CPSC 540: Machine Learning

Convex Functions, Gradient Descent, Convergence Rates

Winter 2016

Admin

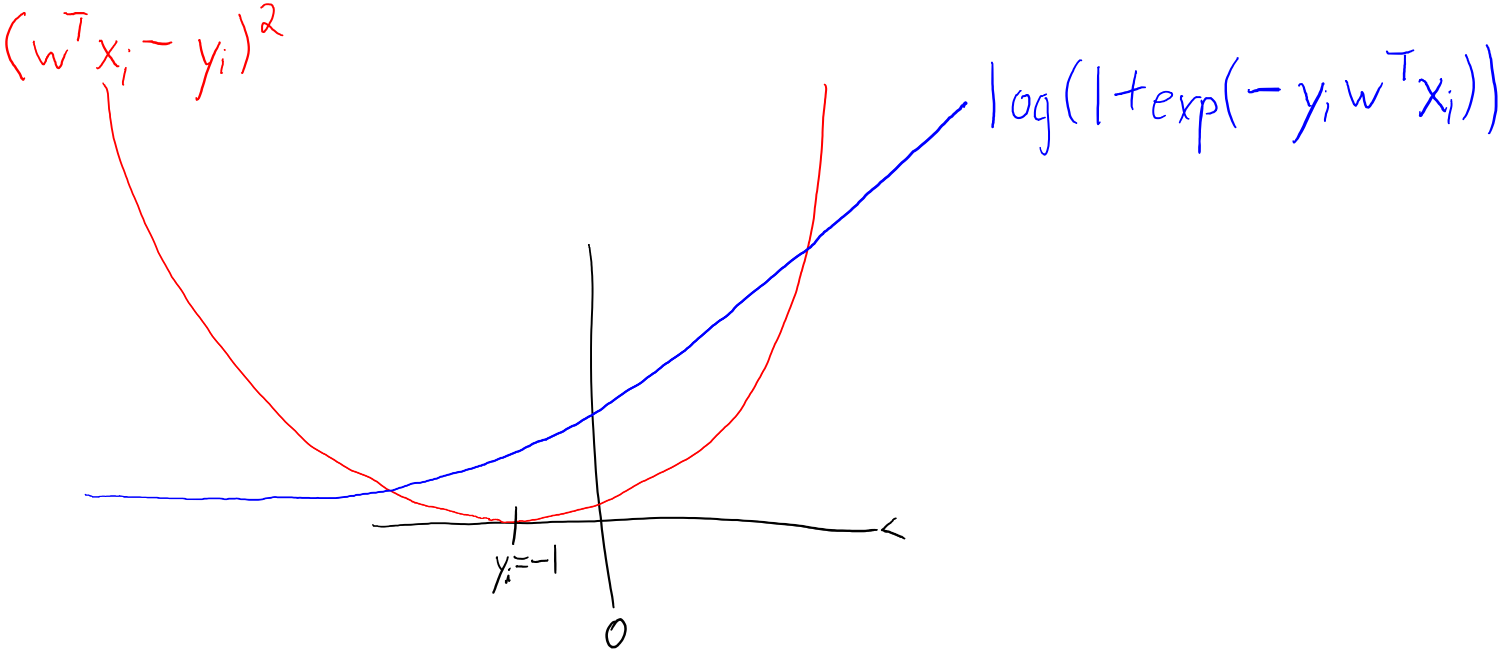
- Auditing/enrollment forms:
  - Drop-off/pickup your forms at the end of class.
    - It will be easier to argue for larger classroom if people are officially enrolled/auditing.
  - Remaining forms can be picked up at the tutorials tomorrow.
- CPSC and EECE graduate students: prereq forms due now.
- Assignment 1: due Tuesday.
  - Hand in one assignment for the group (of 1-3).
- Add/Drop deadline: Monday.
  - Last chance before you are locked in/out.

The ‘Best’ Machine Learning Model

- What is the ‘best’ machine learning model?
  - SVMs? Random forests? Deep learning?
- No free lunch theorem:
  - There is no ‘best’ model that achieves the best test error for every problem.
  - If model A works better than model B on one dataset,
    there is another dataset where model B works better.
- Asking what is the ‘best’ machine learning model is like asking which is ‘best’ among “rock”, “paper”, and “scissors”.
- Caveat of no free lunch (NFL) theorem:
  - The world is very structured, some datasets are more likely than others.
  - Model A could be better than model B on a huge variety of practical applications.
- Machine learning emphasizes models useful across applications.

Last Time: Logistic Regression

- We considered binary labels y_i, and classifying with $\text{sign}(w^Tx_i)$.
  - Squared error $(w^Tx_i – y_i)^2$ is not ideal: penalizes model for “too right”.
  - Minimizing number of errors is also not ideal: NP-hard.
  - Tractable upper bounds are hinge loss and logistic loss.



Last Time: Maximum Likelihood and MAP

- Minimizing a loss function often equivalent to maximum likelihood.
  - For example, least squares is equivalent to using a Gaussian likelihood:

$$\text{If } y_i \sim \mathcal{N}(w^T x_i, \sigma^2), \quad \underset{w \in \mathbb{R}^d}{\text{argmax}} \underbrace{p(y \mid w, X)}_{\text{likelihood}} \Leftrightarrow \underset{w \in \mathbb{R}^d}{\text{argmin}} \frac{1}{2}\|Xw - y\|^2$$

- With a regularizer, often equivalent to MAP estimation:
  - For example, L2-regularization is equivalent to using a Gaussian prior:

$$\text{If } y_i \sim \mathcal{N}(w^T x_i, \sigma^2), \; w_j \sim \mathcal{N}(0, \lambda^{-1}) \quad \underset{w \in \mathbb{R}^d}{\text{argmax}} \underbrace{p(w \mid y, X)}_{\text{posterior}} \Leftrightarrow \underset{w \in \mathbb{R}^d}{\text{argmax}} \underbrace{p(y \mid w, X)}_{\text{likelihood}} \underbrace{p(w)}_{\text{prior}} \Leftrightarrow \underset{w \in \mathbb{R}^d}{\text{argmin}} \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$$

- Gives probabilistic perspective on regularization: prior on 'w'.

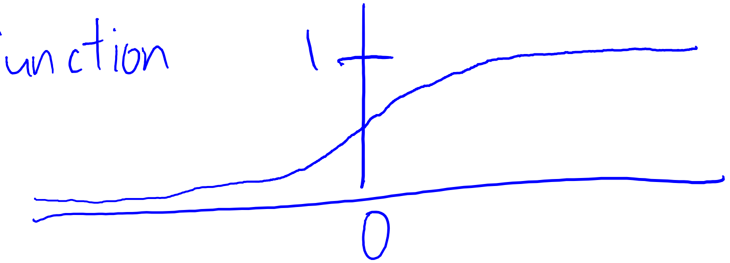
Last Time: Maximum Likelihood and MAP

- Logistic loss is equivalent to maximum likelihood logistic regression:

$$p(y_i \mid w, x_i) = \frac{1}{1+\exp(-y_i w^T x_i)}$$

Based on "sigmoid" function

$$\sigma(z) = \frac{1}{1+\exp(-z)}$$



- L2-regularized logistic is MAP estimate with Gaussian prior:

$$p(y_i \mid w, x_i) = \frac{1}{1+\exp(-y_i w^T x_i)} \qquad p(w_j \mid \lambda) \propto \exp\left(-\frac{\lambda}{2} w_j^2\right)$$

- Advantage of likelihood/MAP perspective:
  - Allows us to define objectives for other distributions of y_i.

Multi-Class Logistic Regression

- Supposed y_i takes values from an unordered discrete set of classes.

$x_i \in \mathbb{R}^d$ $\qquad$ $y_i \in \{1, 2, 3, 4, \ldots, k\}$

 $\longrightarrow$ "itchy" (class 1)

 $\longrightarrow$ "scratchy" (class 2)

 $\longrightarrow$ "poochie" (class 3)

- Standard model:
  - Use a 'd'-dimensional weight vector 'w_c' for each class 'c'.
  - Try to make inner-product $w_c^T x_i$ big when 'c' is the true label 'y_i'.
  - Classify by finding largest inner-product: $\hat{y}_i = \underset{c}{\operatorname{argmax}} \{ w_c^T x_i \}$

Multi-Class Logistic Regression

We have a parameter matrix $W = \underbrace{\left.\begin{bmatrix} | & | & | & & | \\ w_1 & w_2 & w_3 & \cdots & w_k \\ | & | & | & & | \end{bmatrix}\right\} d}_{k}$

To make a prediction, compute $W^T x_i = \begin{bmatrix} w_1^T x_i \\ w_2^T x_i \\ \vdots \\ w_k^T x_i \end{bmatrix}$ and compute maximum.

predict "2"

We want a loss function that will make $w_c^T x_i$ big when c is the true label y_i and will otherwise make $w_c^T x_i$ small.

We can define probability using softmax function:

$$p(y_i = c \mid W, x_i) = \frac{\exp(w_c^T x_i)}{\sum_{c'=1}^{k} \exp(w_{c'}^T x_i)} \propto \exp(w_c^T x_i)$$

If $k=3$:

$$p(y_i \mid W, x_i) = \frac{\exp(w_{y_i}^T x_i)}{\exp(w_1^T x_i) + \exp(w_2^T x_i) + \exp(w_3^T x_i)}$$

$\uparrow$ true

To fit model, use

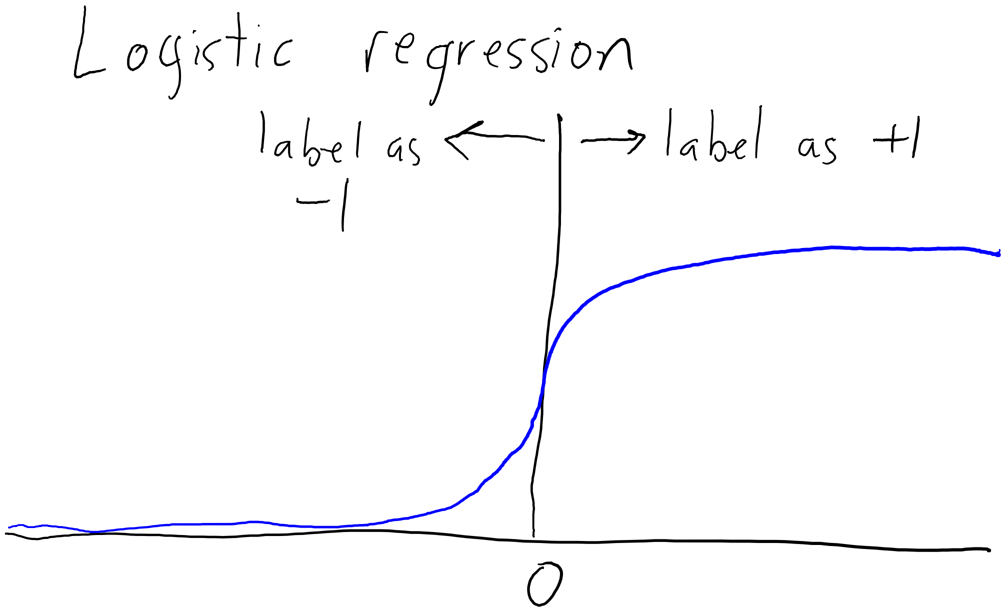
$$-\log p(y_i \mid W, x_i) = -w_{y_i}^T x_i + \log\left(\sum_{c'=1}^{k} \exp(w_{c'}^T x_i)\right)$$

