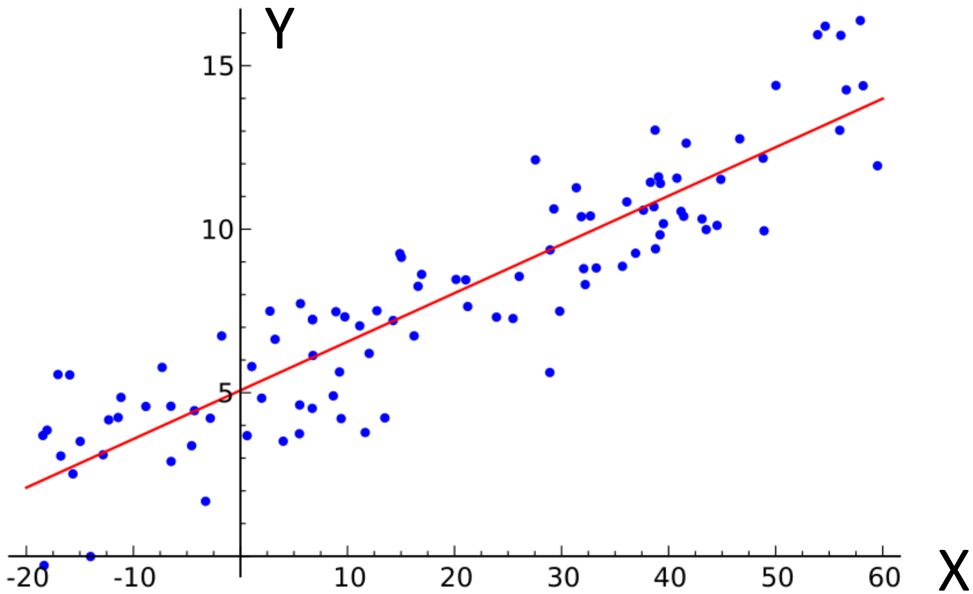
Linear Regression and the Bias Variance Tradeoff

Guest Lecturer
Joseph E. Gonzalez

slides available here: **http://tinyurl.com/reglecture**

Simple Linear Regression



Response Variable

Covariate

Linear Model: $$Y = mX + b$$

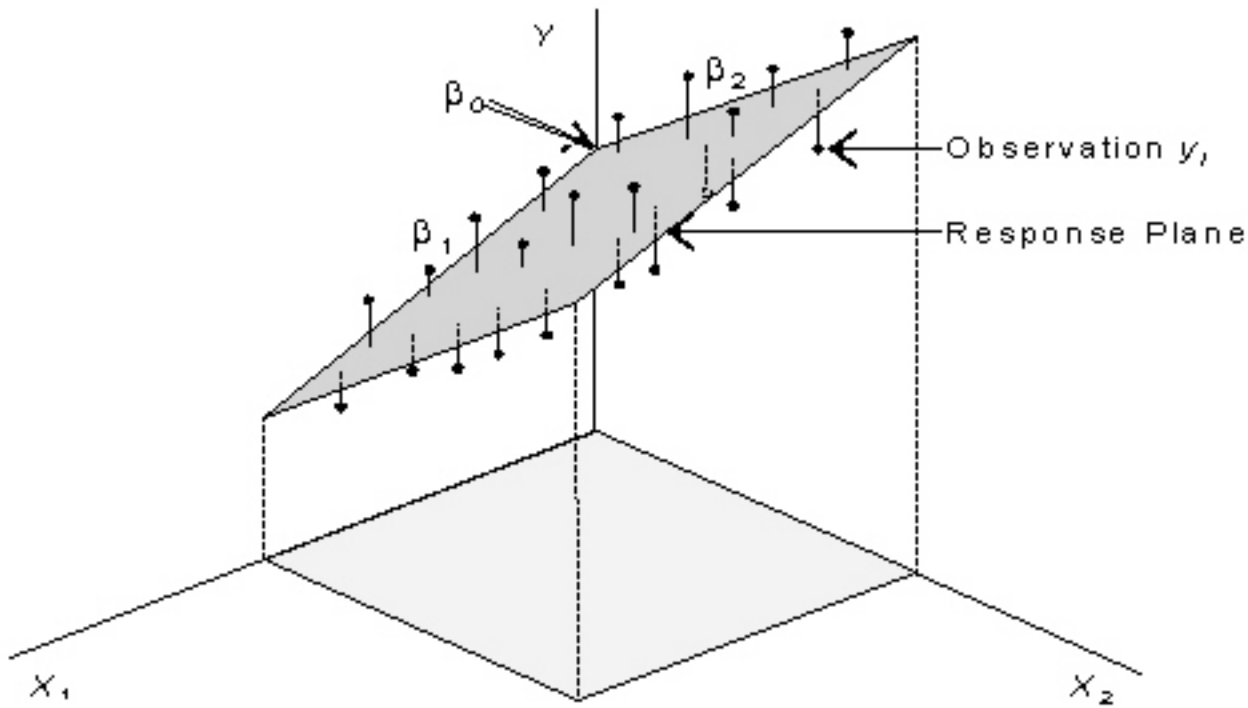
Slope

Intercept (bias)

Motivation

- One of the most widely used techniques
- Fundamental to many larger models
  - Generalized Linear Models
  - Collaborative filtering
- Easy to interpret
- Efficient to solve

Multiple Linear Regression



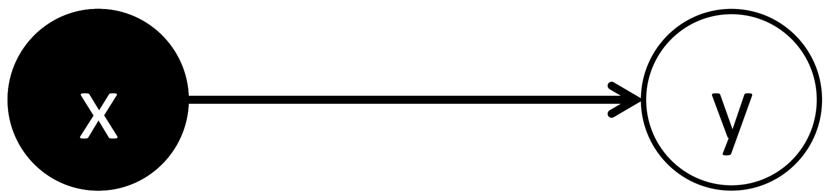
The Regression Model

- For a *single* data point *(x,y)*:

Independent Variable (Vector)

Response Variable (Scalar)

Observe: (Condition)



$x \in \mathbb{R}^p$ $\qquad$ $y \in \mathbb{R}$

- Joint Probability:

$$p(x, y) = p(x)\boxed{p(y|x)}$$

Discriminative Model


END DUALITY GAP
SUPPORT VECTOR MACHINES
BAYESIANS AGAINST DISCRIMINATION

The Linear Model

Vector of Parameters

Vector of Covariates

Scalar Response

$$y = \underbrace{\theta^T x}_{\sum_{i=1}^{p} \theta_i x_i} + \epsilon \quad + \text{b}$$

Real Value Noise

Noise Model:

$$\epsilon \sim N(0, \sigma^2)$$

Linear Combination of Covariates

$$\sum_{i=1}^{p} \theta_i x_i$$

What about bias/intercept term?

$$\text{Define: } x_{p+1} = 1$$

Then redefine p := p+1 for notational simplicity

Conditional Likelihood p(y|x)

- Conditioned on x:

$$y = \overbrace{\theta^T x}^{\text{Constant}} + \boxed{\epsilon \sim \overset{\text{Normal Distribution}}{N}(\underset{\text{Mean}}{0}, \underset{\text{Variance}}{\sigma^2})}$$

- Conditional distribution of Y:

$$Y \sim N(\theta^T x, \sigma^2)$$

$$p(y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\theta^T x)^2}{2\sigma^2}\right)$$

Parameters and Random Variables

Parameters

$$y \sim N(\boxed{\theta^T} x, \boxed{\sigma^2})$$

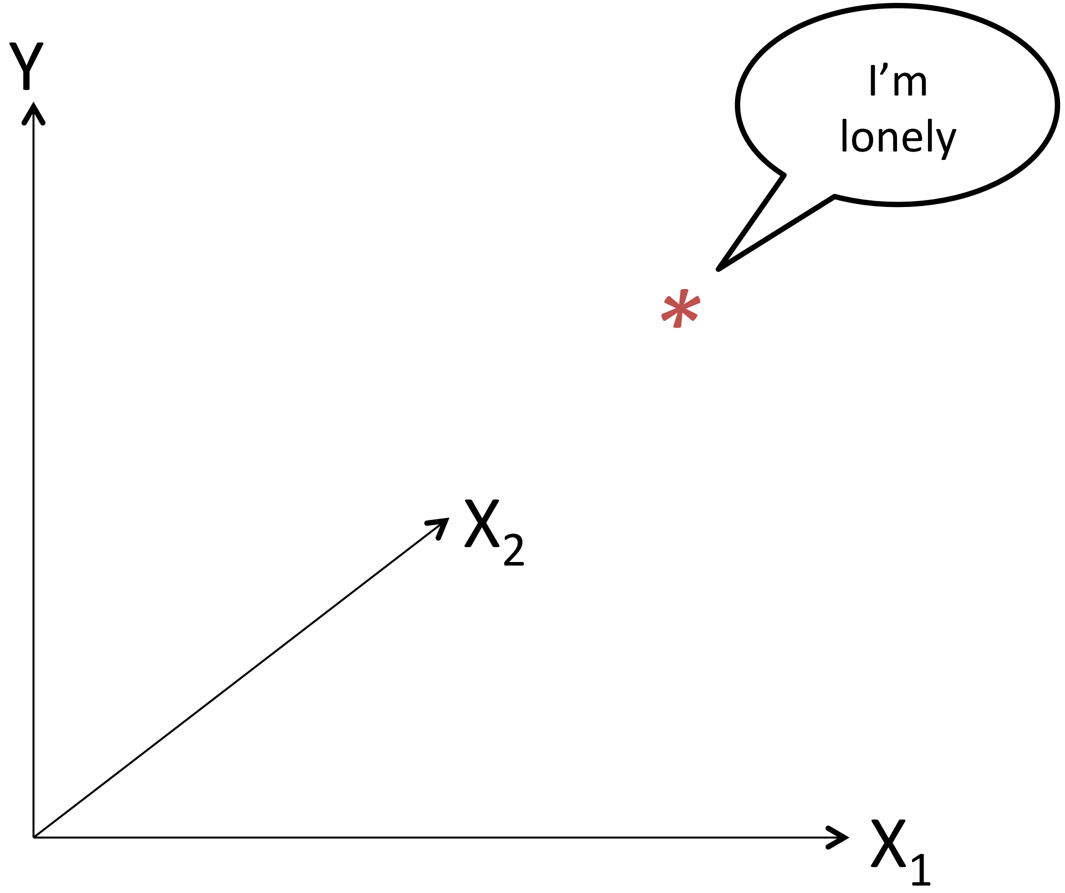
- Conditional distribution of y:
  - Bayesian: parameters as random variables

