Linear Regression

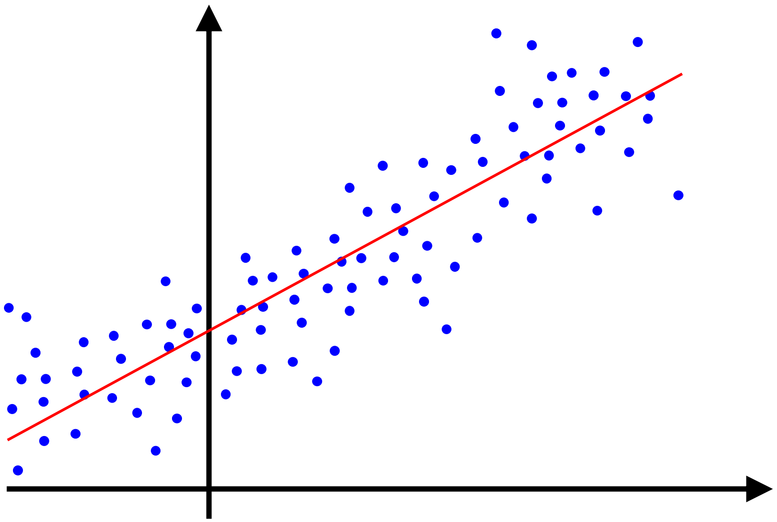
Data Intelligence and Learning (DIAL) Lab
Prof. Jongwuk Lee

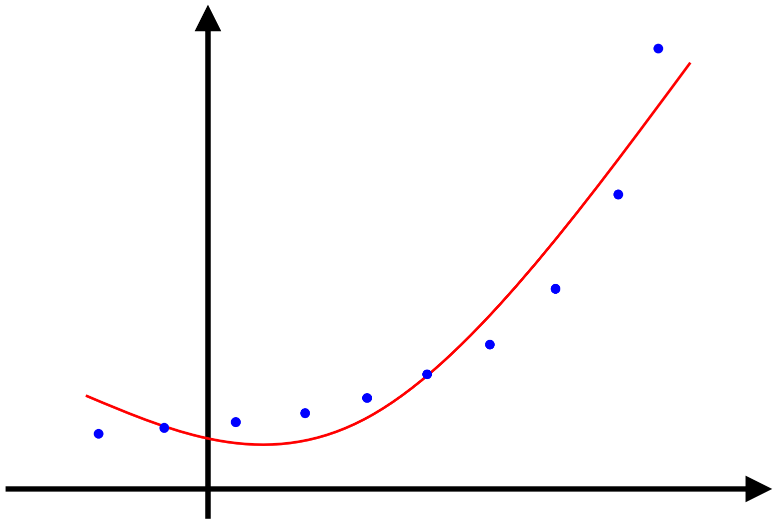
Overview: Linear Regression

Regression Models

- Modeling the relationship between **one or more explanatory variables $\mathbf{x}$** and a **dependent variable y**

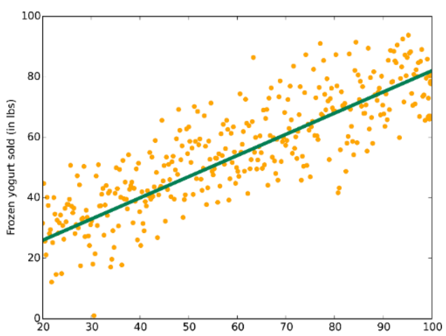
- Widely used for predicting and forecasting **continuous values**

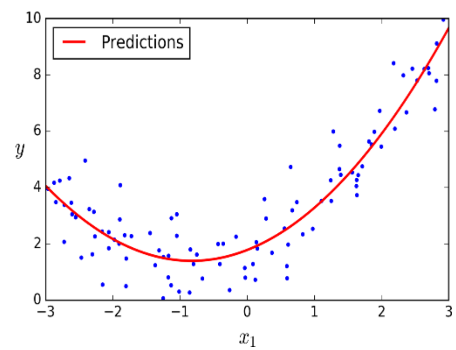


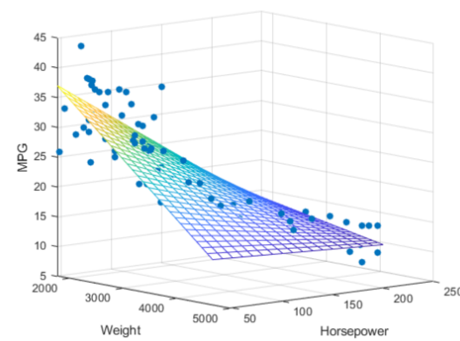


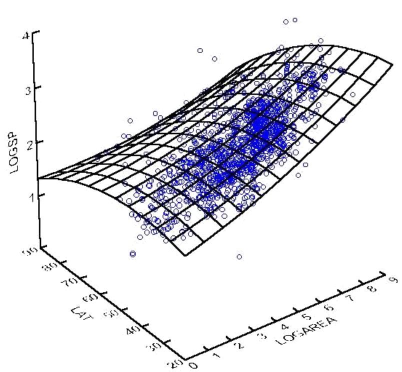
Types of Regression Models

- **Regression Models**
  - **Simple (one variable)**
    - **Linear**
    - **Non-linear**
  - **Multiple (two or more variables)**
    - **Linear**
    - **Non-linear**







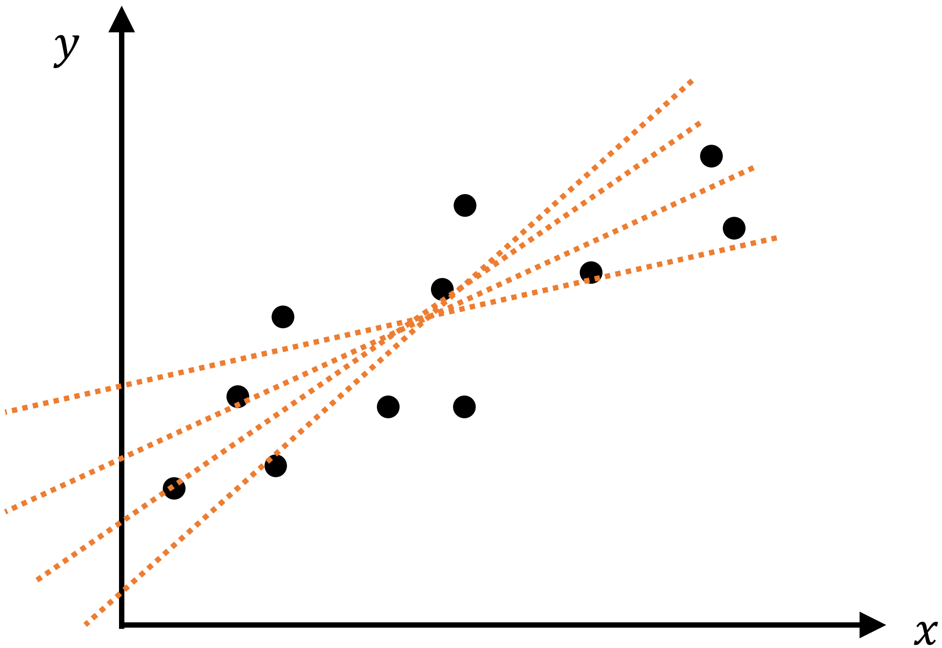


Simple Linear Regression

- Given training data $\{(x^{(i)}, y^{(i)}): 1 \leq i \leq n\}$,
- Model a linear function for given training data.

$$f(x; w_0, w_1) = w_1 x + w_0$$

w_0: bias



Which line is the best?

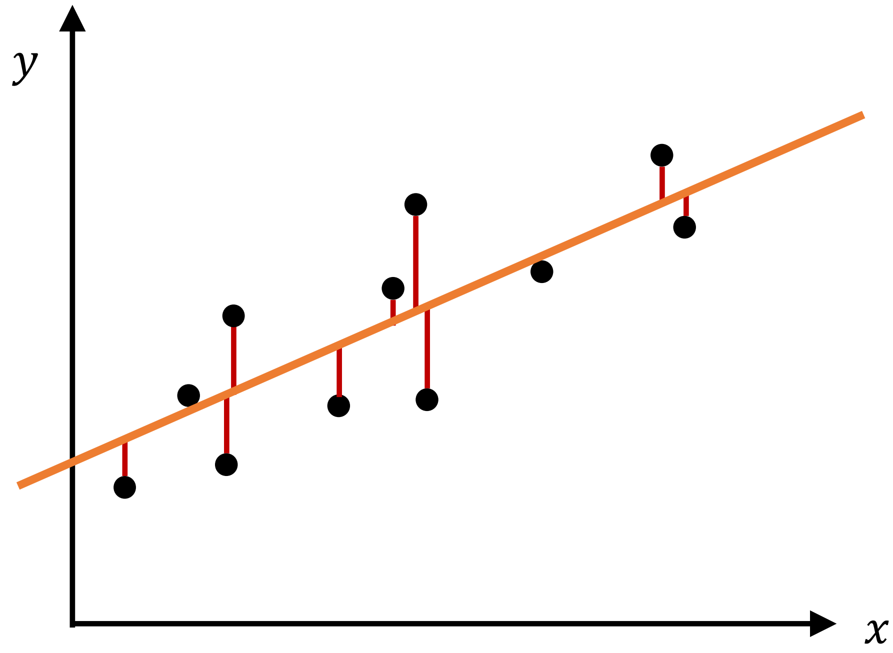
Error Function (or Loss Function)

➢ **Minimize the sum of squared residuals between actual values and predicted values.**

$$E(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

⬇

$$E(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \left(\boldsymbol{w_1} x^{(i)} + \boldsymbol{w_0}\right)\right)^2$$



Solving the Optimization Problem

- **The ML models can be trained analytically (e.g., normal equation) or are solved numerically (e.g., gradient descent).**

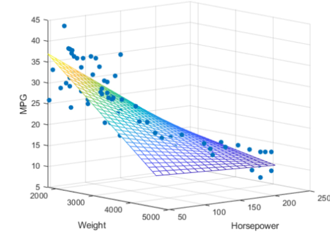
- **Analytical solution**
  - It involves framing the problem in a **well-understood form** and calculating the **exact solution**.
  - In general, it is preferred because it is **faster,** and the solution is **exact**.

- **Numerical solution**
  - **Make guesses for the solution**.
  - **It is necessary to validate whether it is solved well or not**.

