

Lecture 7

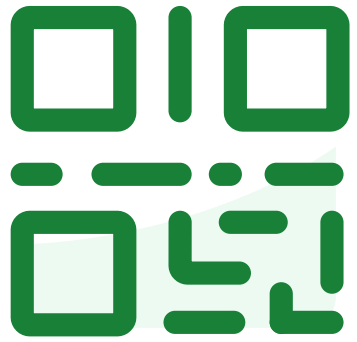
Regression (Cont.) and Bias-Variance Trade-off

The probabilistic view: maximum likelihood, noise models, priors, and the bias-variance trade-off

EECS 189/289, Fall 2026 @ UC Berkeley

Joseph E. Gonzalez and Narges Norouzi

Do not edit
How to change the
design



**Join at slido.com
#4191216**

① The Slido app must be installed on every computer you're presenting from

slido

Roadmap

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs



Regularized Least Squares



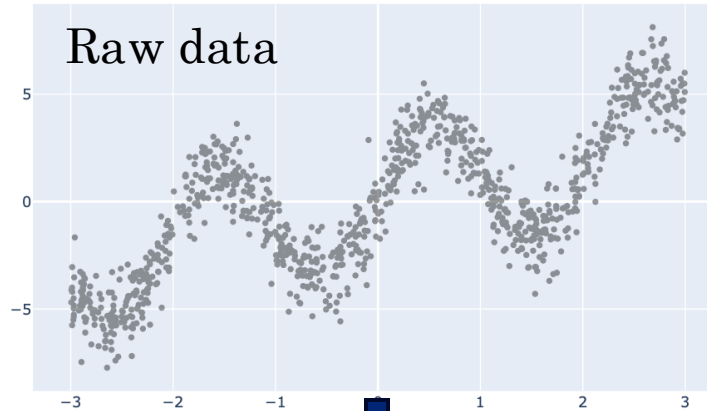
4191216

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs

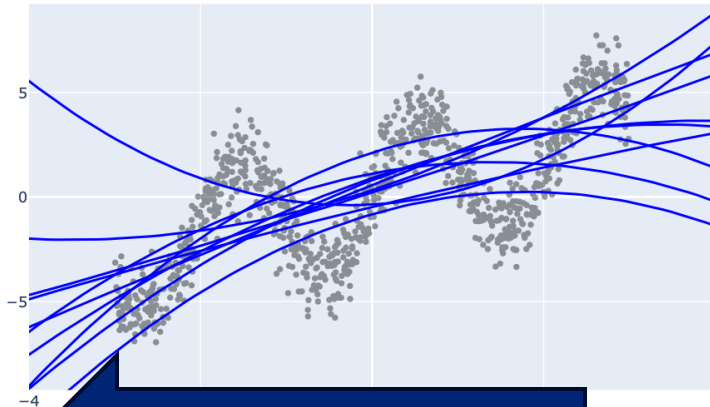
Complexity and Overfitting



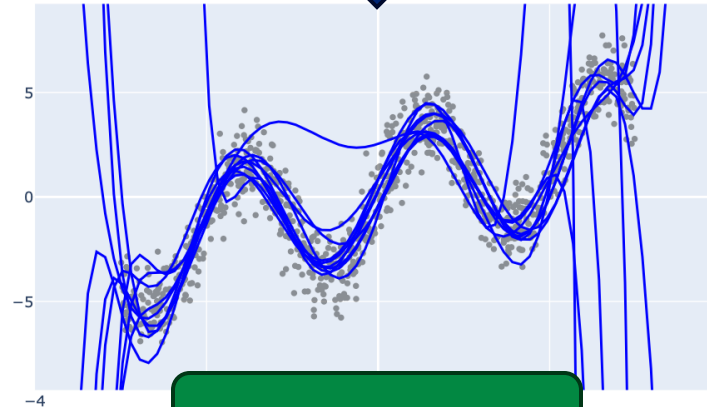
4191216



Fit models to
samples of data

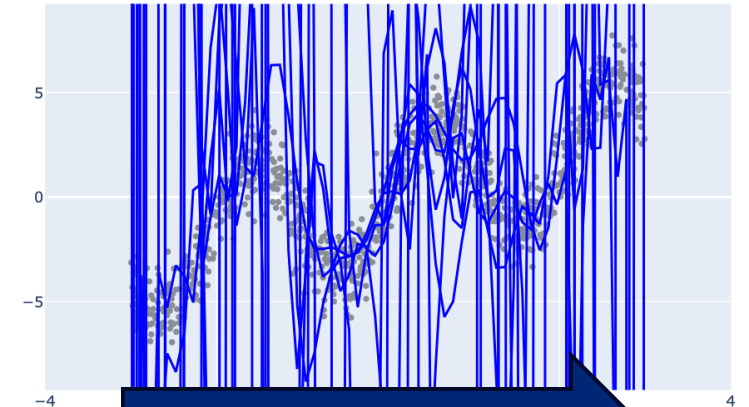


Underfitting



"Sweet Spot"

More complex isn't
always better.



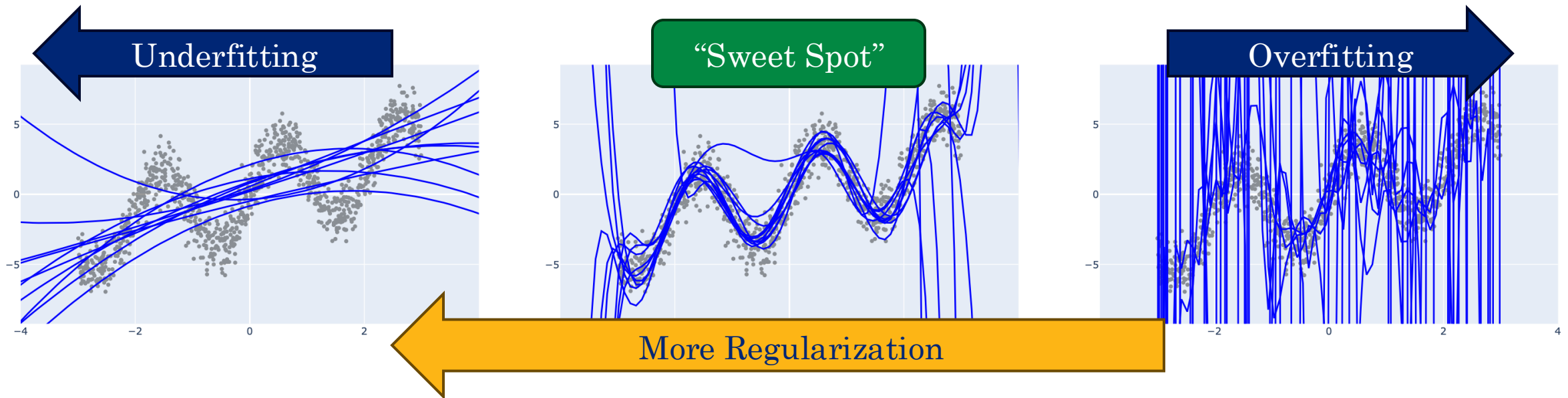
Overfitting

Regularization



4191216

Regularization is the process of adding **constraints** or **penalties** to the learning process to **improve generalization**.