$$p(y|x, \theta, \sigma^2)$$

  - Frequentist: parameters as (unknown) constants

$$p_{\theta, \sigma^2}(y|x)$$

So far …

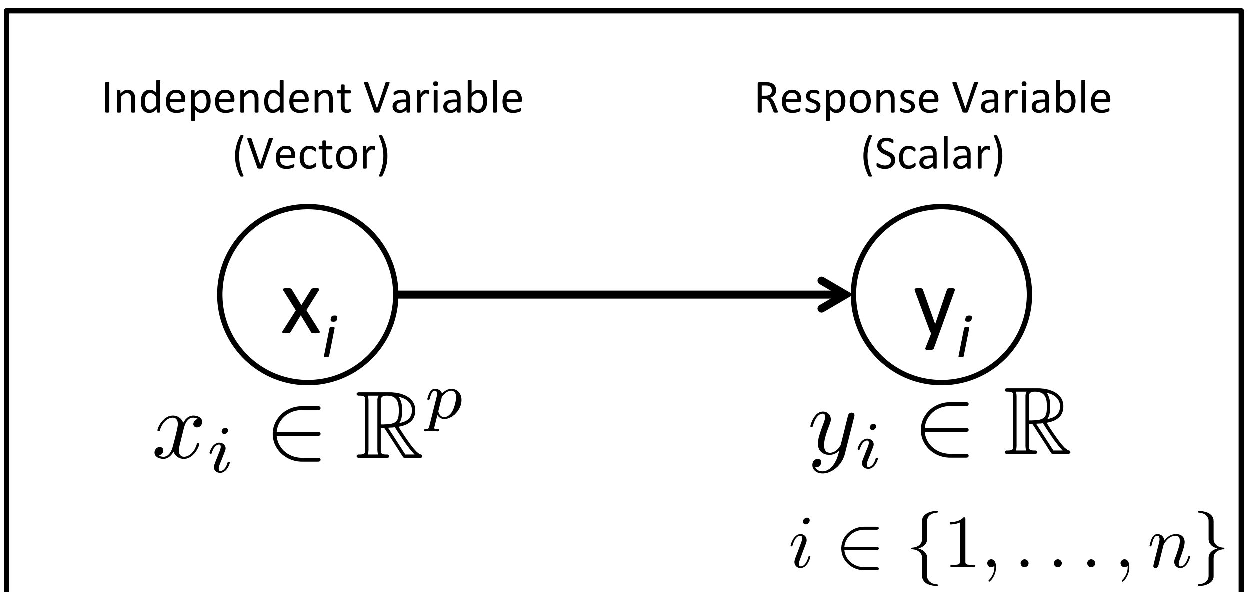


Independent and Identically Distributed (iid) Data

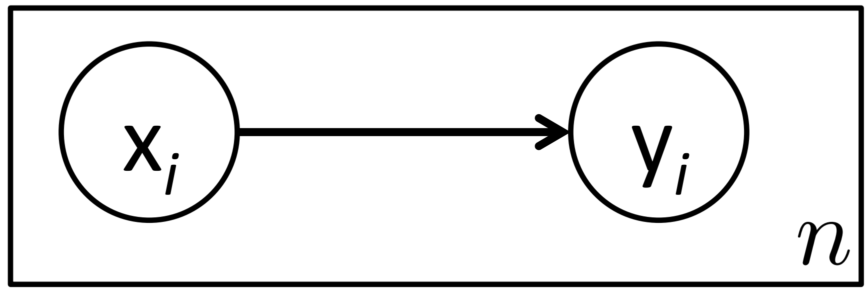
- For *n* data points:

$$\begin{aligned}\mathcal{D} &= \{(x_1, y_1), \ldots, (x_n, y_n)\} \\ &= \{(x_i, y_i)\}_{i=1}^{n}\end{aligned}$$

Plate Diagram



Joint Probability



- For *n* data points **independent and identically distributed (iid)**:

$$p(\mathcal{D}) = \prod_{i=1}^{n} p(x_i, y_i)$$

$$= \prod_{i=1}^{n} p(x_i)p(y_i|x_i)$$

Rewriting with Matrix Notation

- Represent data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ as:

Covariate (Design) Matrix

$$X = n\left\{\underbrace{\begin{pmatrix} — & x_1 & — \\ — & x_2 & — \\ & \dots & \\ — & x_n & — \end{pmatrix}}_{p}\right. \in \mathbb{R}^{np}$$

Assume *X* has rank p (not degenerate)

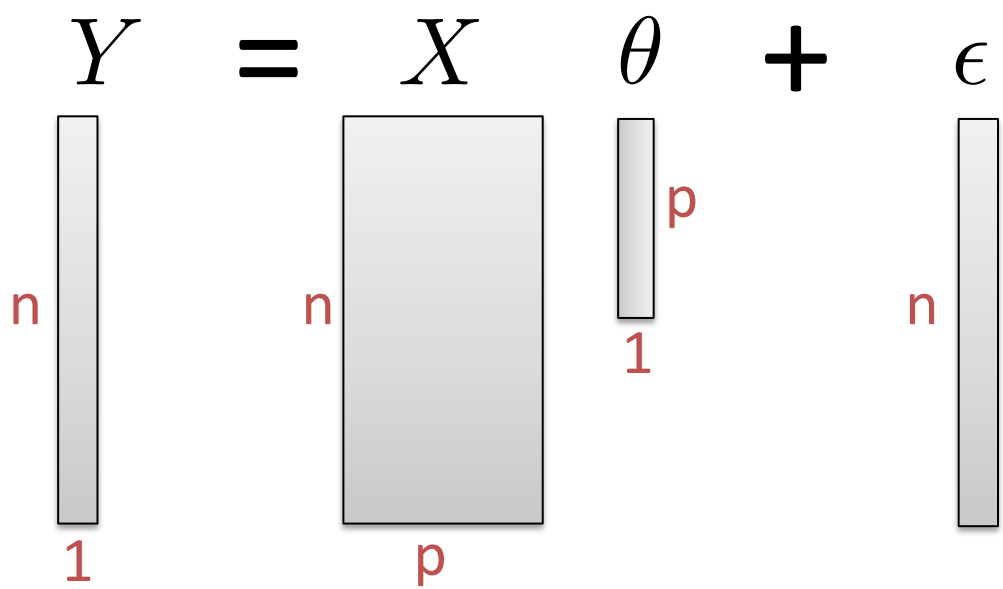
Response Vector

$$Y = n\left\{\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{1}\right. \in \mathbb{R}^{n}$$

Rewriting with Matrix Notation

- Rewriting the model using matrix operations:

$$Y = X\theta + \epsilon$$



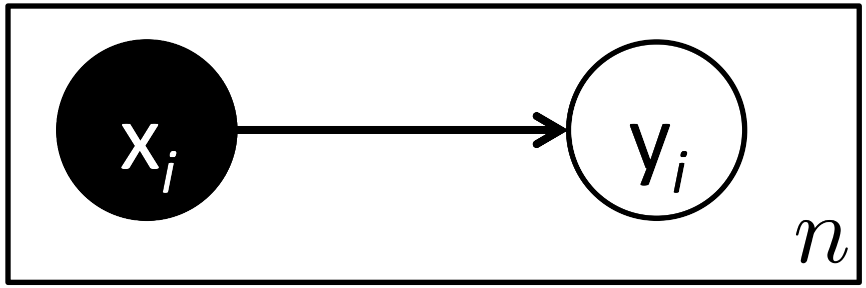
Estimating the Model

- Given data how can we estimate θ?

$$Y = X\theta + \epsilon$$

- Construct maximum likelihood estimator (MLE):
  - Derive the log-likelihood
  - Find θ_{MLE} that maximizes log-likelihood
    - Analytically: Take derivative and set = 0
    - Iteratively: (Stochastic) gradient descent

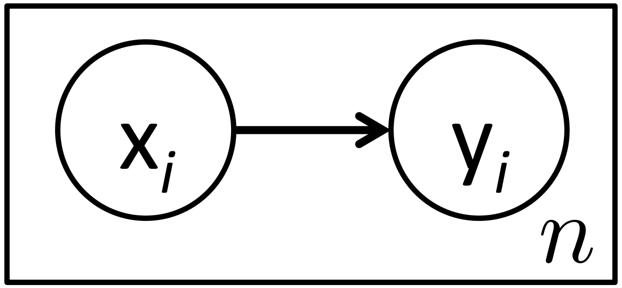
Joint Probability



- For *n* data points:

$$
\begin{aligned}
p(\mathcal{D}) &= \prod_{i=1}^{n} p(x_i, y_i) \\
&= \prod_{i=1}^{n} \cancelto{\text{“1”}}{p(x_i)} \boxed{p(y_i | x_i) \quad \text{Discriminative Model}}
\end{aligned}
$$

