- HW 2 due Friday Feb 11th

Lecture 15: Coordinate Descent (continued)

- How to solve non-smooth optimization like 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 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

$$\underset{w_1}{\arg\min} \underbrace{\left\| \mathbf{X}[:,1]w_1 \; - \; \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{2:d}\right) \right\|_2^2}_{(aw_1 - b)^2 + \text{Constants}} + \lambda\left(|w_1| + \underbrace{\|w_{2:d}\|_1}_{\text{constant}} \right)$$

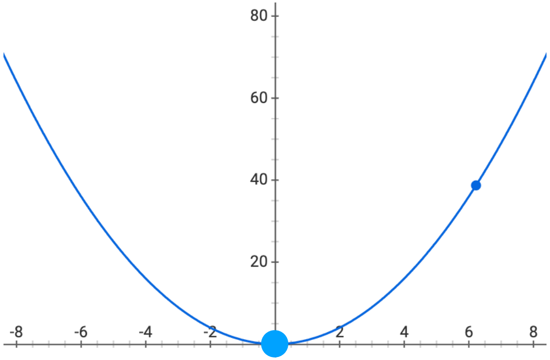
$$+ \lambda |w_1|$$

  - hence, the coordinate descent update boils down to

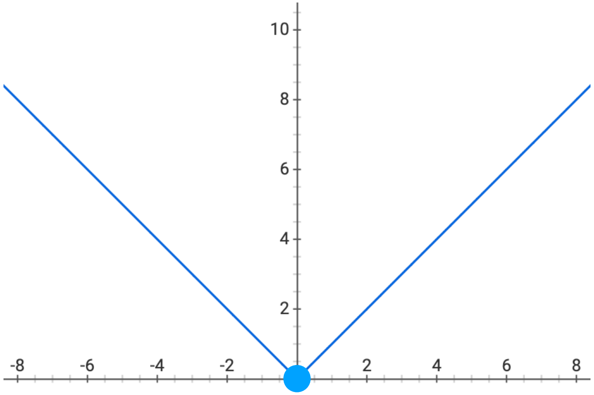
$$w_1^{(t)} \leftarrow \underset{w_1}{\arg\min} \underbrace{\left\| \mathbf{X}[:,1]w_1 \; - \; \left(\mathbf{y} - \mathbf{X}[:,2:d]w_{2:d}^{(t-1)}\right) \right\|_2^2 + \lambda |w_1|}_{f(w_1)}$$

How do we find the minima?

- 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



Finding the minima for $(aw_1 - b)^2 + \lambda | w_1 | = f(w_1)$

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

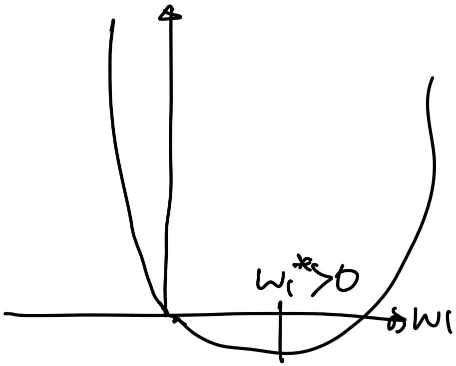
$$\partial f(w_1) \;=\; \begin{cases} 2a(aw_1 - b) + \lambda & \text{for } w_1 > 0 \\ [-2ab - \lambda, -2ab + \lambda] & \text{for } w_1 = 0 \\ 2a(aw_1 - b) - \lambda & \text{for } w_1 < 0 \end{cases}$$

* Case 1.

$w_1^* > 0$, & $2a(aw_1^* - b) + \lambda = 0$

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

$\therefore$ If $2ab - \lambda > 0$, then $w_1^* = \frac{2ab - \lambda}{2a^2}$

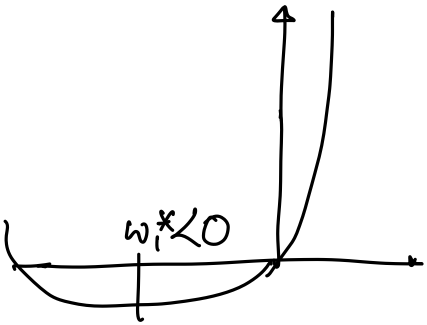


Finding the minima for $(aw_1 - b)^2 + \lambda |w_1|$

- 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.



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

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

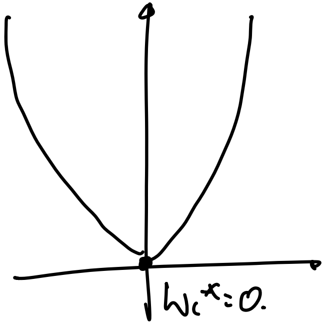
$\therefore 2ab + \lambda < 0 \longrightarrow w_1^* = \frac{2ab + \lambda}{2a^2}$

Finding the minima 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.



