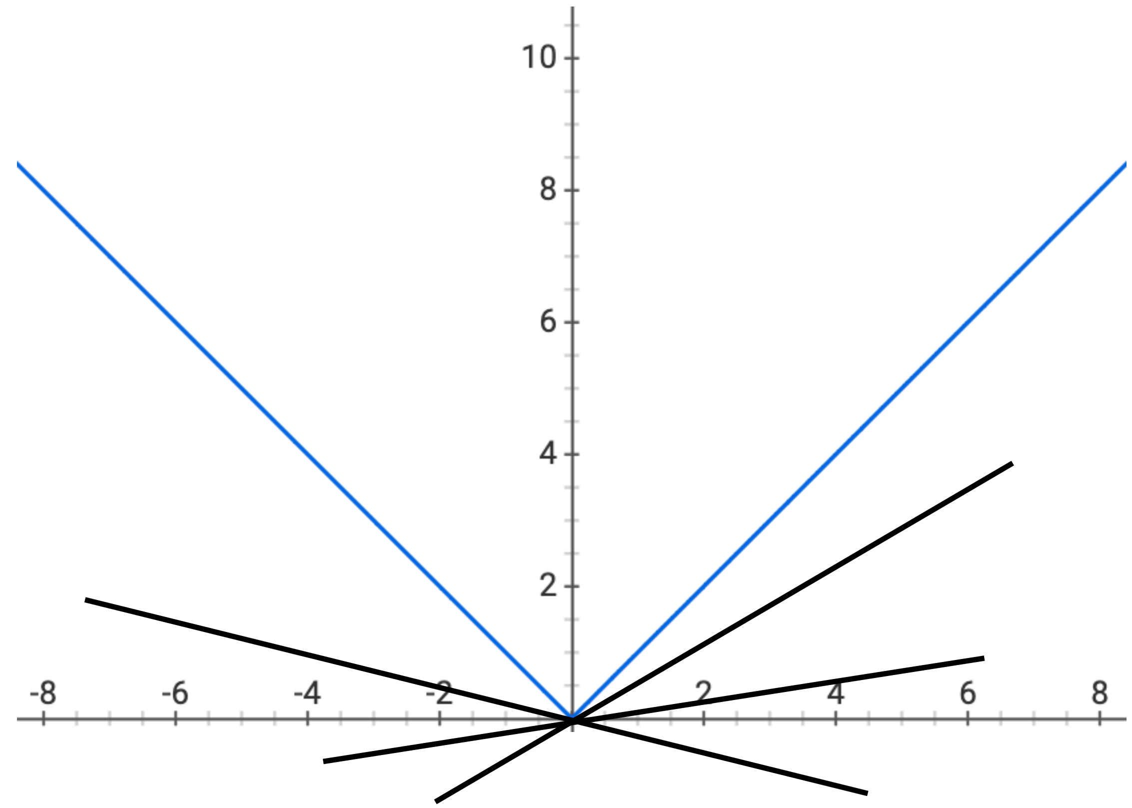
- function $f(x)$ is **differentiable** if and only if
  - partial derivative $\dfrac{\partial f(x)}{\partial x_j}$ exists for all $x$ and $j \in \{1, \ldots, d\}$

- for a differentiable function $f(x)$, there is another definition of **convexity**
  - $f(y) \geq f(x) + \nabla f(x)^T (y - x)$
    for all $x, y$



