Pattern Recognition
Lecture 03-1

Gradient Descent & Kernel Trick

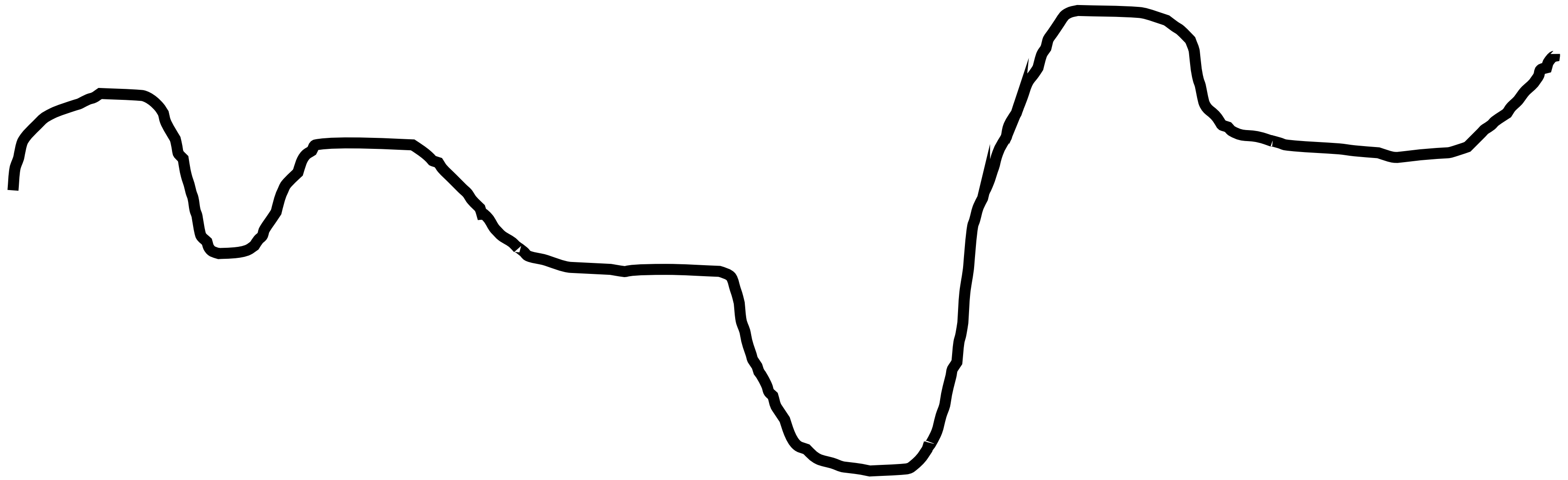
Prof. Jongwon Choi
Chung-Ang University
Fall 2022

This Class

- **Gradient Descent**
- **Robust Regression**
- **Regularization**
- RANSAC
- Kernel Trick