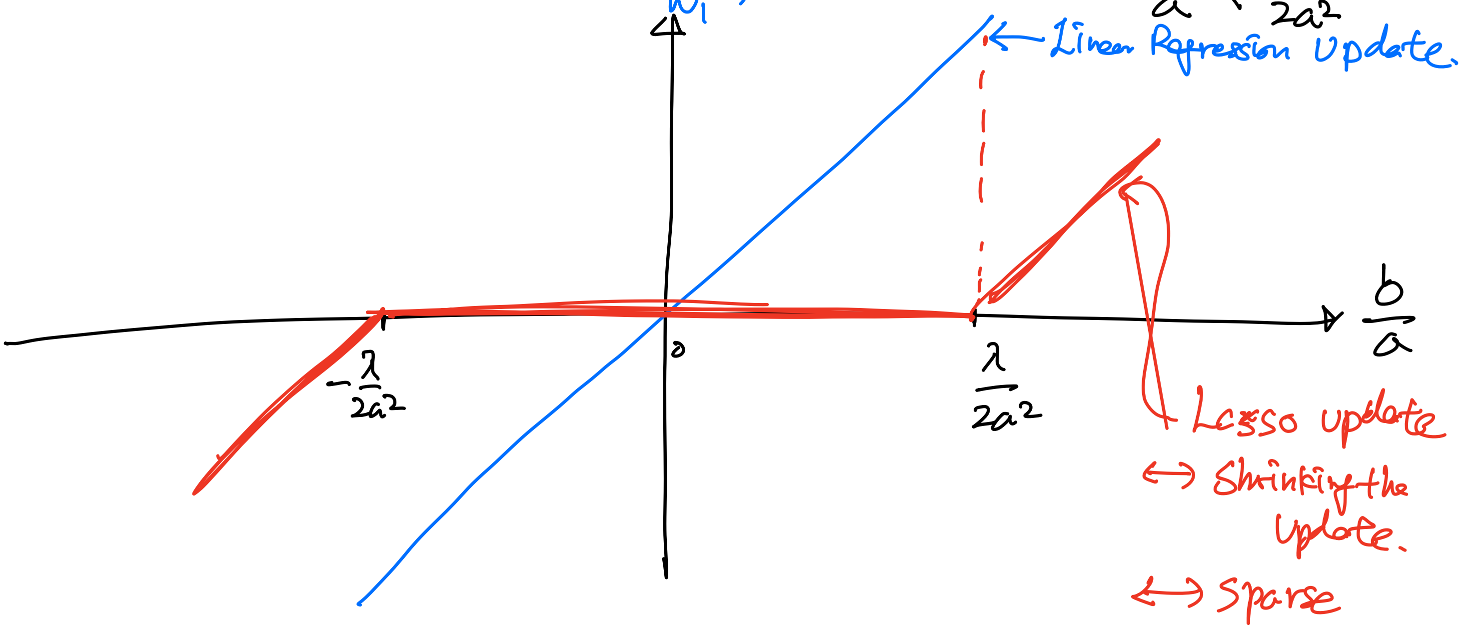
$w_1^* = 0$ & $-2ab - \lambda \le 0 \le -2ab + \lambda$

↳ i.e. $-\lambda \le 2ab \le \lambda$ then $w_1^* = 0.$

Finding the minima 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

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



How do we find the minimizer?

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

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

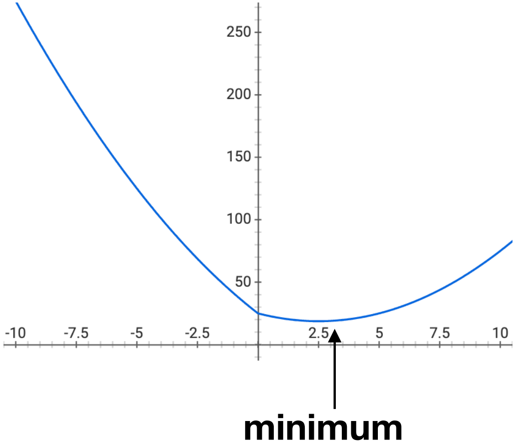
- case 1:
  - $2a(aw_1 - b) + \lambda = 0$ for some $w_1 > 0$
  - this happens when

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

  - hence,

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

if $\lambda < 2ab$



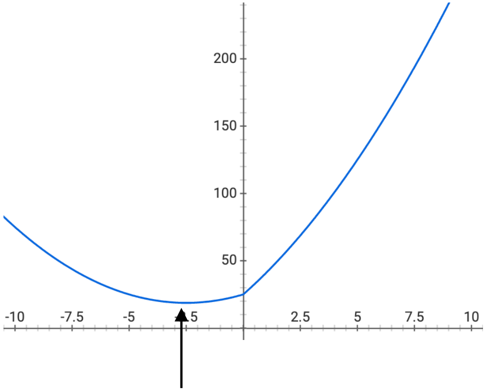
- case 2:
  - $2a(aw_1 - b) - \lambda = 0$ for some $w_1 < 0$
  - this happens when

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

  - hence,

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

if $\lambda < -2ab$

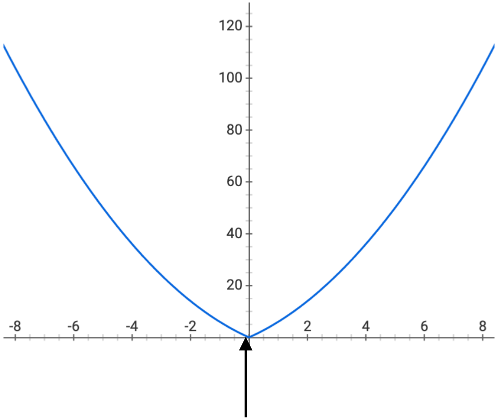


minimum

- case 3:
  - $0 \;\in\; [-2ab - \lambda, -2ab + \lambda]$
  - and $w_1 = 0$
  - hence,