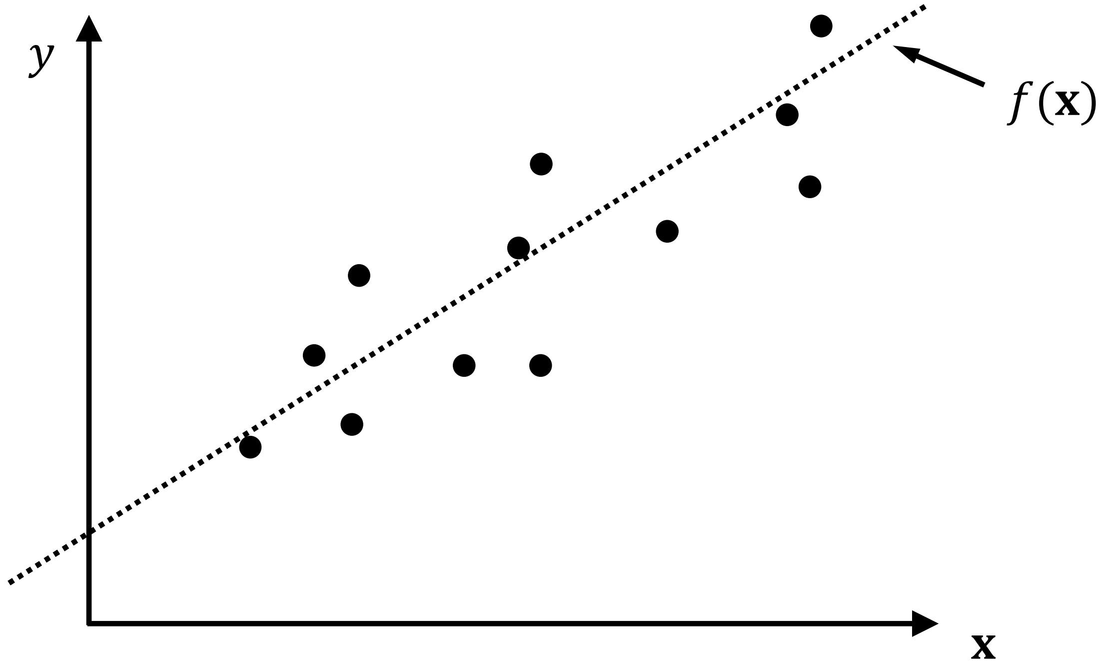
https://machinelearningmastery.com/analytical-vs-numerical-solutions-in-machine-learning/

Analytical Solution

Multiple Linear Regression



- **Finding a hyperplane that best fits the training data on d-dimensional space**



$$f(\mathbf{x}; w_0, w_1, \dots, w_d) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d = w_0 + \sum_{j=1}^{d} w_j x_j$$

Multiple Linear Regression

- **Training data $\mathbf{X} \in \mathbb{R}^{n \times d}$ can be converted to a matrix $\mathbb{R}^{n \times (d+1)}$.**

x_1	…	x_d
2	…	1
5	…	2
5	…	3
3	…	4
2	…	6

x_0	x_1	…	x_d
1	2	…	1
1	5	…	2
1	5	…	3
1	3	…	4
1	2	…	6

Interpreted as x_0, which corresponds to bias w_0

- **It can be easily represented by a matrix including the bias.**

Matrix Form of Linear Regression

- **$\mathbf{X} \in \mathbb{R}^{n\times(d+1)}$ matrix, $\mathbf{y} \in \mathbb{R}^{n\times 1}$ vector**
- **$\mathbf{w} \in \mathbb{R}^{(d+1)\times 1}$ vector**
  - w_0 is interpreted as the bias.

$$
\underbrace{\begin{bmatrix} x_{10} & x_{11} & \dots & x_{1d} \\ x_{20} & x_{21} & \dots & x_{2d} \\ x_{30} & x_{31} & \dots & x_{3d} \\ \dots & \dots & \dots & \dots \\ x_{n0} & x_{n1} & \dots & x_{nd} \end{bmatrix}}_{\mathbf{X}} \cdot \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_d \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \dots \\ \hat{y}_n \end{bmatrix}}_{\hat{\mathbf{y}}} \approx \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \dots \\ y_n \end{bmatrix}}_{\mathbf{y}}
$$

Normal Equation

- It is an analytical solution to linear regression with a **least square error function.**

$$E(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

$$\Downarrow$$

$$\mathbf{w} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

- We can directly find out the optimal parameter.

Recap: Solving Linear Equations

- Define a linear system with the same number of equations.

$$a_1 w_1 + b_1 w_2 = c_1$$
$$a_2 w_2 + b_2 w_2 = c_2$$

$$\Rightarrow \quad \mathbf{A} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

- Then, we can get the following equation. $\mathbf{Aw} = \mathbf{y}$
- By using the invertible matrix,

$$\left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{Aw} = \left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{y}$$

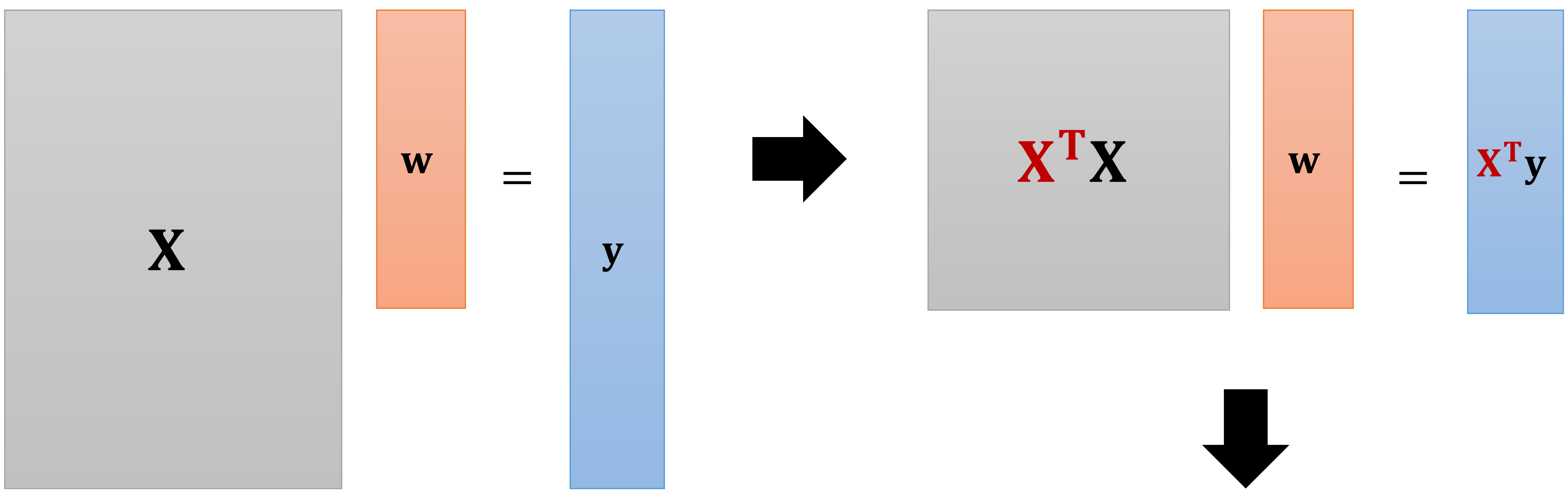
$$\left[\left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{A}\right]\mathbf{w} = \left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{y}$$

$$\boldsymbol{I}\mathbf{w} = \left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{y} \qquad \Rightarrow \qquad \mathbf{w} = \left(\boldsymbol{A}^{-\mathbf{1}}\right)\mathbf{y}$$

Conceptual View: Normal Equation

- $\mathbf{X} \in \mathbb{R}^{n\times(d+1)}$ **matrix,** $\mathbf{y} \in \mathbb{R}^{n\times 1}$ **vector**
- $\mathbf{w} \in \mathbb{R}^{(d+1)\times 1}$ **vector**

Assuming that $\mathbf{X}^\mathrm{T}\mathbf{X}$ is **invertible**,



Normal equation: $\mathbf{w} = \left(\mathbf{X}^\mathrm{T}\mathbf{X}\right)^{-1}\mathbf{X}^\mathrm{T}\mathbf{y}$

Detail: Deriving the Normal Equation

- **Redefine the error function for matrix form.**
  - For simplicity, we ignore the constant and $\|\mathbf{x}\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$.

$$E(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\mathbf{w})$$

$$= \left(\mathbf{y}^{\mathbf{T}} - (\mathbf{X}\mathbf{w})^{\mathbf{T}}\right)(\mathbf{y} - \mathbf{X}\mathbf{w}) \qquad (\mathbf{A}+\mathbf{B})^{\mathrm{T}} = \mathbf{A}^{\mathbf{T}} + \mathbf{B}^{\mathbf{T}}$$

$$= \mathbf{y}^{\mathbf{T}}\mathbf{y} - \mathbf{y}^{\mathbf{T}}(\mathbf{X}\mathbf{w}) - (\mathbf{X}\mathbf{w})^{\mathbf{T}}\mathbf{y} + (\mathbf{X}\mathbf{w})^{\mathbf{T}}(\mathbf{X}\mathbf{w}) \qquad \mathbf{A}(\mathbf{B}+\mathbf{C}) = \mathbf{AB} + \mathbf{AC}$$

$$= \mathbf{y}^{\mathbf{T}}\mathbf{y} - 2(\mathbf{X}\mathbf{w})^{\mathbf{T}}\mathbf{y} + (\mathbf{X}\mathbf{w})^{\mathbf{T}}(\mathbf{X}\mathbf{w}) \qquad \mathbf{A}^{\mathbf{T}}\mathbf{B} = \mathbf{B}^{\mathbf{T}}\mathbf{A}$$

$$= \mathbf{y}^{\mathbf{T}}\mathbf{y} - 2\mathbf{w}^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{y} + \mathbf{w}^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{X}\mathbf{w} \qquad (A\mathbf{B})^{\mathrm{T}} = \mathbf{B}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}$$

$$\frac{\partial E}{\partial \mathbf{w}} = -2\mathbf{X}^{\mathbf{T}}\mathbf{y} + \mathbf{X}^{\mathbf{T}}\mathbf{X}\mathbf{w} + \left(\mathbf{w}^{\mathbf{T}}\mathbf{X}^{\mathbf{T}}\mathbf{X}\right)^{\mathbf{T}} = -2\mathbf{X}^{\mathbf{T}}\mathbf{y} + 2\mathbf{X}^{\mathbf{T}}\mathbf{X}\mathbf{w} = \mathbf{0}$$

$$\Longrightarrow \mathbf{w} = \left(\mathbf{X}^{\mathbf{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{y}$$

Solving Multiple Linear Regression

- **Given a dataset $\mathcal{D} = \left\{\left(\mathbf{x}^{(i)}, \boldsymbol{y}^{(i)}\right): \mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{n}\right\}$,**
  - $\mathbf{x}^{(i)} = (x_{i0}, x_{i1}, x_{2i}, \dots, x_{id})$ is the input on $(d + 1)$-dimensional space.
  - $y^{(i)}$ is the output.