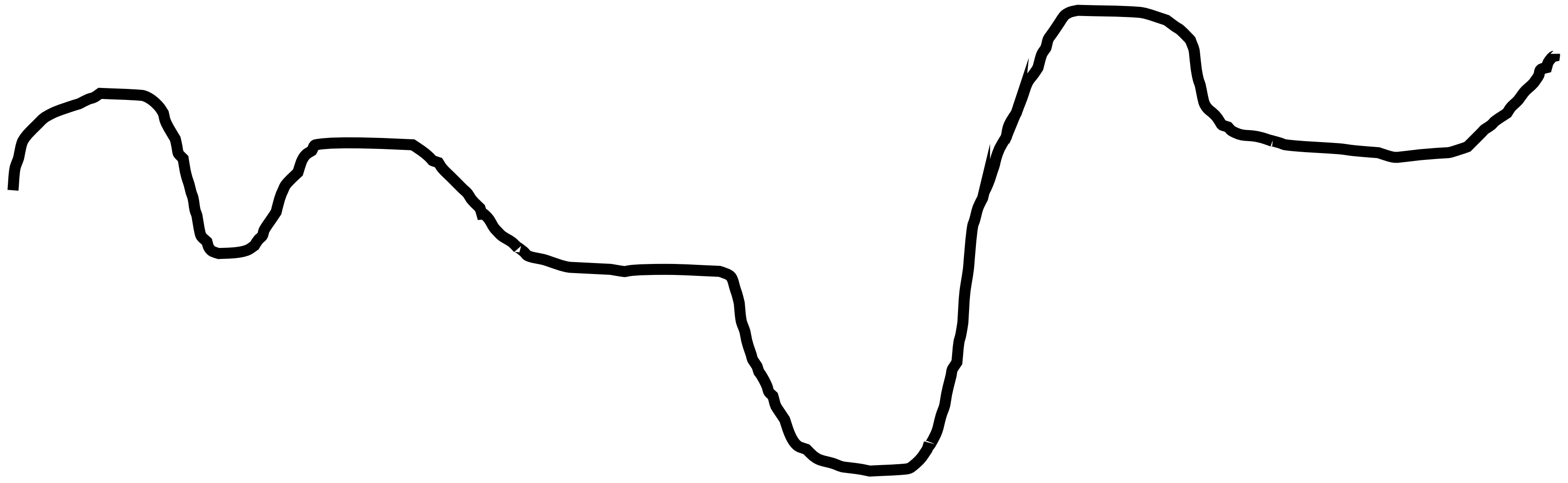
Stationary/Critical Points

- ‘w’ with $\nabla f(w) = 0$ is called a stationary point or critical point
  - The slope is zero so the tangent plane is “flat”



Stationary/Critical Points

- ‘w’ with $\nabla f(w) = 0$ is called a stationary point or critical point
  - The slope is zero so the tangent plane is “flat”



- If we’re minimizing, we would ideally like to find a global minimum!

Gradient Descent

- Motivation – Large-scale Least Squares
  - Normal equations find ‘w’ with $\nabla f(w) = 0$ in $O(nd^2 + d^3)$ time
  - It is very slow if ‘d’ is large
- Alternatively, we can utilize “gradient descent” method
  - The most important class of algorithms in machine learning! (i.e. Deep learning)

Gradient Descent

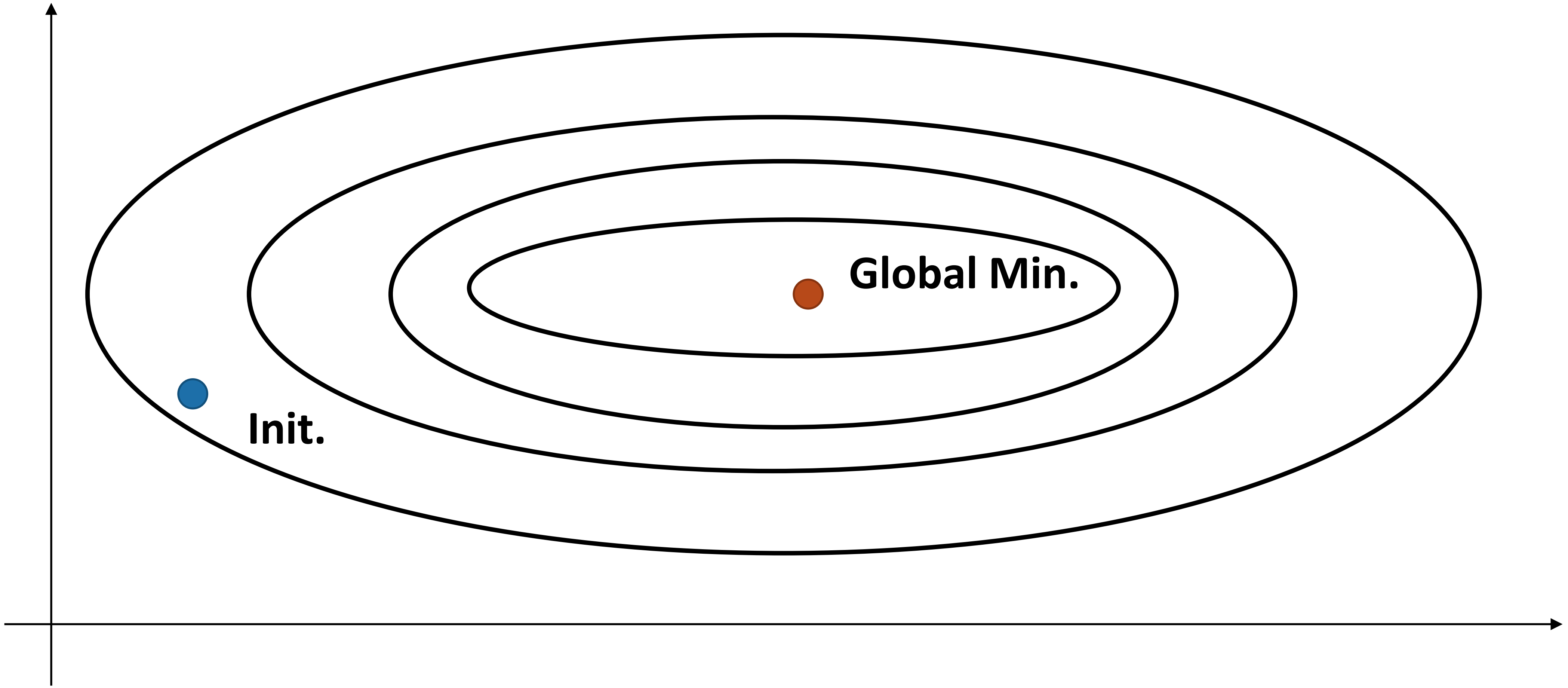
- Mechanism - Iterative optimization algorithm
  - It starts with a "guess" w^0
  - It uses the gradient $\nabla f(w^0)$ to generate a better guess w^1
  - It uses the gradient $\nabla f(w^1)$ to generate a better guess w^2
  - It uses the gradient $\nabla f(w^2)$ to generate a better guess w^3
  - ....
  - The limit of w^t as 't' goes to ∞ has $\nabla f(w^t) = 0$
- It converges to a global optimum if 'f' is "convex"