$$w_1^{(t)} \leftarrow 0,$$

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



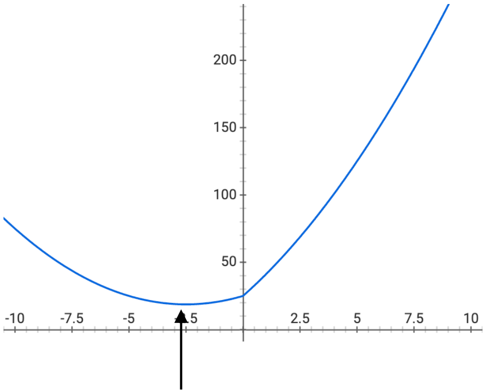
minimum

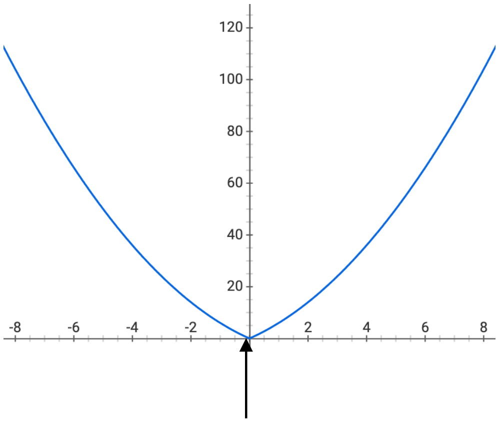
Coordinate descent on Lasso

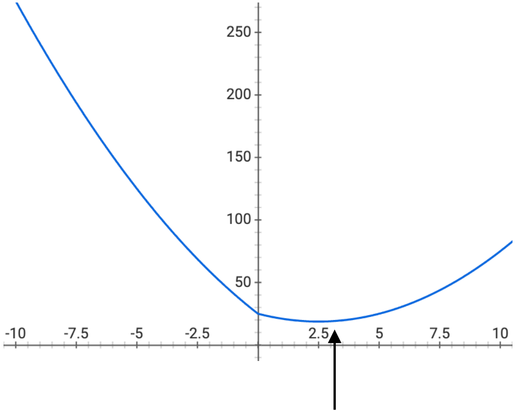
- considering all three cases, we get the following update rule by setting the sub-gradient to zero

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

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

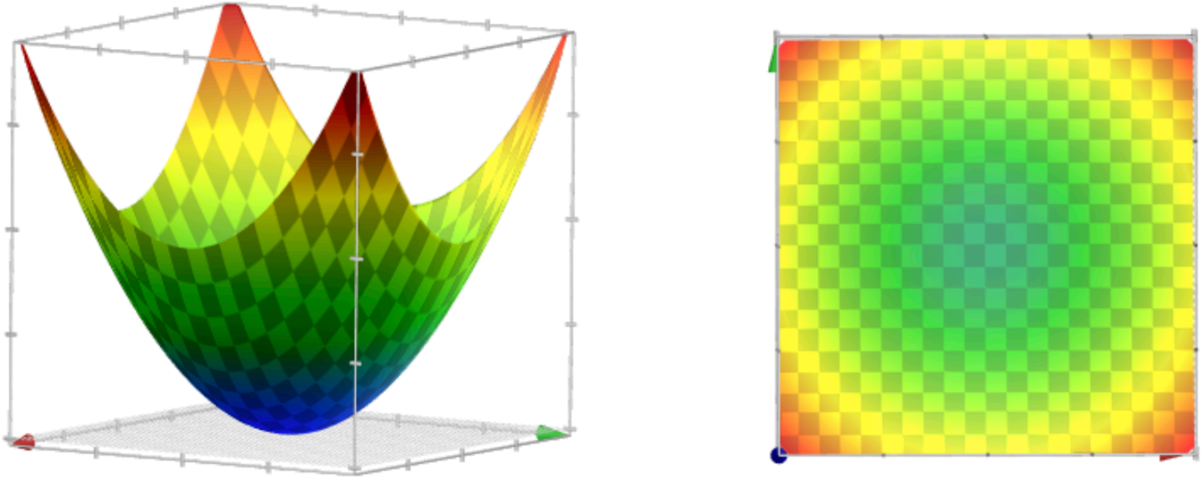






When does coordinate descent work?

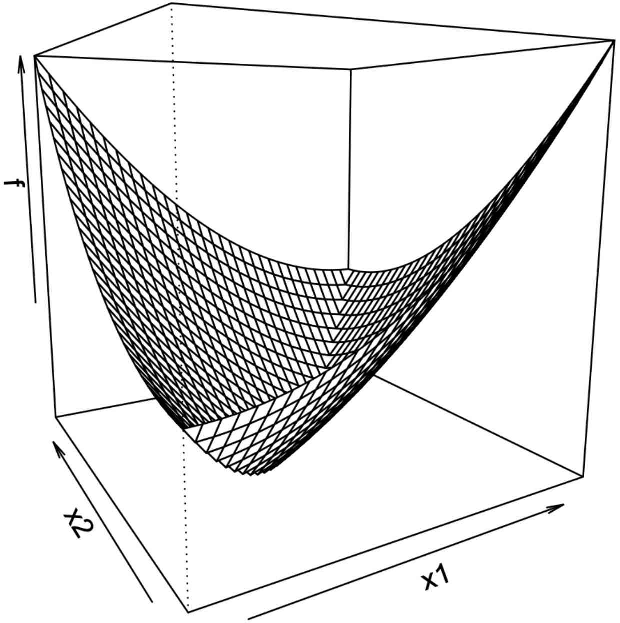
- Consider minimizing a **differentiable convex** function $f(x)$, then coordinate descent converges to the global minima