Defining the Likelihood



$$p_\theta(y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\theta^T x)^2}{2\sigma^2}\right)$$

$$\begin{aligned}
\mathcal{L}(\theta|\mathcal{D}) &= \prod_{i=1}^{n} p_\theta(y_i|x_i) \\
&= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i-\theta^T x_i)^2}{2\sigma^2}\right) \\
&= \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\theta^T x_i)^2\right)
\end{aligned}$$

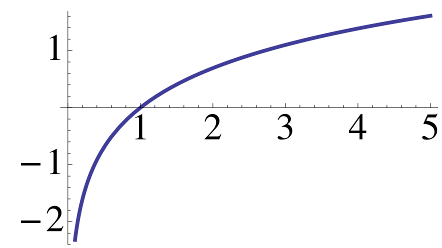
Maximizing the Likelihood

- Want to compute:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} \mathcal{L}(\theta | \mathcal{D})$$

- To simplify the calculations we take the log:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} \boxed{\log} \mathcal{L}(\theta | \mathcal{D})$$



which does not affect the maximization because log is a monotone function.

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \theta^T x_i)^2\right)$$

- Take the log:

$$\log \mathcal{L}(\theta|\mathcal{D}) = -\log(\sigma^n (2\pi)^{\frac{n}{2}}) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \theta^T x_i)^2$$

- Removing constant terms with respect to θ:

$$\log \mathcal{L}(\theta) = -\sum_{i=1}^{n}(y_i - \theta^T x_i)^2$$

Monotone Function
(Easy to maximize)

$$\log \mathcal{L}(\theta) = -\sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

- Want to compute:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} \log \mathcal{L}(\theta | \mathcal{D})$$

- Plugging in log-likelihood:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} -\sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

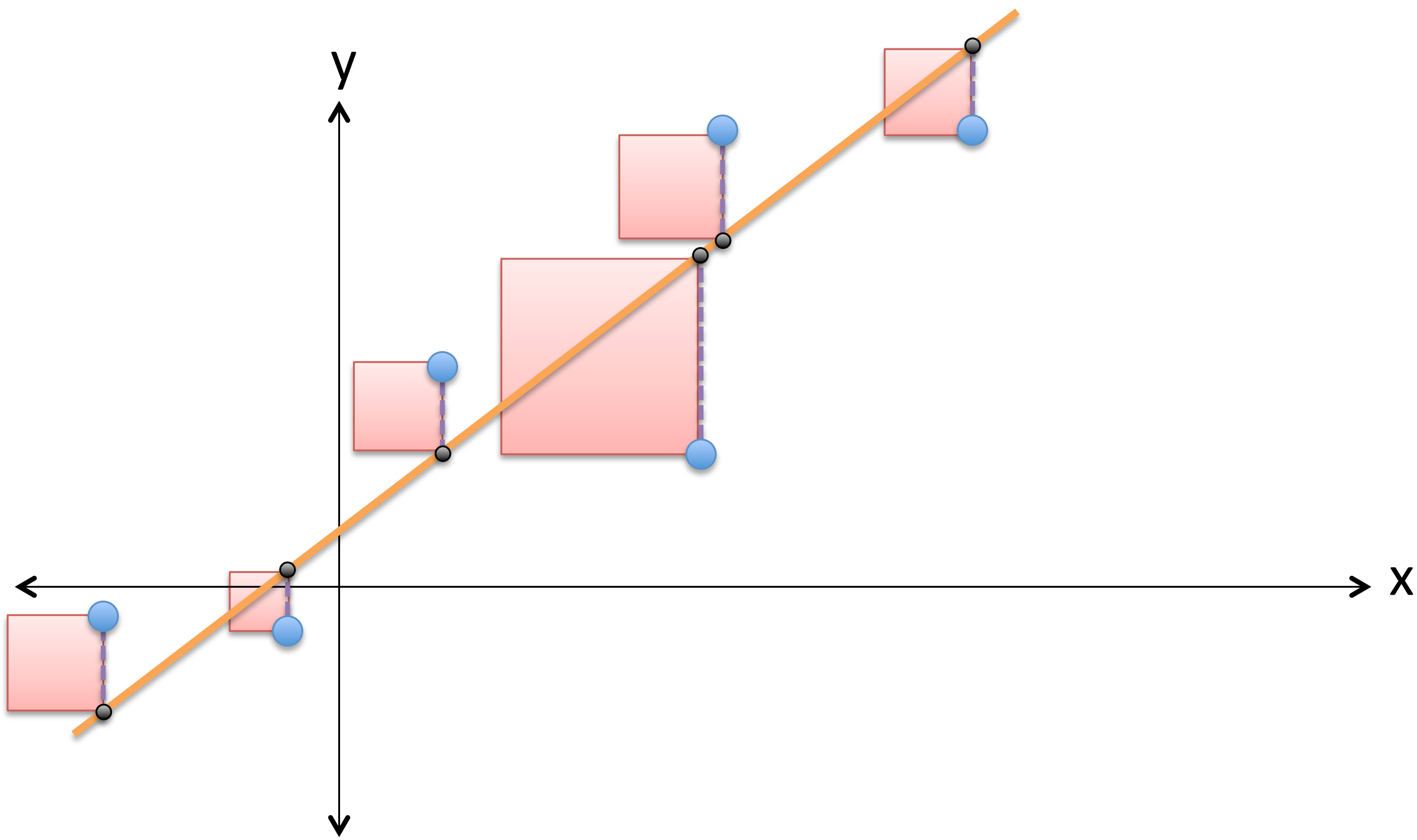
$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} - \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

- Dropping the sign and flipping from maximization to minimization:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} \underbrace{(y_i - \theta^T x_i)^2}_{\text{Minimize Sum (Error)}^2}$$

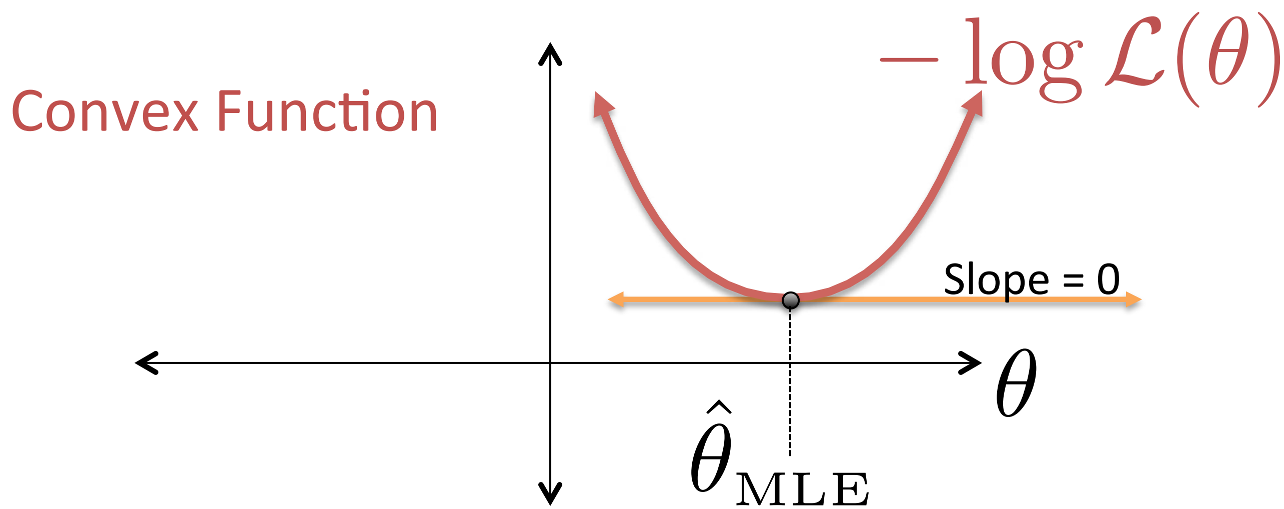
- Gaussian Noise Model → Squared Loss
  - Least Squares Regression

Pictorial Interpretation of Squared Error



Maximizing the Likelihood (Minimizing the Squared Error)

$$\hat{\theta}_{\mathrm{MLE}} = \arg\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$



- Take the gradient and set it equal to zero

Minimizing the Squared Error

$$\hat{\theta}_{\text{MLE}} = \arg\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

- Taking the gradient

$$-\nabla_\theta \log \mathcal{L}(\theta) = \nabla_\theta \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

Chain Rule →

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

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

- Rewriting the gradient in matrix form:

$$
\begin{aligned}
-\nabla_\theta \log \mathcal{L}(\theta) &= -2 \sum_{i=1}^{n} y_i x_i + 2 \sum_{i=1}^{n} (\theta^T x_i) x_i \\
&= -2X^T Y + 2X^T X\theta
\end{aligned}
$$