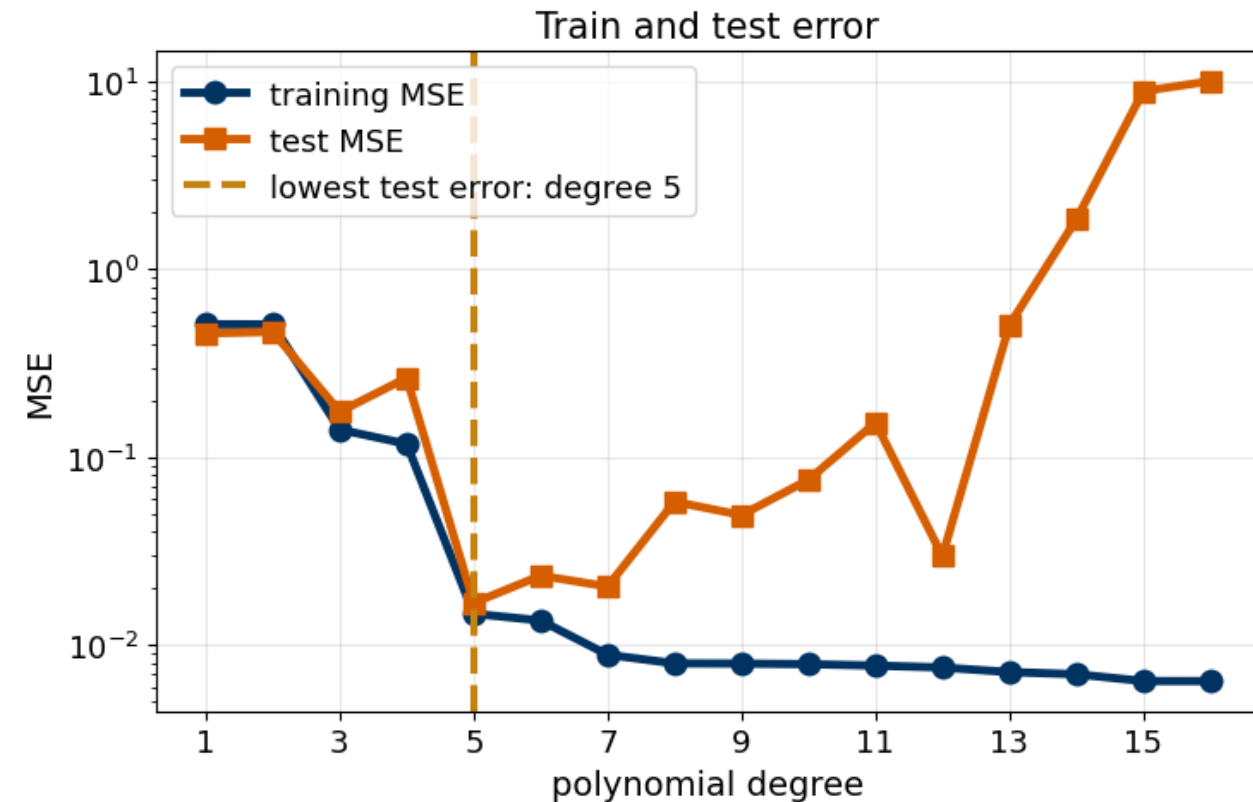
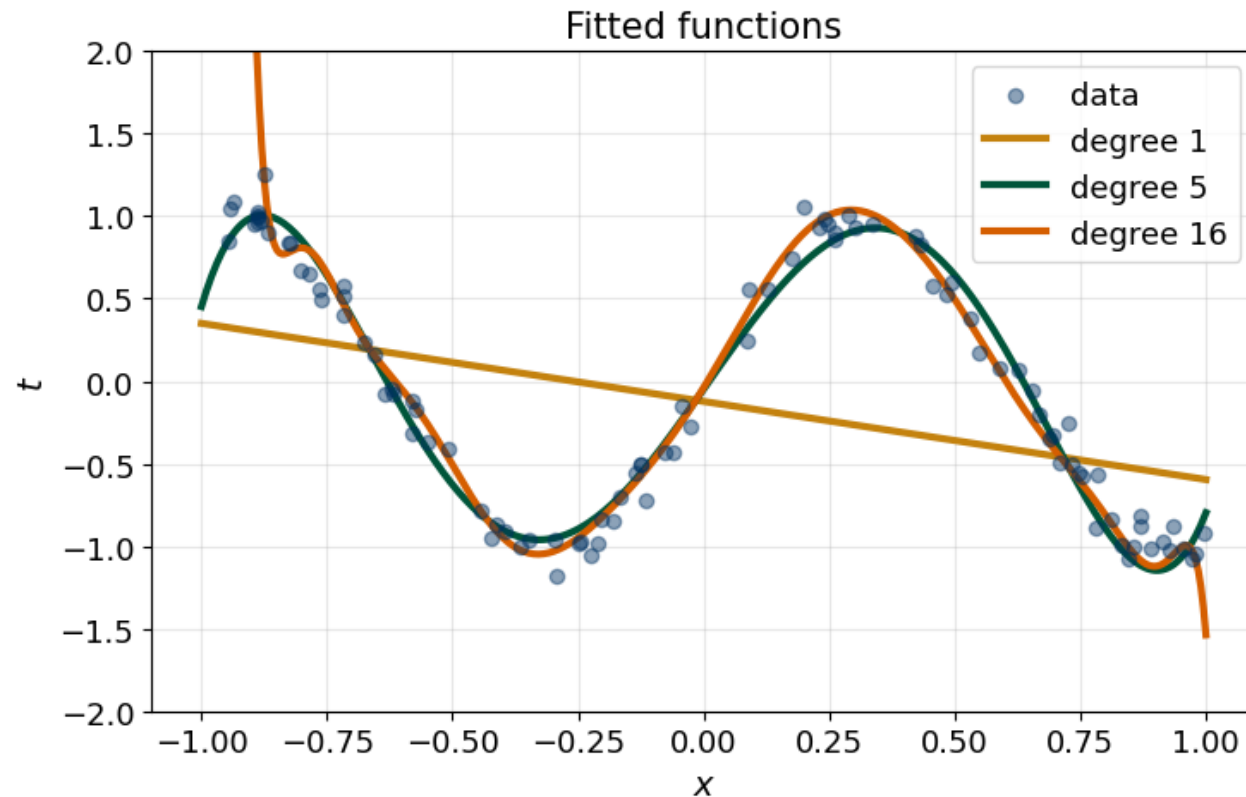


Many **models** and **learning algorithms** have methods to **tune** the **regularization** during the training process.

Training and Test Error vs Model Complexity



4191216

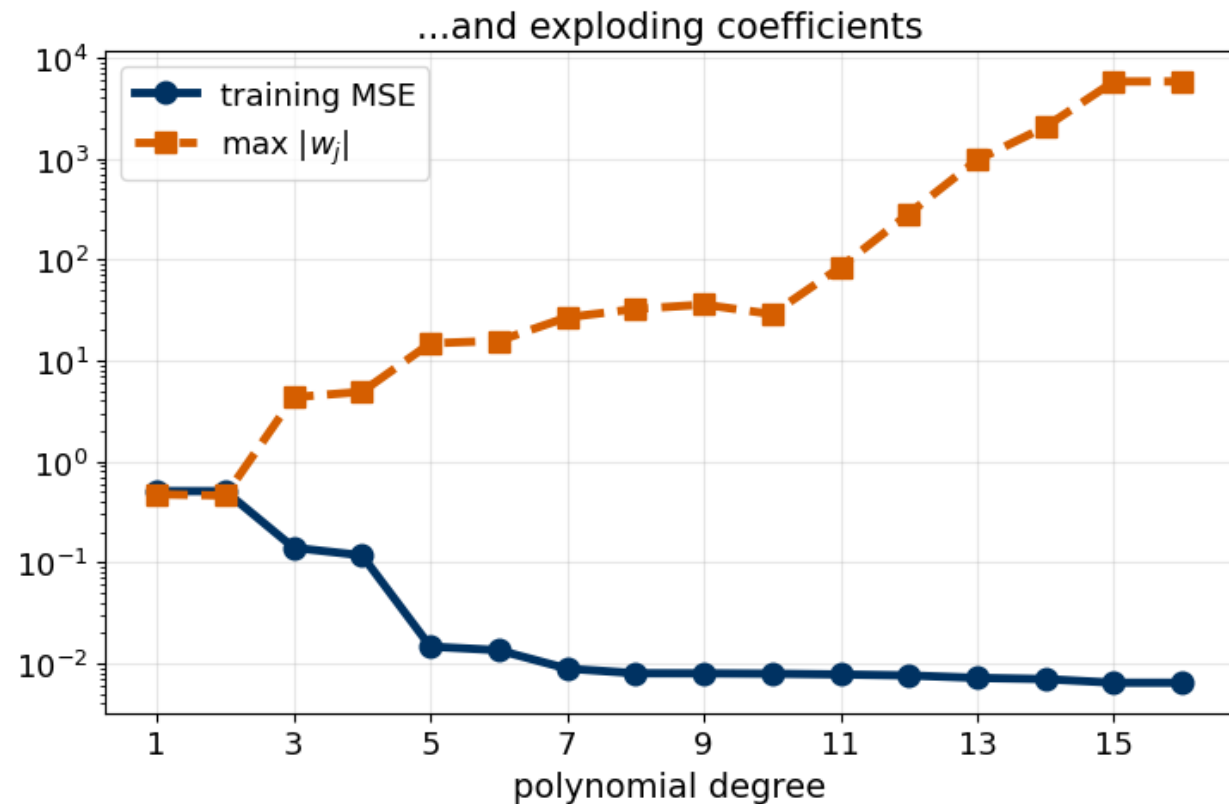
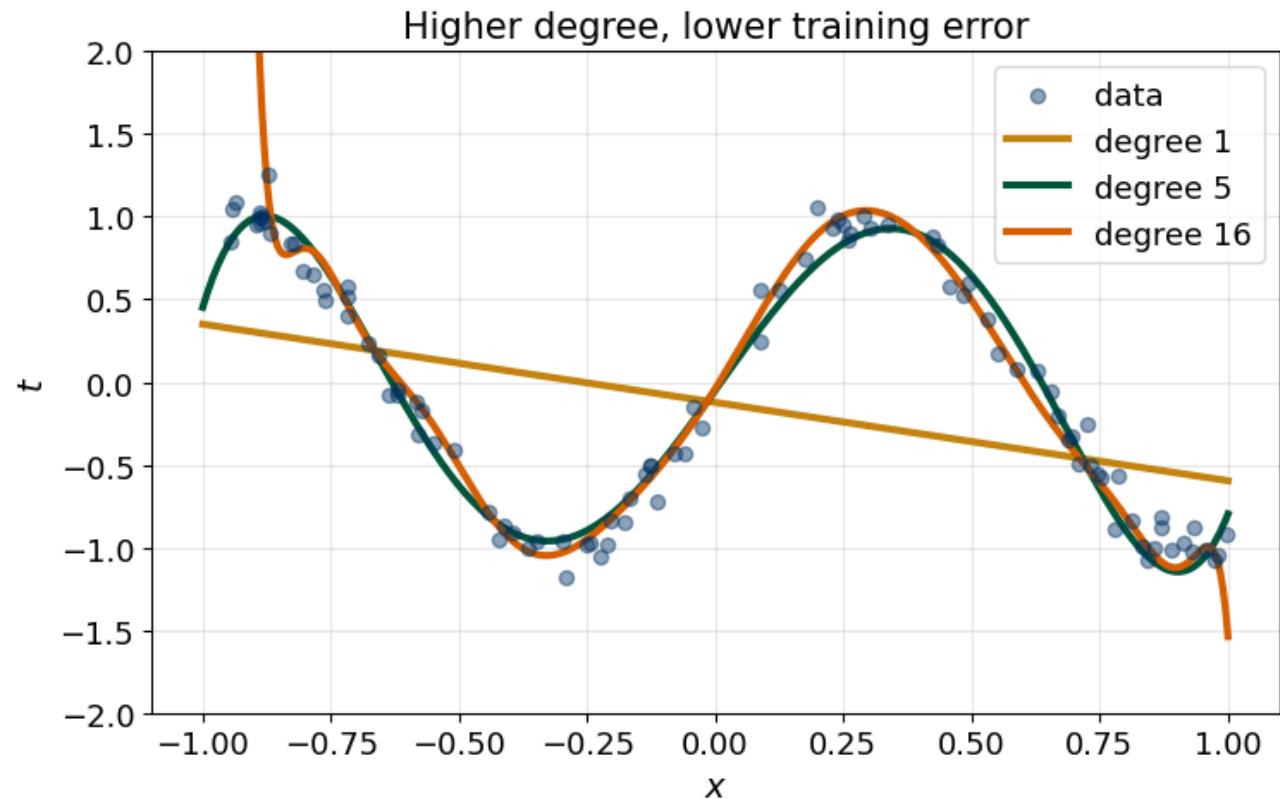