# Convexity

$f(x) = |x|$



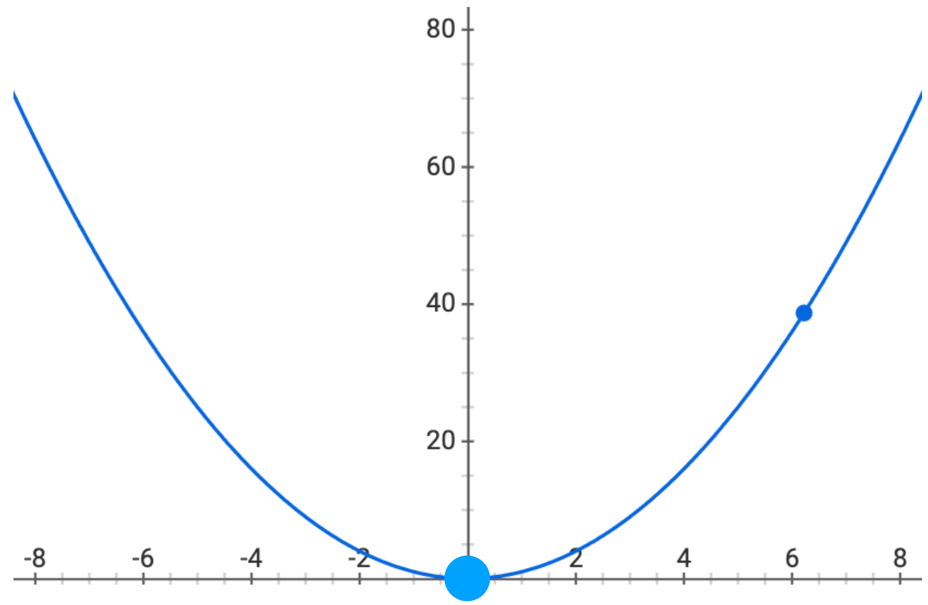

- for a **non-differentiable** function, gradient is not defined at some points, for example at $x = 0$ for $f(x) = |x|$
- at such points, **sub-gradient** plays the role of gradient
  - sub-gradient at a differentiable point is the same as the gradient
  - sub-gradient at a non-differentiable point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

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

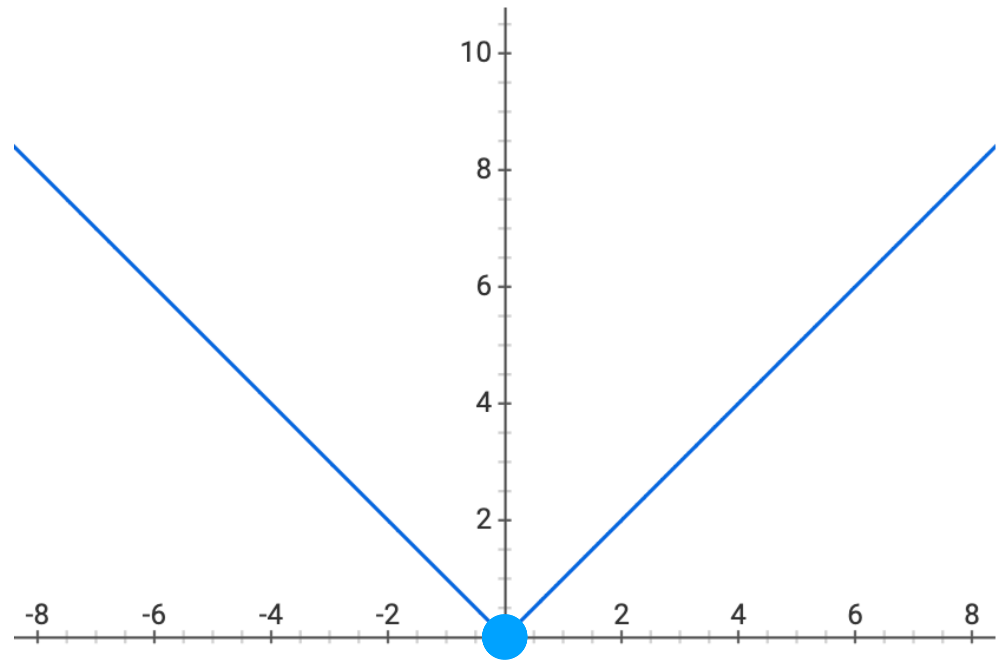
  - for example, $\partial|x| \;=\; \begin{cases} +1 & \text{for } x > 0 \\ [-1, 1] & \text{for } x = 0 \\ -1 & \text{for } x < 0 \end{cases}$