- **Finding the hyperplane $f(\mathbf{x})$ which best fits the dataset $\mathcal{D}$**
  - Make $f(\mathbf{x}^{(i)})$ close to $y^{(i)}$ as much as possible for $i = 1, \dots, n$

$$E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

Solving Multiple Linear Regression

- Given $\mathcal{D} = \{(\mathbf{x}^{(i)}, \boldsymbol{y}^{(i)}) : \mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{n}\}$
- Find $\mathbf{w} = (\boldsymbol{w}_0, \boldsymbol{w}_1, \dots, \boldsymbol{w}_d)$ which minimizes $E(\mathbf{w})$

$$E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)^2$$

$$f\left(\mathbf{x}^{(i)}\right) = \sum_{j=0}^{d} w_j x_{ij} = w_0 x_{i0} + w_1 x_{i1} + \cdots + w_d x_{id}$$

- How to solve this?

Solving Multiple Linear Regression

- When a function is convex, continuous, and differentiable, a necessary and sufficient condition for a point $\mathbf{w}^*$ to be optimal is $\nabla \boldsymbol{E}(\mathbf{w}^*) = \mathbf{0}$.

$$\frac{\partial}{\partial w_j} E(w_0, w_1, \cdots, w_d) = 0 \;\text{ for } j = 0, \cdots, d$$

$$\frac{\partial}{\partial \mathbf{w}} E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(\frac{\partial}{\partial \mathbf{w}} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)^2 \right) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right) \left(-2\mathbf{x}^{(i)} \right) = 0$$

Detail: Computing the Derivative

- For a single training sample $\mathbf{x}^{(i)}$,

$$E(\mathbf{w}) = \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

- Substituting $y^{(i)} - f\left(\mathbf{x}^{(i)}\right)$ to $\boldsymbol{error}$

$$E(\mathbf{w}) = \boldsymbol{error}^2 \quad \text{where } \boldsymbol{error} = y^{(i)} - f\left(\mathbf{x}^{(i)}\right)$$

Detail: Chain Rule for the Derivative

- By using the chain rule,

$$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \boldsymbol{error}} \frac{\partial \boldsymbol{error}}{\partial w_j} = 2\boldsymbol{error} \times \left(-x_{ij}\right) = -2x_{ij}\left(\boldsymbol{y}^{(i)} - \boldsymbol{f}\left(\mathbf{x}^{(i)}\right)\right)$$

$$\boldsymbol{error} = \boldsymbol{y}^{(i)} - \boldsymbol{f}\left(\mathbf{x}^{(i)}\right)$$

- For all samples in the training dataset,

$$\frac{\partial}{\partial w_j} E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(\frac{\partial}{\partial w_j} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2 \right) = \frac{1}{2n} \sum_{i=1}^{n} \left(-2x_{ij}\right)\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)$$

Deriving the Normal Equation

- **Obtain $\mathbf{w} = (w_0, w_1, \cdots, w_d)$ by solving these equations.**

$$\left.\begin{aligned}
&\sum_{i=1}^{n} x_{i0}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0 \\
&\sum_{i=1}^{n} x_{i1}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0 \\
&\qquad\qquad \dots \dots \dots \\
&\sum_{i=1}^{n} x_{id}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0
\end{aligned}\right\}$$

There are $d+1$ variables
$d+1$ equations.

Deriving the Normal Equation

$$\sum_{i=1}^{n} x_{i0}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0$$
$$\sum_{i=1}^{n} x_{i1}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0$$
$$\dots$$
$$\sum_{i=1}^{n} x_{id}\left(f(\mathbf{x}^{(i)}) - y^{(i)}\right) = 0$$

$$\Rightarrow$$

$$\sum_{i=1}^{n} x_{i0}\left(w_0 x_{i0} + w_1 x_{i1} + \cdots + w_d x_{id} - y^{(i)}\right) = 0$$
$$\sum_{i=1}^{n} x_{i1}\left(w_0 x_{i0} + w_1 x_{i1} + \cdots + w_d x_{id} - y^{(i)}\right) = 0$$
$$\dots$$
$$\sum_{i=1}^{n} x_{id}\left(w_0 x_{i0} + w_1 x_{i1} + \cdots + w_d x_{id} - y^{(i)}\right) = 0$$

$$\Rightarrow$$

$$w_0 \sum_{i=1}^{n} x_{i0} x_{i0} + w_1 \sum_{i=1}^{n} x_{i0} x_{i1} + \cdots + w_d \sum_{i=1}^{n} x_{i0} x_{id} = \sum_{i=1}^{n} x_{i0} y^{(i)}$$
$$w_0 \sum_{i=1}^{n} x_{i1} x_{i0} + w_1 \sum_{i=1}^{n} x_{i1} x_{i1} + \cdots + w_d \sum_{i=1}^{n} x_{i1} x_{id} = \sum_{i=1}^{n} x_{i1} y^{(i)}$$
$$\dots$$
$$w_0 \sum_{i=1}^{n} x_{id} x_{i0} + w_1 \sum_{i=1}^{n} x_{id} x_{i1} + \cdots + w_d \sum_{i=1}^{n} x_{id} x_{id} = \sum_{i=1}^{n} x_{id} y^{(i)}$$

Deriving the Normal Equation

$$\boxed{w_0}\sum_{i=1}^{n} x_{i0}x_{i0} + \boxed{w_1}\sum_{i=1}^{n} x_{i0}x_{i1} + \cdots + \boxed{w_d}\sum_{i=1}^{n} x_{i0}x_{id} = \sum_{i=1}^{n} x_{i0}y^{(i)}$$

$$w_0\sum_{i=1}^{n} x_{i1}x_{i0} + w_1\sum_{i=1}^{n} x_{i1}x_{i1} + \cdots + w_d\sum_{i=1}^{n} x_{i1}x_{id} = \sum_{i=1}^{n} x_{i1}y^{(i)}$$

$$\cdots$$

$$w_0\sum_{i=1}^{n} x_{id}x_{i0} + w_1\sum_{i=1}^{n} x_{id}x_{i1} + \cdots + w_d\sum_{i=1}^{n} x_{id}x_{id} = \sum_{i=1}^{n} x_{id}y^{(i)}$$

$$\mathbf{A} = \begin{pmatrix} \sum_{i=1}^{n} x_{i0}x_{i0}, \sum_{i=1}^{n} x_{i0}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i0}x_{id} \\ \sum_{i=1}^{n} x_{i0}x_{i1}, \sum_{i=1}^{n} x_{i1}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i1}x_{id} \\ \cdots \\ \sum_{i=1}^{n} x_{i0}x_{id}, \sum_{i=1}^{n} x_{i1}x_{id}, \cdots, \sum_{i=1}^{n} x_{id}x_{id} \end{pmatrix} \quad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_d \end{pmatrix} \quad = \quad \mathbf{b} = \begin{pmatrix} \sum_{i=1}^{n} x_{i0}y^{(i)} \\ \sum_{i=1}^{n} x_{i1}y^{(i)} \\ \cdots \\ \sum_{i=1}^{n} x_{id}y^{(i)} \end{pmatrix}$$

Deriving the Normal Equation

- Given training matrix $\mathbf{X} \in \mathbb{R}^{n\times(d+1)}$, we can calculate the matrix $\mathbf{A} \in \mathbb{R}^{(d+1)\times(d+1)}$.

$$\mathbf{X}^T = \begin{pmatrix} x_{10} & x_{20} & x_{30} & \cdots & x_{n0} \\ x_{11} & x_{21} & x_{31} & \cdots & x_{n1} \\ x_{12} & x_{22} & x_{32} & \cdots & x_{n2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{1d} & x_{2d} & x_{3d} & \cdots & x_{nd} \end{pmatrix} \cdot \mathbf{X} = \begin{pmatrix} x_{10} & x_{11} & x_{12} & \cdots & x_{1d} \\ x_{20} & x_{21} & x_{22} & \cdots & x_{2d} \\ x_{30} & x_{31} & x_{32} & \cdots & x_{3d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n0} & x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$

$$= \quad \mathbf{A} = \begin{pmatrix} \sum_{i=1}^{n} x_{i0}x_{i0}, \sum_{i=1}^{n} x_{i0}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i0}x_{id} \\ \sum_{i=1}^{n} x_{i0}x_{i1}, \sum_{i=1}^{n} x_{i1}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i1}x_{id} \\ \cdots \\ \sum_{i=1}^{n} x_{id}x_{i0}, \sum_{i=1}^{n} x_{id}x_{i1}, \cdots, \sum_{i=1}^{n} x_{id}x_{id} \end{pmatrix} = \mathbf{X}^T\mathbf{X}$$

Deriving the Normal Equation

- **Given $\mathbf{X}^T \in \mathbb{R}^{(d+1)\times n}$ and $\mathbf{y} \in \mathbb{R}^{n\times 1}$, we can calculate the vector $\mathbf{b} \in \mathbb{R}^{(d+1)\times 1}$.**

$$\mathbf{X}^{\mathbf{T}} = \begin{pmatrix} x_{10} & x_{20} & x_{30} & \dots & x_{n0} \\ x_{11} & x_{21} & x_{31} & \dots & x_{n1} \\ x_{12} & x_{22} & x_{32} & \dots & x_{n2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{1d} & x_{2d} & x_{3d} & \dots & x_{nd} \end{pmatrix} \cdot \mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \\ \vdots \\ y^{(n)} \end{pmatrix} = \mathbf{b} = \begin{pmatrix} \sum_{\mathrm{i=1}}^{\mathrm{n}} x_{i0} y^{(i)} \\ \sum_{\mathrm{i=1}}^{\mathrm{n}} x_{i1} y^{(i)} \\ \dots \\ \sum_{\mathrm{i=1}}^{\mathrm{n}} x_{id} y^{(i)} \end{pmatrix}$$

Solution of Normal Equation

$$f(\mathbf{x}_i) = w_0 x_{i0} + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id} = \sum_{j=0}^{d} w_i x_{ij}$$

$$\mathbf{A} = \begin{pmatrix} \sum_{i=1}^{n} x_{i0}x_{i0}, \sum_{i=1}^{n} x_{i0}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i0}x_{id} \\ \sum_{i=1}^{n} x_{i0}x_{i1}, \sum_{i=1}^{n} x_{i1}x_{i1}, \cdots, \sum_{i=1}^{n} x_{i1}x_{id} \\ \cdots \\ \sum_{i=1}^{n} x_{id}x_{i0}, \sum_{i=1}^{n} x_{id}x_{i1}, \cdots, \sum_{i=1}^{n} x_{id}x_{id} \end{pmatrix} \quad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_d \end{pmatrix} \quad = \quad \mathbf{b} = \begin{pmatrix} \sum_{i=1}^{n} x_{i0}y^{(i)} \\ \sum_{i=1}^{n} x_{i1}y^{(i)} \\ \cdots \\ \sum_{i=1}^{n} x_{id}y^{(i)} \end{pmatrix}$$

- The normal equation is

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{b} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\left(\mathbf{X}^{\mathrm{T}}\mathbf{y}\right)$$

Invertability of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$

- $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ may **not be invertible if**

- **Case 1: too many features, i.e.,** $n < d$
  - Delete some features because we have too many features for the number of samples in the training dataset.