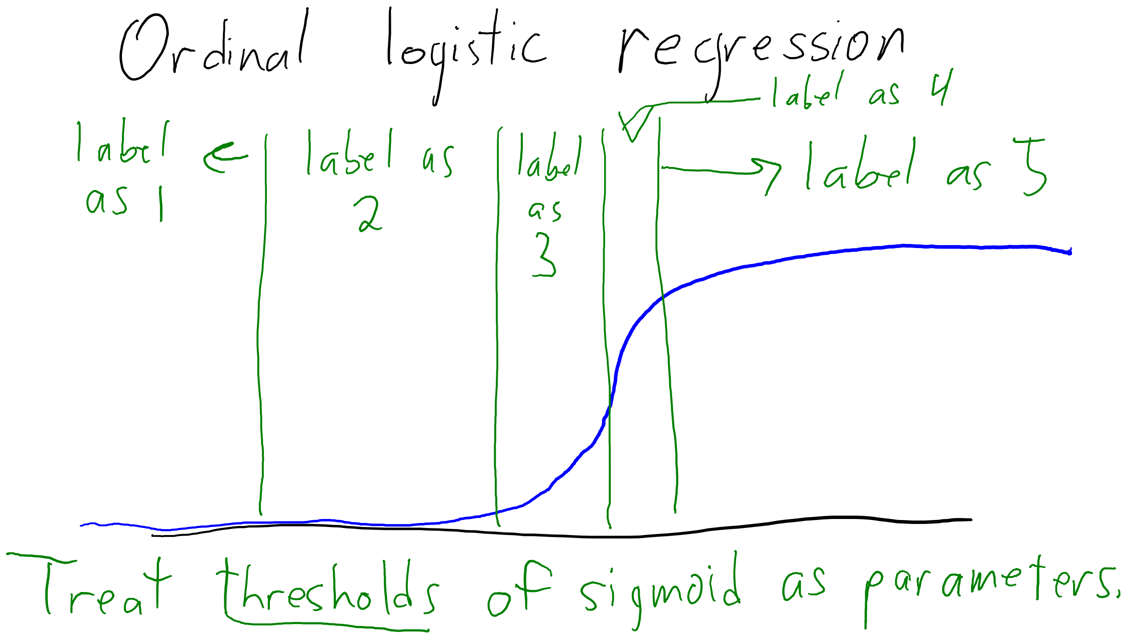
Generalizes sigmoid:

- special case of $k=2$ and $w_2 = 0$

Ordinal Labels

- Ordinal data: categorical data where the order matters:
  - Rating hotels as {'1 star', '2 stars', '3 stars', '4 stars', '5 stars'}.
  - Softmax would ignore order.
- 'Proportional odds' or 'ordinal logistic regression':



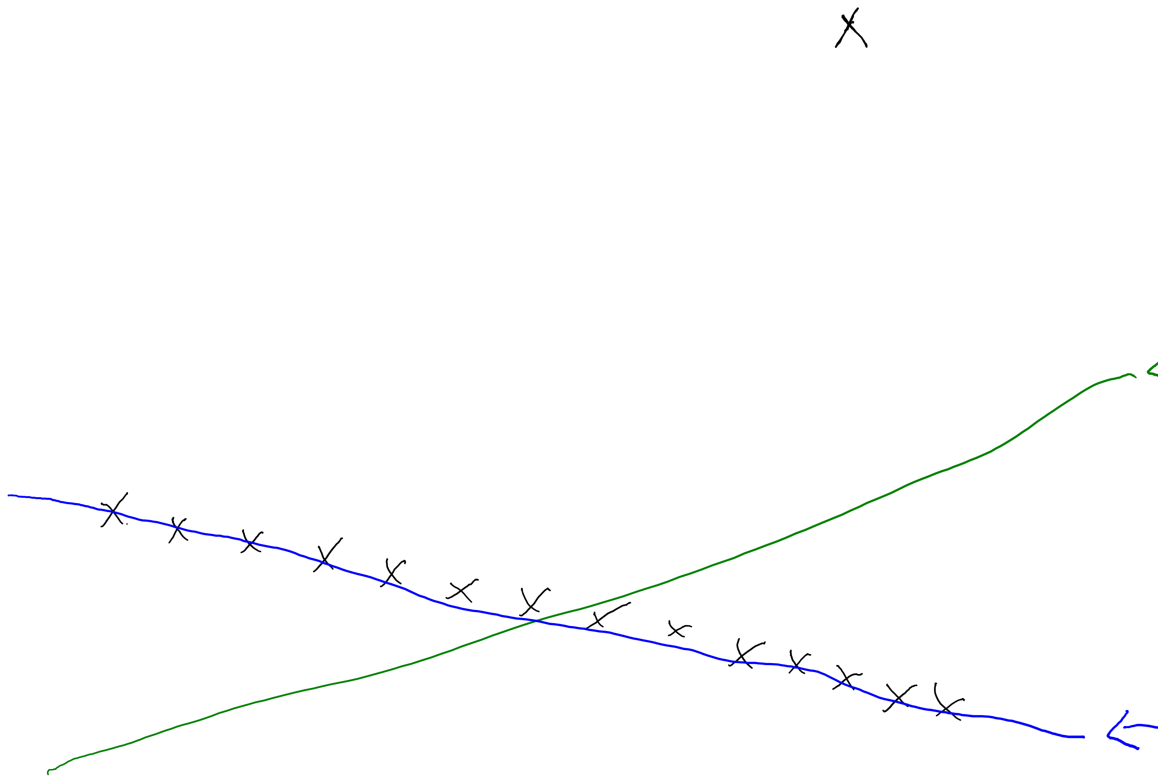


Count Labels

- Count data: predict the number of times something happens.
  - For example, y_i = "602" Facebok likes.
- Softmax/ordinal require finite number of categories.
- We probably don't want separate parameter for '654' and '655'.
- Poisson regression: use probability from Poisson count distribution.
  - Many variations exist.

Last Time: Robust Regression

- We said that squared error is sensitive to outliers:
  - Absolute error is less sensitive: can be solved as a linear program.



minimizes squared error

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

minimizes absolute error

$$\sum_{i=1}^{n} |w^T x_i - y_i|$$

‘Brittle’ Regression

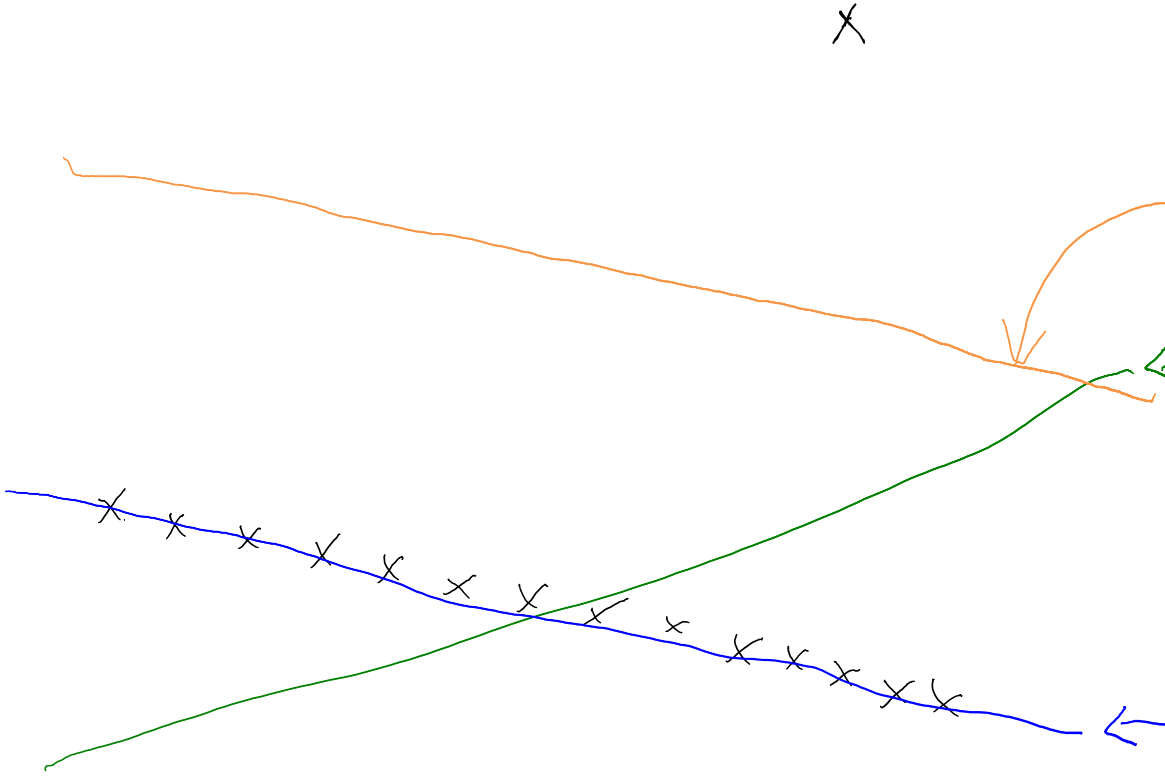
- What if you really care about getting the outliers right?
  - You want best performance on worst training example.
  - For example, if in worst case the plane can crash.
- In this case you can use something like the infinity-norm:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; \|Xw - y\|_\infty \qquad\qquad \|z\|_\infty = \max_i \{|z_i|\}$$

- Very sensitive to outliers (brittle), but worst case will be better.

Last Time: Robust Regression

- We said that squared error is sensitive to outliers:
  - Absolute error is less sensitive: can be solved as a linear program.
  - Maximum error is more sensitive: can also be solved as linear program.



minimizes maximum error:

$\max_i \{|w^Tx_i - y_i|\}$

minimizes squared error

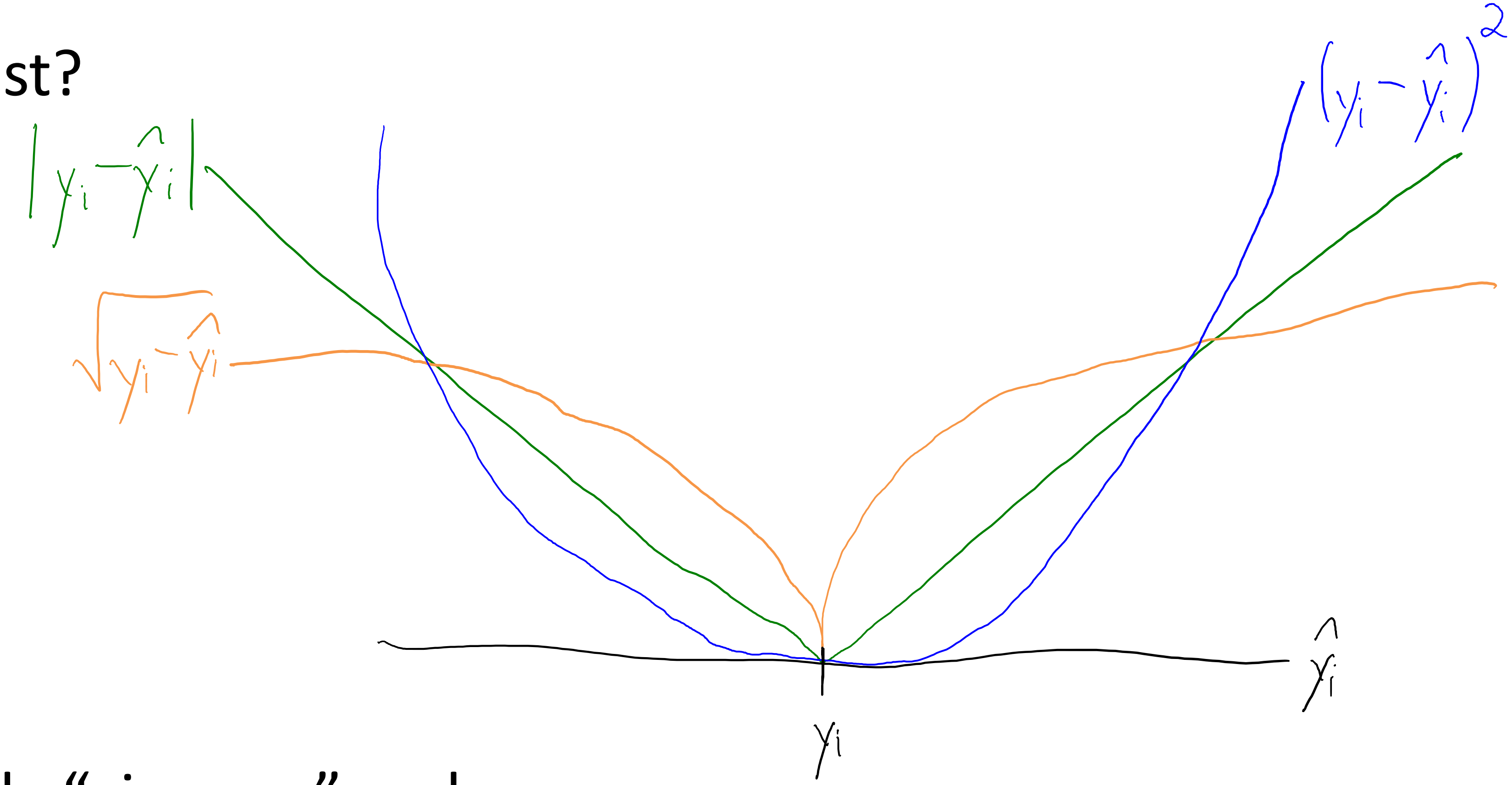
$\sum_{i=1}^{n} (w^Tx_i - y_i)^2$

minimizes absolute error

$\sum_{i=1}^{n} |w^Tx_i - y_i|$

Very Robust Regression

- Can we be more robust?



- Very robust: eventually "gives up" on large errors.
- But finding optimal 'w' is NP-hard.
  - Absolute value is the most robust that is not NP-hard.

Course Roadmap

- Topics we discussed in part 1:
  - Linear models: change of basis, regularization, loss functions.
  - Basics of learning theory: Training vs. test error, bias-variance, fundamental trade-off, no free lunch.
  - Probabilistic learning principles: Maximum likelihood, MAP estimation.
- Part 2: Large-scale machine learning.
  - Why are SVMs/logistic easy while minimizing number of errors is hard?
  - How do we fit these models to huge datasets?

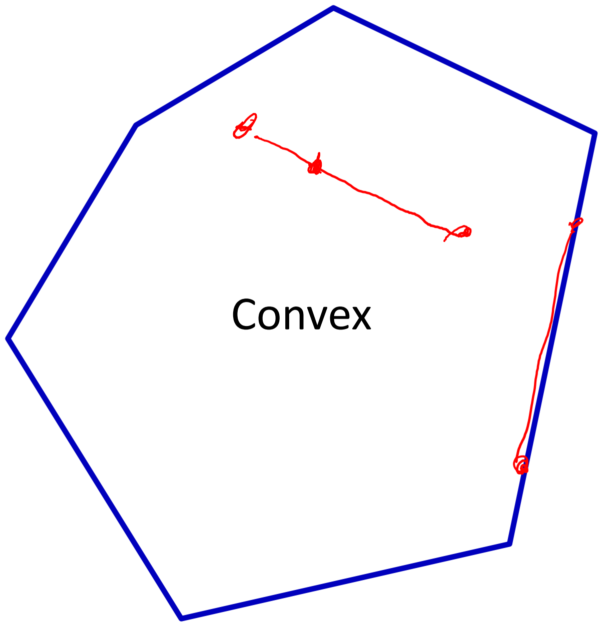
Convex Functions

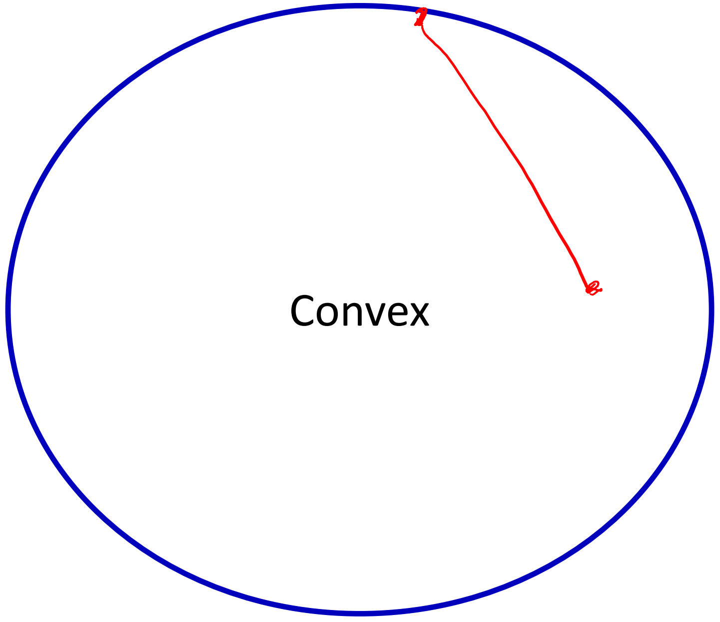
- We are first going to discuss convex functions:
  - Minimizing convex functions is usually easy.
  - Minimizing non-convex functions is usually hard.
- The ‘easy’ problems we have discussed are convex:
  - Least squares, robust regression, logistic regression, support vector machines, multi-class logistic, brittle regression, Poisson regression.
  - All of the above with L2-regularization.
- The ‘hard’ problems we have discussed are non-convex:
  - 0-1 loss, “very robust” regression.

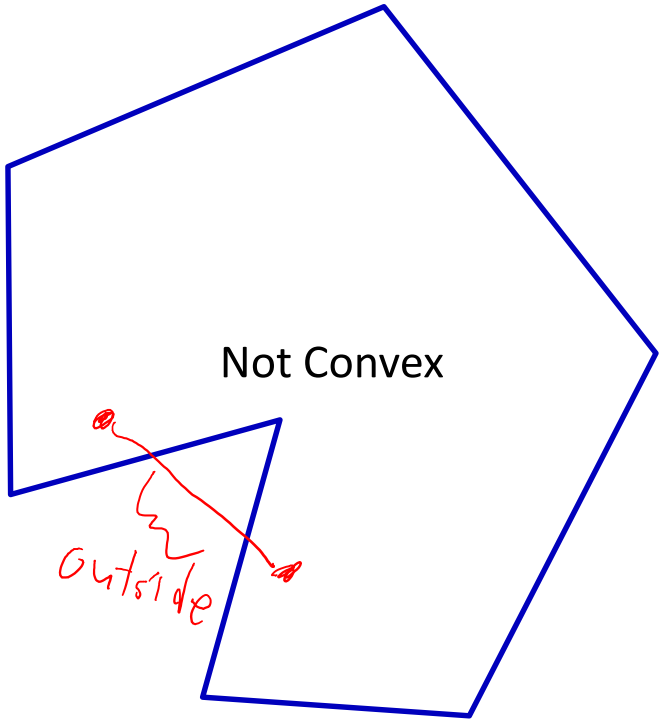
Convex Sets

- First we need to define a convex set:
  - A set is convex if the line between any two points stays in the set.

For all $x \in C$ and $y \in C$ we have $\theta x + (1-\theta)y \in C$ for $0 \leq \theta \leq 1$







Convex Sets

- Examples:

Real-space: $\mathbb{R}^d$

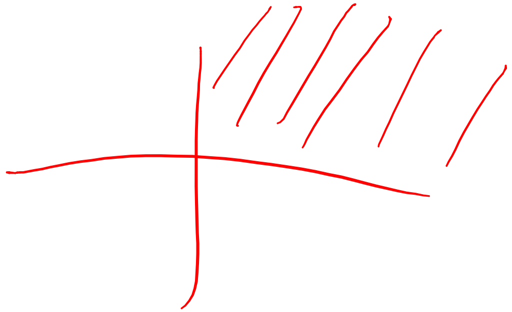
Positive orthant $\mathbb{R}^d_+$: $\{x \mid x \geq 0\}$

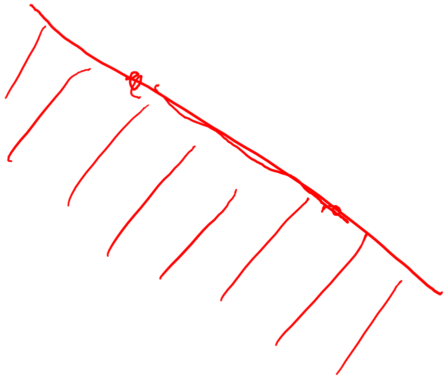
Hyper-plane: $\{x \mid a^T x = b\}$

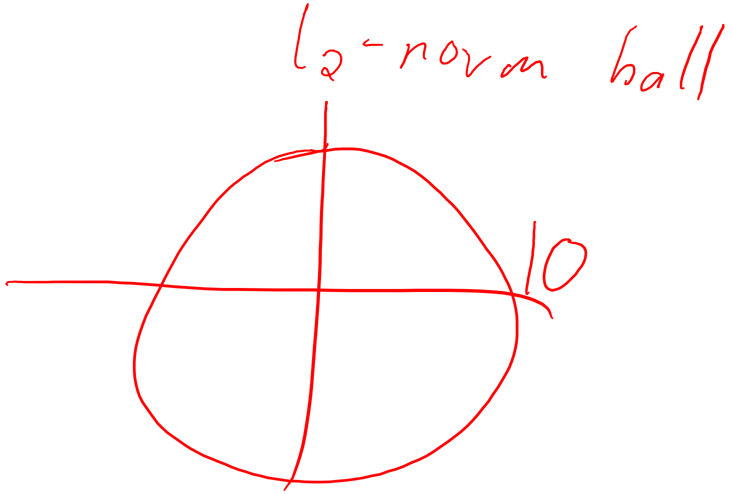
Half-space: $\{x \mid a^T x \leq b\}$

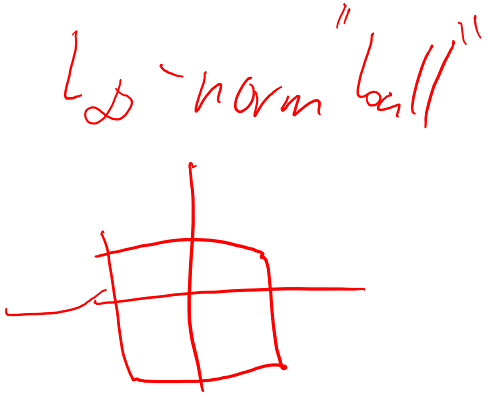
Norm-ball: $\{x \mid \|x\| \leq r\}$

Norm-cone: $\{(x, r) \mid \|x\| \leq r\}$









Showing a Set is Convex

How to prove a set is convex?