# 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



$$w_1^{(t)} \leftarrow \arg\min_{w_1} \underbrace{\| X_1 \cdot w_1 - (y - X_{2:d} \cdot w_{-1}) \|_2^2 + \lambda |w_1|}_{f(w_1)}$$

$$f(w_1) = (a w_1 - b)^2 + \lambda |w_1| \quad + \text{const}$$

$$= \underbrace{X_1^T X_1}_{\in \mathbb{R}} \cdot w_1^2 - w_1 \cdot 2 \underbrace{X_1^T (y - X_{2:d} \cdot w_{-1})}_{\in \mathbb{R}} + \lambda |w_1| + \text{const}$$

$$= \Big( \underbrace{\sqrt{X_1^T X_1}}_{a} \cdot w_1 - \underbrace{\frac{X_1^T (y - X_{2:d} w_{-1})}{\sqrt{X_1^T X_1}}}_{b} \Big)^2 + \lambda |w_1| + \text{const}$$

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

$$= 2a(aw_1 - b) + \lambda \cdot \begin{cases} +1 & w_1 > 0 \\ [-1, +1] & w_1 = 0 \\ -1 & w_1 < 0 \end{cases}$$

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

# Coordinate descent update on Lasso

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

- 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

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

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

$$\partial f(w_1) = \begin{cases} 2a(aw_1 - b) + \lambda & \text{if } w_1 > 0 \\ \hline \end{cases}$$

Case 1: if $2a(aw_1 - b) + \lambda = 0$ for some $w_1 > 0$

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

$$w_1^{(t)} \leftarrow \frac{b}{a} - \frac{\lambda}{2a^2} \quad \text{if } 2ab > \lambda$$

---

$$\partial f(w_1) = 2a(aw_1 - b) - \lambda \quad \text{if } w_1 < 0$$

Case 2: $w_1^* = \dfrac{\lambda + 2ab}{2a^2} < 0$

$$w_1^{(t)} \longleftarrow \frac{b}{a} + \frac{\lambda}{2a^2} \quad \text{if } 2ab < -\lambda$$

---

$$\partial f(w_1) = [-2ab - \lambda, -2ab + \lambda], \; w_1^* = 0$$

$$\Downarrow \text{(contains) } 0$$

Case 3:

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

$$w_1^{(t)} \longleftarrow 0, \; -\lambda \leq 2ab \leq \lambda$$

# How do we find the minimizer?

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

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

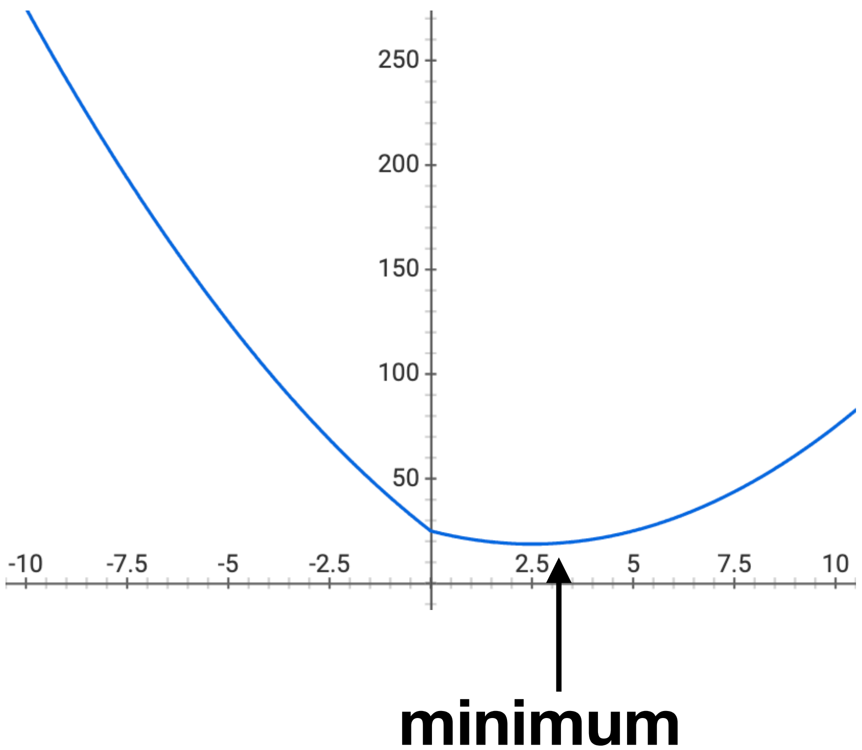
- case 1:
  - $2a(aw_1 - b) + \lambda = 0$ for some $w_1 > 0$
  - this happens when