- **Case 2: some columns in $\mathbf{X}$ are linearly dependent.**
  - We have some redundant features.

- **We check the rank of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ by using the rank of $\mathbf{X}$.**
  - $rank(\mathbf{X}) = rank(\mathbf{X}^{\mathrm{T}}\mathbf{X})$
  - The rank of the matrix refers to **the number of linearly independent rows or columns in the matrix**.

Quiz: Rank of the Matrix

- Consider the matrix $\mathbf{X}$ as below.

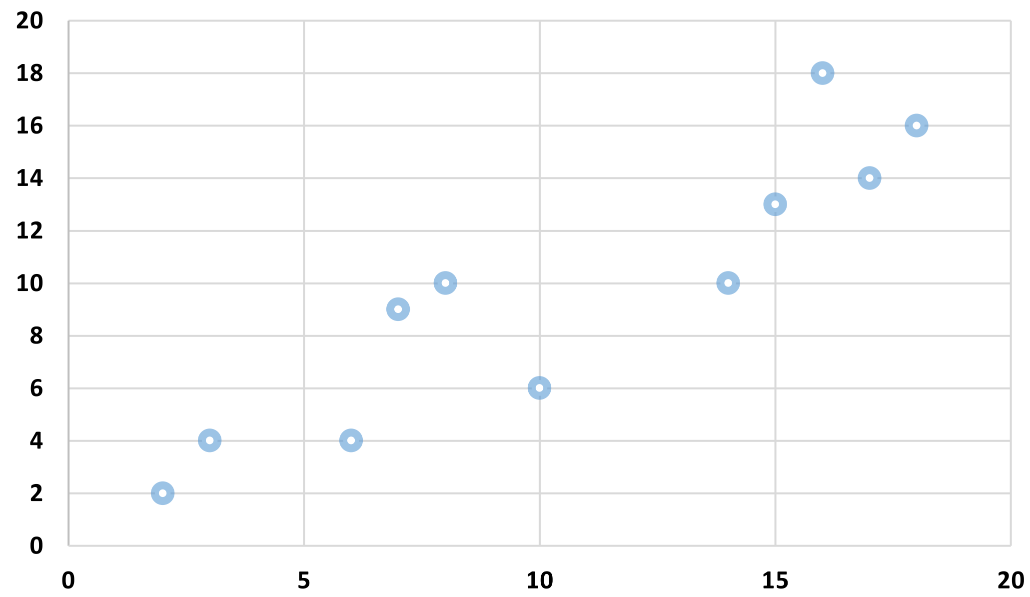
$$\mathbf{X} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \\ 4 & 8 \\ 4 & 0 \end{bmatrix}$$

- What is the rank of $\mathbf{X}$?

https://www.emathhelp.net/calculators/linear-algebra/reduced-row-echelon-form-rref-caclulator/

Example: Normal Equation

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16



Example: Normal Equation

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$X = \begin{bmatrix} \boxed{\begin{matrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{matrix}} & \begin{matrix} 2 \\ 3 \\ 6 \\ 7 \\ 8 \\ 10 \\ 14 \\ 15 \\ 16 \\ 17 \\ 18 \end{matrix} \end{bmatrix} \qquad y = \begin{bmatrix} 2 \\ 4 \\ 4 \\ 9 \\ 10 \\ 6 \\ 10 \\ 13 \\ 18 \\ 14 \\ 16 \end{bmatrix}$$

x_0

$$\mathbf{A} = \mathbf{X}^T\mathbf{X} = \begin{pmatrix} 11, 116 \\ 116, 1552 \end{pmatrix}$$

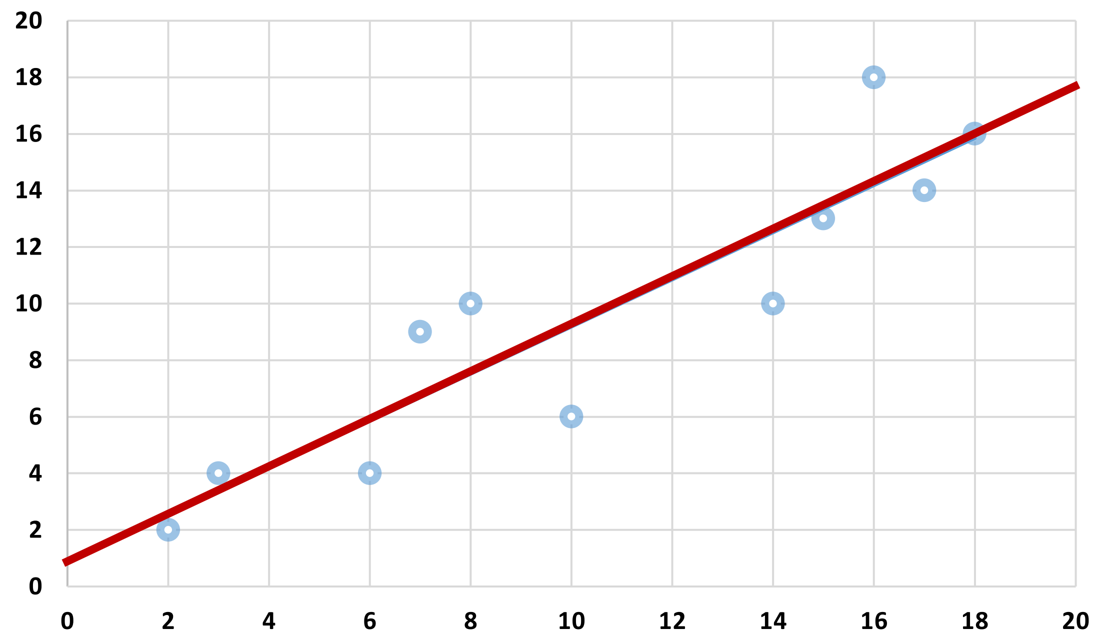
$$\mathbf{b} = \mathbf{X}^T\mathbf{y} = \begin{pmatrix} 106 \\ 1392 \end{pmatrix}$$

$$\mathbf{w} = (A)^{-1}(b) = \begin{pmatrix} 0.840708 \\ 0.834071 \end{pmatrix}$$

Example: Normal Equation

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$y = 0.834071x + 0.840708$$

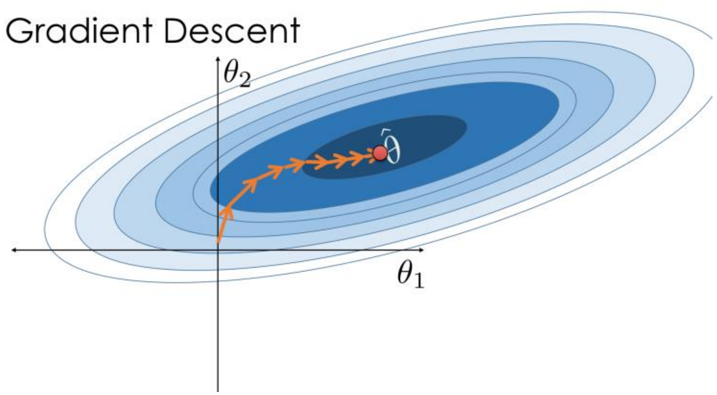


Numerical Solution

Recap: Gradient Descent (GD)

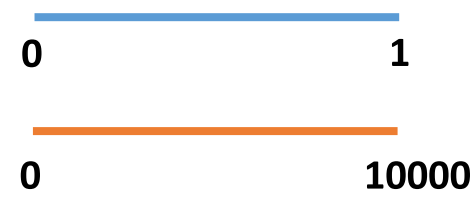
- **Simple concept: follow the gradient *downhill***
- **Process:**
  1. Pick a starting position: $\mathbf{w}^0 = (w_0, w_1, w_2, \dots, w_d)$
  2. Determine the descent direction: $\Delta\mathbf{w} = \nabla E(\mathbf{w}^t)$
  3. Choose a learning rate: η
  4. Update your position: $\mathbf{w}^{t+1} = \mathbf{w}^t - \eta\Delta\mathbf{w}$
  5. Repeat from 2) until the stopping criterion is satisfied.

- **Key issues in GD**
  - **How to compute $\Delta\mathrm{w}$?**
    - **Batch size in $\mathcal{D}$**
  - **How to determine η?**

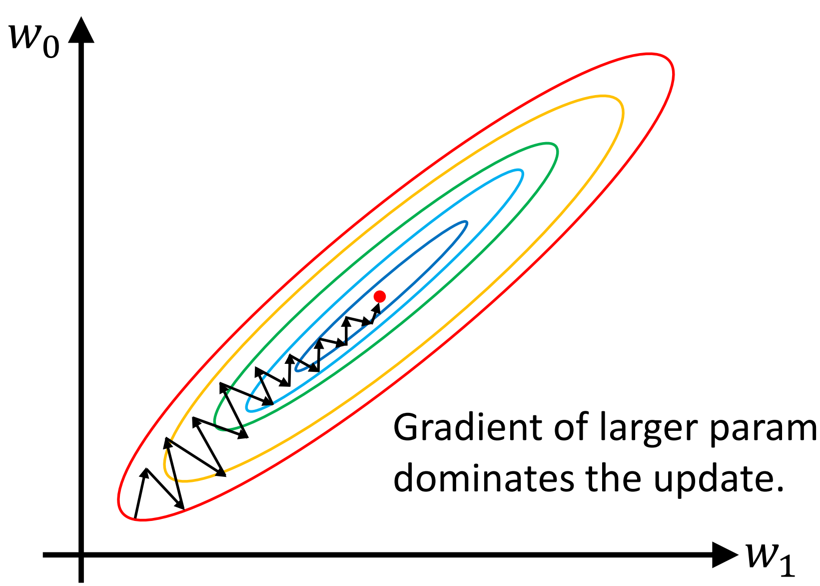


Data Normalization in GD

- **Make sure to scale the data if it is on different scales.**
  - The first input x_1 varies from 0 to 1.
  - The second input x_2 varies from 0 to 10,000.