Degree 1 to 16 on 45 training points. Training MSE falls from 0.509 to 0.006. Test MSE is lowest at degree 5 (0.0168) and reaches 9.97 at degree 16.

Training and Test Error vs Model Complexity



4191216



At degree 16, where $\max |w_j|$ is 5780.



4191216

Regularization – Lagrangian Duality

Our Error function becomes a constrained function, so we minimize:

$$\text{Error}[h_w; \mathcal{D}_{\text{training}}] \quad \text{such that} \quad \text{Reg}[w] < \mathcal{C}$$

This constraint complicates our optimization.

What if we assign a penalty for each violation of the constraint?

Idea of Lagrangian

Introduce a penalty for breaking the constraint.

$$\underbrace{\lambda}_{\text{Imposed penalty}} \underbrace{(\text{Reg}[w] - \mathcal{C})}_{\text{Violation magnitude}}$$

The updated Error function

$$E(w) = \text{Error}[h_w; \mathcal{D}_{\text{training}}] + \lambda \text{Reg}[w]$$

Note: we did not include $-\lambda\mathcal{C}$ since it is a constant and won't impact the optimization



4191216

From **Reg**[w] $\rightarrow$ Norms

- One simple example of **Reg**[w] is the sum of the squares of the weight vector:

$$\frac{1}{2} \sum w_d^2 = \frac{1}{2} w^T w$$

- Generally, **norms** are the standard choices for **Reg**[w].
- What is a norm?

Norm definition: $L_p(w): \|w\|_p = (\sum_d |w_d|^p)^{\frac{1}{p}}$



4191216

From **Reg**[w] $\rightarrow$ Norms

- Norm definition: $L_p(w): \|w\|_p = (\sum_d |w_d|^p)^{\frac{1}{p}}$
- Example of norms:
 - $L_1(w): \|w\|_1 = \sum_d |w_d|$
 - $L_2(w): \|w\|_2 = \sqrt{\sum_d w_d^2}$
 - $L_\infty(w): \|w\|_\infty = \max_d |w_d|$



4191216

Reg[w] $\rightarrow$ Norms $\rightarrow L_1$ Regularization

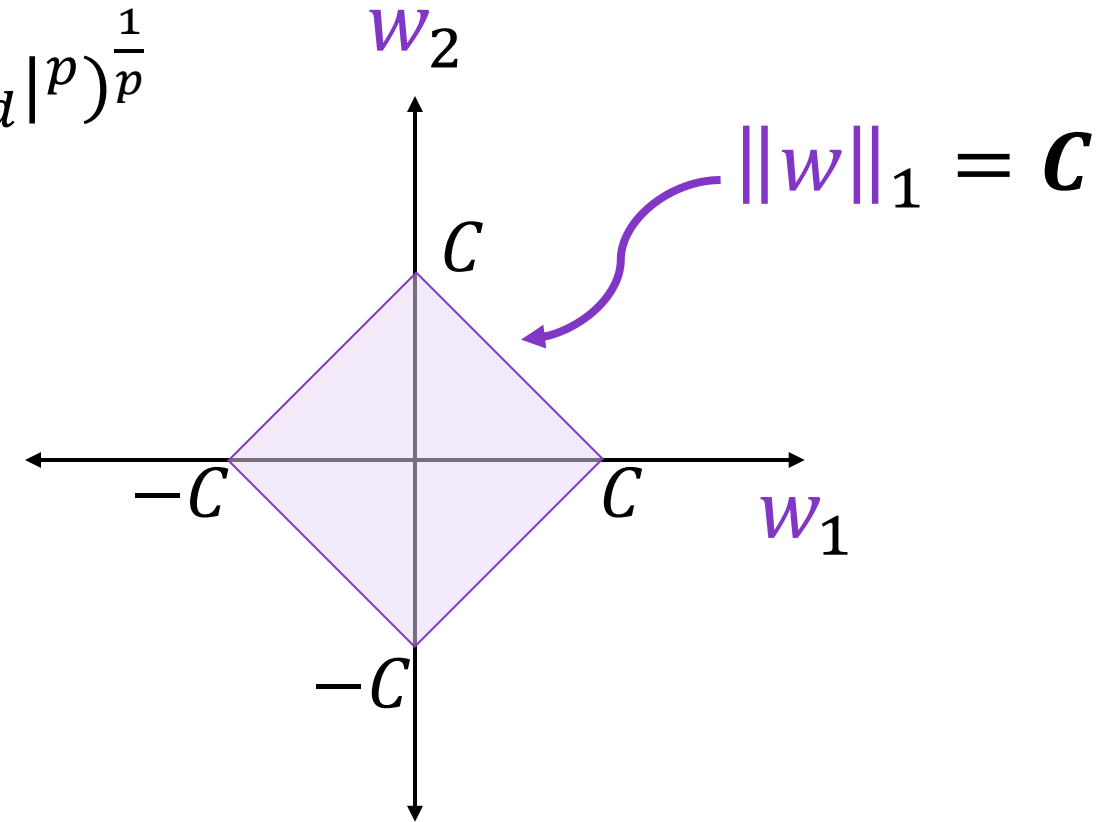
- Norm definition: $L_p(w): \|w\|_p = (\sum_d |w_d|^p)^{\frac{1}{p}}$

- Example of norms:

- $L_1(w): \|w\|_1 = \sum_d |w_d|$

- $L_2(w): \|w\|_2 = \sqrt{\sum_d w_d^2}$

- $L_\infty(w): \|w\|_\infty = \max_d |w_d|$





4191216

Reg[w] $\rightarrow$ Norms $\rightarrow L_2$ Regularization

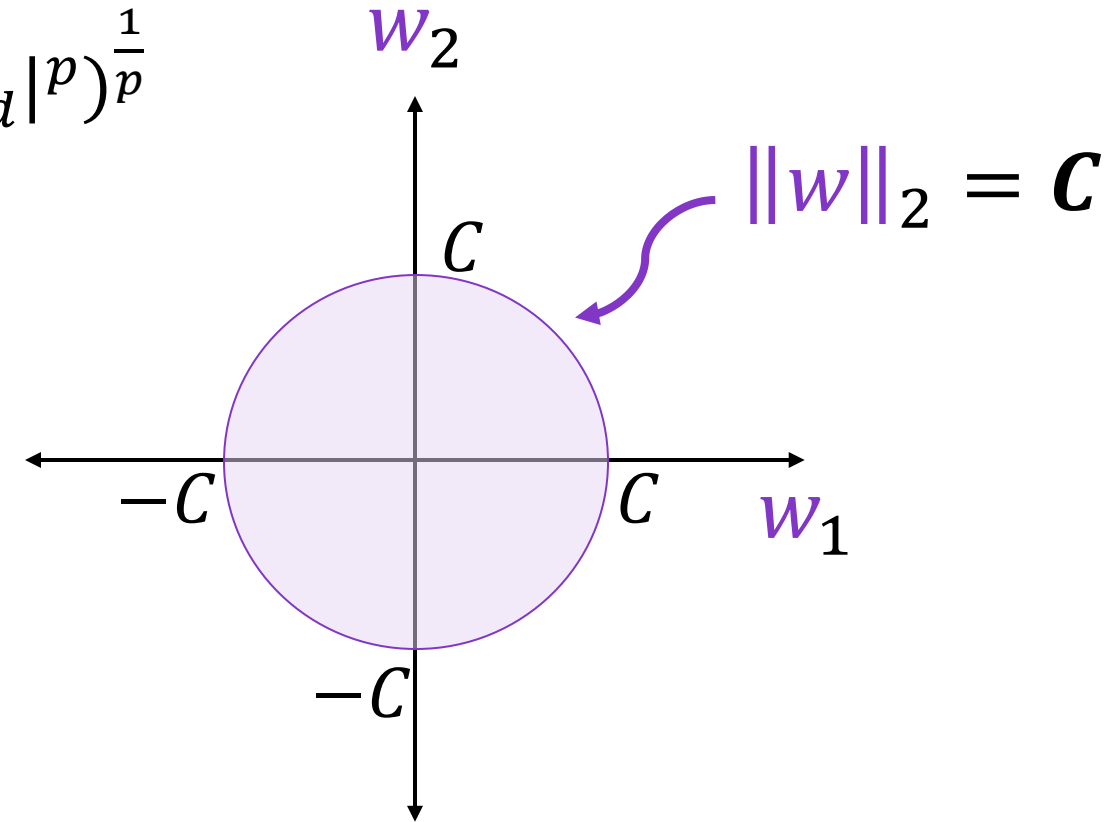
- Norm definition: $L_p(w): \|w\|_p = (\sum_d |w_d|^p)^{\frac{1}{p}}$

- Example of norms:

- $L_1(w): \|w\|_1 = \sum_d |w_d|$

- $L_2(w): \|w\|_2 = \sqrt{\sum_d w_d^2}$

- $L_\infty(w): \|w\|_\infty = \max_d |w_d|$





4191216

Regularization

We typically use **iterative optimization algorithms** to train (fit) the model by solving the following problem:

$$w^* = \arg \min_w \left(\underbrace{\mathbf{E}[w; \mathcal{D}_{\text{training}}]}_{\text{Data-dependent Error}} + \lambda \mathbf{Reg}[w] \right)$$



4191216

Regularization

We typically use **iterative optimization algorithms** to train (fit) the model by solving the following problem:

$$w^* = \arg \min_w \left(\underbrace{E_D(w)}_{\text{Data-dependent Error}} + \lambda \text{Reg}[w] \right)$$

Data-dependent
Error

Regularization



4191216

We typically use **iterative optimization algorithms** to train (fit) the model by solving the following problem:

$$w^* = \arg \min_w \left(\underbrace{E_D(w)}_{\text{Data-dependent Error}} + \lambda \underbrace{E_w[w]}_{\text{Regularization Function}} \right)$$



4191216

Regularization

We typically use **iterative optimization algorithms** to train (fit) the model by solving the following problem:

$$w^* = \arg \min_w (\underbrace{E_D(w)}_{\text{Data-dependent Error}} + \underbrace{\lambda E_w[w]}_{\text{Regularization Function}})$$

Data-dependent
Error

Regularizer
Hyperparameter

Regularization
Function

L2 Regularization (Ridge)

$$w^* = \arg \min_w \underbrace{E_D[w] + \lambda E_w[w]}_{\text{L2 Regularization (Ridge)}}$$

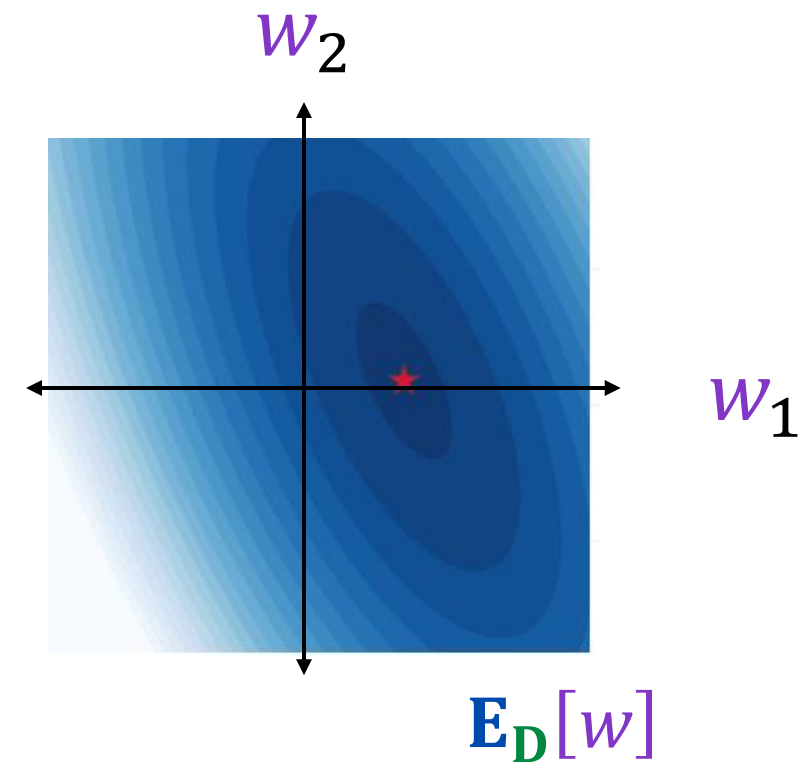
$$\frac{1}{2} \sum_{n=1}^N (t_n - y(x_n, w))^2$$

$$\frac{1}{2} (\|t_n - y(x_n, w)\|_2)^2$$

$$\frac{1}{2} (\|e_n\|_2)^2$$

length of the
residual vector

$$L_2[w] = \sqrt{\sum_d w_d^2}$$



4191216

L2 Regularization (Ridge)



$$w^* = \arg \min_w \underbrace{E_D[w]} + \underbrace{\lambda E_w[w]}$$

$$\frac{1}{2} \sum_{n=1}^N (t_n - y(x_n, w))^2$$

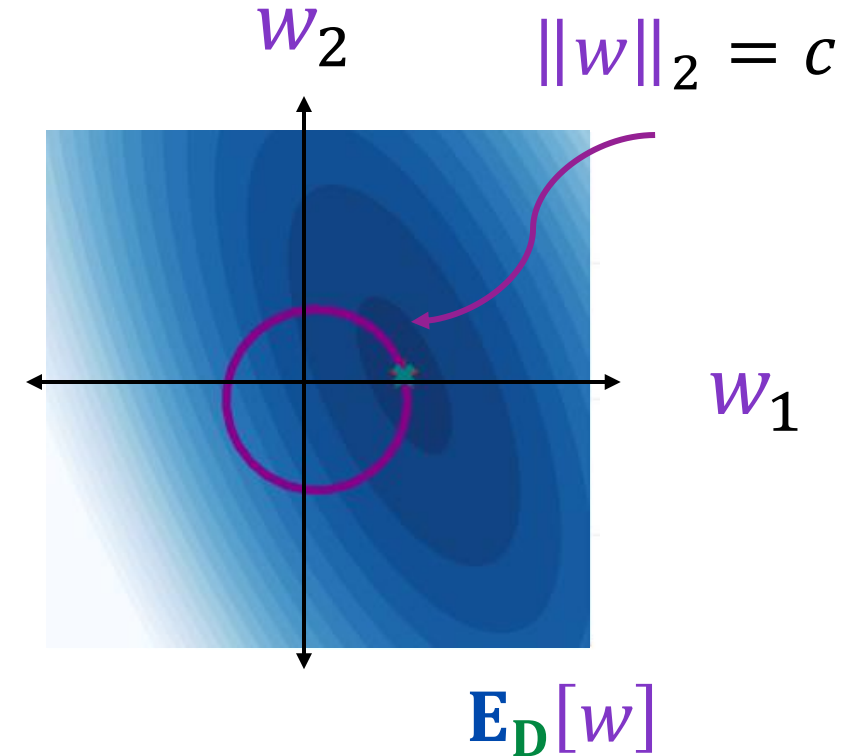
$$\frac{1}{2} L_2[w]^2 = \frac{1}{2} \sum_d w_d^2$$