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

  - hence,

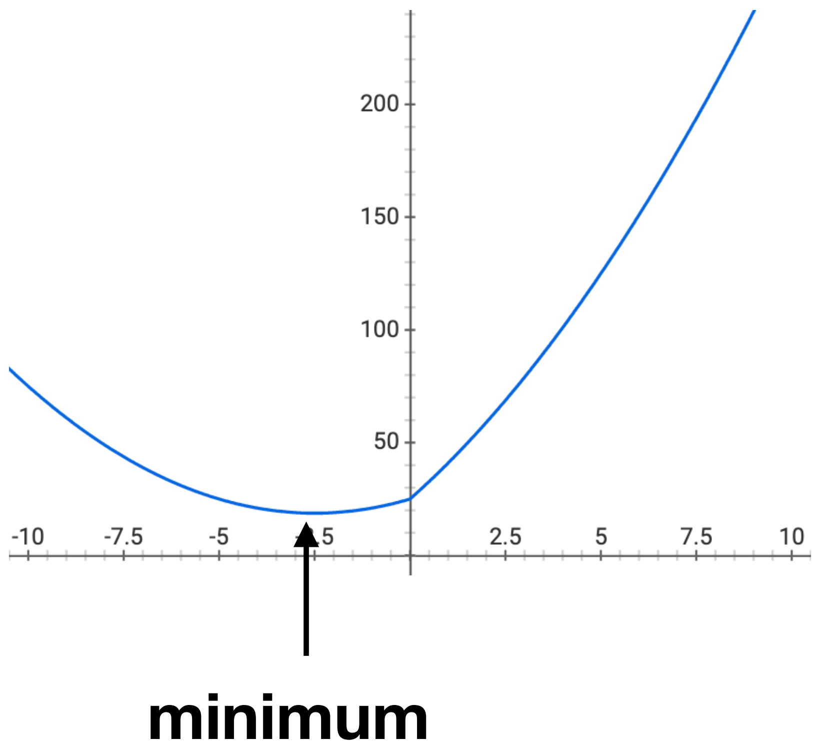
    $$w_1^{(t)} \leftarrow \frac{b}{a} - \frac{\lambda}{2a^2},$$

    if $\lambda < 2ab$



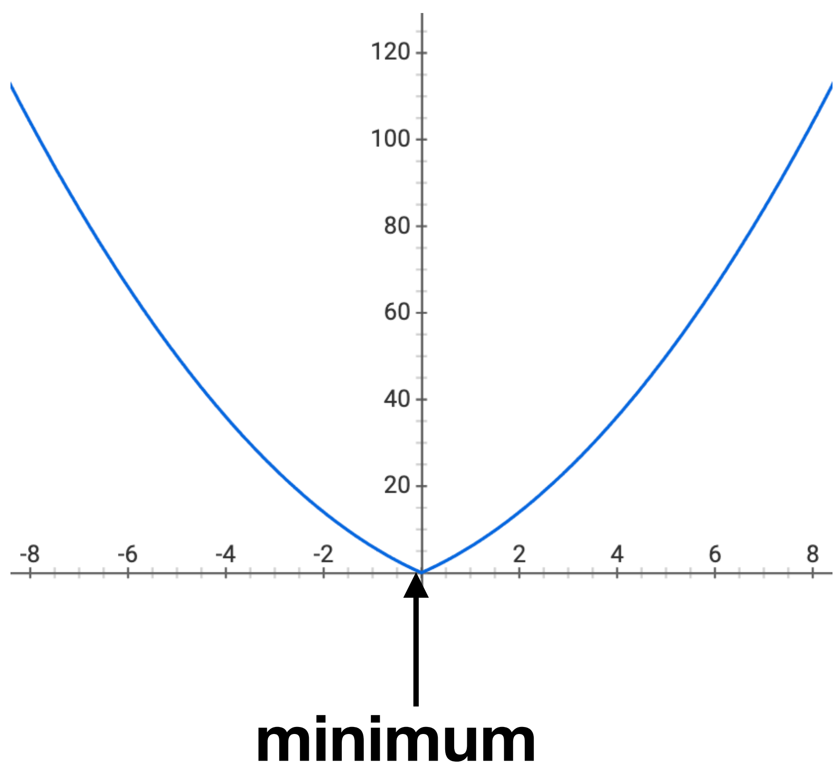
- case 2:
  - $2a(aw_1 - b) - \lambda = 0$ for some $w_1 < 0$
  - this happens when
    $$w_1 = \frac{\lambda + 2ab}{2a^2} < 0$$
  - hence,
    $$w_1^{(t)} \leftarrow \frac{b}{a} + \frac{\lambda}{2a^2},$$