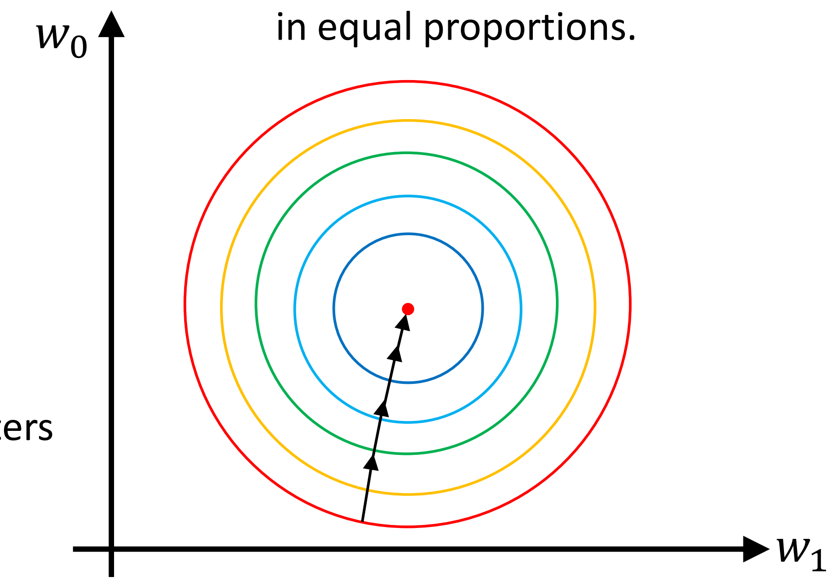


- **Otherwise, the curve would be narrower and taller.**



Unnormalized

Both parameters can be updated in equal proportions.



Normalized

https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

Batch Gradient Descent

- **Most machine learning models rely on the gradient descent and its variants.**

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - \eta \nabla_t E(\mathbf{w})$$

- **Gradient on a full training set → Batch gradient descent**
  - Computed empirically from **all training samples**.

$$\frac{1}{2n}\sum_{i=1}^{n} \nabla E_i(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)\mathbf{x}^{(i)}$$

Detail: Chain Rule for the Derivative

$$\frac{\partial E}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}} \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)^2$$

$$= \frac{1}{2n} \sum_{i=1}^{n} \frac{\partial}{\partial \mathbf{w}} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)^2$$

$$= \frac{1}{2n} \sum_{i=1}^{n} 2 \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right) \frac{\partial}{\partial \mathbf{w}} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right) \frac{\partial}{\partial \mathbf{w}} \left(y^{(i)} - \mathbf{w}^{\mathrm{T}} \mathbf{x}^{(i)} \right)$$

$$\frac{\partial E}{\partial \mathbf{w}} = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right) \left(-\mathbf{x}^{(i)} \right) = -\frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right) \left(\mathbf{x}^{(i)} \right)$$

Example: Multiple Linear Regression

Randomly choose an initial solution $\mathbf{w}^0$,

Repeat

$$\Delta\mathbf{w} = -\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)\mathbf{x}^{(i)}$$

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - \eta\Delta\mathbf{w}$$

Until the stopping condition is satisfied

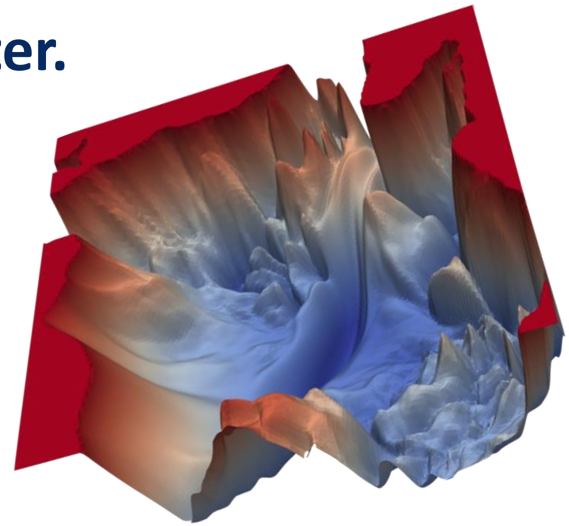
Disadvantages of Batch GD Learning

- Data is often too large to compute the full gradient, the training is too so **slow**.

- Sample gradient $\rightarrow$ **Only an approximation to the true gradient g^t if we knew the true data distribution**.

- No guarantee that it will converge faster.

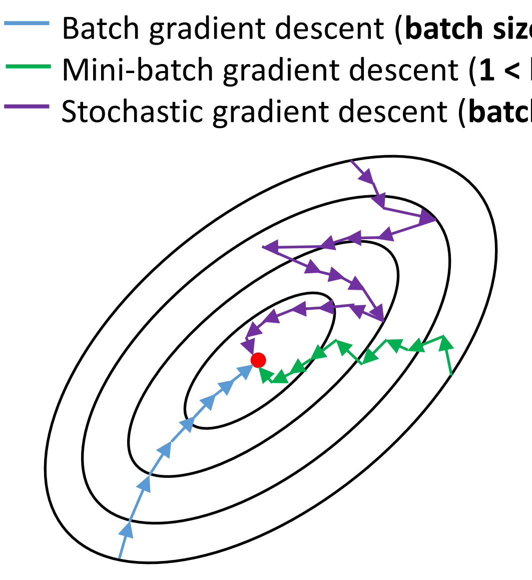
- The loss surface is highly non-convex, so cannot compute the true gradient.



https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/

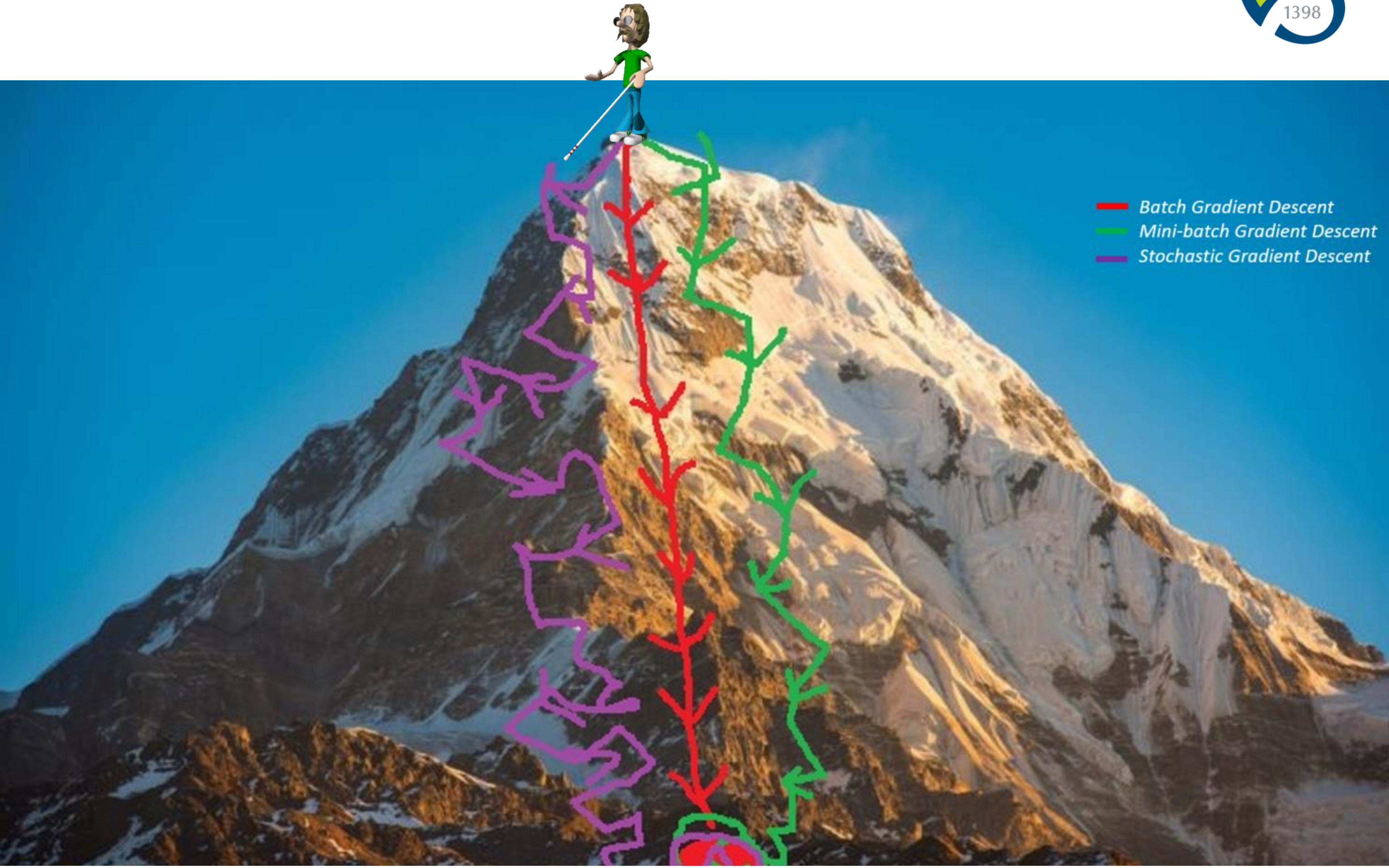
Batch Size in GD

- **The number of training samples is a hyperparameter for learning the model.**
  - **Batch Gradient Descent**: Batch size is set to the **total number of examples** in the training data.
  - **Stochastic Gradient Descent**: Batch size is set to **one**.
  - **Minibatch Stochastic Gradient Descent**: Batch size is set to more than one and less than the total number of examples in the training data.



https://machinelearningmastery.com/how-to-control-the-speed-and-stability-of-training-neural-networks-with-gradient-descent-batch-size/

Batch Size in GD



https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

Mini-batch Stochastic Gradient Descent

- **Introducing an approximation in computing the gradients**
  - Stochastically sample **mini-batches** from a training dataset

$$\mathcal{B} = sample(\mathcal{D})$$

$$\mathbf{w}^{t+1} = \mathbf{w}^{t} - \frac{\eta^{t}}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_{t} E_{i}(\mathbf{w})$$

Example: Multiple Linear Regression

- Use a **small subset or mini-batch** of the data and use it to compute a gradient which is added to the model.

$$E(\mathbf{w}) = \frac{1}{2n}\sum_{i=1}^{n}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

$$\Downarrow$$

$$E(\mathbf{w}) = \frac{1}{2|\mathcal{B}|}\sum_{\left(\mathbf{x}^{(i)},y^{(i)}\right)\in\mathcal{B}}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

- It is possible to compute high-quality models with very few passes in a large-scale dataset.