- One way: choose two generic x and y in the set, show that generic z between them is also in the set.
- Another way: show that set is intersection of sets that you know are convex.

E.g. if $C = \{x \mid a^T x = b\}$

then for $x \in C$ and $y \in C$ and $0 \leq \theta \leq 1$

we have $a^T(\theta x + (1-\theta)y)$

$$= \theta(a^T x) + (1-\theta)(a^T y)$$

$$= \theta b + (1-\theta) b = b$$

E.g., if $C = \{x \mid \|x\| \leq 10\}$

then for $x \in C$ and $y \in C$ and $0 \leq \theta \leq 1$

we have $\|\theta x + (1-\theta)y\|$

$$\leq \|\theta x\| + \|(1-\theta)y\| \quad \text{(triangle inequality)}$$

$$= |\theta| \cdot \|x\| + |1-\theta| \cdot \|y\| \quad \text{(homogeneity)}$$

$$= \theta\|x\| + (1-\theta)\|y\| \quad \theta \geq 0$$

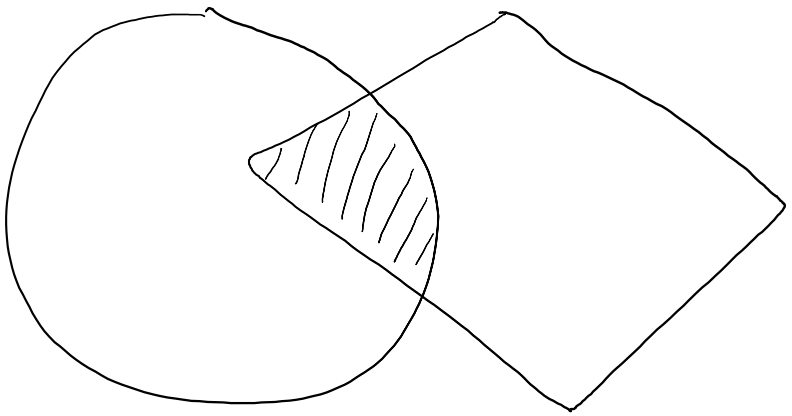
$$\leq \theta \max\{\|x\|, \|y\|\} + (1-\theta)\max\{\|x\|, \|y\|\}$$

$a \leq \max\{a, b\}$

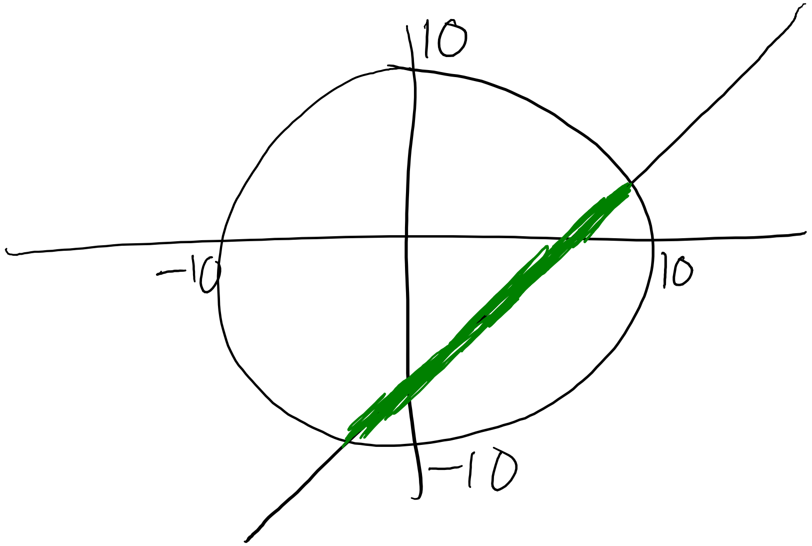
$$= \max\{\|x\|, \|y\|\} \leq 10$$

Intersection of Convex Sets

- Intersection of convex sets is convex:



E.g., $\{x \mid a^T x = b\} \cap \{x \mid \|x\| \leq 10\}$
is a convex set.



For example, x satisfying linear program constraints are a convex set:

$$Ax \leq b$$

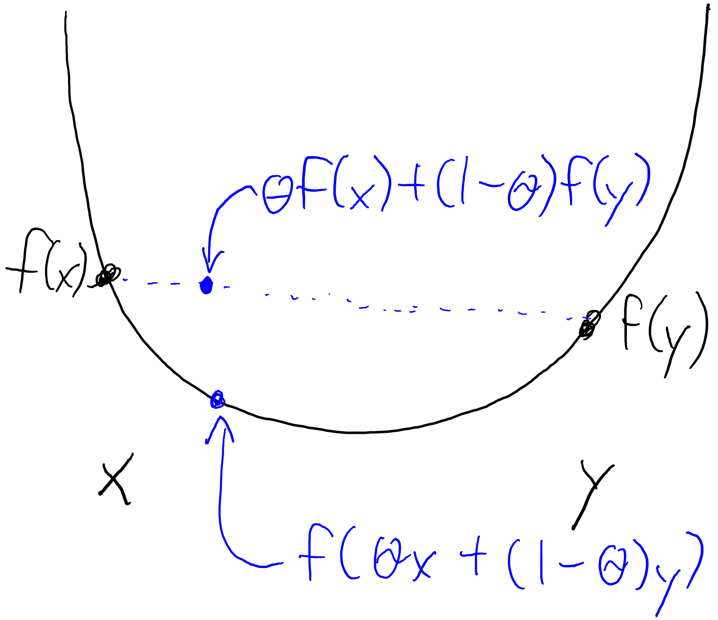
$$A_{eq} x = b_{eq}$$

$$LB \leq x \leq UB$$

Convex Functions

- A function 'f' is convex if:
  1. The domain of 'f' is a convex set.
  2. The function is always below 'chord' between two points.

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y) \quad \text{for all } x \in C, y \in C, \text{ and } 0 \leq \theta \leq 1$$



Implication: all local minima are global minima.

We can minimize a convex function by finding any stationary point.

Convex Functions

- Examples:

Quadratic functions: $f(x) = ax^2 + bx + c$, $a > 0$.

Linear functions $f(x) = a^T x + b$

Exponential: $f(x) = \exp(ax)$

Negative logarithm: $f(x) = -\log(x)$

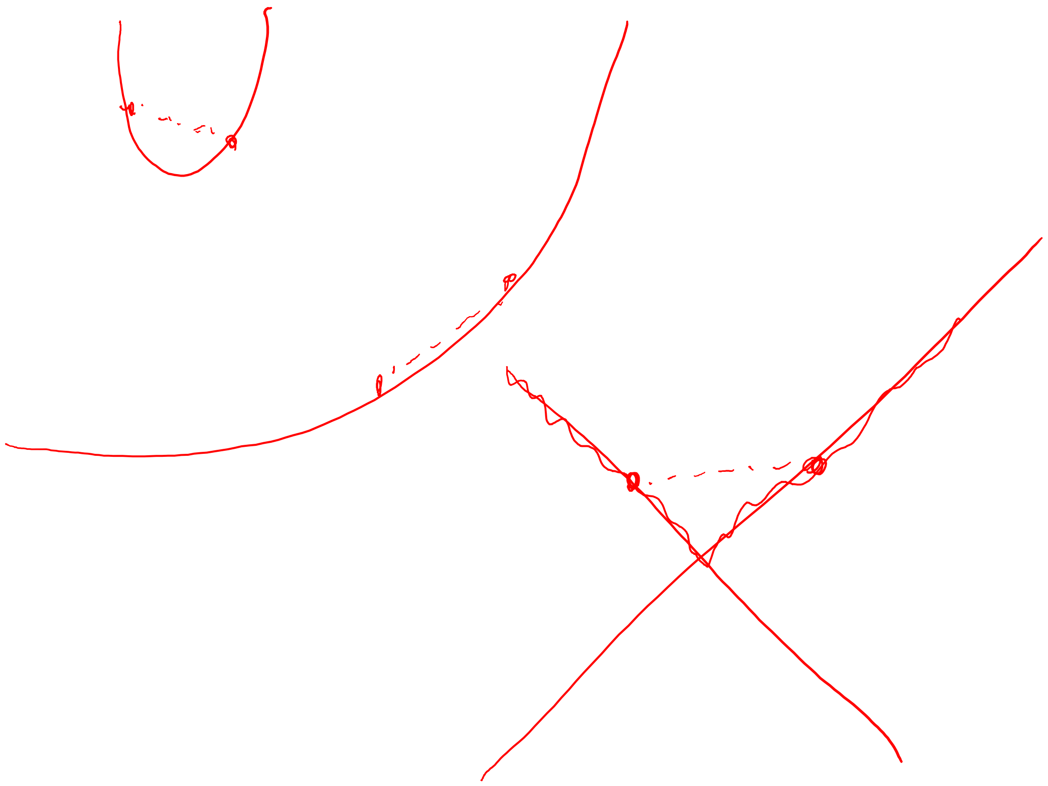
Absolute value: $f(x) = |x|$

Max function: $f(x) = \max_i \{x_i\}$

Negative entropy: $f(x) = x\log(x)$, $x > 0$

Logistic loss: $f(x) = \log(1 + \exp(-z))$

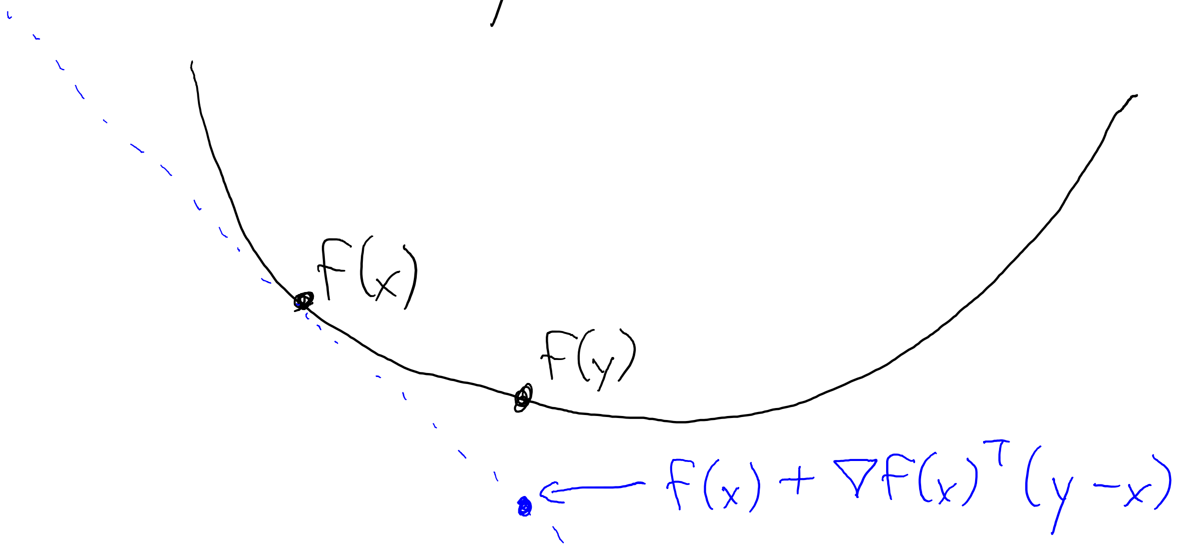
Log-sum-exp: $f(x) = \log(\sum_{i=1}^{d} \exp(x_i))$



Differentiable Convex Functions

- A *differentiable* 'f' is convex iff 'f' is always above tangent:

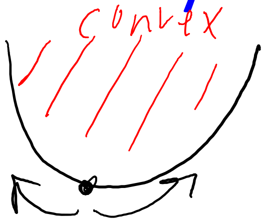
$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \quad \text{for all } x \in C \text{ and } y \in C$$



If $\nabla f(x) = 0$, this implies $f(y) \geq f(x)$ for <u>all</u> y so x is a global minimizer.