Gradient Descent

Gradient Descent for a Local Minimum

- We start with some initial guess, w^0
- Generate new guess by moving in the negative gradient direction:
  - $w^1 = w^0 - \alpha^0 \nabla f(w^0)$
    - This decreases 'f' if the "step size" α^0 is small enough
    - Usually, we decrease α^0 if it increases 'f'
- Repeat to successively refine the guess:
  - $w^{t+1} = w^t - \alpha^t \nabla f(w^t)$
- Stop if not making progress
  - $\|\nabla f(w^t)\| \leq \epsilon$

Gradient Descent in 2D



Gradient Descent for Least Squares

- The least squares objective and gradient:
  - $f(\mathbf{w}) = \frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$ -> $\nabla f(\mathbf{w}) = \mathbf{X}^{\mathrm{T}}(\mathbf{X}\mathbf{w} - \mathbf{y})$
- Gradient descent iterations for least squares:
  - $\mathbf{w}^{t+1} = \mathbf{w}^{t} - \alpha^{t}\mathbf{X}^{\mathrm{T}}(\mathbf{X}\mathbf{w}^{\mathrm{t}} - \mathbf{y})$
- **Cost of gradient descent iteration is O(nd)**
  - Much smaller than the normal equations of $O(nd^2 + d^3)$ with large d
- **Can cover many problems other than the least square!**

Beyond Gradient Descent

- There are many variations on gradient descent
  - Netwon’s method – uses second derivative for the step size
  - Quasi-Newton and Hessia-free Newton methods – small computation
  - Stochastic gradient – sample-wise approach

Gradient Descent for Least Squares

- The least squares objective and gradient:
  - $f(\mathbf{w}) = \frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$ -> $\nabla f(\mathbf{w}) = \mathbf{X}^{\mathrm{T}}(\mathbf{X}\mathbf{w} - \mathbf{y})$
- Gradient descent iterations for least squares:
  - $\mathbf{w}^{t+1} = \mathbf{w}^{t} - \alpha^{t}\mathbf{X}^{\mathrm{T}}(\mathbf{X}\mathbf{w}^{\mathrm{t}} - \mathbf{y})$
- **Cost of gradient descent iteration is O(nd)**

Gradient Descent

- Sequence of iterations of the form:
  - $w^{t+1} = w^t - \alpha^t \nabla f(w^t)$
- Converges to a stationary point where $\nabla f(w)$ under weak conditions
  - Will be a global minimum if the function is “convex”
- Convex?
  - Second derivative is non-negative (1D functions)
  - Closed under addition, multiplication by non-negative, maximization
  - Any [squared-] norm is convex
  - Composition of convex function with linear function is convex

Convex?