- To make sure the log-likelihood is convex compute the second derivative (Hessian)

$$
-\nabla^2 \log \mathcal{L}(\theta) = 2X^T X
$$

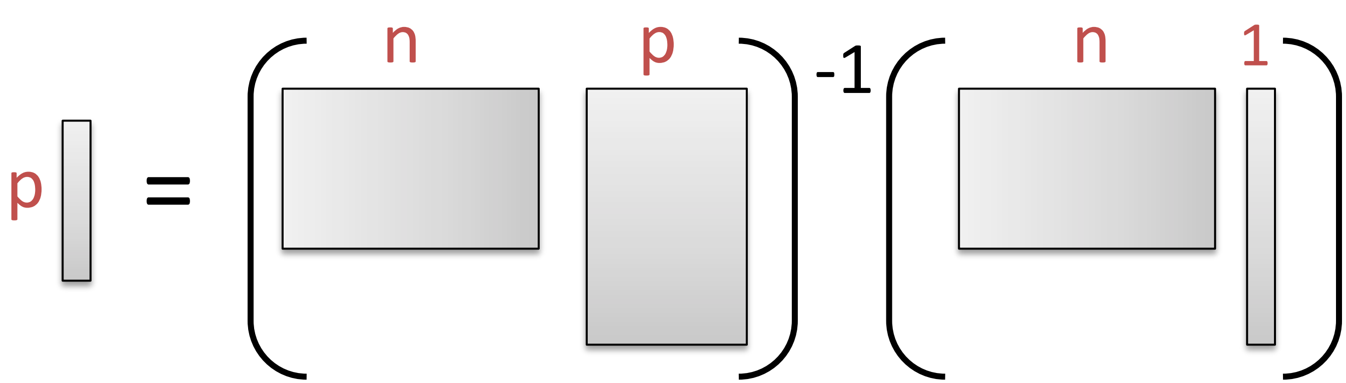
- If X is full rank then $X^T X$ is positive definite and therefore θ_{MLE} is the minimum
  - Address the degenerate cases with regularization

$$-\nabla_\theta \log \mathcal{L}(\theta) = -2X^T y + 2X^T X\theta = 0$$

- Setting gradient equal to 0 and solve for θ_{MLE}:

$$\boxed{\begin{aligned}(X^T X)\hat{\theta}_{\text{MLE}} &= X^T Y \\ \hat{\theta}_{\text{MLE}} &= (X^T X)^{-1} X^T Y\end{aligned}}$$

Normal Equations
(Write on board)

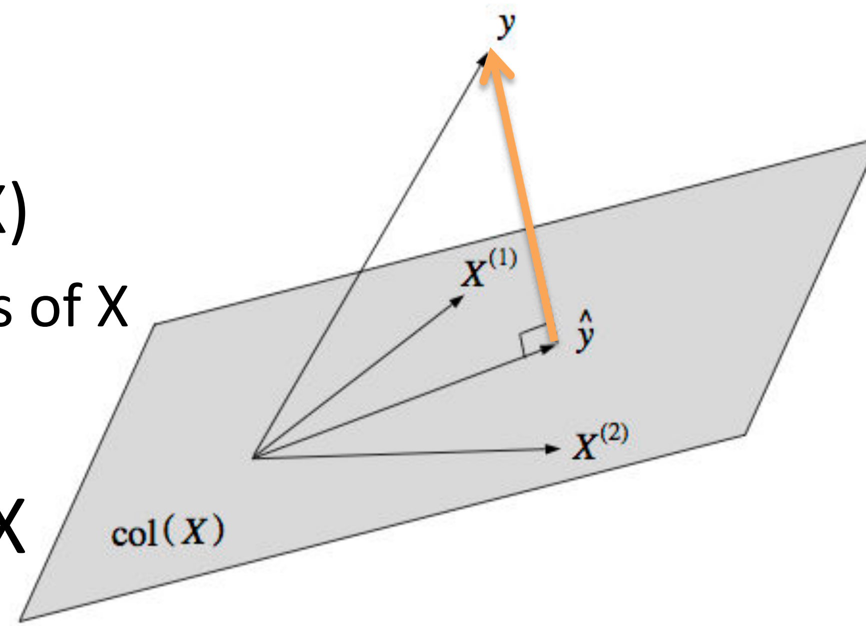


Geometric Interpretation

- View the MLE as finding a projection on col(X)
  - Define the estimator:

$$\hat{Y} = X\theta$$

  - Observe that Ŷ is in col(X)
    - linear combination of cols of X
  - Want to Ŷ closest to Y
- Implies (Y-Ŷ) normal to X



$$X^T(Y - \hat{Y}) = X^T(Y - X\theta) = 0$$

$$\Rightarrow X^T X\theta = X^T Y$$

Connection to Pseudo-Inverse

$$\hat{\theta}_{\mathrm{MLE}} = \underbrace{(X^T X)^{-1} X^T}_{X^{\dagger}} Y$$

Moore-Penrose Psuedoinverse $X^{\dagger}$

- Generalization of the inverse:
  - Consider the case when X is square and invertible:

$$X^{\dagger} = (X^T X)^{-1} X^T = X^{-1} (X^T)^{-1} X^T = X^{-1}$$

  - Which implies $\theta_{\mathrm{MLE}} = X^{-1} Y$ the solution to $X\theta = Y$ when X is square and invertible

Computing the MLE

$$\hat{\theta}_{\mathrm{MLE}} = (X^T X)^{-1} X^T Y$$

- **Not** typically solved by inverting $X^T X$
- Solved using direct methods:
  - Cholesky factorization:
    - Up to a factor of 2 faster
  - QR factorization:
    - More numerically stable

or use the
built-in solver
in your math library.
R: solve(Xt %*% X, Xt %*% y)

- Solved using various iterative methods:
  - Krylov subspace methods
  - (Stochastic) Gradient Descent

http://www.seas.ucla.edu/~vandenbe/103/lectures/qr.pdf

Cholesky Factorization

$$\text{solve}\ \hat{\theta}_{\text{MLE}} \quad \underbrace{(X^T X)}_{C}\hat{\theta}_{\text{MLE}} = \underbrace{X^T Y}_{d}$$

- Compute symm. matrix $C = X^T X$ $O(np^2)$
- Compute vector $d = X^T Y$ $O(np)$
- Cholesky Factorization $LL^T = C$ $O(p^3)$
  - L is lower triangular
- Forward subs. to solve: $Lz = d$ $O(p^2)$
- Backward subs. to solve: $L^T \hat{\theta}_{\text{MLE}} = z$ $O(p^2)$

Connections to graphical model inference:
http://ssg.mit.edu/~willsky/publ_pdfs/185_pub_MLR.pdf and
http://yaroslavvb.blogspot.com/2011/02/junction-trees-in-numerical-analysis.html with illustrations

Solving Triangular System

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ & A_{22} & A_{23} & A_{24} \\ & & A_{33} & A_{34} \\ & & & A_{44} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Solving Triangular System

$A_{11}x_1$	$A_{12}x_2$	$A_{13}x_3$	$A_{14}x_4$	b_1
	$A_{22}x_2$	$A_{23}x_3$	$A_{24}x_4$	b_2
		$A_{33}x_3$	$A_{34}x_4$	b_3
			$A_{44}x_4$	b_4

$$x_1 = \frac{b_1 - A_{12}x_2 - A_{13}x_3 - A_{14}x_4}{A_{11}}$$

$$x_2 = \frac{b_2 - A_{23}x_3 - A_{24}x_4}{A_{22}}$$

$$x_3 = \frac{(b_3 - A_{34}x_4)}{A_{33}}$$

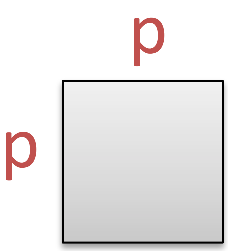
$$x_4 = b_4 / A_{44}$$

Bonus Content

Distributed Direct Solution (Map-Reduce)

$$\hat{\theta}_{\mathrm{MLE}} = (X^T X)^{-1} X^T Y$$

- Distribution computations of sums:



$$C = X^T X = \sum_{i=1}^{n} x_i x_i^T \qquad O(np^2)$$

$$d = X^T y = \sum_{i=1}^{n} x_i y_i \qquad O(np)$$

- Solve system $C\,\theta_{\mathrm{MLE}} = d$ on master. $O(p^3)$

Gradient Descent:

What if p is large? (e.g., n/2)