$$\frac{1}{2} (\|t_n - y(x_n, w)\|_2)^2$$

$$\frac{1}{2} (\|e_n\|_2)^2$$

length of the
residual vector

$$L_2[w] = \sqrt{\sum_d w_d^2}$$

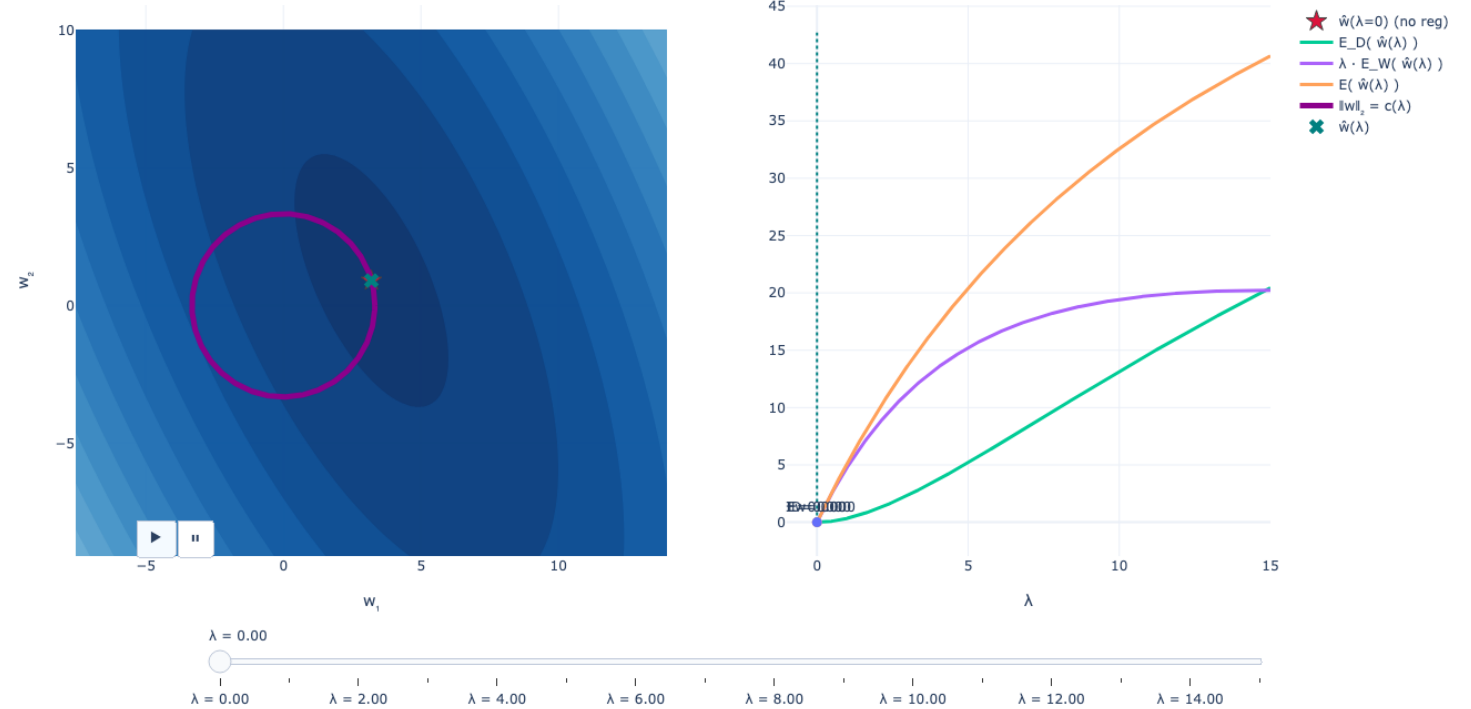


L2 Regularization Demo

[Interactive link](#)



4191216





4191216

L1 Regularization (Lasso)

$$L_1[w] = \sum_j |w_j|$$

$$w^* = \arg \min_w \underbrace{\mathbf{E}_D[w]} + \underbrace{\lambda \mathbf{E}_w[w]}$$

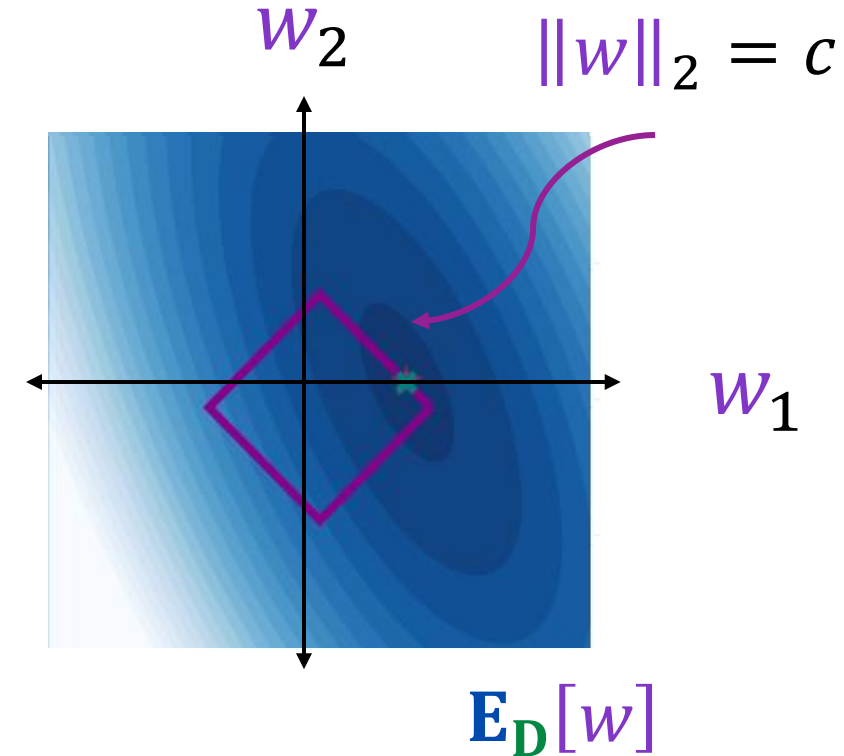
$$\frac{1}{2} \sum_{n=1}^N (t_n - y(x_n, w))^2$$

$$\frac{1}{2} L_1[w] = \frac{1}{2} \sum_d |w_d|$$

$$\frac{1}{2} (\|t_n - y(x_n, w)\|_2)^2$$

$$\frac{1}{2} (\|e_n\|_2)^2$$

length of the
residual vector

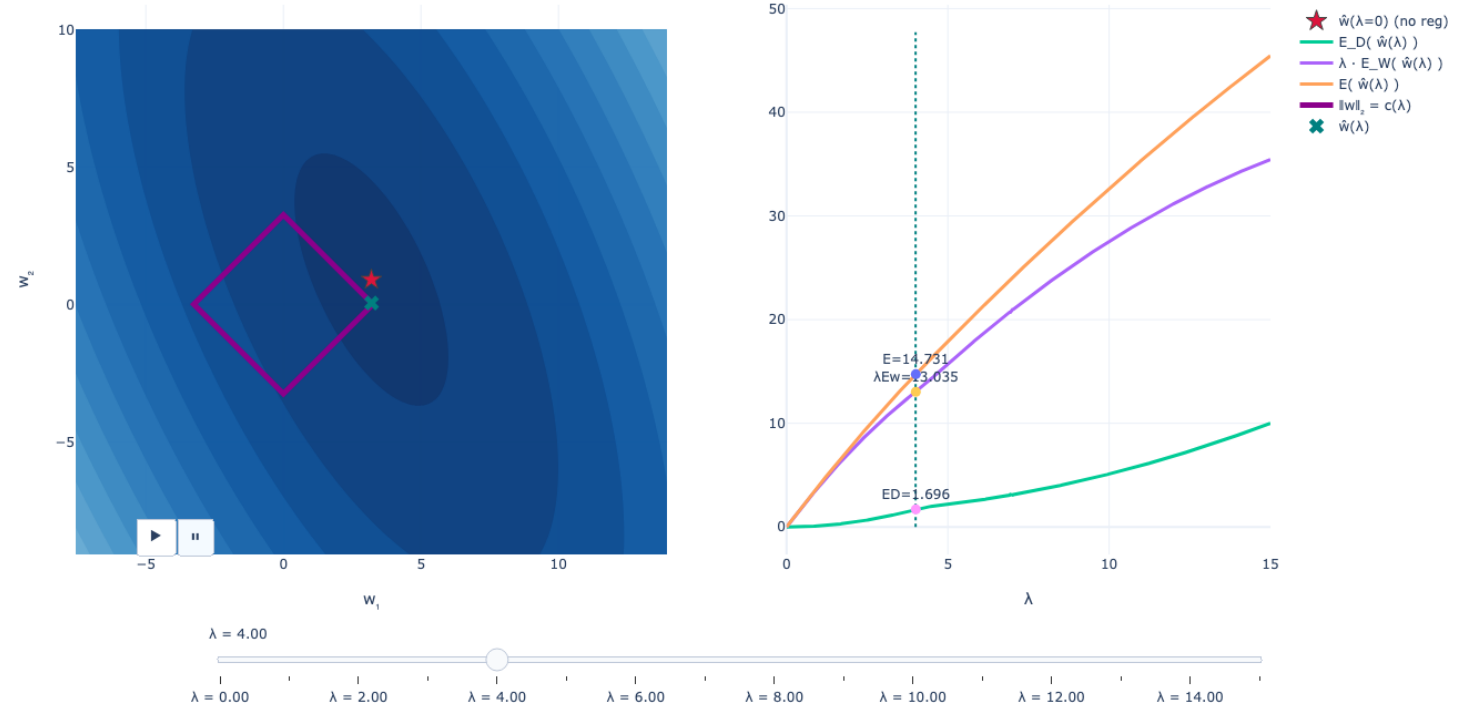


L1 Regularization Demo

[Interactive link](#)