Twice-Differentiable Convex Functions

- A *twice-differentiable* 'f' is convex iff it's curved upwards everywhere.



For one-dimensional functions, reduces to $f''(x) \geq 0$.

- usually, this is the easiest way to show a function is convex.

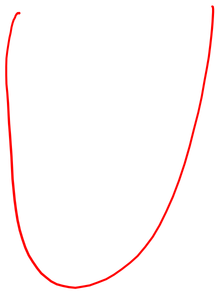
For multivariate functions, generalization is

$$\nabla^2 f(x) \succeq 0 \text{ for all } x \in C.$$

$A \succeq 0$ means A is symmetric and positive semi-definite: $y^T A y \geq 0$ for all y

Showing Functions are Convex

- Examples:



If $f(x) = x^2$

then $f'(x) = 2x$

and $f''(x) = 2$.

Since $2 \geq 0$ we've shown x^2 is convex.

If $f(x) = \frac{1}{2}x^T A x + b^T x + c$ with $A \succeq 0$

then $\nabla f(x) = Ax + b$

and $\nabla^2 f(x) = A$

Since $\nabla^2 f(x) \succeq 0$ we've shown $f(x)$ is convex.

Showing Functions are Convex

- Examples:

$$f(w) = \frac{1}{2}\|Xw - y\|^2$$

$$\nabla f(w) = X^T(Xw - y)$$

$$\nabla^2 f(w) = X^T X$$

Want to show that $\nabla^2 f(w) \succeq 0$, or equivalently $y^T \nabla^2 f(w) y \geq 0$.

We have $y^T \nabla^2 f(w) y = y^T X^T X y$

$$= (Xy)^T (Xy)$$

$$= \|Xy\|^2 \geq 0.$$

So least squares is convex and setting $\nabla f(w) = 0$ gives global minimum.

Strictly-Convex Functions

- A function is strictly-convex if these inequalities strictly hold:

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y) \quad \text{for} \quad 0 < \theta < 1.$$

$$f(y) > f(x) + \nabla f(x)^T (y - x)$$

$$\nabla^2 f(x) \succ 0 \qquad (y^T \nabla^2 f(x) y > 0 \text{ for all } y \neq 0)$$

- Strict convexity implies at most one global minimum:

Points 'x' and 'y' can't both be global minima if $x \neq y$, since this would imply $f(\theta x + (1-\theta)y)$ is below global min.

- This implies L2-regularized least squares has unique solution:

for $y \neq 0$

$$y^T \nabla^2 f(w) y = y^T (X^T X + \lambda I) y = y^T X^T X y + y^T (\lambda I) y = (Xy)^T (Xy) + \lambda y^T y = \|Xy\|^2 + \lambda \|y\|^2 > 0.$$

Operations that Preserve Convexity

- There are a few operations preserve convexity.
  - Often lets us avoid calculating Hessian.
  - Often lets us prove convexity of non-smooth functions.
- If f_1 and f_2 are convex, then convexity is preserved under:
  1. Non-negative weighted sum:

     $f(x) = z_1 f_1(x) + z_2 f_2(x)$ is convex if $z_1 \geq 0$ and $z_2 \geq 0$
  2. Composition with affine function:

     $f(x) = f_1(Ax + b)$ is convex.
  3. Pointwise maximum:

     $f(x) = \max\{f_1(x), f_2(x)\}$ is convex.

Example: SVMs

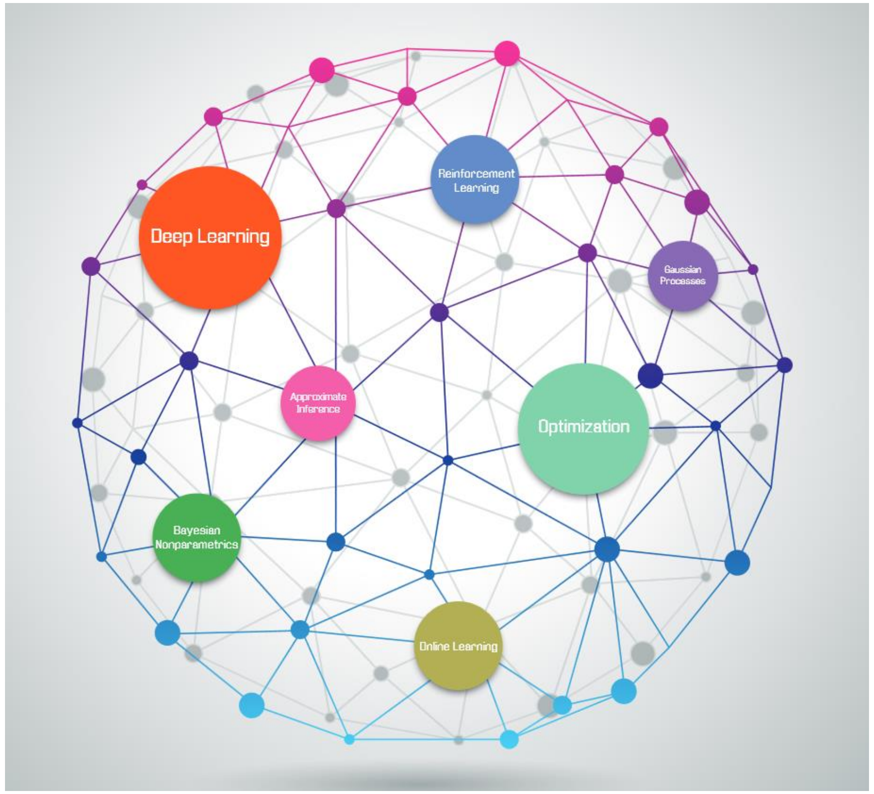
$$f(x) = \underbrace{\sum_{i=1}^{n} \underbrace{\max\{\underbrace{0}_{\substack{\text{linear} \\ \text{(convex)}}}, \underbrace{1 - y_i w^T x_i}_{\substack{\text{linear} \\ \text{(convex)}}}\}}_{\substack{\max \text{(convex)} \\ \text{so convex}}} + \underbrace{\frac{\lambda}{2}\|w\|^2}_{}}_{\text{non-negative sum of convex}}$$

$\nabla^2\left[\frac{\lambda}{2}\|w\|^2\right] = \lambda I \succeq 0$ so convex

(pause)

Current Hot Topics in Machine Learning

- Graph of most common keywords among ICML papers last year:



- Why is there so much focus on deep learning and optimization?

Why Study Optimization in CPSC 540?

- In machine learning, training is typically written as optimization:
  – Numerically optimize parameters of model, given data.
- There are some exceptions:
  1. Counting- and distance-based methods (random forests, KNN).
     - See CPSC 340.
  2. Integration-based methods (Bayesian learning).
     - Later in course.

  Although you still need to tune parameters in those models.
- But why study optimization? Can't I just use Matlab functions?
  – '\', linprog, quadprog, fmincon, CVX,…

The Effect of Big Data and Big Models

- Datasets are getting huge, we might want to train on:
  - Entire medical image databases.
  - Every webpage on the internet.
  - Every product on Amazon.
  - Every rating on Netflix.
  - All flight data in history.
- With bigger datasets, we can build bigger models:
  - This is where deep learning comes in.
  - Complicated models can address complicated problems.
- Now optimization becomes a problem because of time/memory:
  - We can' afford $O(d^2)$ memory, or an $O(d^2)$ operation.
  - Going through huge datasets 100s of times is too slow.
  - Evaluating huge models too many times is too slow.

Fitting Logistic Regression Models

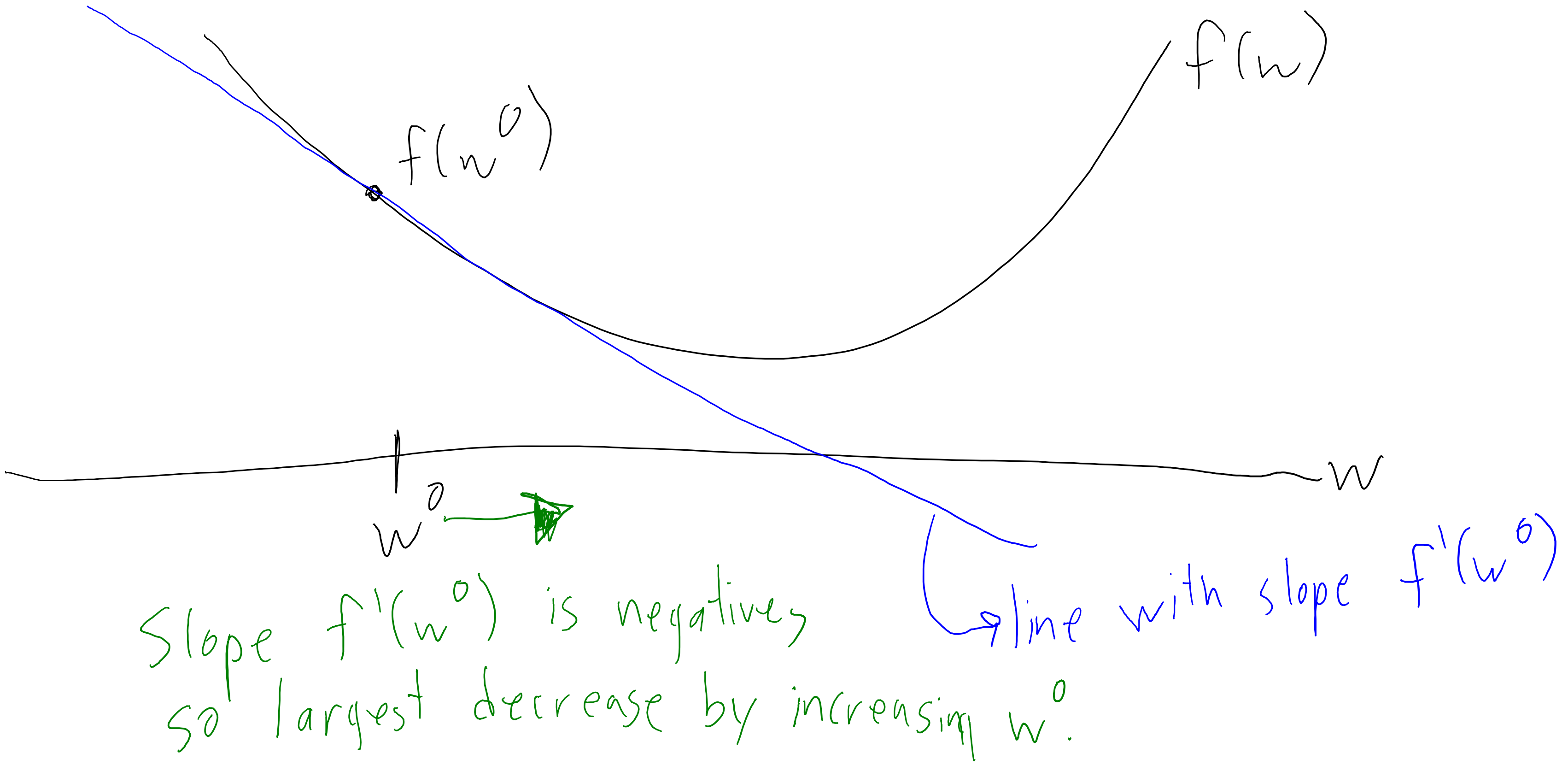
- Recall the L2-regularized logistic regression objective function:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{d} \log(1+\exp(-y_i w^T x_i)) + \frac{\lambda}{2}\|w\|^2$$

- This objective function is strictly-convex and differentiable.
- But we can't formulate as linear system or linear program.
- Nevertheless, we can efficiently solve this problem.
- There are many ways to do this, but we focus on gradient descent:
  - Iteration cost is linear in 'd' (not true of IRLS/Newton's method).
  - We can prove that we don't need too many iterations:
    - Number of iterations does not directly depend on 'd'.