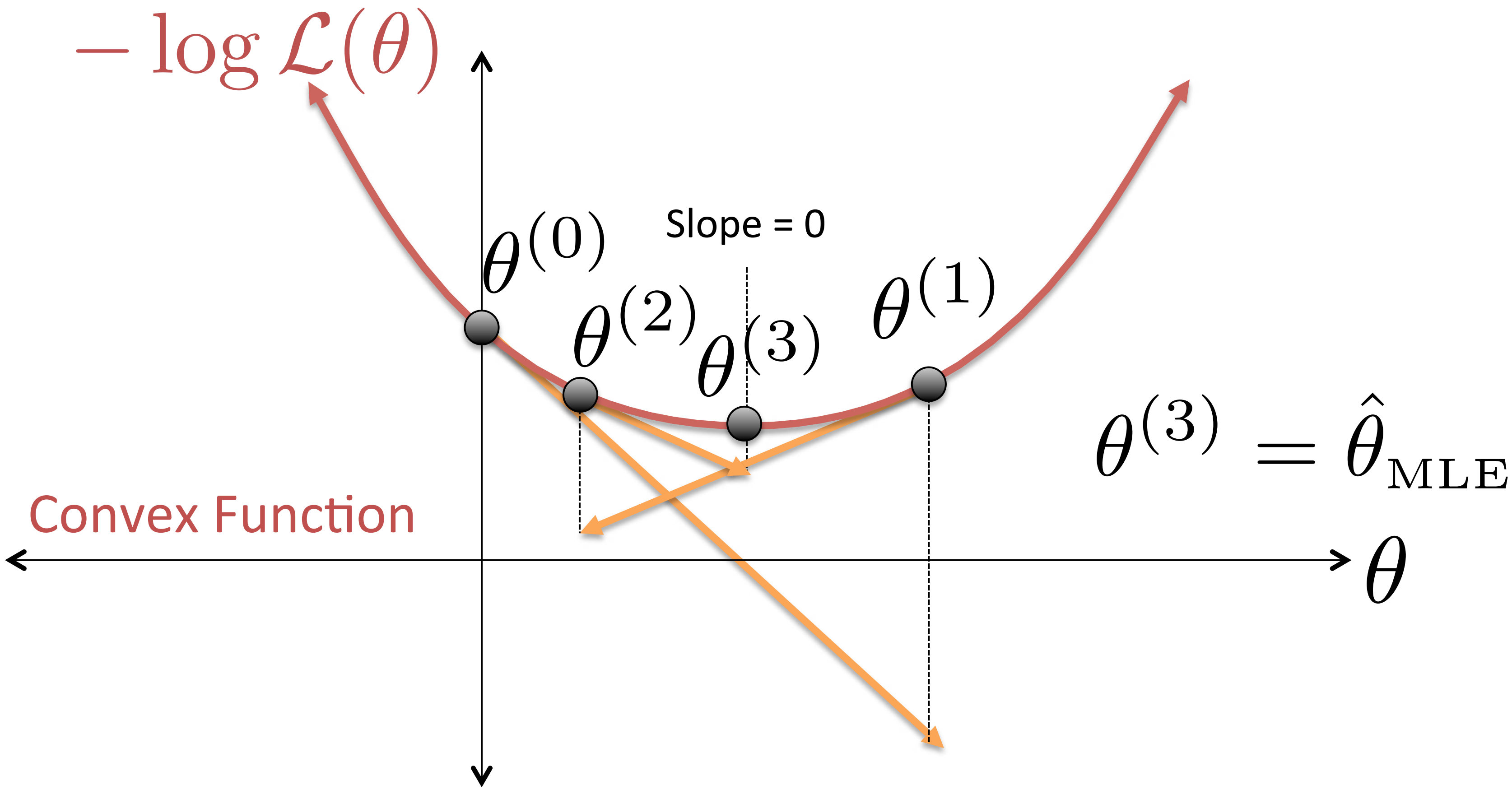
- The cost of $O(np^2) = O(n^3)$ could by prohibitive
- Solution: Iterative Methods
  - Gradient Descent:

For τ from 0 until *convergence*

$$\theta^{(\tau+1)} = \theta^{(\tau)} - \rho(\tau)\nabla \log \mathcal{L}(\theta^{(\tau)} | D)$$

Learning rate

Gradient Descent Illustrated:



Gradient Descent:

What if p is large? (e.g., n/2)

- The cost of $O(np^2) = O(n^3)$ could by prohibitive
- Solution: Iterative Methods
  - Gradient Descent:

For τ from 0 until *convergence*

$$\theta^{(\tau+1)} = \theta^{(\tau)} - \rho(\tau)\nabla \log \mathcal{L}(\theta^{(\tau)}|D)$$

$$= \theta^{(\tau)} + \rho(\tau)\underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - \theta^{(\tau)T}x_i)x_i}_{\text{Estimate of the Gradient}} \qquad O(np)$$

- Can we do better?

Stochastic Gradient Descent

- Construct noisy estimate of the gradient:

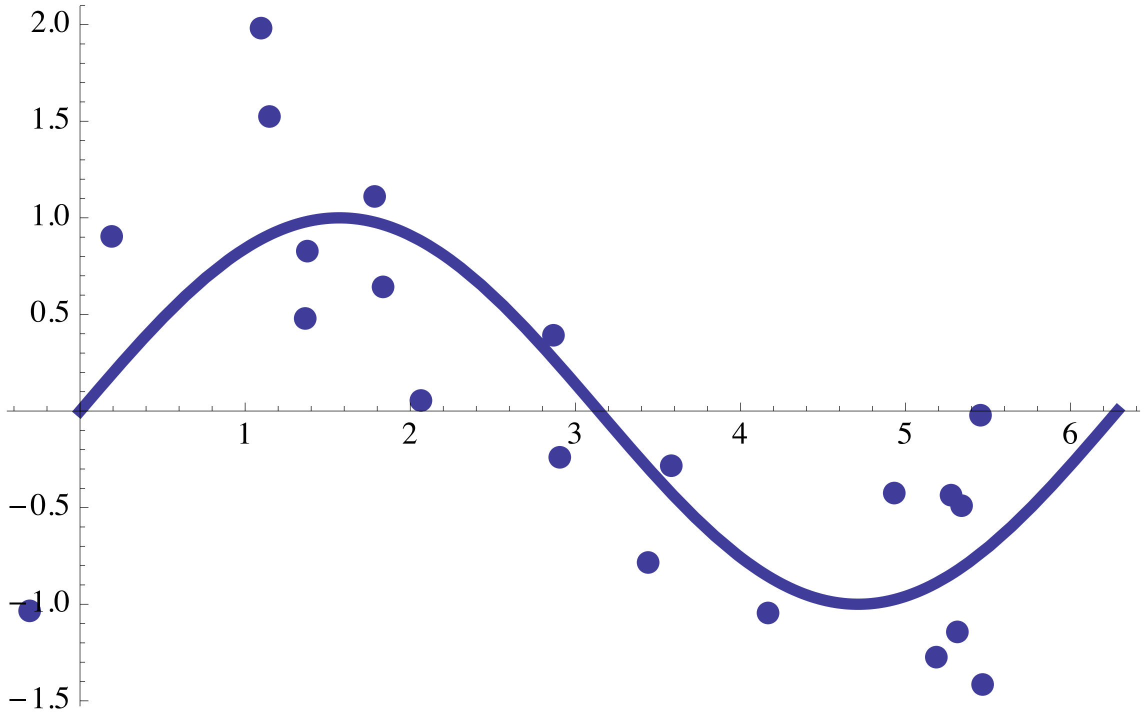
```
For τ from 0 until convergence
    1) pick a random i
    2)
```

$$\theta^{(\tau+1)} = \theta^{(\tau)} + \rho(\tau)\boxed{(y_i - \theta^{(\tau)T} x_i) x_i} \quad O(p)$$

- Sensitive to choice of $\rho(\tau)$ typically ($\rho(\tau)=1/\tau$)
- Also known as Least-Mean-Squares (LMS)
- Applies to streaming data *O(p)* storage

Fitting Non-linear Data

- What if Y has a non-linear response?



- Can we still use a linear model?

Transforming the Feature Space

- Transform features x_i

$$x_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,p})$$

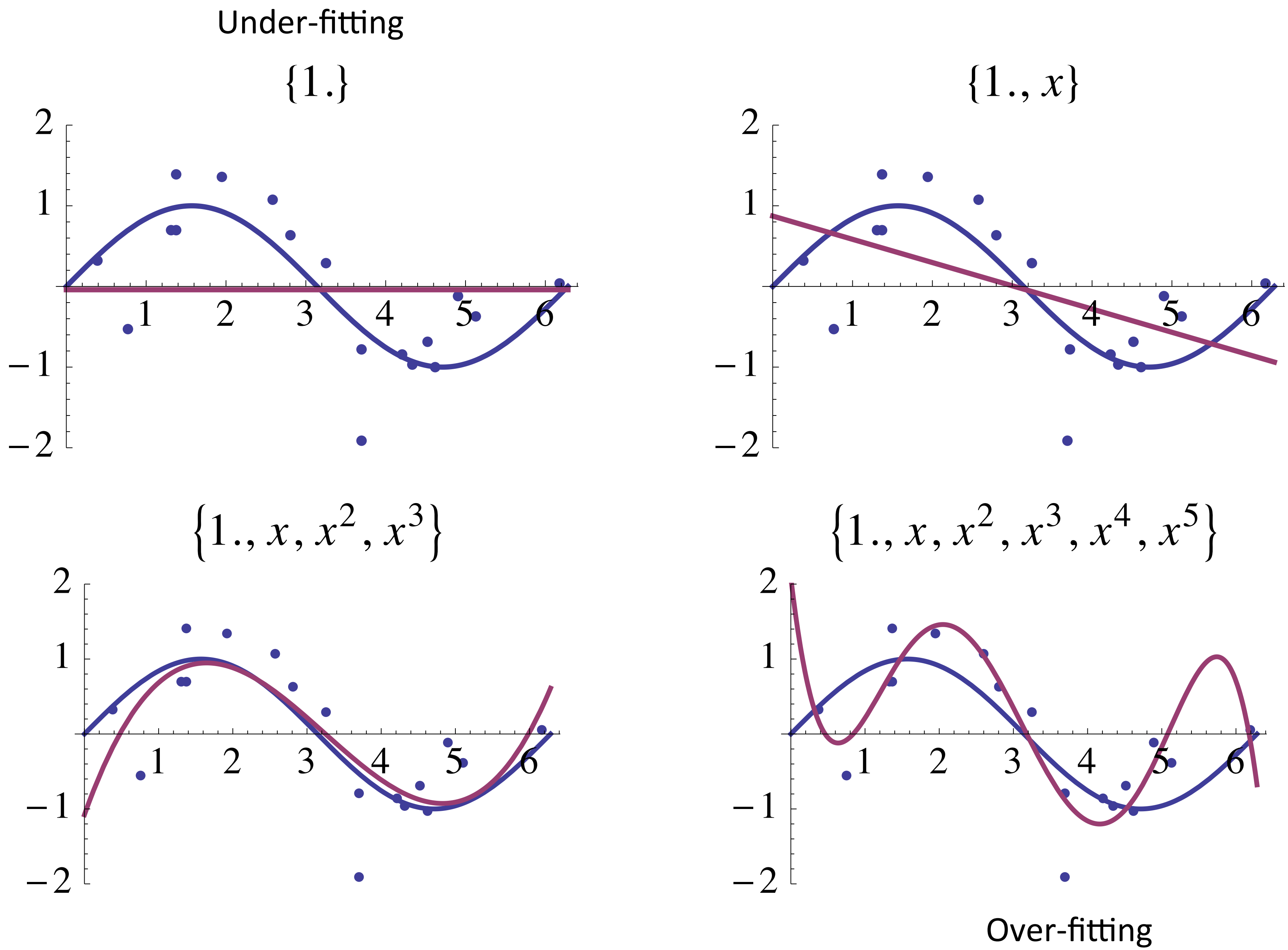
- By applying non-linear transformation ϕ:

$$\phi : \mathbb{R}^p \rightarrow \mathbb{R}^k$$

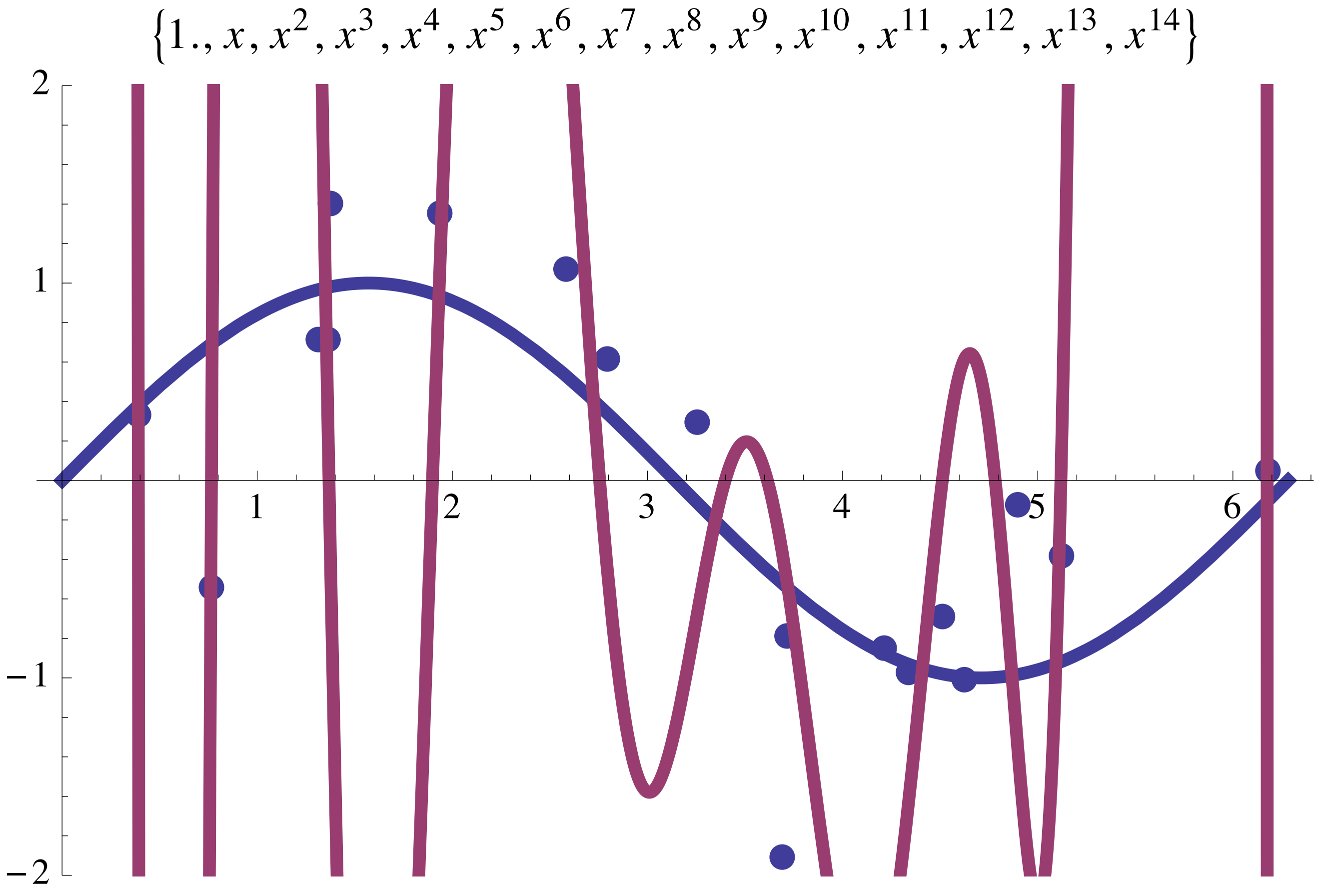
- Example:

$$\phi(x) = \{1, x, x^2, \ldots, x^k\}$$

  – others: splines, radial basis functions, …
  – Expert engineered features (modeling)


Under-fitting
{1.}
{1., x}
{1., x, x^2, x^3}
{1., x, x^2, x^3, x^4, x^5}
Over-fitting

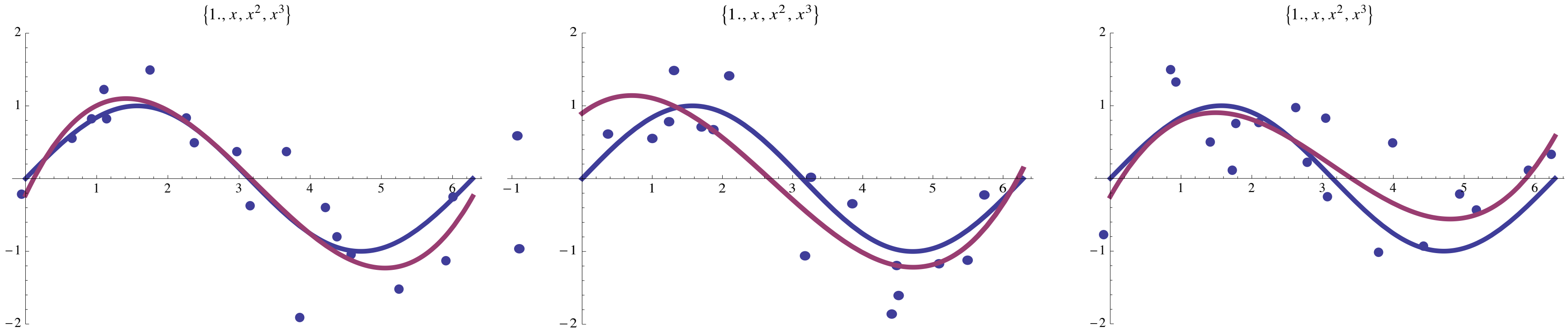
Really Over-fitting!



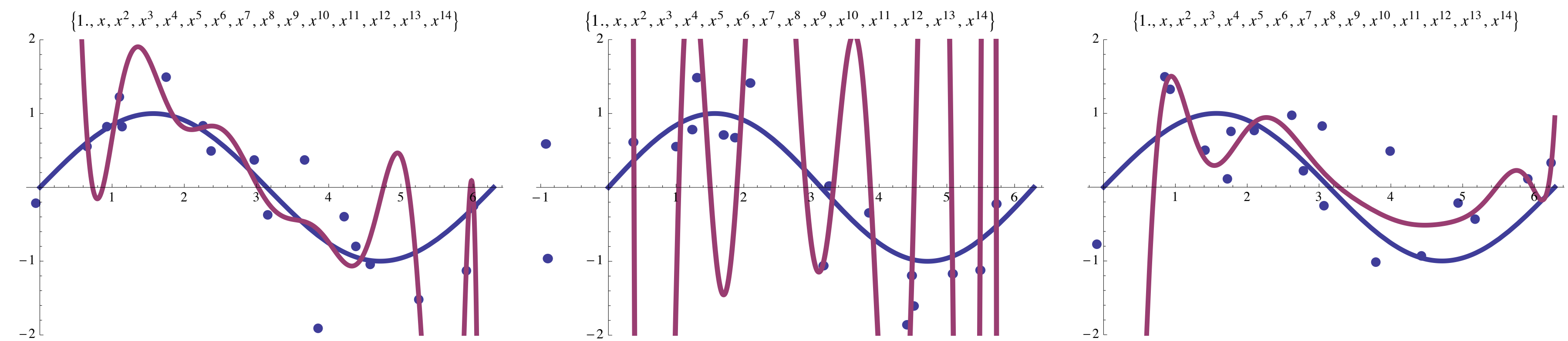
- Errors on training data are small
- But errors on new points are likely to be large

What if I train on different data?