4191216

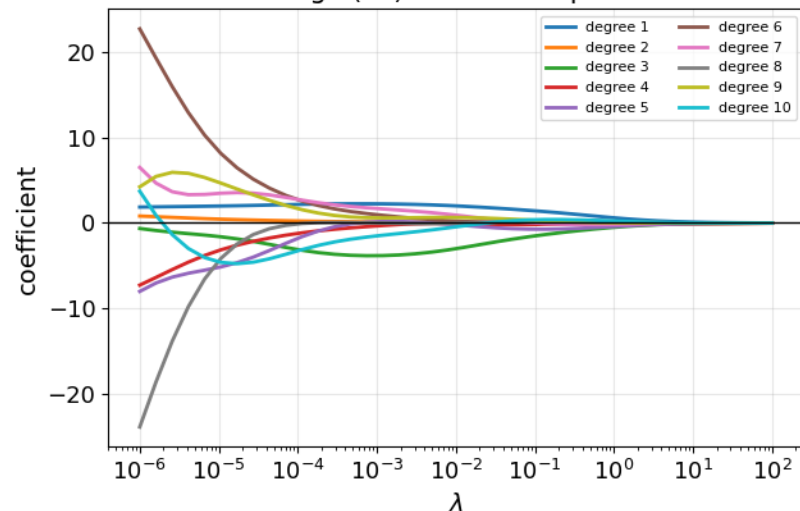


Ridge and Lasso Coefficient Paths

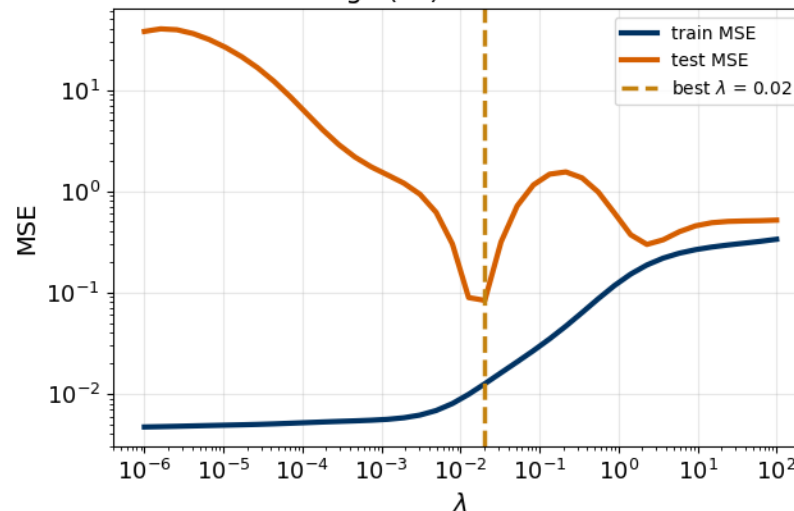


4191216

Ridge (L2): coefficient paths



Ridge (L2): train vs test

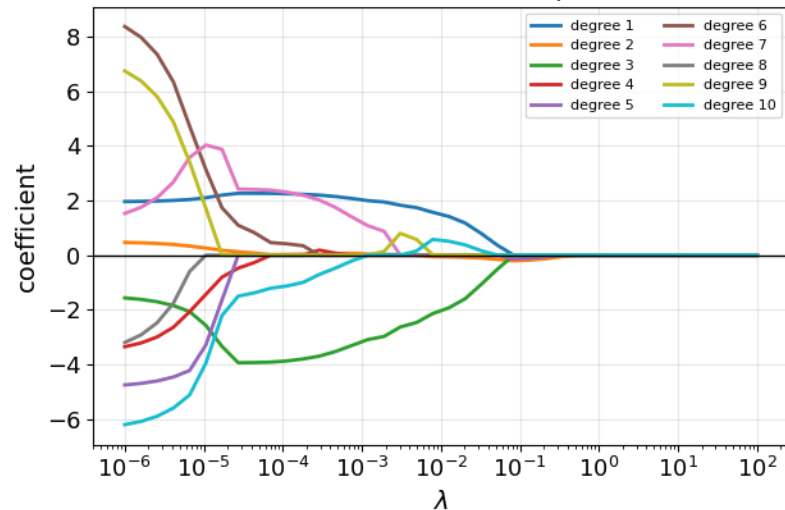


Degree-10 polynomial on 15 training points.

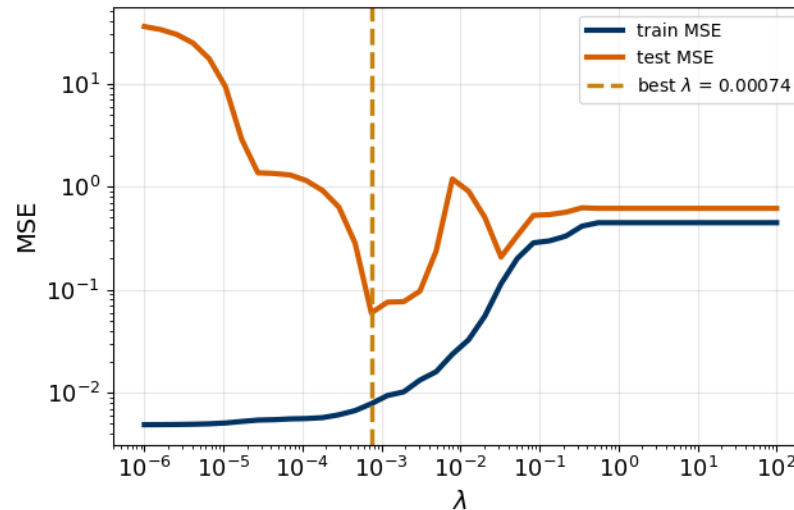
Unregularized test MSE 37.70; ridge 0.084 and lasso 0.060 at their best λ .

Lasso has 5 of 10 coefficients exactly zero there; ridge has none.

Lasso (L1): coefficient paths



Lasso (L1): train vs test



Lasso
Least Absolute Shrinkage and Selection Operator



4191216

When Normal Equation Gets Tricky

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs



4191216

When Normal Equation Gets Tricky

- When is $\mathbf{X}^T \mathbf{X}$ invertible?
 - Full column rank: $\text{rank}(\mathbf{X}) = D + 1$
 - Requires $N \geq D + 1$ (at least as many samples as features)
 - No perfect collinearity: $\det(\mathbf{X}^T \mathbf{X}) \neq 0$
 - All singular values must be non-zero: $\sigma_1, \sigma_2, \dots, \sigma_{D+1} > 0$
- Illconditioning (numerical instability)
 - Even if $\det(\mathbf{X}^T \mathbf{X}) \neq 0$
 - **Condition number** $\kappa = \frac{\sigma_{\max}}{\sigma_{\min}} \gg 10^8$ small noise in the data creates huge weight swing
 - Why?
 - Rounding error will be amplified when calculating $\kappa \epsilon_{\text{machine}}$.



4191216

Fix #1: Ridge Trick

Stabilize with $(\mathbb{X}^T \mathbb{X} + \lambda I) \mathbf{w} = \mathbb{X}^T \mathbf{t}$

- $\lambda > 0$ boosts all singular values $\rightarrow$ lower κ .
- Closed-form solution: $\mathbf{w}^* = (\mathbb{X}^T \mathbb{X} + \lambda I)^{-1} \mathbb{X}^T \mathbf{t}$



4191216

Future lectures

Fix 2: Sequential Learning

- In practice, we use sequential algorithms, also known as online algorithms, in which the data points are considered one at a time and the model parameters updated after each such presentation.
- **Algorithm:** Stochastic gradient descent, also known as sequential gradient descent:
 - If we assume that the error function is calculated based on all samples, then $E = \sum_n E_n$.
 - After presentation of datapoint n , the gradient descent algorithm updates parameters using $w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n$

Known as Least
Mean Squares or
LMS

$$w^{(\tau+1)} = w^{(\tau)} - \eta \left(t_n - w^{(\tau)T} \phi(x_n) \right) \phi(x_n)$$

Learning rate



4191216

MLE Recap: Least Squares as Maximum Likelihood

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs



4191216

Least Squares $\equiv$ Maximum Likelihood

Recall: We fit a **polynomial** by *Minimizing the sum-of-squares error*.

$$E_D(w) = \frac{1}{2} \sum_{n=1}^N (t_n - y(x_n, w))^2$$

Is this just algebraic convenience ... or is there a probabilistic justification?