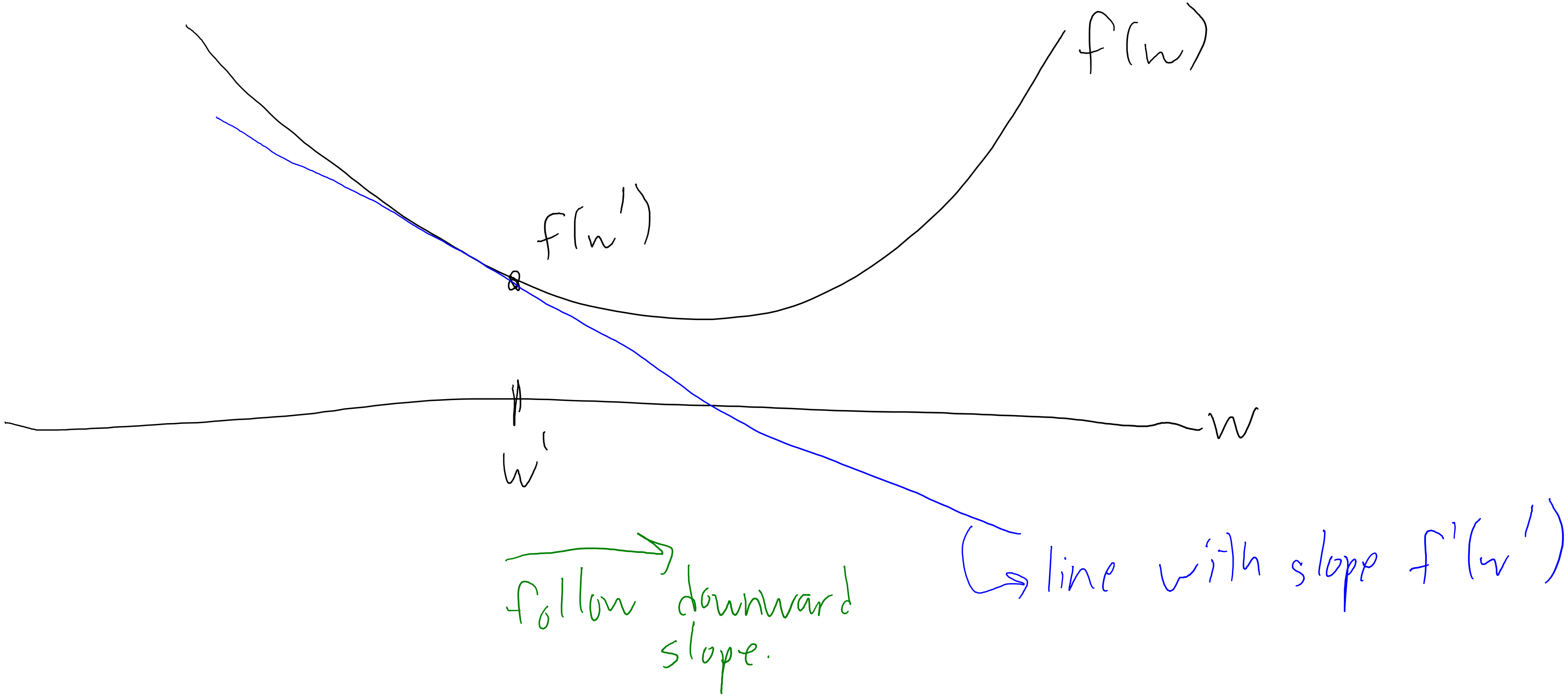
Gradient Descent

- Gradient descent is based on a simple observation:
  - Given parameters ‘w^0’, direction of largest decrease is $-\nabla f(w^0)$.



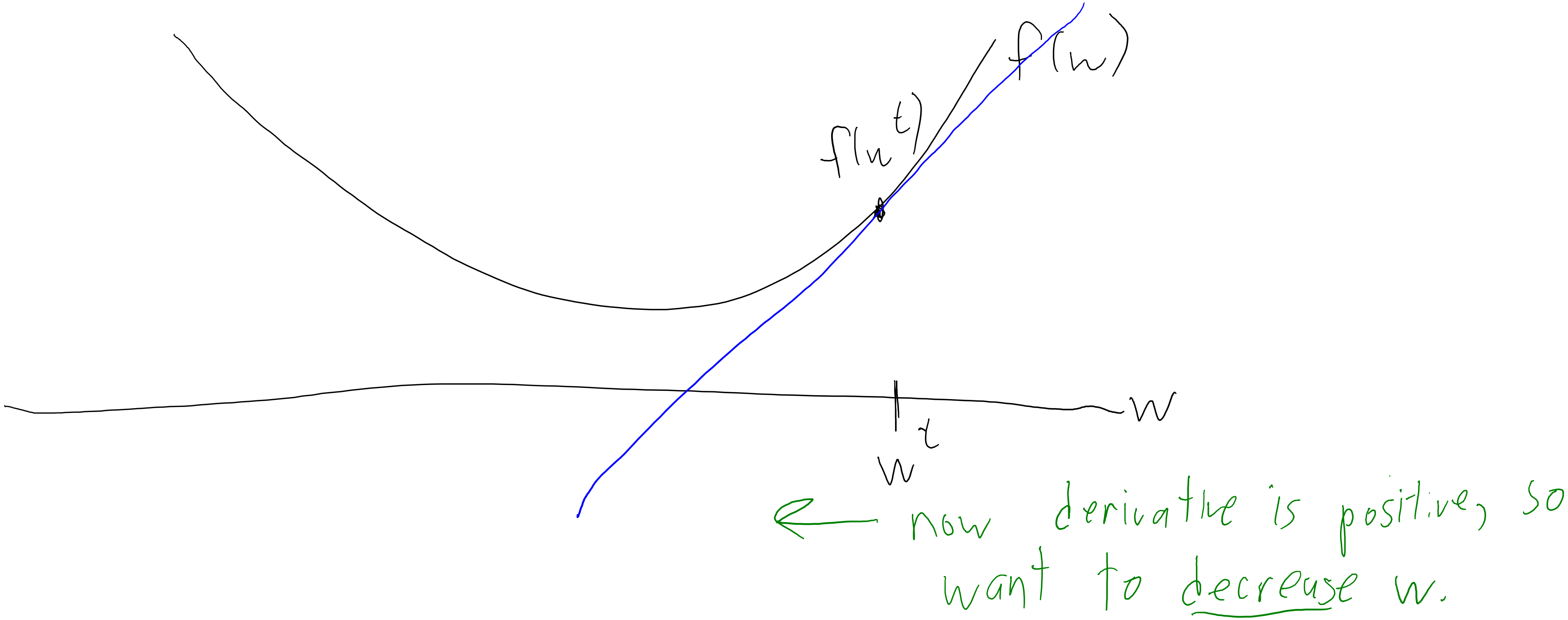
Gradient Descent

- Gradient descent is based on a simple observation:
  - Given parameters ‘w^0’, direction of largest decrease is $-\nabla f(w^0)$.



Gradient Descent

- Gradient descent is based on a simple observation:
  - Given parameters 'w^0', direction of largest decrease is $-\nabla f(w^0)$.



Gradient Descent

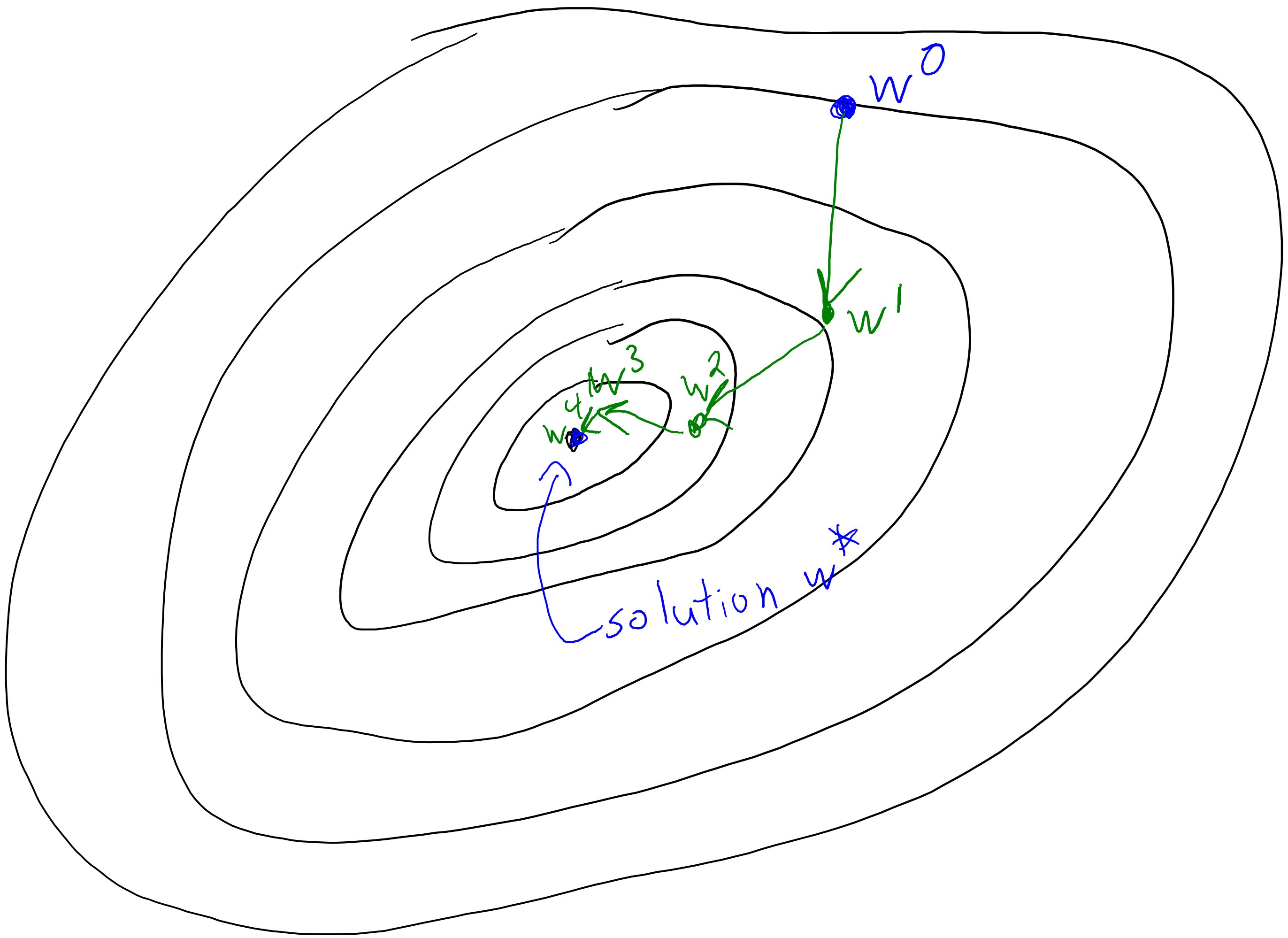
- Gradient descent is an iterative algorithm:
  - We start with some initial guess, w^0.
  - Generate new guess by moving in the negative gradient direction:

$$w^1 = w^0 - \alpha_0 \nabla f(w^0).$$

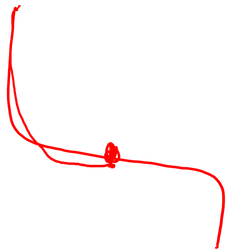
  (The scalar α_0 is the `step size'.)
  - Repeat to successively refine the guess: $w^{t+1} = w^t - \alpha_t \nabla f(w^t).$

    Generate $w^2, w^3, w^4, \ldots$
  - Stop if not making progress or $\|\nabla f(w^t)\| \leq \delta$ (some small number)

Gradient Descent in 2D



Gradient Descent



- If α_t is small enough and $\nabla f(w^t) \neq 0$, guaranteed to decrease 'f':

$$f(w^{t+1}) < f(w^t)$$

- Under weak conditions, procedure converges to a stationary point.

If 'f' is convex, converges to global minimum.

- Least squares via linear system vs. gradient descent:
  - Solving linear system cost $O(nd^2 + d^3)$.
  - Gradient descent costs $O(ndt)$ to run for 't' iterations.
    - Will be faster if $t < d$.

$O(nd)$

$X^T(Xw) + X^Ty$

Convergence Rate of Gradient Descent

- How many iterations do we need?
  - Let x^* be the optimal solution and ε be the accuracy we want.
  - What is the smallest number of iterations 't' such that:

$$f(x^t) - f(x^*) \leq \varepsilon$$

Notation:
- In optimization, we usually talk about optimizing x.

- To answer this question, need assumptions:
  - Lets assume $\underbrace{\mu I \preceq}_{\text{"strongly convex"}} \nabla^2 f(x) \underbrace{\preceq L I}_{\text{"strongly smooth"}}$ for all x and some $L < \infty$ and $\mu > 0$.

"strongly convex" $y^T \nabla^2 f(x) y \geq \mu \|y\|^2$

"strongly smooth" $y^T \nabla^2 f(x) y \leq L \|y\|^2$

strongly-convex
$\Rightarrow$ strictly-convex
$\Rightarrow$ convex.

Example:

If $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2$, then $\mu = \min \text{eig}(X^TX) + \lambda \geq \lambda$ and $L = \max \text{eig}(X^TX) + \lambda$.

Bonus Slide: Constants for Least Squares

- Consider least squares: $f(x) = \frac{1}{2}\|Ax - b\|^2$

What are 'L' and 'μ' such that $\mu I \preceq \nabla^2 f(x) \preceq LI$?

Note that $\nabla^2 f(x) = A^TA$, and since it's symmetric we can use spectral decomposition:

$A^TA = \sum_{j=1}^{d} \lambda_j q_j q_j^T$ where $q_j^T q_j = 1$ and $q_i^T q_j = 0$ for $i \neq j$. (Assume $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$)

We can write any y as linear combination of orthogonal basis, $y = \alpha_1 q_1 + \alpha_2 q_2 + \cdots + \alpha_d q_d$.

So we have $y^T \nabla^2 f(x) y = y^T A^T A y = y^T \left(\sum_{j=1}^{d} \lambda_j q_j q_j^T\right) y = \sum_{j=1}^{d} \lambda_j \underbrace{y^T q_j}_{=\alpha_j} \underbrace{q_j^T y}_{} = \sum_{j=1}^{d} \lambda_j \alpha_j^2$

Note that we can assume $\|y\| = 1$ or $y^T y = \sum_{j=1}^{d} \alpha_j^2 = 1$.

So $y^T \nabla^2 f(x) y$ is maximized when $\alpha_1^2 = 1$ and minimized when $\alpha_d^2 = 1$, giving $L = \lambda_1 = \max(\mathrm{eig}(A^TA))$ and $\mu = \lambda_n = \min(\mathrm{eig}(A^TA))$

Convergence Rate of Gradient Descent

- The gradient descent iteration:

$$x^{t+1} = x^t - \alpha_t \nabla f(x^t)$$

- Assumptions:
  - Function ‘f’ is L-strongly smooth and μ-strongly convex.
  - We set the step-size to $\alpha_t = 1/L$.
- Then gradient descent has a linear convergence rate:

$$f(x^t) - f(x^*) \leq O(\rho^t) \quad \text{for } \rho < 1.$$

  - It follows that we need $t = O(\log(1/\varepsilon))$ iterations.
    - This is good! We want ‘t’ to grow slowly in accuracy $1/\varepsilon$.
  - Also called ‘exponential’ convergence rate.

$f(x^t) - f(x^*) = \varepsilon \leq O(\rho^t)$

means $\varepsilon \leq c\rho^t$ for t large

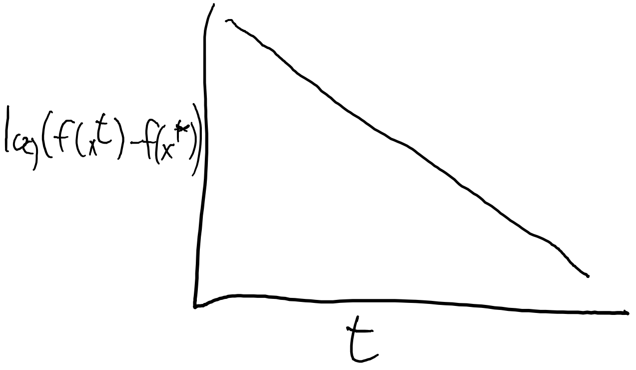
or $\log(\varepsilon) \leq \log(c\rho^t)$

$= \log(c) + t\log(\rho)$

or $t \geq \frac{\log(\varepsilon)}{\log(\rho)} - \text{const}$

or $t \geq O(\log(1/\varepsilon))$

(since $\rho < 1$)

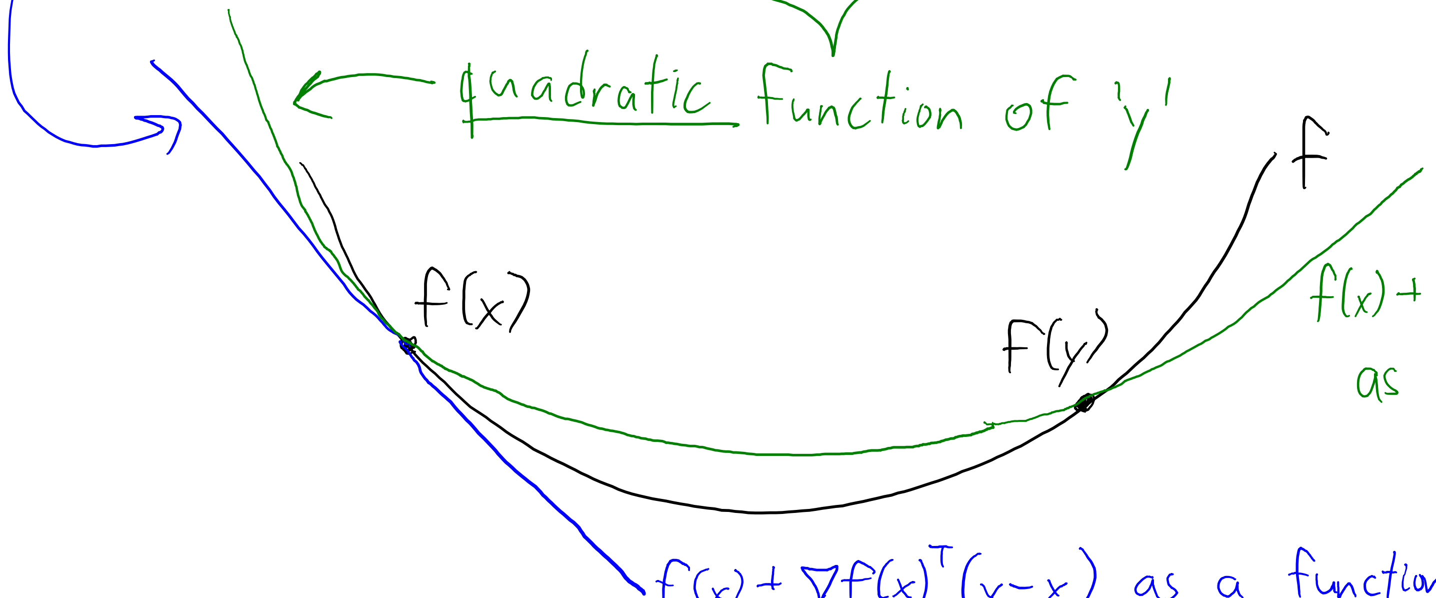


Convergence Rate of Gradient Descent

- One version of Taylor expansion:

$$f(y) = \underbrace{\underbrace{f(x) + \nabla f(x)^T(y-x)}_{\text{linearization of 'f' at 'x'}} + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)}_{\text{quadratic function of 'y'}}$$

for some z
for all x and y.



$f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$ as function of y.

$f(x) + \nabla f(x)^T(y-x)$ as a function of 'y'

Using Strong-Smoothness

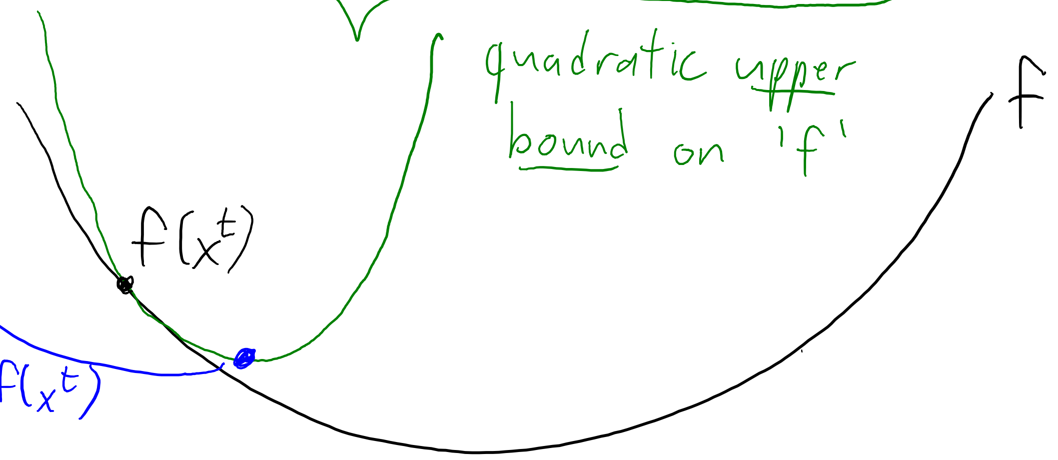
- One version of Taylor expansion:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(z)(y-x)$$ for <u>some</u> z
for <u>all</u> x and y.