Example: Multiple Linear Regression

Randomly choose an initial solution $\mathbf{w}^0$,

Repeat

Choose a random sample set $\mathcal{B} \subseteq \mathcal{D}$.

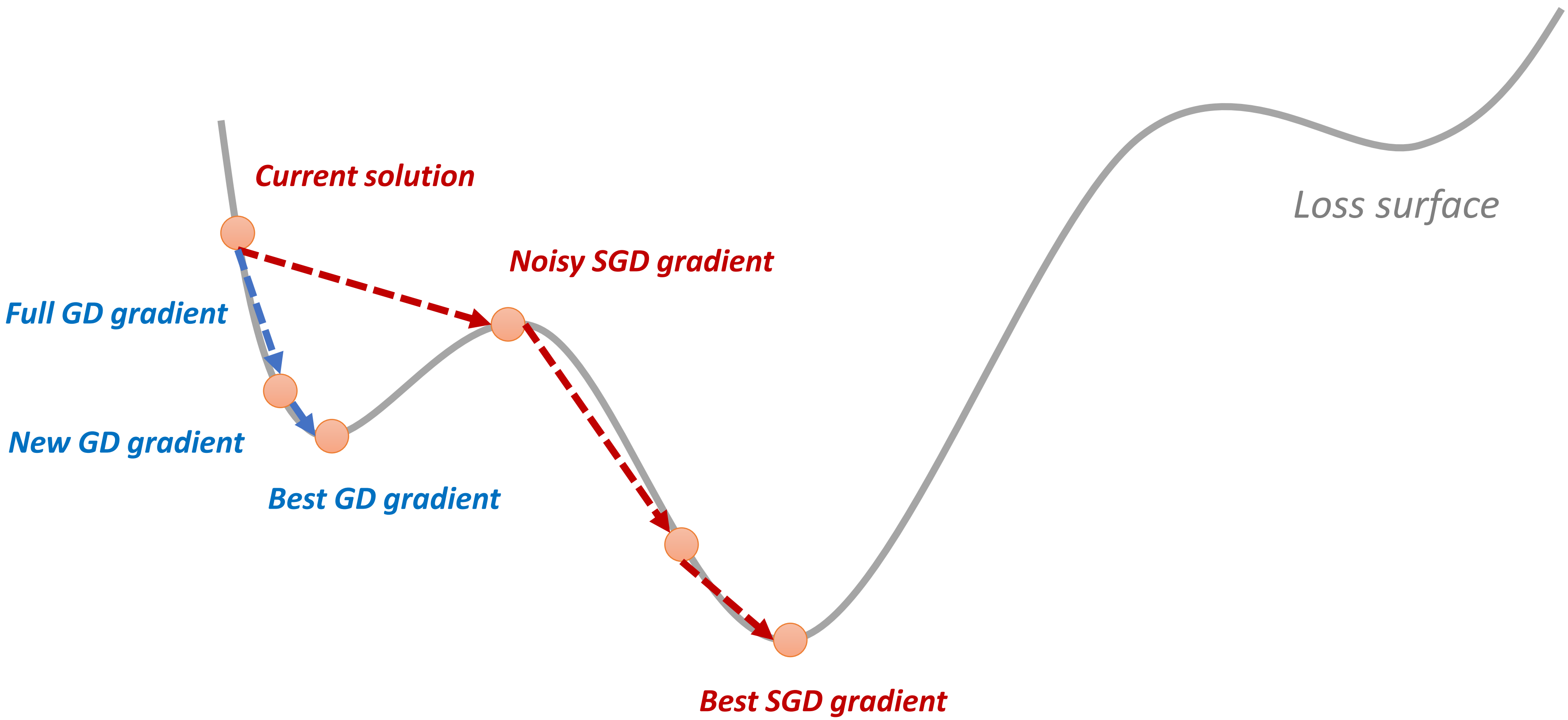
$$\Delta\mathbf{w} = -\frac{1}{|\mathcal{B}|} \sum_{\left(\mathbf{x}^{(i)}, y^{(i)}\right) \in \mathcal{B}} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right) \mathbf{x}^{(i)}$$

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \Delta\mathbf{w}$$

Until the stopping condition is satisfied

SGD is Often Better.

- **Sometimes, noisy SGD can help escaping local optima.**
  - No guarantee that it is what is going to always happen.



Effect of Smaller Batch Sizes

- Smaller batch sizes are used for two main reasons.

- Smaller batch sizes make it easier to **fit one batch worth of training data in GPU memory**.

- Stochastic or mini-batch gradients → **sampled training data sample roughly representative gradients**.

Advantages of SGD

- Random sampling allows being much **faster** than batch gradient descent.
- In practice, the **accuracy** is often **better**.
- Variance of gradients increases as the batch size decreases.

In Practice, SGD vs. GD

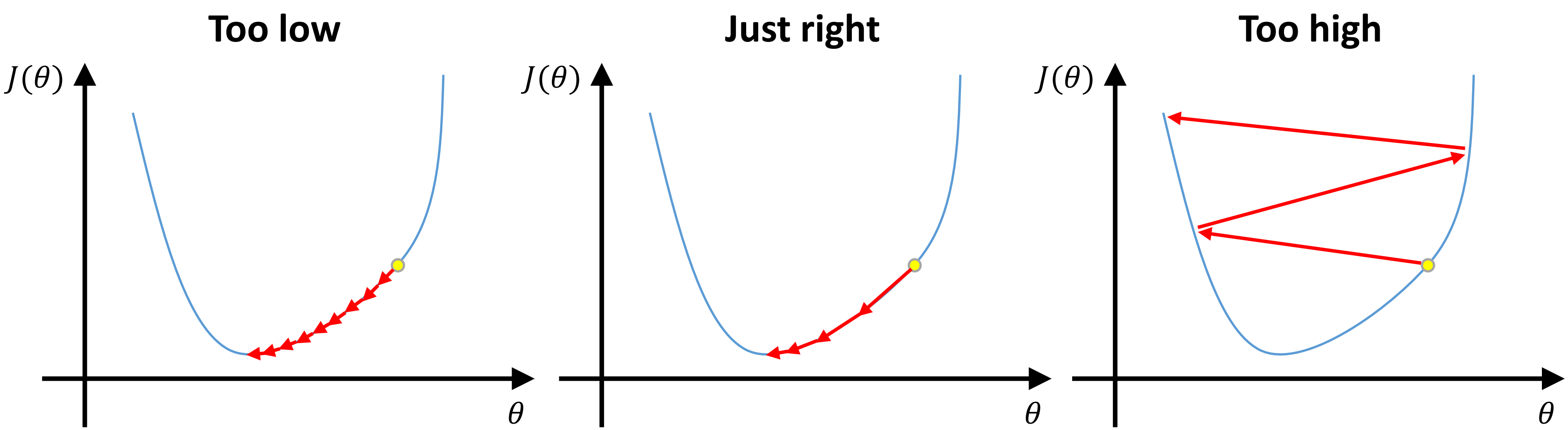
- **SGD is preferred to GD.**

- **Training is faster.**

- **How many samples per mini-batch?**
  - Usually use between 32~256 samples.
  - A good rule of thumb → as many as your GPU fits.

Learning Rate

- **Right learning rate η is very important for fast convergence.**
  - Too big gradients → gradients **overshoot** and **bounce**
  - Too small gradients → **slow training**



Too low: A **small learning rate** requires **many updates** before reaching the minimum point.

Just right: The **optimal** learning rate **swiftly reaches** the minimum point.

Too high: Too **large learning rates** cause **drastic updates** which lead to divergent behaviors.

https://www.jeremyjordan.me/nn-learning-rate/

In Practice, Learning Rate is

- **Try several log-spaced values $10^{-1}, 10^{-2}, 10^{-3}$,… on a smaller set.**

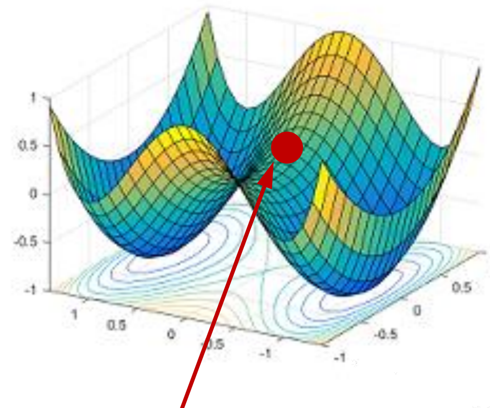
- **Yon can narrow it down from there around where you get the lowest error.**

- **You can decrease the learning rate every 10 (or some other value) full training set epochs.**
  - Although it highly depends on your data.

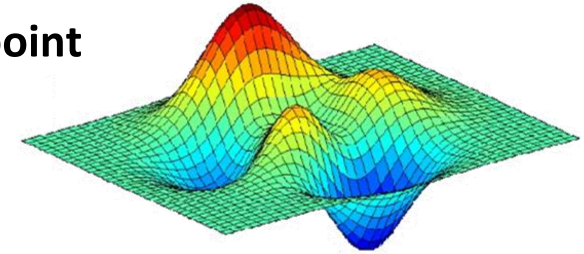
Adjusting Learning Rates

- **Challenges**
  - Choosing an optimal learning rate is **non-trivial**.
  - Updating the **same** learning rate with **all parameters** is problematic.

- **Solutions**
  - **Learning rate schedules** try to adjust the learning rate during training.
  - The learning rate is **different** depending on **parameters**.
  - Use **early stopping** if the error does not improve enough.