4191216

Additive-Gaussian Noise Assumption

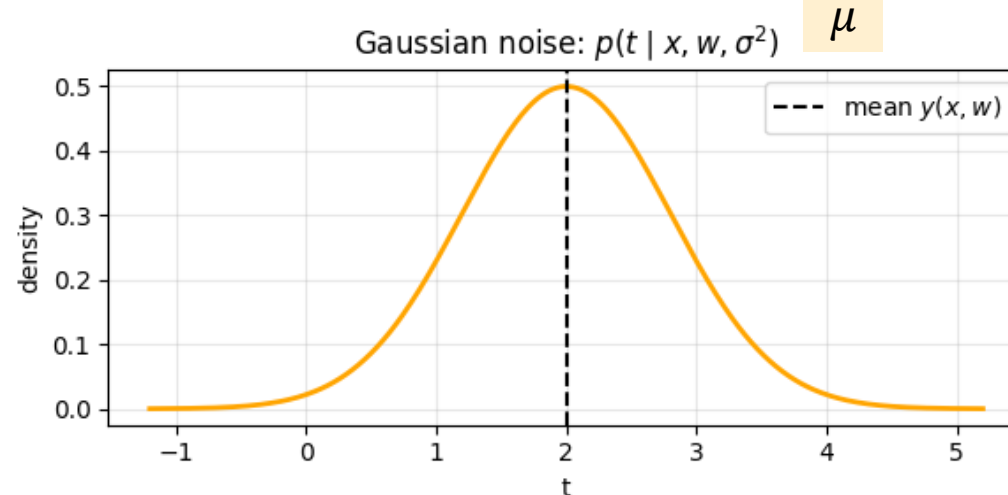
- Assume the target t is the true signal plus noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ (zero-mean Gaussian random variable) $\rightarrow$ i.i.d. across points.

- The conditional likelihood of one observation is:

$$p(t | \mathbf{x}, \mathbf{w}, \sigma^2) = \mathcal{N}(t | y(\mathbf{x}, \mathbf{w}), \sigma^2)$$





4191216

Maximum Likelihood: Multiple Inputs

- Consider the data set of inputs $\mathbb{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ with corresponding target values $t_1, \dots, t_N$. We group the target variables $\{t_n\}$ into a column vector that we denote by $\mathbf{t}$.

$$p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbb{X}, \mathbf{w}), \sigma^2)$$

Datapoints are
independent

$$= \prod_{n=1}^N \mathcal{N}(t_n|\mathbf{w}^T \phi(\mathbf{x}_n), \sigma^2)$$

$$p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2) = \prod_{n=1}^N \mathcal{N}(t_n|\mathbf{w}^T \phi(\mathbf{x}_n), \sigma^2)$$



4191216

Least Squares $\equiv$ Maximum Likelihood

$$p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2) = \prod_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \sigma^2)$$

The optimum value of $\mathbf{w}$ in the above equation equals the optimum value of the log of the function (because log is a monotonically increasing function)

$$\ln(p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2)) = \sum_{n=1}^N \ln(\mathcal{N}(t_n | \overset{\mu}{\mathbf{w}^T \phi(\mathbf{x}_n)}, \sigma^2))$$

Writing the equation for a Normal distribution

$$= \sum_{n=1}^N \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(t_n - \mathbf{w}^T \phi(\mathbf{x}_n))^2}\right)$$

Applying $\log a \times b = \log a + \log b$

$$= \sum_{n=1}^N \left(\ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{1}{2\sigma^2}(t_n - \mathbf{w}^T \phi(\mathbf{x}_n))^2 \right)$$

Simplify and separate two terms

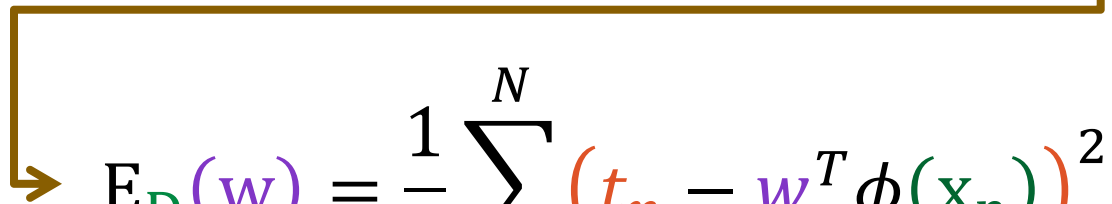
$$\ln(p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2)) = -\frac{N}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^N (t_n - \mathbf{w}^T \phi(\mathbf{x}_n))^2$$



4191216

Least Squares $\equiv$ Maximum Likelihood

$$\begin{aligned}\ln(p(\mathbf{t}|\mathbb{X}, \mathbf{w}, \sigma^2)) &= -\frac{N}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^N (t_n - \mathbf{w}^T \phi(\mathbf{x}_n))^2 \\ &= -\frac{N}{2} \ln(2\pi) - \frac{N}{2} \ln(\sigma^2) - \frac{1}{\sigma^2} E_D(\mathbf{w}) \quad (*)\end{aligned}$$


$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N (t_n - \mathbf{w}^T \phi(\mathbf{x}_n))^2$$

- The first two terms in can be treated as constants when determining $\mathbf{w}$ because they are independent of $\mathbf{w}$.
- Therefore, maximizing the likelihood function under a Gaussian noise distribution is equivalent to minimizing the sum-of-squares error function.

Choosing Different Noise Models

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs





4191216

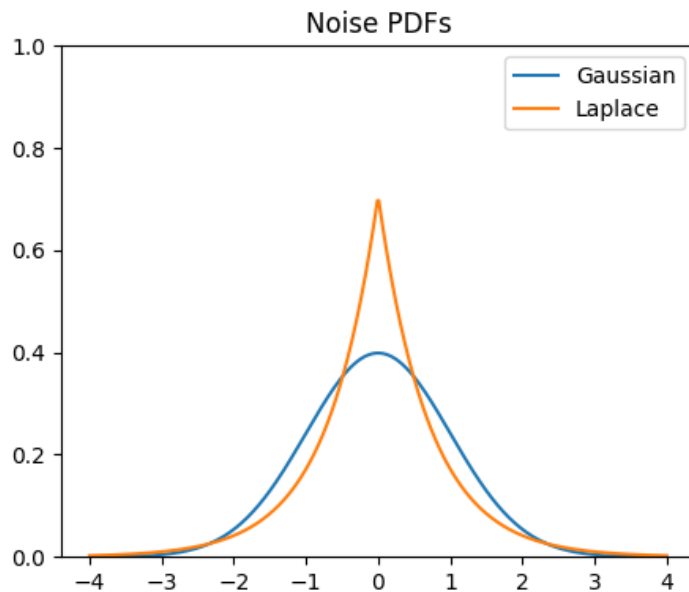
Noise Model $\Leftrightarrow$ Error Function

Zero-mean Gaussian Noise

$$p(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\epsilon^2}{2\sigma^2}\right)$$