- Second derivative is non-negative (1D functions)
- Closed under addition, multiplication by non-negative, maximization
- Any [squared-] norm is convex
- Composition of convex function with linear function is convex

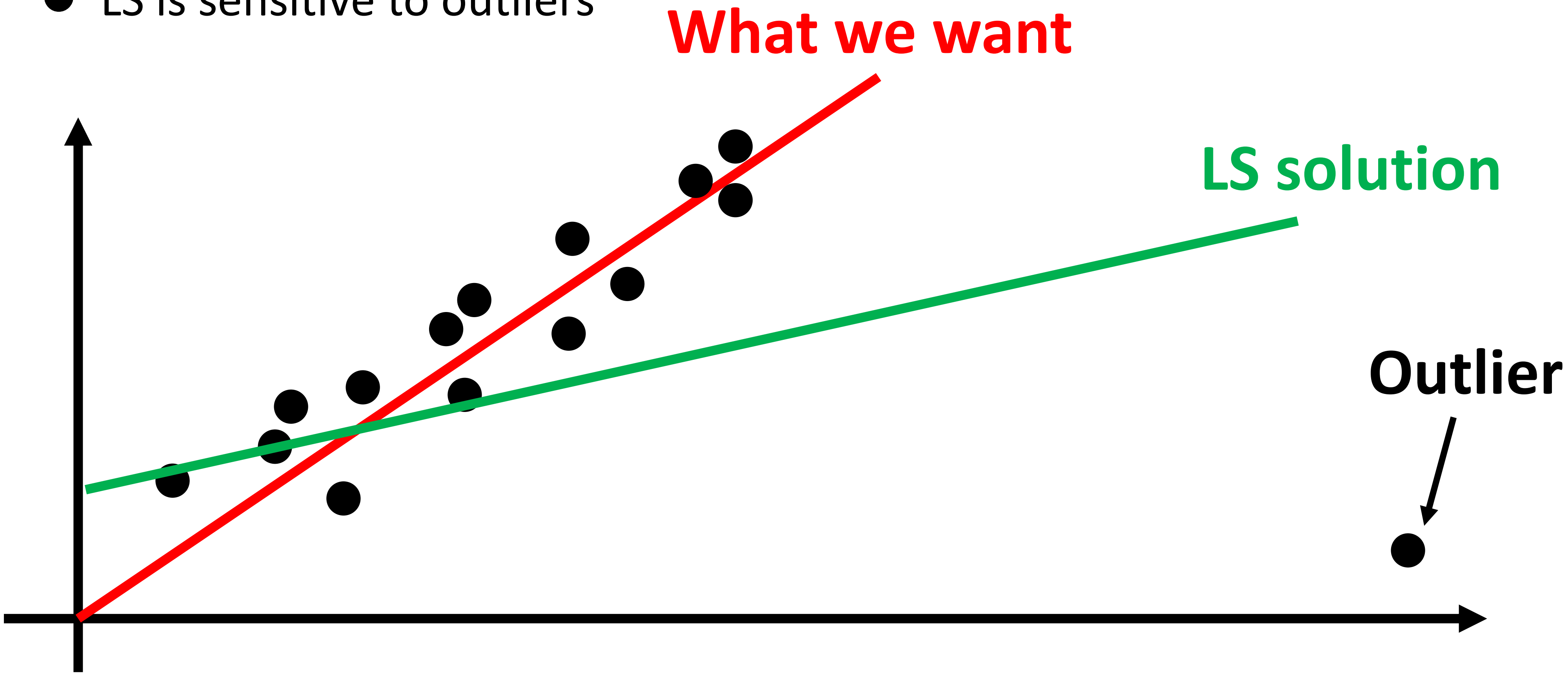
Convex and Gradient Descent

- With the convex function,
  - The gradient descent can converge to the global minimum!
  - The stochastic gradient descent also converges to the global minimum

- Unfortunately, many real applications cannot be represented as a convex form
  - Approximate the function by a convex form
  - Find the local minimum that is close to the global minimum (Deep learning)