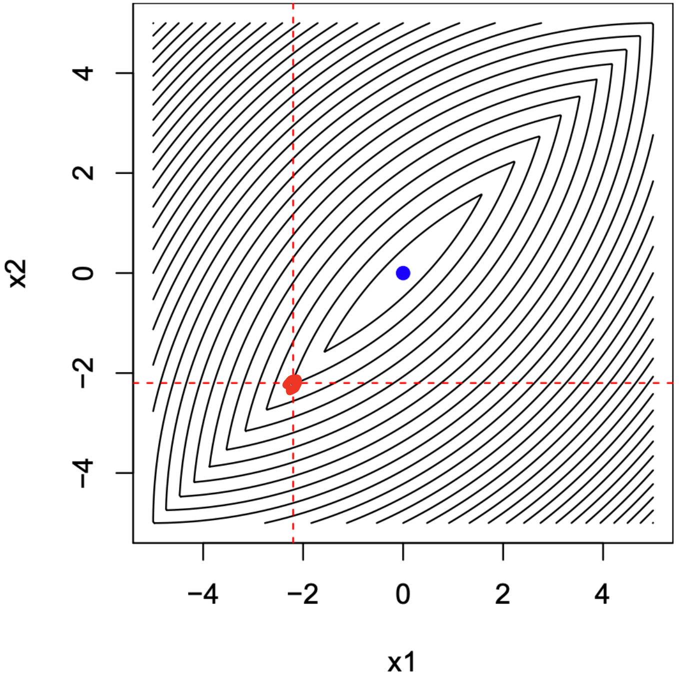


- when coordinate descent has stopped, that means
  $$\frac{\partial f(x)}{\partial x_j} = 0 \text{ for all } j \in \{1, \ldots, d\}$$
- this implies that the gradient $\nabla_x f(x) = 0$, which happens only at minimum

When does coordinate descent work?

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

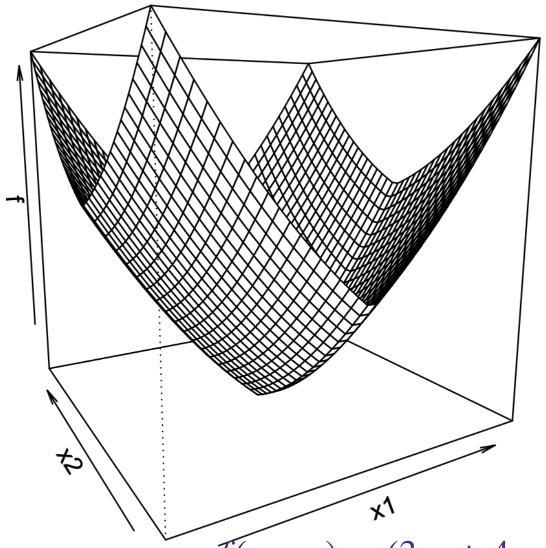


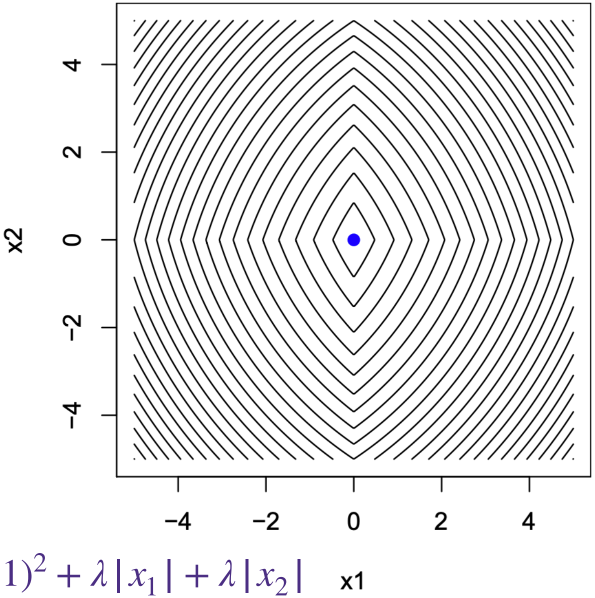


$$f(x_1, x_2) = (3x_1 + 4x_2 + 1)^2 + \lambda \,|\, x_1 - x_2 \,|$$

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





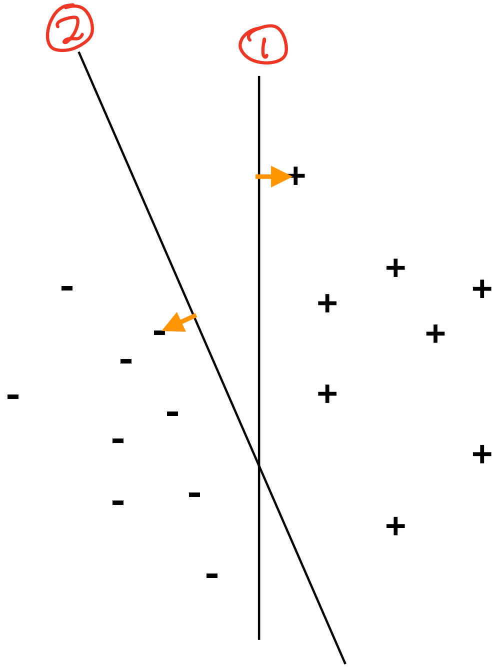
- $f(x_1, x_2) = (3x_1 + 4x_2 + 1)^2 + \lambda |x_1| + \lambda |x_2|$

Questions?