From <u>strong-smoothness</u> we have: $v^T \nabla^2 f(z) v \leq L\|v\|^2$ for <u>any</u> z and v.

$$f(y) \leq \underbrace{f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|^2}$$ for <u>all</u> x and y.

If we set x^{t+1} to <u>minimize bound</u>, we get $x^{t+1} = x^t - \frac{1}{L}\nabla f(x^t)$



Let's find <u>min</u> of quadratic upper bound:

Let $g(y) = f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|^2$

$\nabla g(y) = 0 + \nabla f(x) - 0 + L(y-x)$

or

$y = x - \frac{1}{L}\nabla f(x)$.

Using Strong-Smoothness

- One version of Taylor expansion:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(z)(y-x)$$ for some z for all x and y.

From strong-smoothness we have: $v^T \nabla^2 f(z) v \leq L\|v\|^2$ for any z and v.

$$f(y) \leq f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|^2$$ for all x and y.

Set $x = x^t$ and $y = x^{t+1}$:

$$f(x^{t+1}) \leq f(x^t) + \nabla f(x^t)^T(x^{t+1} - x^t) + \frac{L}{2}\|x^{t+1} - x^t\|^2$$

$x^{t+1} = x^t - \frac{1}{L}\nabla f(x^t)$

(minimum of upper bound)

$$= f(x^t) + \nabla f(x^t)^T\left(-\frac{1}{L}\nabla f(x^t)\right) + \frac{L}{2}\left\|-\frac{1}{L}\nabla f(x^t)\right\|^2$$

$$= f(x^t) - \frac{1}{L}\nabla f(x^t)^T \nabla f(x^t) + \frac{1}{2L}\|\nabla f(x^t)\|^2$$

$\nabla f(x)^T \nabla f(x) = \|\nabla f(x)\|^2$

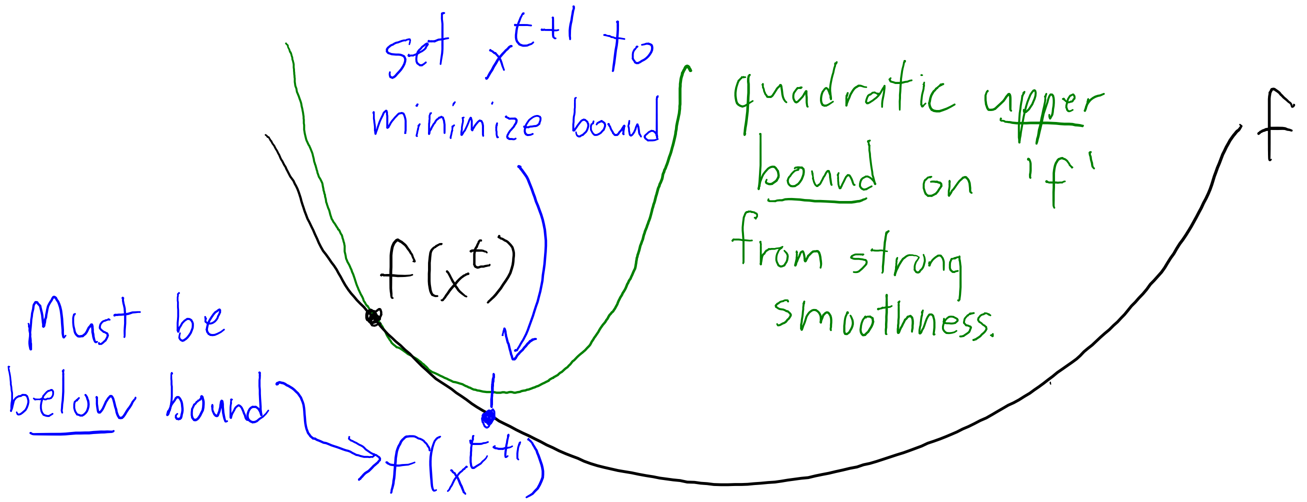
$$= f(x^t) - \frac{1}{2L}\|\nabla f(x^t)\|^2$$

Using Strong-Smoothness

- We've derived a bound on guaranteed progress at iteration 't':

$$f(x^{t+1}) \leq f(x^t) - \frac{1}{2L}\|\nabla f(x^t)\|^2$$

  – If gradient is non-zero, guaranteed to decrease objective.
  – Amount we decrease grows with the size of the gradient.
  – Note: bound involves for any strongly-smooth function (e.g., non-convex)



Using Strong-Convexity

- One version of Taylor expansion:

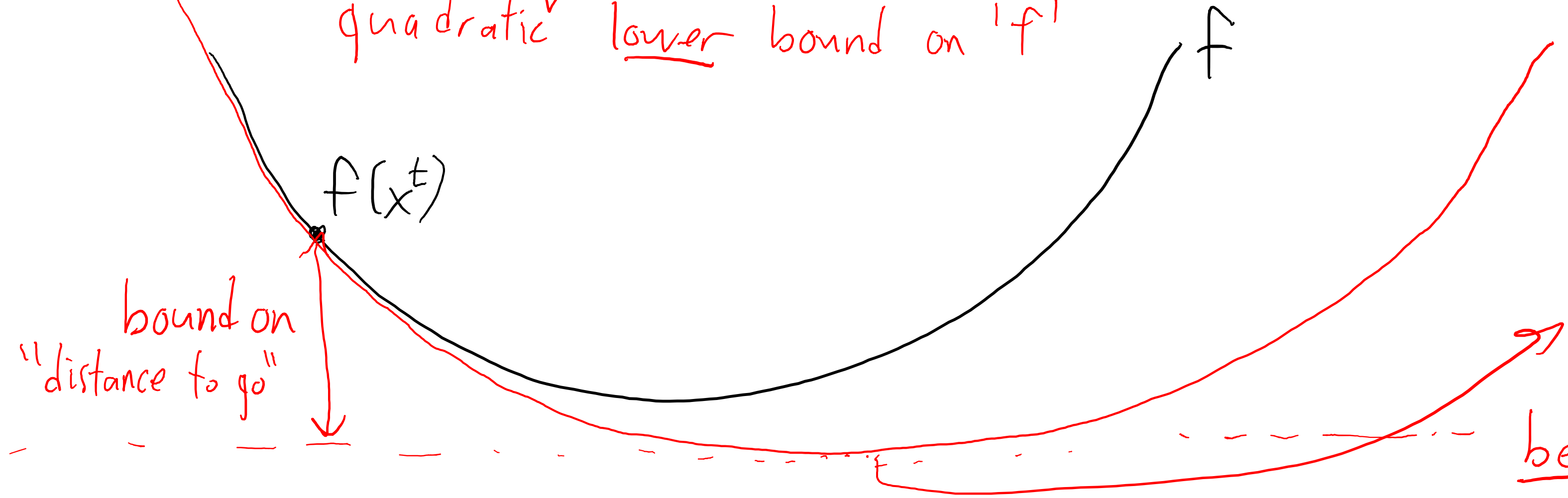
$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

for some z

for all x and y.

By strong-convexity we have $v^T\nabla^2 f(z)v \geq \mu$ for all v and z.

$$f(y) \geq \underbrace{f(x) + \nabla f(x)^T(y-x) + \frac{\mu}{2}\|y-x\|^2}_{\text{quadratic lower bound on 'f'}}$$



We know that $f(x^*)$ cannot be below minimum of bound.

Using Strong-Convexity

- One version of Taylor expansion:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$ for some z for all x and y.

By strong-convexity we have $v^T\nabla^2 f(z)v \geq \mu$ for all v and z.

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\mu}{2}\|y-x\|^2$$

Minimize both sides with respect to y:

$$f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2$$

Combining Strong-Smoothness and Convexity

- Our bound on guaranteed progress:

$$f(x^{t+1}) \leq f(x^t) - \frac{1}{2L}\|\nabla f(x^t)\|^2$$

- Our bound on 'distance to go':

$$f(x^*) \geq f(x^t) - \frac{1}{2\mu}\|\nabla f(x^t)\|^2 \iff -\frac{1}{2}\|\nabla f(x^t)\|^2 \leq -\mu(f(x^t) - f(x^*))$$

- Use 'distance to go' bound in guaranteed progress bound:

$$f(x^{t+1}) \leq f(x^t) - \frac{1}{L}\left(-\mu(f(x^t) - f(x^*))\right)$$

- Subtract f(x*) from both sides and simplify:

$$f(x^{t+1}) - f(x^*) \leq f(x^t) - f(x^*) - \frac{\mu}{L}(f(x^t) - f(x^*))$$

$$= \left(1 - \frac{\mu}{L}\right)\left[f(x^t) - f(x^*)\right]$$

Combining Strong-Smoothness and Convexity

- We've shown that:

$$f(x^t) - f(x^*) \leq \left(1 - \frac{\mu}{L}\right)\left[f(x^{t-1}) - f(x^*)\right]$$

- Applying this recursively:

$$\begin{aligned} f(x^t) - f(x^*) &\leq \left(1 - \frac{\mu}{L}\right)\left[\left(1 - \frac{\mu}{L}\right)\left[f(x^{t-2}) - f(x^*)\right]\right] \\ &= \left(1 - \frac{\mu}{L}\right)^2\left[f(x^{t-1}) - f(x^*)\right] \\ &= \left(1 - \frac{\mu}{L}\right)^3\left[f(x^{t-2}) - f(x^*)\right] \\ &\;\vdots \\ &= \underbrace{\left(1 - \frac{\mu}{L}\right)^t\left[f(x^0) - f(x^*)\right]} \rightarrow O(\rho^t) \end{aligned}$$

- Since μ ≤ L, we've shown linear convergence rate.

Discussion of Linear Convergence Rate

- We've shown that gradient descent under certain settings has:

$$f(x^t) - f(x^*) \leq \left(1 - \frac{\mu}{L}\right)^t [f(x^0) - f(x^*)]$$

- The number L/μ is called the 'condition number' of 'f'.
- Connection to matrix condition number:
  - For least squares, condition number of 'f' is condition number of X^TX.
- This rate is dimension-independent:
  - It does not directly depend on dimensions 'd'.
  - In principle, applies to infinite-dimensional problems.
  - But, L and μ may be larger in high-dimensional spaces.
- In practice, typically you don't have 'L'.
  - We'll discuss practical issues next time.

Summary

- No free lunch: there is no 'best' machine learning model.
- Softmax loss to model discrete yi, other losses can be derived.
- Convex functions: all stationary points are global minima.
- Show functions are convex.
- Gradient descent finds stationary point of differentiable function.
- Rate of convergence of gradient descent is linear.

- Next time:
  – What if we don't know which features are relevant or which basis to use?