    if $\lambda < -2ab$




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

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

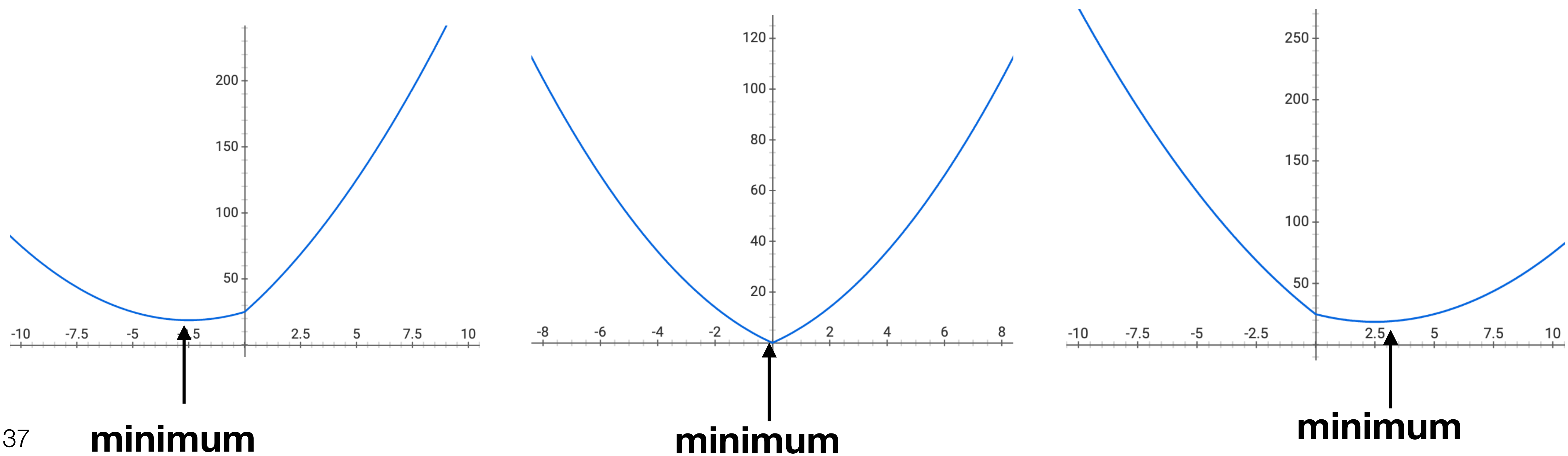


# Coordinate descent on Lasso

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

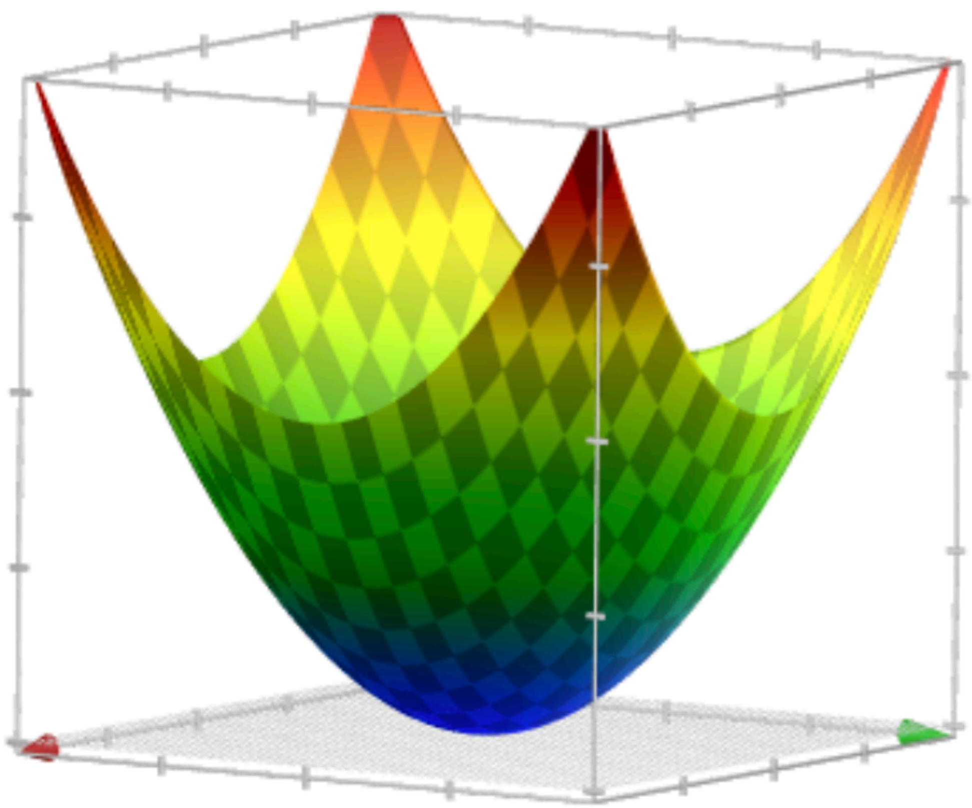
$$w_1^{(t)} \leftarrow \begin{cases} \frac{b}{a} - \frac{\lambda}{2a^2} & \text{for } 2ab > \lambda \\ 0 & \text{for } -\lambda \leq 2ab \leq \lambda \\ \frac{b}{a} + \frac{\lambda}{2a^2} & \text{for } \lambda < -2ab \end{cases}$$

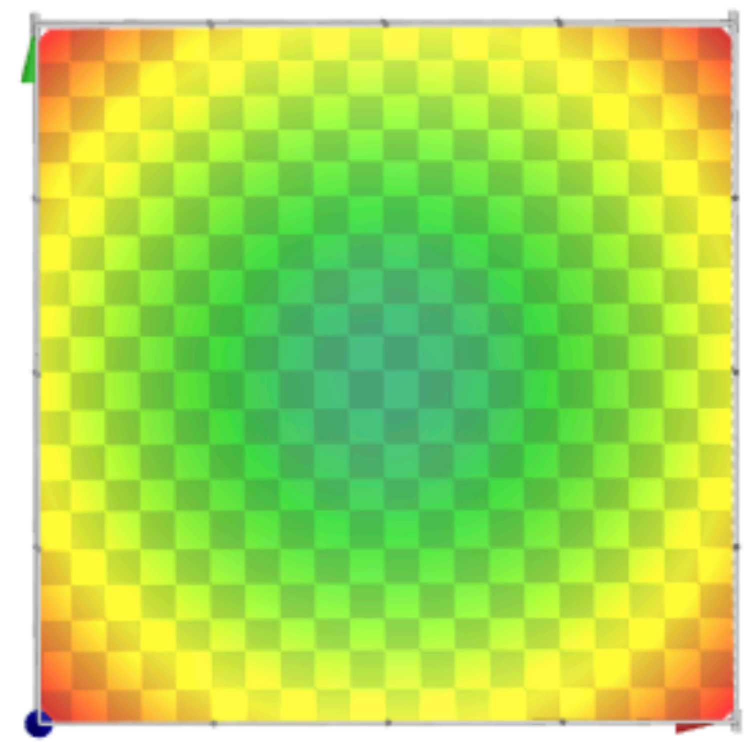
- where $a = \sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}$, and $b = \dfrac{\mathbf{X}[:,1]^T \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{-1}\right)}{\sqrt{\mathbf{X}[:,1]^T \mathbf{X}[:,1]}}$