Support Vector Machines

How do we choose the best linear classifier?

- informally, **margin** of a set of examples to a decision boundary is the distance to the closest point to the decision boundary
- for linearly separable datasets, **maximum margin** classifier is a natural choice
- large margin implies that the decision boundary can change without losing accuracy, so the learned model is more robust against new data points



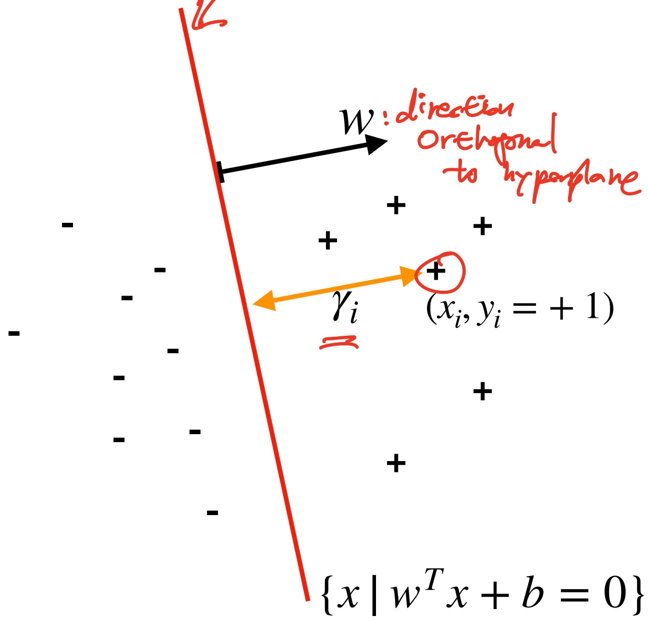
Geometric margin

- given a set of training examples $\{(x_i, y_i)\}_{i=1}^n$, $y_i \in \{-1, +1\}$
- and a linear classifier $(w, b) \in \mathbb{R}^d \times \mathbb{R}$
- such that the decision boundary is
  a separating hyperplane $\{x \mid \underbrace{b + w_1x[1] + w_2x[2] + \cdots + w_dx[d]}_{w^Tx+b} = 0\}$,

  which is the set of points that are orthogonal to w with a shift of b
- we define **functional margin** of (b, w) with respect to a training example (x_i, y_i) as the distance from the point (x_i, y_i) to the decision boundary, which is

$$\gamma_i = y_i \frac{(w^Tx_i + b)}{\|w\|_2}$$

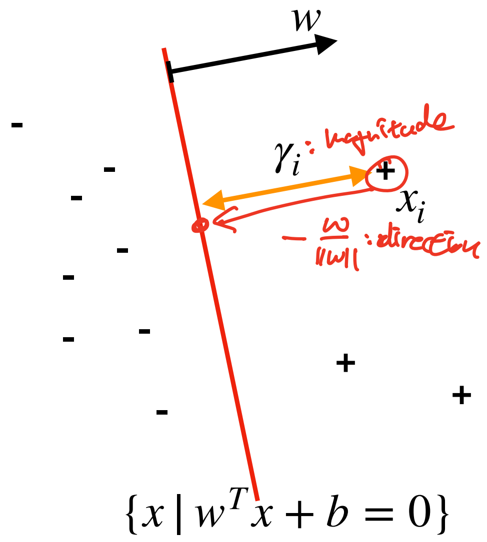
(The proof is on the next slide)



Geometric margin

- the distance γ_i from a hyperplane $\{x \mid w^T x + b = 0\}$ to a point x_i can be computed geometrically as follows
- We know that if you move from x_i in the negative direction of w by length γ_i, you arrive at the line, which can be written as

$$\left(x_i - \frac{w}{\|w\|_2}\gamma_i \right) \text{ is in } \{x \mid w^T x + b = 0\}$$



- so we can plug the point in the formula:

$$w^T\left(x_i - \frac{w}{\|w\|_2}\gamma_i \right) + b \;=\; 0$$

which is

$$w^T x_i - \frac{\|w\|_{\cancel{2}}^{\cancel{2}}}{\cancel{\|w\|_2}}\gamma_i + b \;=\; 0$$

and hence

$$\gamma_i \;=\; \frac{w^T x_i + b}{\|w\|_2},$$

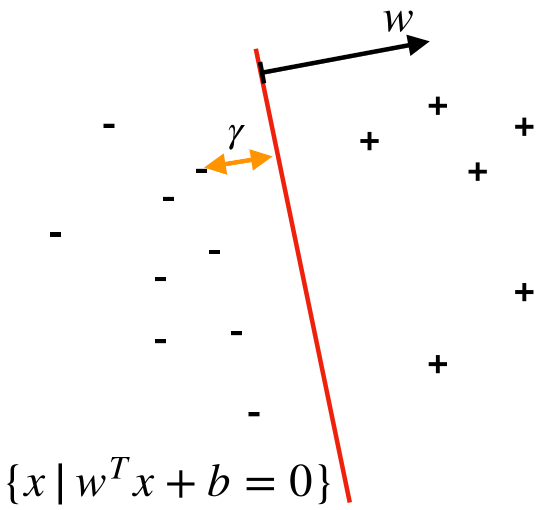
and we multiply it by y_i so that for negative samples we use the opposite direction of $-w$ instead of w