Low Variability:



High Variability



Bias-Variance Tradeoff

- So far we have minimized the error (loss) with respect to **training data**
  - Low training error does not imply good expected performance: **over-fitting**
- We would like to reason about the **expected loss (Prediction Risk)** over:
  - Training Data: $\{(y_1, x_1), \ldots, (y_n, x_n)\}$
  - Test point: (y_*, x_*)
- We will decompose the expected loss into:

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right] = \text{Noise} + \text{Bias}^2 + \text{Variance}$$

- Define (unobserved) the true model (*h*):

$$y_* = h(x_*) + \epsilon_*$$

Assume 0 mean noise
[bias goes in *h(x*)*]

- Completed the squares with: $\boxed{h(x_*) = h_*}$

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right]$$ Expected Loss

$$= \mathbf{E}_{D,(y_*,x_*)}\left[(\underbrace{y_* \boxed{- h(x_*)}}_{a} \underbrace{\boxed{+ h(x_*)} - f(x_*|D)}_{b})^2\right]$$

$$(a+b)^2 = a^2 + b^2 + 2ab$$

$$= \mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right] + \mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$
$$+ 2\mathbf{E}_{D,(y_*,x_*)}\left[y_* h_* - y_* f_* - h_* h_* + h_* f_*\right]$$

- Define (unobserved) the true model (*h*):

$$y_* = h(x_*) + \epsilon_*$$

- Completed the squares with: $h(x_*) = h_*$

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right]$$ Expected Loss

$$= \mathbf{E}_{D,(y_*,x_*)}\left[(y_* - h(x_*) + h(x_*) - f(x_*|D))^2\right]$$

$$= \mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right] + \mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

$$+ 2\underbrace{\cancel{\mathbf{E}_{D,(y_*,x_*)}\left[y_*h_* - y_*f_* - h_*h_* + h_*f_*\right]}}_{\text{Substitute defn. } y_* = h_* + e_*}$$

$$\mathbf{E}\left[(h_* + \epsilon_*)h_* - (h_* + \epsilon_*)f_* - h_*h_* + h_*f_*\right] =$$

$$\cancel{h_*h_*} + \cancel{\mathbf{E}\left[\epsilon_*\right]h_*} - \cancel{h_*\mathbf{E}\left[f_*\right]} - \cancel{\mathbf{E}\left[\epsilon_*\right]f_*} - \cancel{h_*h_*} + \cancel{h_*\mathbf{E}\left[f_*\right]}$$

- Define (unobserved) the true model (*h*):

$$y_* = h(x_*) + \epsilon_*$$

- Completed the squares with: $h(x_*) = h_*$

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right] \quad \text{Expected Loss}$$

$$= \mathbf{E}_{D,(y_*,x_*)}\left[(y_* - h(x_*) + h(x_*) - f(x_*|D))^2\right]$$

$$= \underbrace{\mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right]}_{\substack{\text{Noise Term} \\ \text{(out of our control)} \\ ☹}} + \boxed{\underbrace{\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]}_{\substack{\text{Model Estimation Error} \\ \text{(we want to minimize this)}}} \quad \text{Expand}}$$

- Minimum error is governed by the noise.

- Expanding on the model estimation error:

$$\mathbf{E}_D\left[\left(h(x_*) - f(x_*|D)\right)^2\right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\mathbf{E}_D\left[\left(h(x_*) - f(x_*|D)\right)^2\right]$$

$$= \mathbf{E}\left[\left(h(x_*) - \mathbf{E}\left[f(x_*|D)\right] + \mathbf{E}\left[f(x_*|D)\right] - f(x_*|D)\right)^2\right]$$

$$= \mathbf{E}\left[\left(h(x_*) - \mathbf{E}\left[f(x_*|D)\right]\right)^2\right] + \mathbf{E}\left[\left(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right]\right)^2\right]$$

$$+\, \cancel{2\mathbf{E}\underbrace{\left[h_*\bar{f}_* - h_*f_* - \bar{f}_*f_* + \bar{f}_*^2\right]}_{= h_*\bar{f}_* - h_*\mathbf{E}\left[f_*\right] - \bar{f}_*\mathbf{E}\left[f_*\right] + \bar{f}_*^2 =}}$$

$$h_*\bar{f}_* - h_*\bar{f}_* - \bar{f}_*\bar{f}_* + \bar{f}_*^2 = 0$$

- Expanding on the model estimation error:

$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\begin{aligned}
&\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right] \\
&\quad = \underbrace{\mathbf{E}\left[(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2\right]}_{(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2} + \mathbf{E}\left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2\right]
\end{aligned}$$

- Expanding on the model estimation error:

$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\begin{aligned}
&\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right] \\
&\quad = \underbrace{(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2}_{\text{(Bias)}^2} + \underbrace{\mathbf{E}\left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2\right]}_{\text{Variance}}
\end{aligned}$$

- Tradeoff between bias and variance:
  - **Simple Models:** High Bias, Low Variance
  - **Complex Models:** Low Bias, High Variance

Summary of Bias Variance Tradeoff

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right] =$$ Expected Loss

$$\mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right]$$ Noise

$$+ \left(h(x_*) - \mathbf{E}_D\left[f(x_*|D)\right]\right)^2$$ (Bias)^2

$$+ \mathbf{E}_D\left[\left(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right]\right)^2\right]$$ Variance

- Choice of models balances bias and variance.
  - Over-fitting ➔ Variance is too High
  - Under-fitting ➔ Bias is too High

Bias Variance Plot

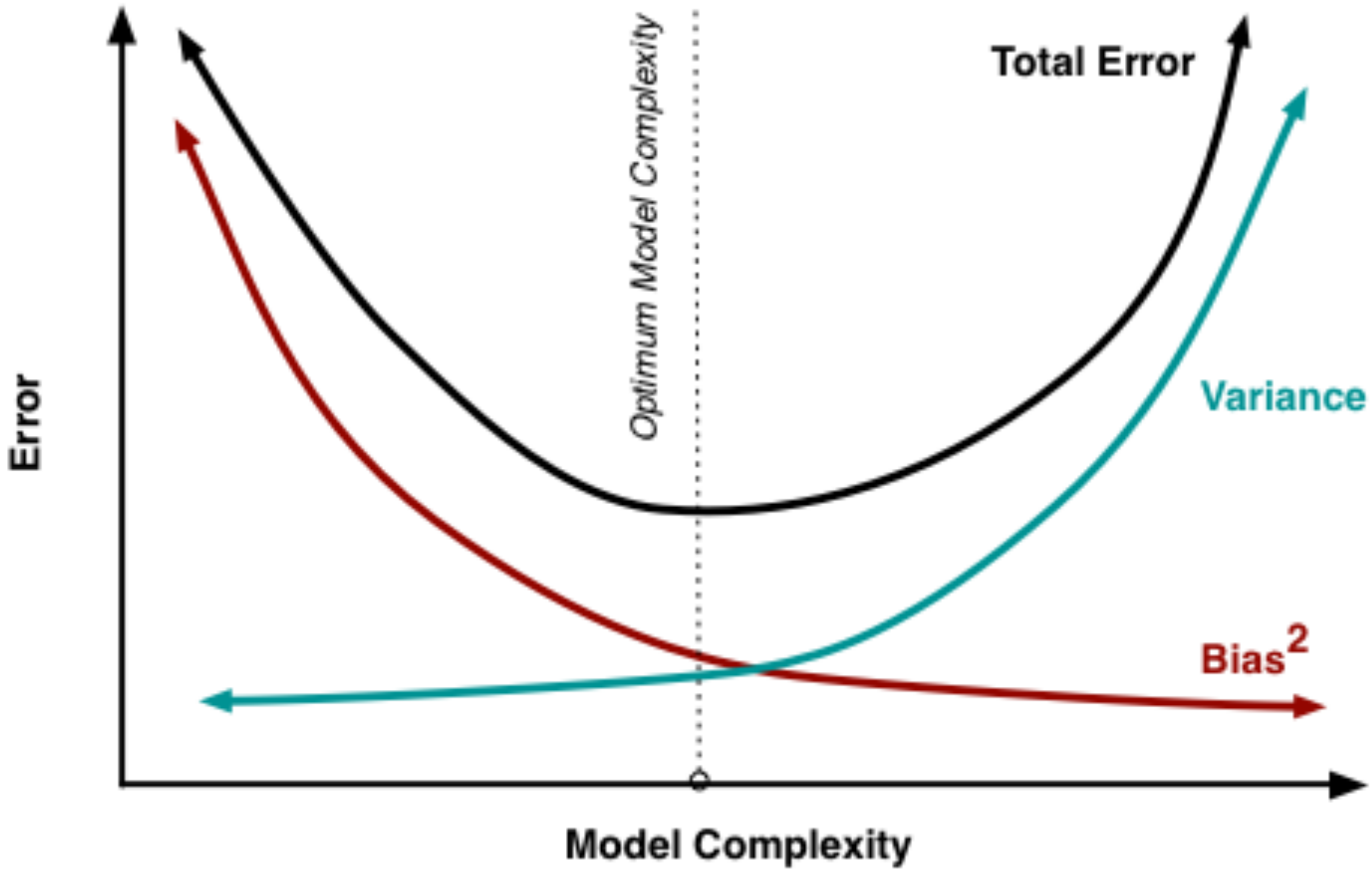


Image from http://scott.fortmann-roe.com/docs/BiasVariance.html

Analyze bias of $f(x_*|D) = x_*^T \hat{\theta}_{\text{MLE}}$

- Assume a true model is linear: $h(x_*) = x_*^T \theta$

$$
\begin{aligned}
\text{bias} &= h(x_*) - \mathbf{E}_D\left[f(x_*|D)\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T \hat{\theta}_{\text{MLE}}\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T Y\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T (X\theta + \epsilon)\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T X\theta + x_*^T (X^T X)^{-1} X^T \epsilon\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T\theta + x_*^T (X^T X)^{-1} X^T \epsilon\right] \\
&= x_*^T\theta - x_*^T\theta + x_*^T (X^T X)^{-1} X^T \mathbf{E}_D\left[\epsilon\right] \\
&= x_*^T\theta - x_*^T\theta = 0
\end{aligned}
$$

Substitute MLE

Plug in definition of Y

Expand and cancel

Assumption:

$\mathbf{E}_D\left[\epsilon\right] = 0$

$\hat{\theta}_{\text{MLE}}$ is unbiased!