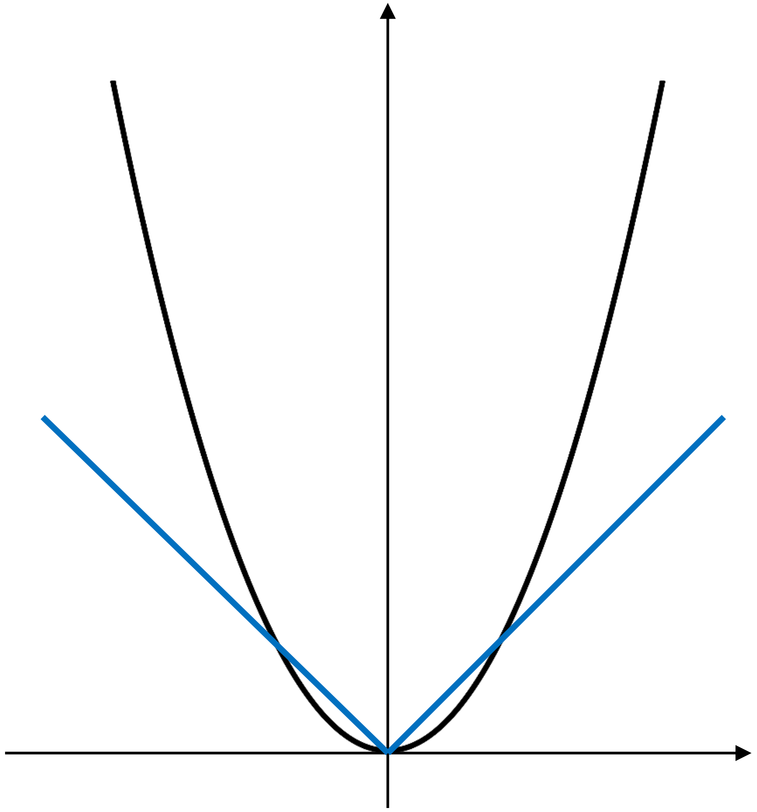
Least Squares with Outliers

- Let’s consider least square problem with outliers in ‘y’:
  - LS is sensitive to outliers



Least Squares with Outliers

- Because squaring error shrinks small errors, and magnifies large errors:
  - Thus, outliers (large error) influence 'w' much more than other points

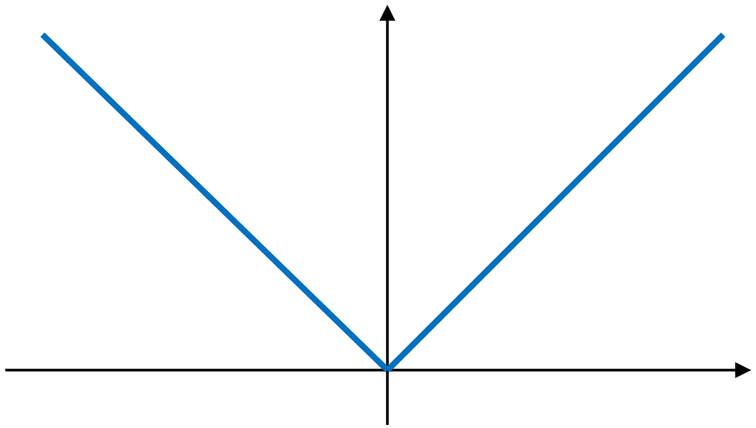


Robust Regression

- Objectives : Focus less on large errors (outliers)
- For example, the absolute error can be a good alternative:
  - $f(\mathbf{w}) = \sum_{i=1}^{n} \left| \mathbf{w}^{\mathrm{T}} \mathbf{x}_i - y_i \right|$
  - Then, decreasing ‘small’ and ‘large’ errors is equally important

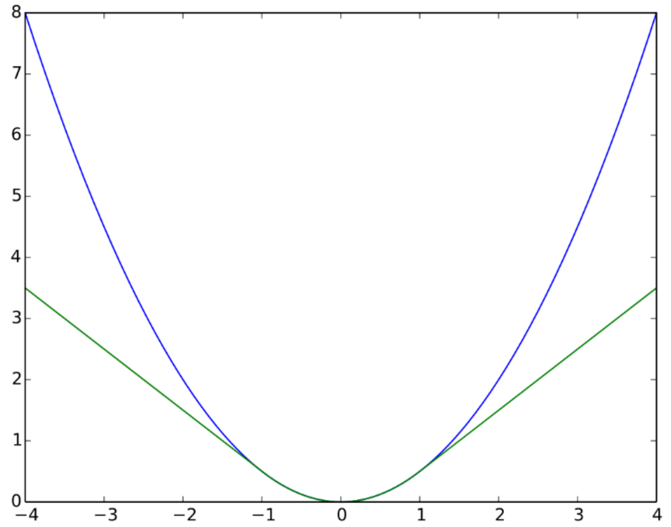
Robust Regression with L1-Norm

- Unfortunately, minimizing the absolute error is harder
  - We don’t have “normal equations”
  - Absolute value is non-differentiable at 0
  - Generally, harder to minimize non-smooth than smooth functions
    - Unlike smooth functions, the gradient may not get smaller near a minimizer
  - To apply gradient descent, we’ll use a smooth approximation