Geometric margin

- the **margin** with respect to a set is defined as

$$\gamma = \min_{i=1}^{n} \gamma_i$$

- among all linear classifiers, we would like to find one that has the **maximum margin**



Maximum margin classifier

- we propose the following optimization problem:

*Series of optimal solutions.
$(w^*, b^*, r^*) \rightarrow \{(c\cdot w^*, c\cdot b^*, r^*)\}_{c\in\mathbb{R}^+}$

$$(*) \quad \text{maximize}_{(w\in\mathbb{R}^d, b\in\mathbb{R}), \gamma\in\mathbb{R}} \quad \gamma$$

(maximize the margin)

$$\text{subject to} \quad \frac{2x \rightarrow y_i(w^T x_i + b)}{2x \rightarrow \|w\|_2} \geq \gamma \quad \text{for all } i \in \{1,\ldots,n\}$$

(s.t. γ is a lower bound on the margin)

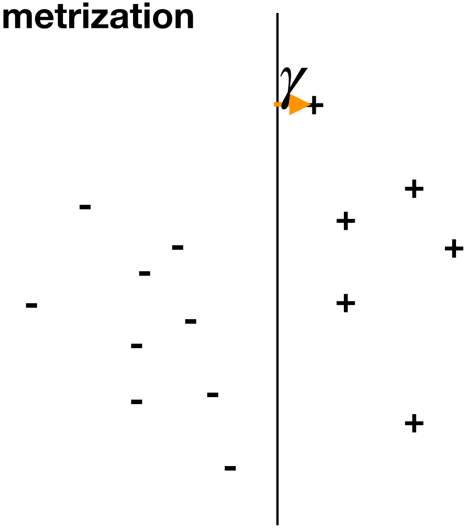
- if we fix (w, b), the optimal solution of the optimization is the margin
- together with (w, b), this finds the classifier with the maximum margin
- note that this problem is **scale invariant** in (w, b), i.e. changing a (w, b) to $(2w, 2b)$ does not change either the feasibility or the objective value, hence the following reparametrization is valid
- the above optimization looks difficult, so we transform it using **reparametrization**

*direction of w matters.
and $\|w\|$ can be set to anything

$$(**) \quad \text{maximize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \gamma\in\mathbb{R}} \quad \gamma$$

$$\text{subject to} \quad \frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \gamma \quad \text{for all } i \in \{1,\ldots,n\}$$

$$\boxed{\|w\|_2 = \frac{1}{\gamma}}$$



- Because of scale invariance, the optimal solution does not change, as the solutions to the original problem did not depend on $\|w\|_2$, and only depends on the direction of w

- (**) maximize$_{w\in\mathbb{R}^d, b\in\mathbb{R}, \gamma\in\mathbb{R}}$ γ

  subject to $$\frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \gamma \text{ for all } i \in \{1,\ldots,n\}$$
  $$\|w\|_2 = \frac{1}{\gamma}$$

- the above optimization still looks difficult, but can be transformed into $\gamma = \frac{1}{\|w\|}$

  ~~maximize~~ min$_{w\in\mathbb{R}^d, b\in\mathbb{R}}$ $\frac{1}{\|w\|_2}$ **(maximize the margin)**

  subject to $$\frac{y_i(w^T x_i + b)}{\cancel{\|w\|_2}} \geq \frac{1}{\cancel{\|w\|_2}} \text{ for all } i \in \{1,\ldots,n\}$$ **(now $\frac{1}{\|w\|_2}$ plays the role of a lower bound on the margin)**

  which simplifies to

  (***) minimize$_{w\in\mathbb{R}^d, b\in\mathbb{R}}$ $\|w\|_2^2$ ← Quadratic objective

  subject to $y_i(w^T x_i + b) \geq 1$ for all $i \in \{1,\ldots,n\}$ ← Linear Constraint

- this is a **quadratic program with linear constraints**, which can be easily solved