Analyze Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Assume a true model is linear: $h(x_*) = x_*^T \theta$

$$\mathrm{Var.} = \mathbf{E}\left[\left(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right]\right)^2\right]$$

$$= \mathbf{E}\left[\left(x_*^T \hat{\theta}_{\mathrm{MLE}} - x_*^T \theta\right)^2\right]$$

Substitute MLE + unbiased result

$$= \mathbf{E}\left[\left(x_*^T (X^T X)^{-1} X^T Y - x_*^T \theta\right)^2\right]$$

Plug in definition of Y

$$= \mathbf{E}\left[\left(x_*^T (X^T X)^{-1} X^T (X\theta + \epsilon) - x_*^T \theta\right)^2\right]$$

$$= \mathbf{E}\left[\left(x_*^T \theta + x_*^T (X^T X)^{-1} X^T \epsilon - x_*^T \theta\right)^2\right]$$

$$= \mathbf{E}\left[\left(x_*^T (X^T X)^{-1} X^T \epsilon\right)^2\right]$$

Expand and cancel

- Use property of scalar: $a^2 = a\, a^T$

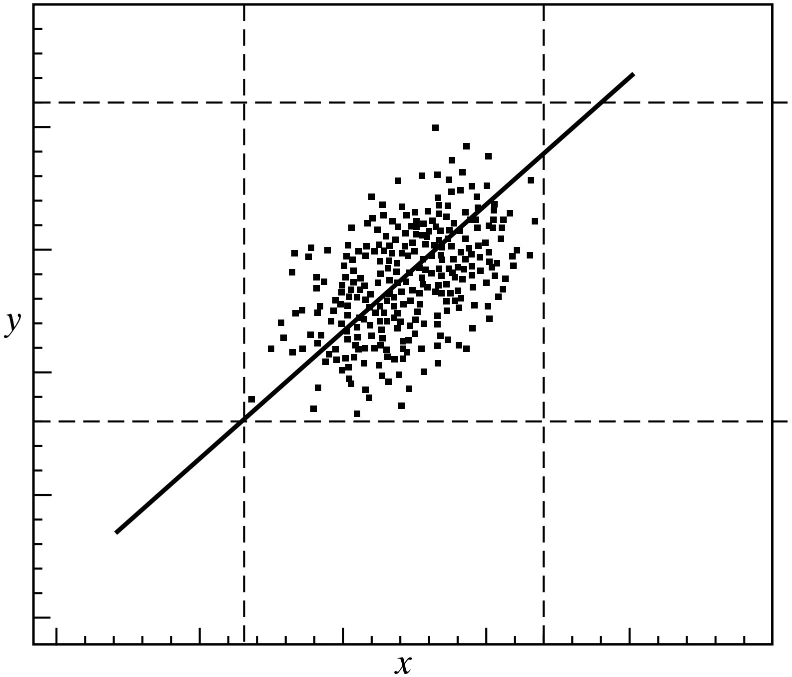
Analyze Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Use property of scalar: $a^2 = a\,a^T$

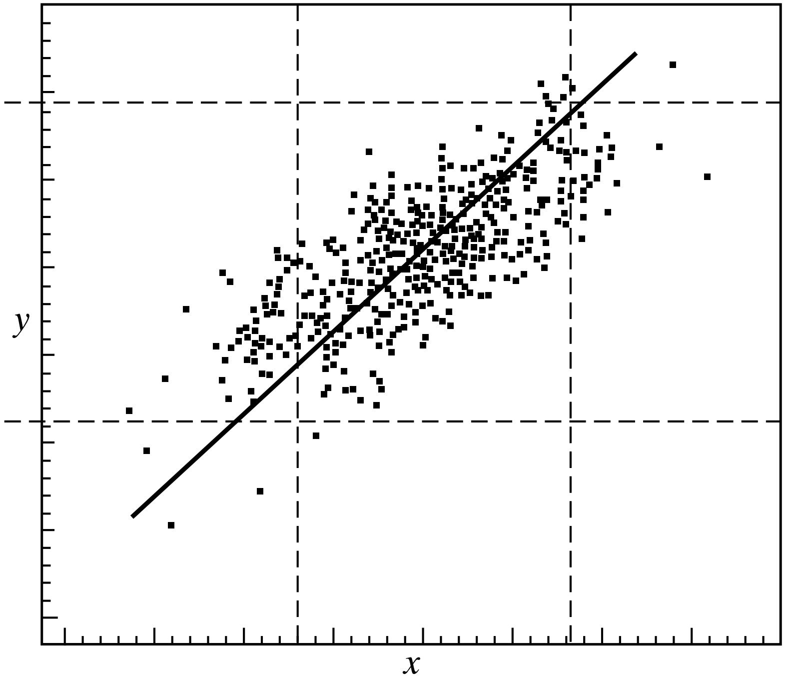
$$
\begin{aligned}
\mathrm{Var.} &= \mathbf{E}\left[(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right])^2\right] \\
&= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\epsilon)^2\right] \\
&= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\epsilon)(x_*^T(X^TX)^{-1}X^T\epsilon)^T\right] \\
&= \mathbf{E}\left[x_*^T(X^TX)^{-1}X^T\epsilon\epsilon^T(x_*^T(X^TX)^{-1}X^T)^T\right] \\
&= x_*^T(X^TX)^{-1}X^T\mathbf{E}\left[\epsilon\epsilon^T\right](x_*^T(X^TX)^{-1}X^T)^T \\
&= x_*^T(X^TX)^{-1}X^T\sigma_\epsilon^2 I(x_*^T(X^TX)^{-1}X^T)^T \\
&= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}X^TX(x_*^T(X^TX)^{-1})^T \\
&= \sigma_\epsilon^2 x_*^T(x_*^T(X^TX)^{-1})^T \\
&= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}x_*
\end{aligned}
$$

Consequence of Variance Calculation

$$\text{Var.} = \mathbf{E}\left[\left(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right]\right)^2\right]$$
$$= \sigma_\epsilon^2 x_*^T (X^T X)^{-1} x_*$$



Higher Variance



Lower Variance

Figure from http://people.stern.nyu.edu/wgreene/MathStat/GreeneChapter4.pdf

Summary

- Least-Square Regression is Unbiased:

$$\mathbf{E}_D\left[x_*^T\hat{\theta}_{\mathrm{MLE}}\right] = x_*^T\theta$$

- Variance depends on:

$$\mathbf{E}\left[\left(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right]\right)^2\right] = \sigma_\epsilon^2 x_*^T (X^T X)^{-1} x_*$$
$$\approx \sigma_\epsilon^2 \frac{p}{n}$$

  – Number of data-points *n*
  – Dimensionality *p*
  – Not on observations *Y*

Deriving the final identity

- Assume x_i and x_* are N(0,1)

$$\begin{aligned}
\mathbf{E}_{X,x_*}\left[\text{Var.}\right] &= \sigma_\epsilon^2 \mathbf{E}_{X,x_*}\left[x_*^T (X^T X)^{-1} x_*\right] \\
&= \sigma_\epsilon^2 \mathbf{E}_{X,x_*}\left[tr(x_* x_*^T (X^T X)^{-1})\right] \\
&= \sigma_\epsilon^2 tr(\mathbf{E}_{X,x_*}\left[x_* x_*^T (X^T X)^{-1}\right]) \\
&= \sigma_\epsilon^2 tr(\mathbf{E}_{x_*}\left[x_* x_*^T\right] \mathbf{E}_X\left[(X^T X)^{-1}\right]) \\
&= \frac{\sigma_\epsilon^2}{n} tr(\mathbf{E}_{x_*}\left[x_* x_*^T\right]) \\
&= \frac{\sigma_\epsilon^2}{n} p
\end{aligned}$$

Gauss-Markov Theorem

- The linear model:

$$f(x_*) = x_*^T \hat{\theta}_{\mathrm{MLE}} = x_*^T (X^T X)^{-1} X^T Y$$

has the **minimum variance** among all **unbiased** linear estimators

– Note that this is linear in Y

- **BLUE: B**est **L**inear **U**nbiased **E**stimator

Summary

- Introduced the Least-Square regression model
  - Maximum Likelihood: Gaussian Noise
  - Loss Function: Squared Error
  - Geometric Interpretation: Minimizing Projection
- Derived the normal equations:
  - Walked through process of constructing MLE
  - Discussed efficient computation of the MLE
- Introduced basis functions for non-linearity
  - Demonstrated issues with over-fitting
- Derived the classic bias-variance tradeoff
  - Applied to least-squares model


SUPPORT VECTOR MACHINES
REPEA
POWE
LAWS
END DUALITy GAP
Map Reduce
Map Reuse
Map Recycle
BAYeSIANS AGAINST DISCRIMINATION
FREE VARIABLES!
BAN GENETIC ALGORITHMS

Additional Reading I found Helpful

- http://www.stat.cmu.edu/~roeder/stat707/lectures.pdf
- http://people.stern.nyu.edu/wgreene/MathStat/GreeneChapter4.pdf
- http://www.seas.ucla.edu/~vandenbe/103/lectures/qr.pdf
- http://www.cs.berkeley.edu/~jduchi/projects/matrix_prop.pdf