Saddle point



http://ruder.io/optimizing-gradient-descent/index.html#additionalstrategiesforoptimizingsgd

Learning Rate Schedulers

- **Constant**

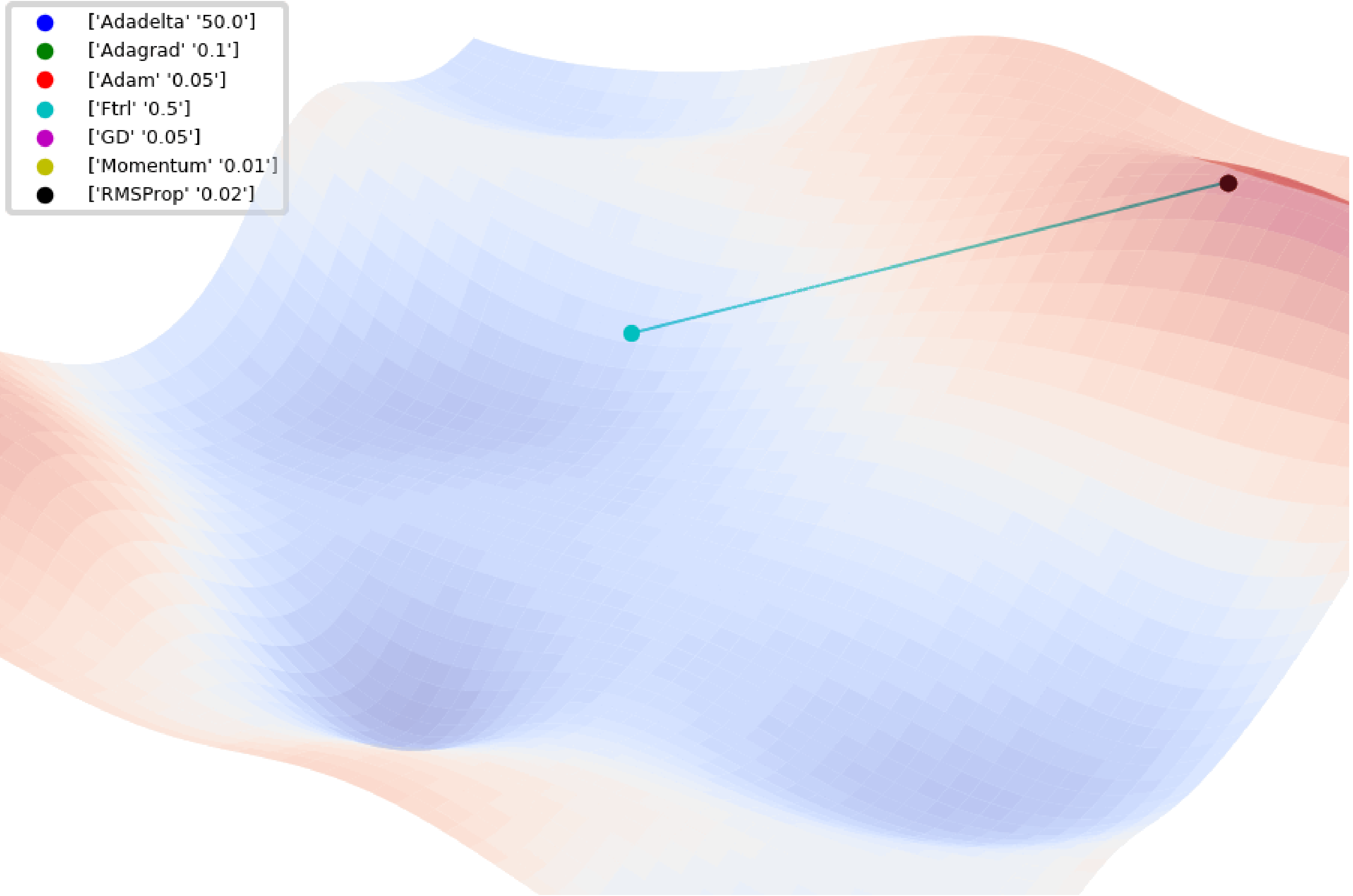
  - Learning rate remains the same for all epochs
- **Step decay**

  - Decrease (e.g., η_t/T) every T number of epochs
- **Inverse decay** $\boldsymbol{\eta_t = \frac{\eta_0}{1+\varepsilon t}}$

- **Exponential decay** $\boldsymbol{\eta_t = \eta_0 e^{-\varepsilon t}}$

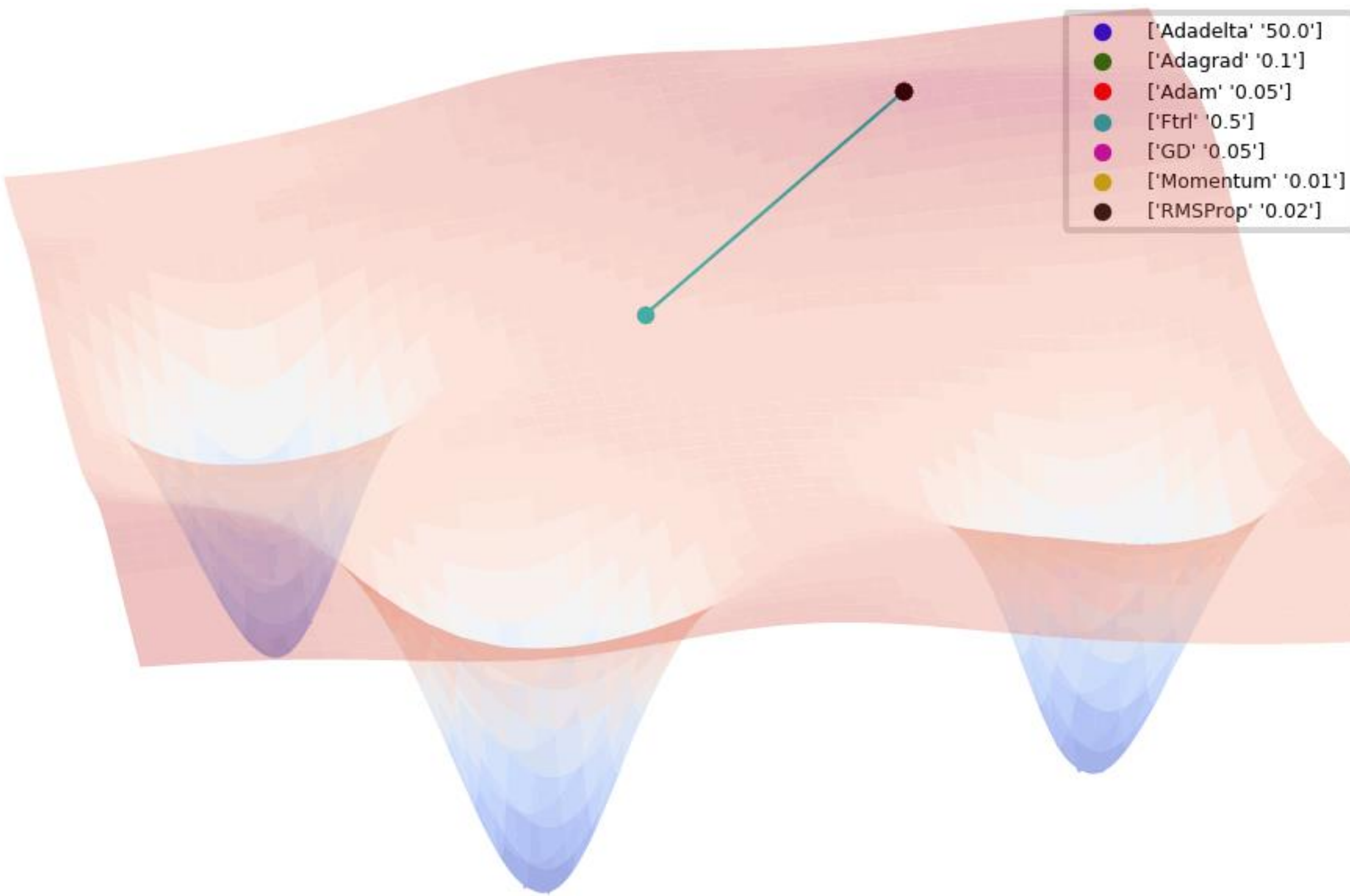
- **Often step decay preferred**
  - Works well and only a single extra hyperparameter T (T =2, 10)

Visualizing Advanced Optimizers



https://github.com/Jaewan-Yun/optimizer-visualization

Visualizing Advanced Optimizers



https://github.com/Jaewan-Yun/optimizer-visualization

Generalized Linear Regression

Generalized Linear Regression (GLM)

Linear regression

- Find $\mathbf{w}$ so that $f(\mathbf{x})$ best fits a given data

$$f(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d = w_0 + \sum_{j=1}^{d} w_j x_j$$

Generalized linear regression

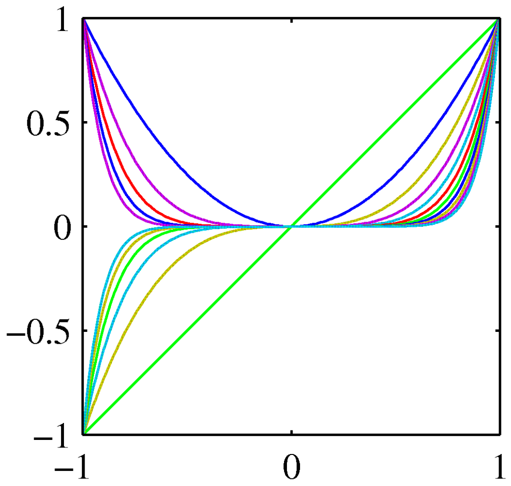
- Instead of using variables, use a **basis function** $\phi_i(\mathbf{x})$ of $\mathbf{x}$.

$$f(\mathbf{x}) = w_0 + w_1 \phi_1(\mathbf{x}) + w_2 \phi_2(\mathbf{x}) + \cdots + w_d \phi_d(\mathbf{x}) = w_0 + \sum_{j=1}^{d} w_j \phi_j(\mathbf{x})$$

Basis Functions in GLM

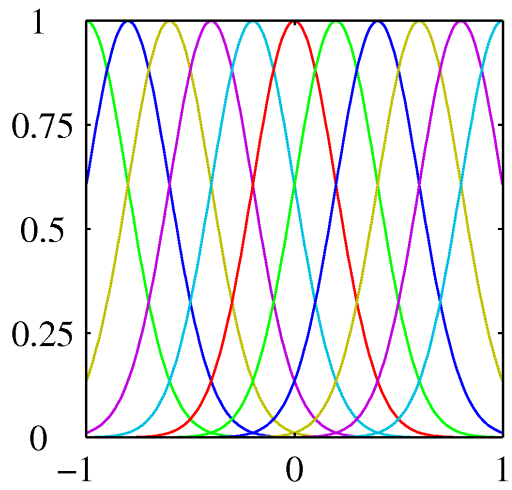
- We can utilize various basis functions.