Robust Regression with L1-Norm

- There are differentiable approximations to absolute value
  - Common example is Huber loss:
    - $f(w) = \sum_{i=1}^{n} h(w^T x_i - y_i)$
    - $h(r_i) = \begin{cases} \frac{1}{2} r_i^2 & , for \ \ |r_i| \leq \epsilon \\ \epsilon\left(|r_i| - \frac{1}{2}\epsilon\right) & , otherwise \end{cases}$



    - Note that 'h' is differentiable
    - This 'f' is convex but setting $\nabla f(x) = 0$ does not give a linear system
      - But, we can minimize the Huber loss using gradient descent!

Infinite Norm Regression

- What if we should care about the outliers?
- Then, we can consider the infinity-norm:
  - $f(w) = \|\mathbf{Xw} - \mathbf{y}\|_\infty$ where $\|r\|_\infty = \max_i\{|r_i|\}$
  - Very sensitive to outliers, but worst case will be better!
  - However, L_∞-norm is convex but non-smooth as L1-norm
    - We approximate the max function by log-sum-exp function
    - $\max_i\{|z_i|\} \approx \log\left(\sum_i \exp(z_i)\right)$
      - Intuition: $\sum_i \exp(z_i) \approx \max_i\{\exp(z_i)\}$ (largest element is magnified exponentially!)

Controlling Complexity

- Usually “true” mapping from x_i to y_i is complex
  - Might need high-degree polynomial (i.e. n=p)
- But complex models can overfit!!
- Solutions
  - Model averaging: average over multiple models to decrease variance
  - **Regularization: add a penalty on the complexity of the model**

L2-Regularization

- One of standard regularization strategies
  - $f(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x}_i - y_i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\mathbf{w}_j^2$
  - $f(\mathbf{w}) = \frac{1}{2}\|\mathbf{Xw} - \mathbf{y}\|^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$
- Intuition: Large slopes tend to lead to overfitting
  - Consider only a part of features!
- The regularization parameter $\lambda > 0$ controls “strength” of regularization
  - λ is a kind of hyperparameters (need cross-validation)

L2-Regularization and Normal Equations

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

- $\nabla f(w) = \mathbf{X}^{\mathrm{T}}\mathbf{Xw} - \mathbf{X}^{\mathrm{T}}\mathbf{y} + \lambda\mathbf{w}$

- $(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I})\mathbf{w} = \mathbf{X}^{\mathrm{T}}\mathbf{y}$
  - Interestingly, unlike $\mathbf{X}^{\mathrm{T}}\mathbf{X}$, $\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I}$ is always invertible!
  - Thus, $\mathbf{w} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I})^{-\mathbf{1}}\mathbf{X}^{\mathrm{T}}\mathbf{y}$

L2-Regularization and Gradient Descent

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

- $\nabla f(w) = \mathbf{X}^{\mathrm{T}}(\mathbf{Xw} - \mathbf{y}) + \lambda\mathbf{w}$

- $\mathbf{w}^{\mathbf{t+1}} = \mathbf{w}^{\mathbf{t}} - \boldsymbol{\alpha}^{\boldsymbol{t}}[\mathbf{X}^{\mathrm{T}}(\mathbf{Xw} - \mathbf{y}) + \lambda\mathbf{w}]$

Other types of regularization

- L1-Regularization - $f(w) = \frac{1}{2}\|\mathbf{Xw} - \mathbf{y}\|^2 + \frac{\lambda}{2}\|\mathbf{w}\|_1$
  - Outlier-robust regularization
  - Approximate the L1-regularization by Hubor norm
  - In deep learning, ignore the non-differentiable point due to the large parameters

- L0-Regularization - $f(w) = \frac{1}{2}\|\mathbf{Xw} - \mathbf{y}\|^2 + \frac{\lambda}{2}\|\mathbf{w}\|_0$
  - Weight sparsity regularization
  - Approximate the L0-regularization by **standard sigmoid function** / L1 norm
  - In deep learning, this sparsity is considered in the activation function

This Class

- **Gradient Descent**
- **Robust Regression**
- **Regularization**
- RANSAC
- Kernel Trick