# When does coordinate descent work?

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

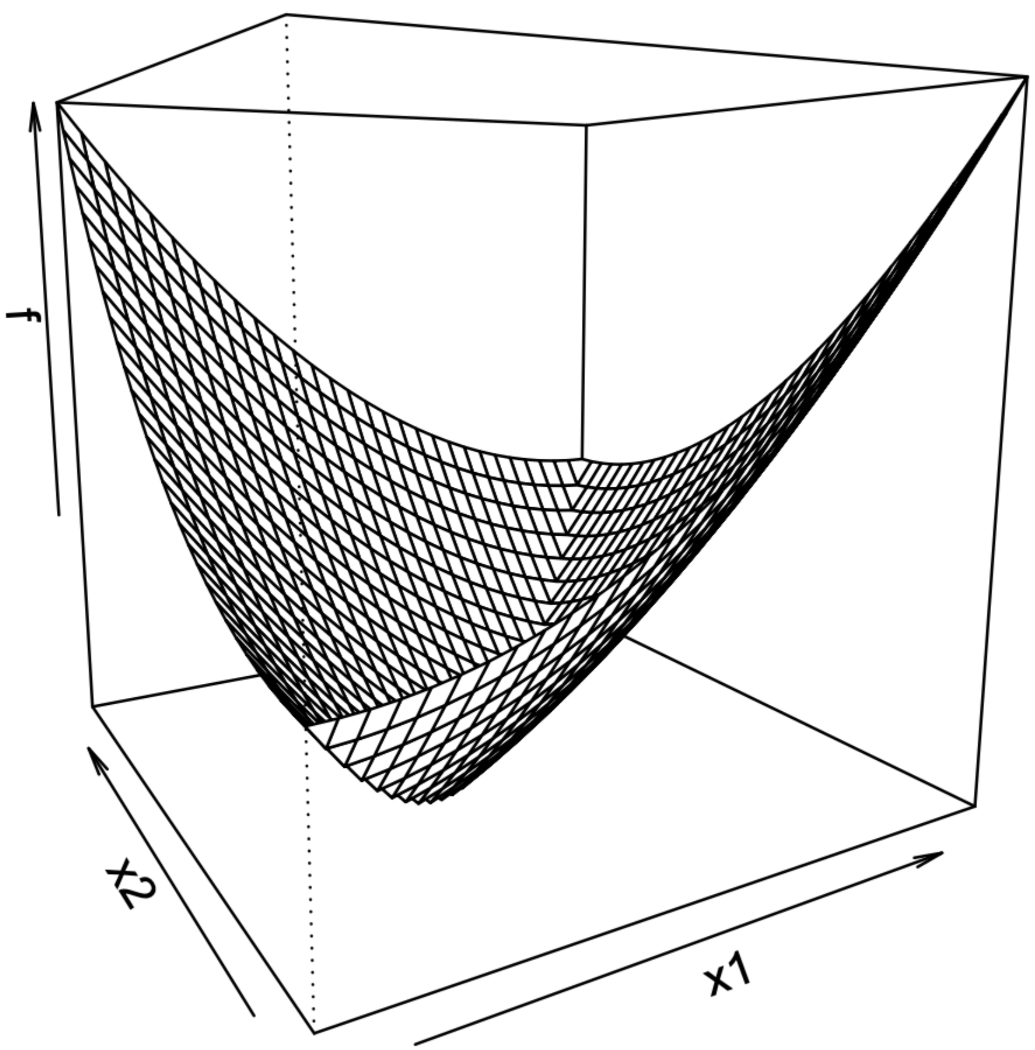


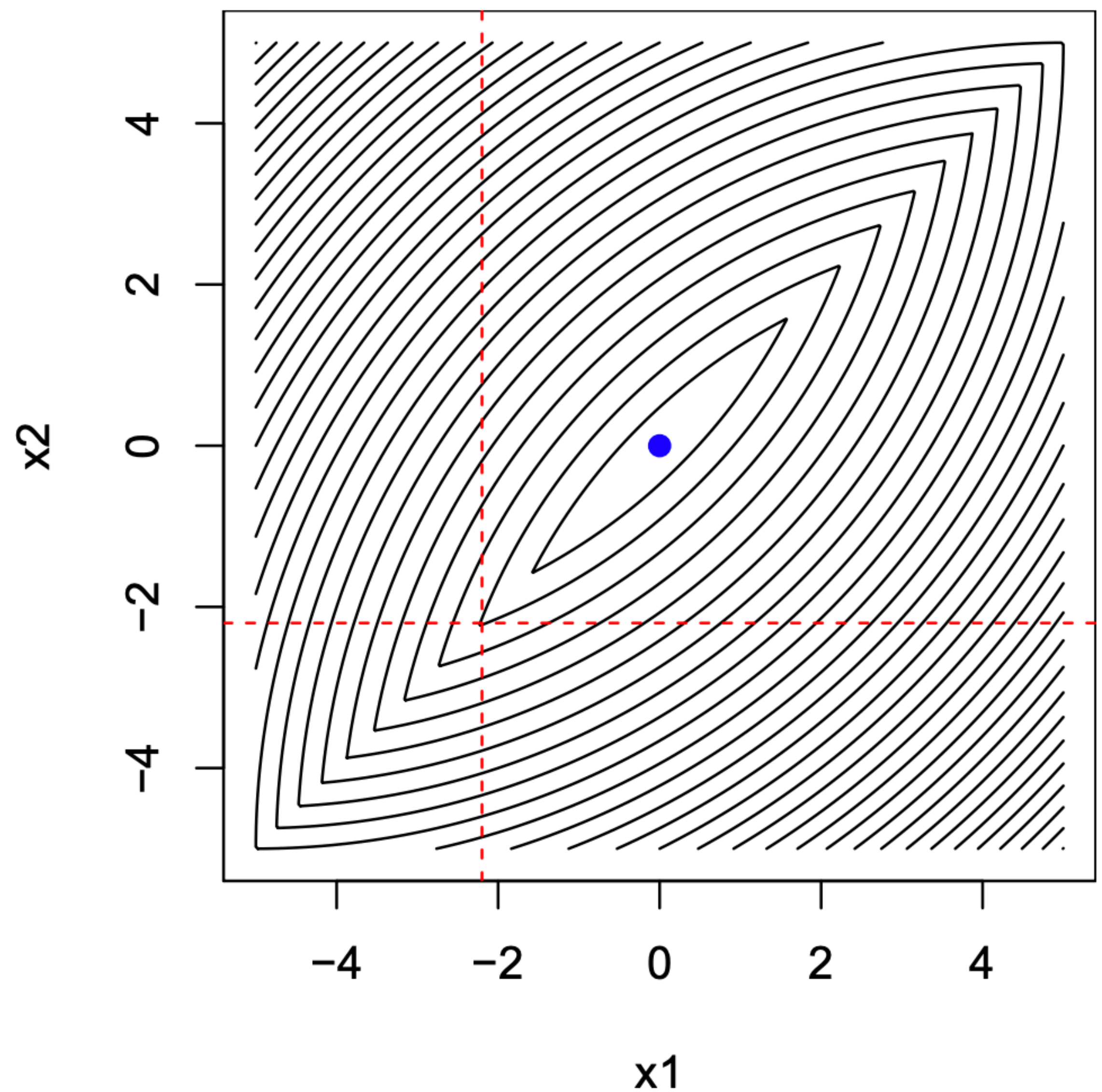


- when coordinate descent has stopped, that means $$\frac{\partial f(x)}{\partial x_j} = 0 \text{ for all } j \in \{1,\ldots,d\}$$
- this implies that the gradient $\nabla_x f(x) = 0$, which happens only at minimum

# 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
-