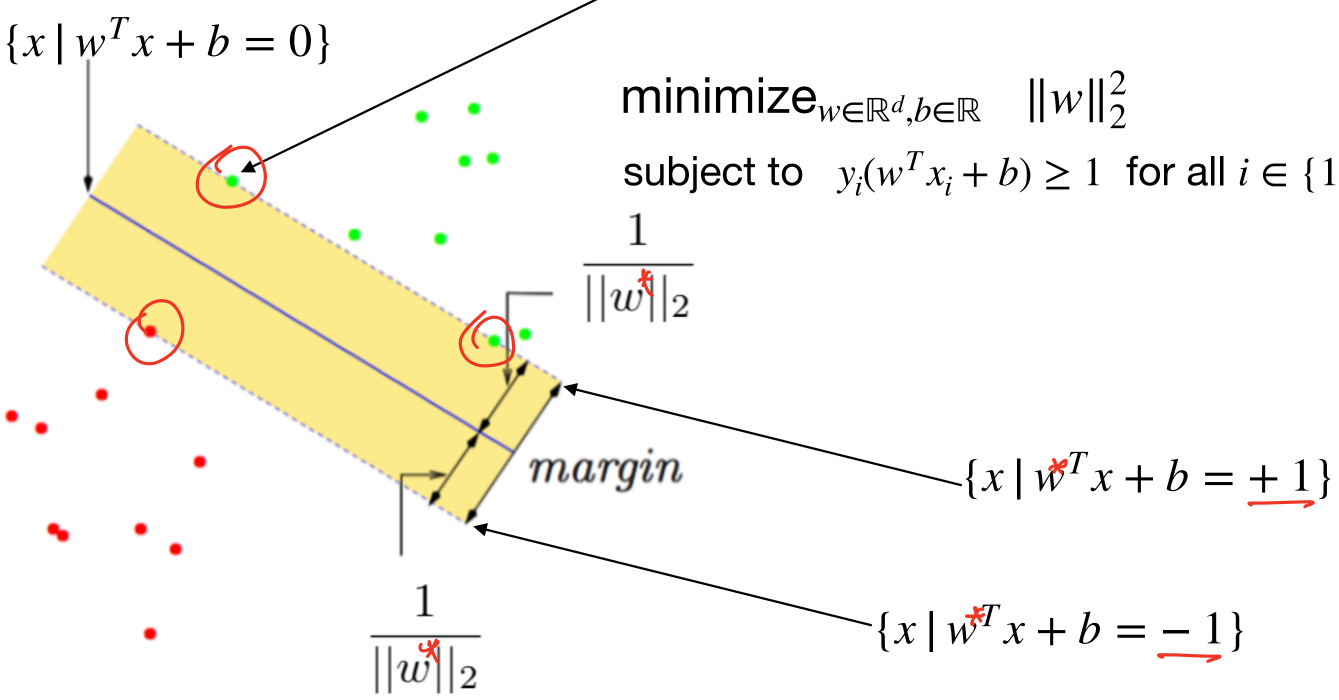
- once the optimal solution is found, the margin of that classifier (w, b) is $\frac{1}{\|w^*\|_2}$

What if the data is not separable?

- we cheated a little in the sense that the reparametrization of $\|w\|_2 = \frac{1}{\gamma}$ is possible only if the the margins are positive,
  i.e. the data is linearly separable with a positive margin
- otherwise, there is no feasible solution to (***)
- the examples at the margin are called **support vectors**

$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \|w\|_2^2$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 \quad \text{for all } i \in \{1, \ldots, n\}$$



$\{x \mid w^T x + b = 0\}$

$\frac{1}{||w^*||_2}$

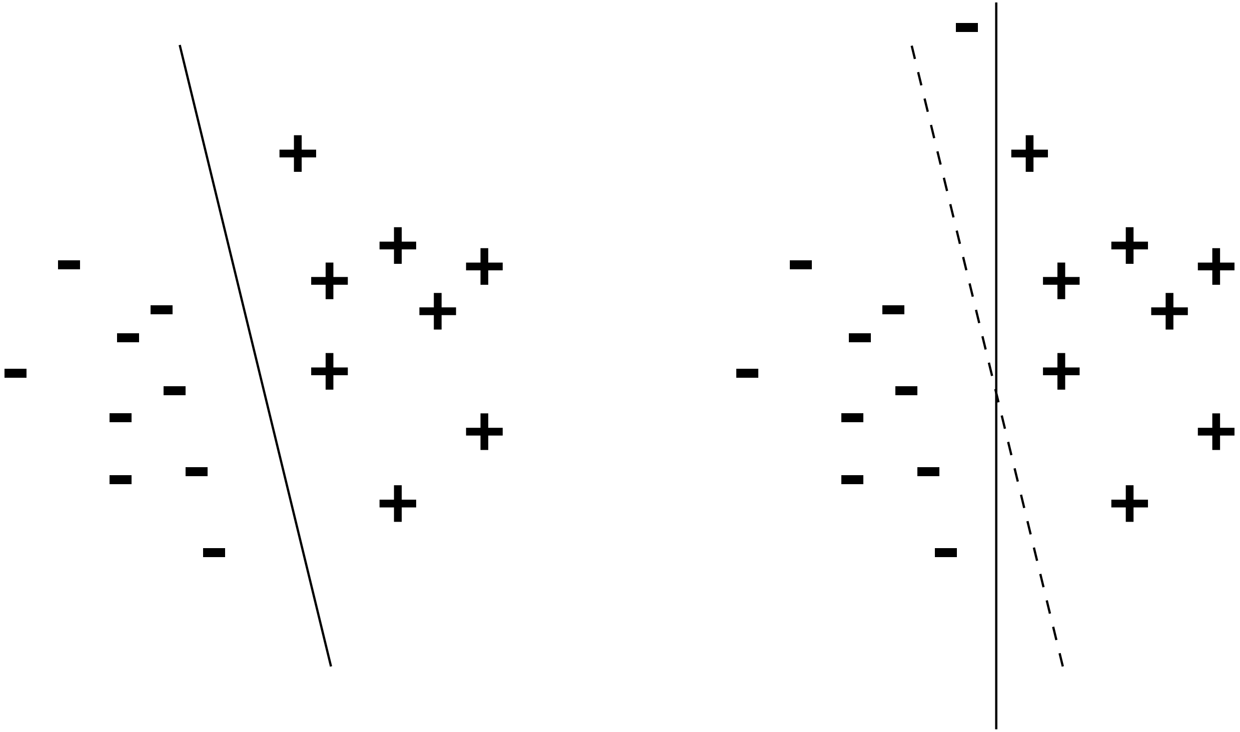
$\{x \mid w^{*T} x + b = \underline{+1}\}$