$$-\log p(\epsilon) = \frac{\epsilon^2}{2\sigma^2} + \text{const}$$

$\Rightarrow$ Sum of the squares error

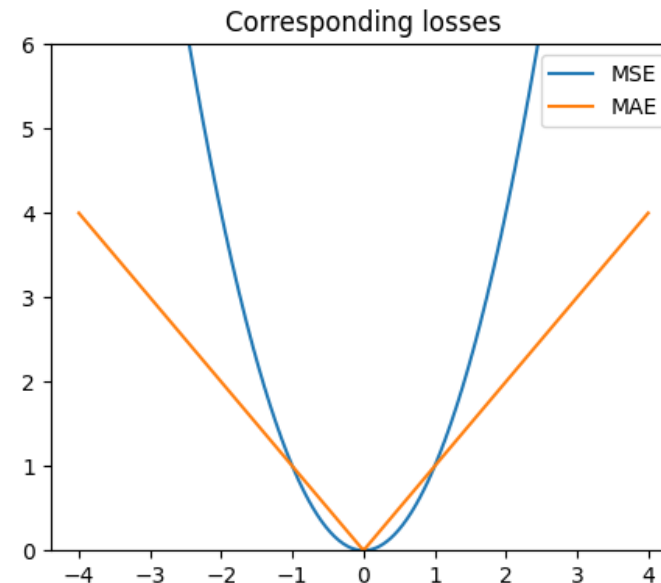


Zero-mean Laplacian Noise

$$p(\epsilon) = \frac{1}{2b} \exp\left(-\frac{|\epsilon|}{b}\right)$$

$$-\log p(\epsilon) = \frac{|\epsilon|}{b} + \text{const}$$

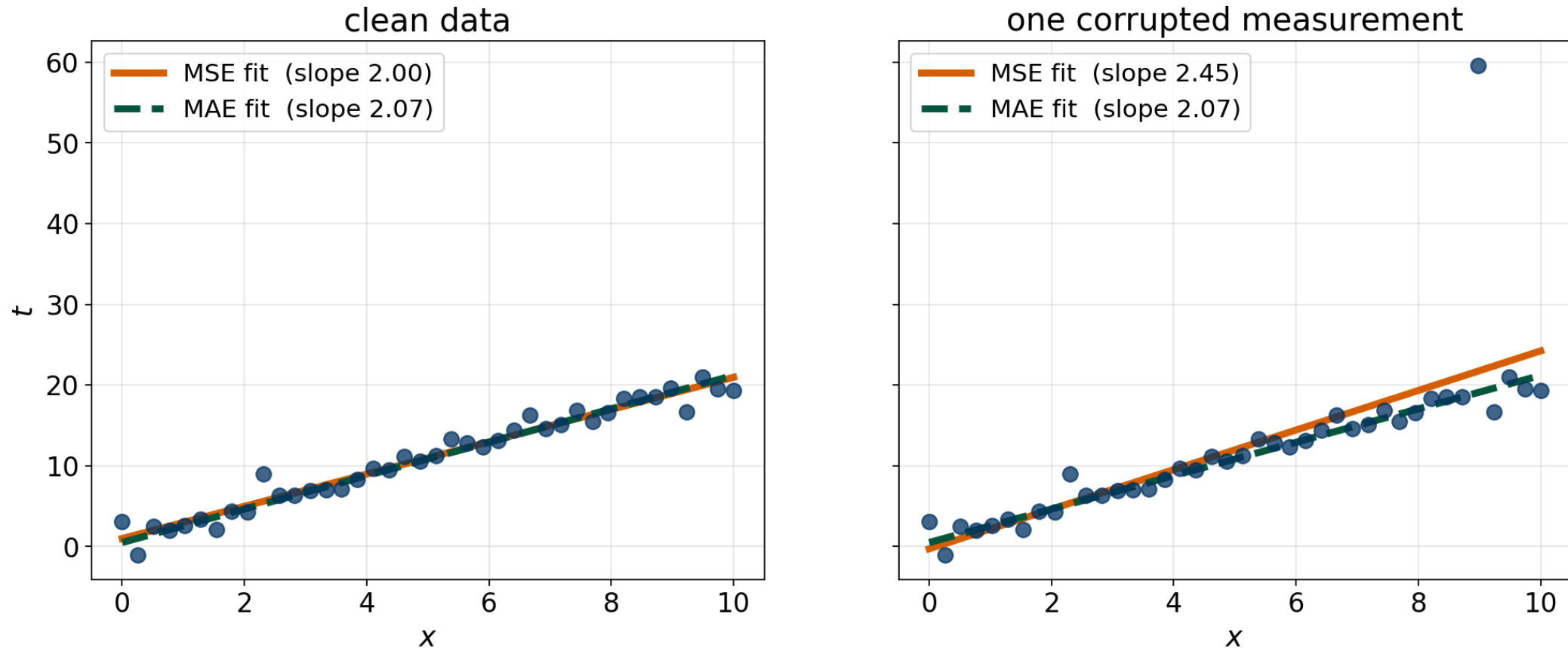
$\Rightarrow$ Sum of the Absolute error



Sensitivity of MSE and MAE to a Single Outlier



4191216



One corrupted measurement out of 40. The squared-error fit moves from slope 2.00 to 2.45; the absolute-error fit stays at 2.07.



4191216

Prior Beliefs

- Regularized Least Squares
- When Normal Equation Gets Tricky
- MLE Recap: Least Squares as Maximum Likelihood
- Choosing Different Noise Models
- Prior Beliefs



4191216

Recall: Beliefs and Priors

In the coin flip example, assume the true probability of head is θ .

- **Frequentist:** Use data only (MLE).
- **Bayesian:** Combine data with belief.

$$\hat{\theta} = \frac{\#heads}{\#flips}$$

- Prior belief: $h_0 = t_0 = 5$

- See HH:

Posterior $p(\theta|HH) \propto p(\theta) \times \text{likelihood}$

Flip twice, see HH, then:

$$\hat{\theta}_{MLE} = \frac{2}{2} = 1$$

Strong prior ensures alignment with beliefs

Chases the data
 $p(HH|\theta) = \theta^2$

Overconfident with $n = 2!!!$

$$\hat{\theta}_{posterior\ mean} = \frac{5 + 2}{10 + 2} = 0.58$$

0 ——— 0.5 (prior) ——— 0.58 (Posterior mean) ——— 1.0 (MLE)

What About Regression? MLE for Weights: No Prior



4191216

- In regression, MLE finds weights $\mathbf{w}$ by maximizing likelihood of data under noise model.
- **Problem:** with few/noisy data points $\rightarrow$ can fit extreme weights.
- No mechanism to encode “weights should be small/reasonable.”
- Why small?
 - With inputs in SD units we don't expect huge effects unless data is persuasive.
 - Shrinking $\mathbf{w}$ lowers prediction variance, especially when features are correlated.



Belief about Parameters

- Just like coins: we can encode a prior belief about weights $\mathbf{w}$.
- Common prior: $w_j \sim \mathcal{N}(0, \sigma_w^2) = \mathcal{N}(0, \tau^2)$
 - Centered at 0 $\rightarrow$ no strong effect unless data supports it.
 - Variance σ_w^2 controls how strongly we expect small weights.
- This prior = belief that simpler models are preferred.

Likelihood function

$$p(\mathbf{t} | \mathbb{X}, \mathbf{w}, \sigma^2) = \prod_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \sigma^2)$$

Prior (small, centered)

$$p(\mathbf{w} | \tau^2) = \frac{1}{(2\pi\tau^2)^{\frac{M}{2}}} e^{-\frac{\|\mathbf{w}\|^2}{2\tau^2}}$$

Posterior (from Bayes rule) up to a constant:

$$p(\mathbf{w} | \mathbf{t}) \propto$$



4191216

What Is MAP?

Bayes Rule

$$p(\mathbf{w} \mid \mathbf{t}) \propto \underbrace{\prod_{n=1}^N \mathcal{N}(t_n \mid \mathbf{w}^T \phi(\mathbf{x}_n), \sigma^2)}_{\text{Likelihood}} \underbrace{\frac{1}{(2\pi\tau^2)^{\frac{M}{2}}} e^{-\frac{\|\mathbf{w}\|^2}{2\tau^2}}}_{\text{Prior}}$$

Maximum A Posteriori (MAP):

$$\hat{\mathbf{w}}_{MAP} = \underset{\mathbf{w}}{\operatorname{argmax}} p(\mathbf{w} \mid \mathbf{t}) = \underset{\mathbf{w}}{\operatorname{argmax}} [p(\mathbf{t} \mid \mathbf{w}) \times p(\mathbf{w})]$$



4191216

What Is MAP?

Maximum A Posteriori (MAP):

$$\hat{w}_{MAP} = \underset{w}{\operatorname{argmax}} p(w | t) = \underset{w}{\operatorname{argmax}} [p(t | w) \times p(w)]$$

Taking $-\log$ of the objective turns maximization problem to a minimization problem:

$$\hat{w}_{MAP} = \underset{w}{\operatorname{argmin}} [-\ln p(t | w) - \ln p(w)]$$



4191216

Plugging in Distributions

$$\begin{aligned}\hat{w}_{MAP} &= \underset{w}{\operatorname{argmin}} \left[\underbrace{-\ln p(\mathbf{t} | w)}_{-\ln \prod_{n=1}^N \mathcal{N}(t_n | w^T \phi(\mathbf{x}_n), \sigma^2)} - \underbrace{\ln p(w)}_{-\ln \frac{1}{(2\pi\tau^2)^{\frac{M}{2}}} e^{-\frac{\|w\|^2}{2\tau^2}}} \right] \\ &= \frac{1}{2\sigma^2} \sum_n (t_n - y(\phi(\mathbf{x}), w))^2 = \frac{\|w\|^2}{2\tau^2} + \text{const} \\ \hat{w}_{MAP} &= \underset{w}{\operatorname{argmin}} \left[\frac{1}{2\sigma^2} \sum_n (t_n - y(\phi(\mathbf{x}), w))^2 + \frac{\|w\|^2}{2\tau^2} \right]\end{aligned}$$



4191216

Does This Look Like a Ridge Regression?

$$\hat{w}_{MAP} = \underset{w}{\operatorname{argmin}} \left[\frac{1}{2\sigma^2} \sum_n \left(t_n - y(\phi(x), w) \right)^2 + \frac{\|w\|^2}{2\tau^2} \right]$$

Multiply by σ^2

$$\hat{w}_{MAP} = \underset{w}{\operatorname{argmin}} \left[\underbrace{\frac{1}{2} \sum_n \left(t_n - y(\phi(x), w) \right)^2}_{E_D(w)} + \underbrace{\frac{\sigma^2 \|w\|^2}{2\tau^2}}_{E_w(w)} \right]$$

λ

Least-Squares + Gaussian Prior $\Rightarrow$ Ridge Regression



What is the relationship between MAP and ridge regression?

Lecture 7

Regression (Cont.) and Bias-Variance Trade-off

Credit: Joseph E. Gonzalez and Narges Norouzi

Reference Book Chapters: Chapter 4.1 - 4.3