$\{x \mid w^{*T} x + b = \underline{-1}\}$

$\frac{1}{||w^*||_2}$

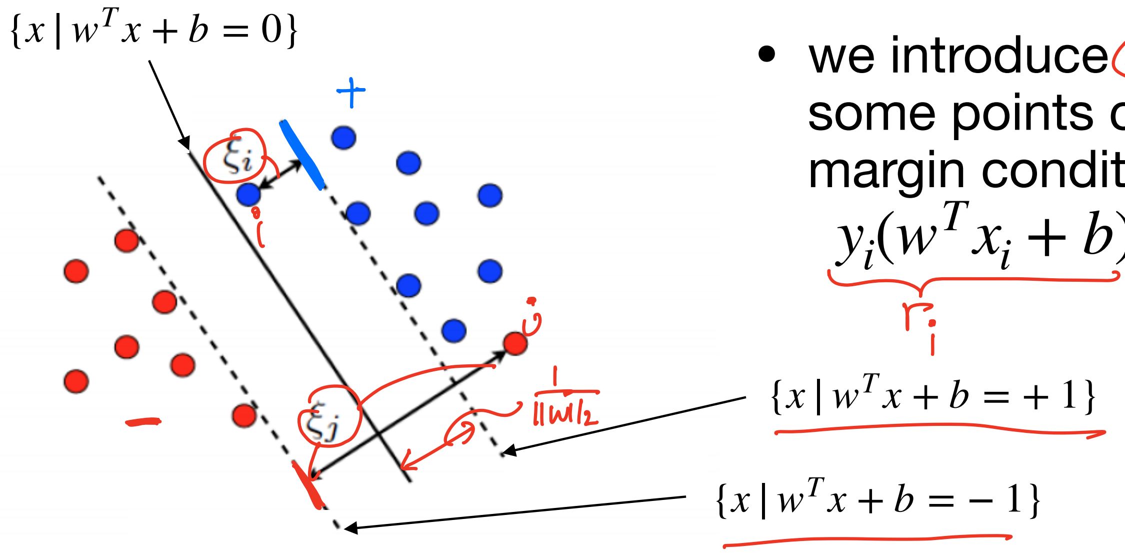
Two issues

- max-margin formulation we proposed is sensitive to outliers



- it does not generalize to non-separable datasets

What if the data is not separable?



- we introduce slack so that some points can violate the margin condition

$$\underbrace{y_i(w^T x_i + b)}_{r_i} \geq 1 \boxed{- \xi_i}$$

- this gives a new optimization problem with some positive constant $c \in \mathbb{R}$

$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \quad \|w\|_2^2 + \underbrace{c \sum_{i=1}^{n} \xi_i}\; = c\,\|\xi\|_{L1}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 - \xi_i \quad \text{for all } i \in \{1,\ldots,n\}$$ ← relaxed with slack

$$\xi_i \geq 0 \quad \text{for all } i \in \{1,\ldots,n\}$$

the (re-scaled) margin (for each sample) is allowed to be less than one,
but you pay $c\xi_i$ in the cost, and c balances the two goals:
maximizing the margin for most examples vs. having small number of violations

Support Vector Machine (SVM)

- for the optimization problem

$$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \|w\|_2^2 + c\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad y_i(w^Tx_i + b) \geq 1 - \xi_i \qquad \text{for all } i \in \{1,\ldots,n\}$$

$$\xi_i \geq 0 \qquad \text{for all } i \in \{1,\ldots,n\}$$

notice that at optimal solution, ξ_i's satisfy

- $\xi_i = 0$ if margin is big enough $y_i(w^Tx_i + b) \geq 1$, or
- $\xi_i = 1 - y_i(w^Tx_i + b)$, if the example is within the margin $y_i(w^Tx_i + b) < 1$

- so one can write
  - $\xi_i = \max\{0, 1 - y_i(w^Tx_i + b)\}$, which gives

$$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{c}\|w\|_2^2 + \sum_{i=1}^{n}\max\{0, 1 - y_i(w^Tx_i + b)\}$$

Sub-gradient descent for SVM

- SVM is the solution of

$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \frac{1}{c}\|w\|_2^2 + \sum_{i=1}^{n} \max\{0, 1 - y_i(w^T x_i + b)\}$$

- as it is non-differentiable, we solve it using sub-gradient descent
- which is exactly the same as gradient descent, except when we are at a non-differentiable point, we take one of the sub-gradients instead of the gradient (recall sub-gradient is a set)
- this means that we can take (a generic form derived from previous page)

$$\partial_w \ell(w^T x_i + b, y_i) \;=\; \mathbf{I}\{y_i(w^T x_i + b) \leq 1\}(-y_i x_i)$$

and apply

$$w^{(t+1)} \;\leftarrow\; w^{(t)} - \eta \left(\sum_{i=1}^{n} \mathbf{I}\{y_i((w^{(t)})^T x_i + b^{(t)}) \leq 1\}(-y_i x_i) \;+\; \frac{2}{c} w^{(t)} \right)$$

$$b^{(t+1)} \;\leftarrow\; b^{(t)} - \eta \sum_{i=1}^{n} \mathbf{I}\{y_i((w^{(t)})^T x_i + b^{(t)}) \leq 1)\}(-y_i)$$

Questions?