Polynomial basis functions



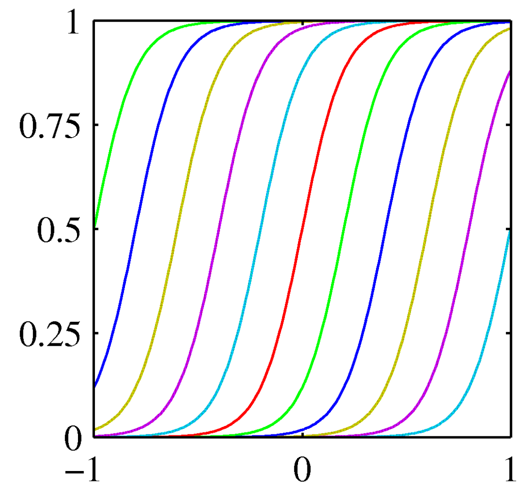
$$\phi_j(x) = x^j$$

Gaussian basis functions



$$\phi_j(x) = \exp\left\{\frac{\left(x-\mu_j\right)^2}{2\sigma^2}\right\}$$

Sigmoidal basis functions



$$\phi_j(x) = \frac{1}{1+\exp\left(-\frac{x-\mu_j}{\sigma}\right)}$$

Possible Models of GLM

➢ Linear regression

$$f(\mathbf{x}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \cdots + w_d\phi_d(\mathbf{x})$$
$$\text{where } \phi_1(\mathbf{x}) = x_1, \phi_2(\mathbf{x}) = x_2, \cdots, \phi_d(\mathbf{x}) = x_d$$

➢ Polynomial regression (2nd order)

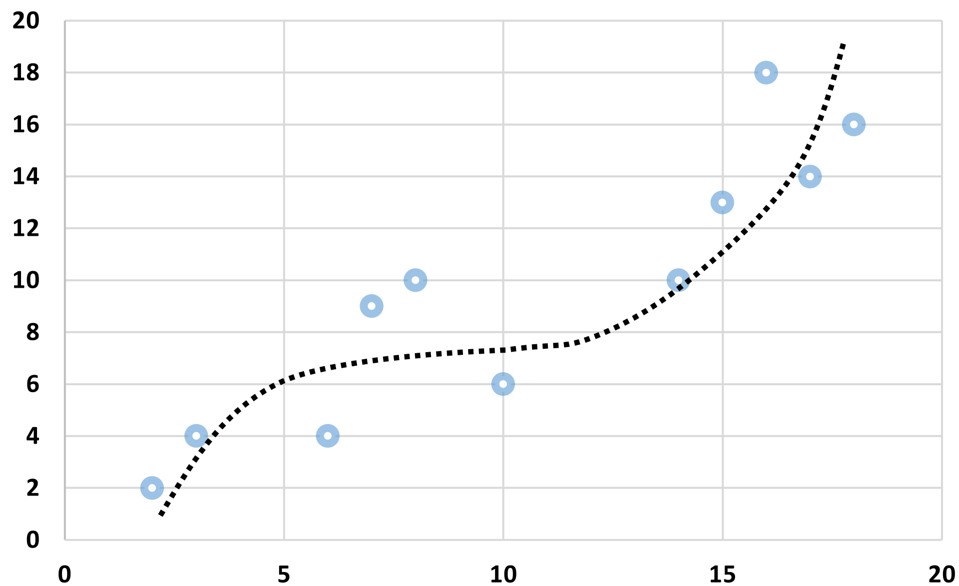
$$f(\mathbf{x}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \cdots + w_{d^2+d}\phi_{d^2+d}(\mathbf{x})$$
$$\begin{aligned}\text{where } & \phi_1(\mathbf{x}) = x_1, \phi_2(\mathbf{x}) = x_2, \cdots, \phi_d(\mathbf{x}) = x_d \\ & \phi_{d+1}(\mathbf{x}) = x_1x_1, \phi_{d+2}(\mathbf{x}) = x_1x_2, \cdots, \phi_{2d}(\mathbf{x}) = x_1x_d \\ & \cdots \\ & \phi_{d^2+1}(\mathbf{x}) = x_dx_1, \phi_{d^2+1}(\mathbf{x}) = x_dx_2, \cdots, \phi_{d^2+d}(\mathbf{x}) = x_dx_d\end{aligned}$$

Example: Generalized Linear Regression

- Finding the 3rd order polynomial which best fits the data

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x)$$
where $\phi_1(x) = x, \phi_2(x) = x^2, \phi_3(x) = x^3$

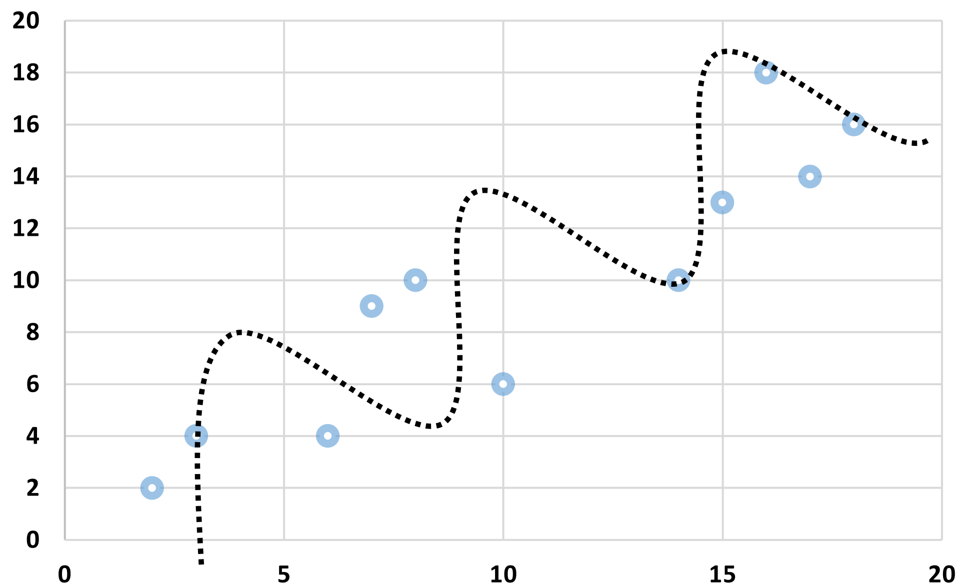


Example: Generalized Linear Regression

➢ **Finding the periodic curve which best fits the data**

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = w_0 + w_1\phi_1(x) + w_2\phi_2(x)$$
where $\phi_1(x) = x, \phi_2(x) = \sin(x)$



Solving Generalized Linear Regression

- **Given a dataset** $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)}): 1 \leq i \leq n\}$,
  - $\mathbf{x}^{(i)} = (x_{i0}, x_{i1}, x_{i2}, \dots, x_{id})$ is the input on $(d+1)$-dimensional space.
  - $y^{(i)}$ is the output.

- **Finding the hyperplane** $f(\mathbf{x})$ **which best fits the data** $\mathcal{D}$
  - Make $f(\mathbf{x}^{(i)})$ close to $y^{(i)}$ as much as possible for $i = 1, \dots, n$

$$E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f(\mathbf{x}^{(i)}) \right)^2$$

$$f(\mathbf{x}^{(i)}) = w_0 \phi_0(\mathbf{x}^{(i)}) + w_1 \phi_1(\mathbf{x}^{(i)}) + \cdots + w_k \phi_k(\mathbf{x}^{(i)}) \quad \text{where } \phi_0(\mathbf{x}^{(i)}) = 1$$

Solving Generalized Linear Regression

- Given $\mathcal{D} = \left\{\left(x^{(i)}, y^{(i)}\right): 1 \leq i \leq n\right\}$
- Find $\mathbf{w} = [w_0, w_1, \dots, w_k]$ which minimizes $E(\mathbf{w})$

$$E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2$$

$$f(\mathbf{x}^{(i)}) = w_0 \phi_0(\mathbf{x}^{(i)}) + w_1 \phi_1(\mathbf{x}^{(i)}) + \cdots + w_k \phi_k(\mathbf{x}^{(i)}) \quad \text{where } \phi_0(\mathbf{x}^{(i)}) = 1$$

- How to solve this?

Solving Multiple Linear Regression

- **When a function is convex, continuous, and differentiable, a necessary and sufficient condition for a point $\mathbf{w}^*$ to be optimal is $\nabla E(\mathbf{w}^*) = \mathbf{0}$.**

$$\frac{\partial}{\partial w_j} E(w_0, w_1, \cdots, w_k) = 0 \;\; \text{for } j = 0, \cdots, k$$

$$\Downarrow$$

$$\frac{\partial}{\partial \mathbf{w}} E(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} \left(\frac{\partial}{\partial \mathbf{w}} \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)^2 \right) = \frac{1}{2n} \sum_{i=1}^{n} \left(-2\phi_j\left(\mathbf{x}^{(i)}\right) \right) \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right) \right)$$

Deriving the Normal Equation

➢ **Obtain $\mathbf{w} = (w_0, w_1, \cdots, w_k)$ by solving these equations.**

$$\left.\begin{aligned}
&\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0 \\
&\sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0 \\
&\qquad\qquad \cdots \cdots \cdots \\
&\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0
\end{aligned}\right\}$$

There are $k+1$ variables
$k+1$ equations.

- The derivation of generalized linear regression is almost similar to that of the existing linear regression.

Deriving the Normal Equation

$$\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0$$

$$\sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0$$

$$\dots$$

$$\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\big(f(\mathbf{x}^{(i)} - y^{(i)}\big) = 0$$

$$\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\big(w_0\phi_0(\mathbf{x}^{(i)}) + w_1\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\phi_k(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0$$

$$\sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\big(w_0\phi_0(\mathbf{x}^{(i)}) + w_1\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\phi_k(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0$$

$$\dots$$

$$\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\big(w_0\phi_0(\mathbf{x}^{(i)}) + w_1\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\phi_k(\mathbf{x}^{(i)}) - y^{(i)}\big) = 0$$

$$w_0\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) + w_1\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) = \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\, y^{(i)}$$

$$w_0\sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) + w_1\sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\sum_{i=1}^{n} \phi_1\big(\mathbf{x}^{(i)}\big)\phi_k\big(\mathbf{x}^{(i)}\big) = \sum_{i=1}^{n} \phi_1\big(\mathbf{x}^{(i)}\big)y^{(i)}$$

$$\dots$$

$$w_0\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) + w_1\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) + \cdots + w_k\sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) = \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\, y^{(i)}$$

Deriving the Normal Equation

$$w_0 \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) \quad + w_1 \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) \quad + \cdots + w_k \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \quad = \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\, y^{(i)}$$

$$w_0 \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) \quad + w_1 \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) \quad + \cdots + w_k \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \quad = \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})y^{(i)}$$

$$\cdots$$

$$w_0 \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}) \quad + w_1 \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}) \quad + \cdots + w_k \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \quad = \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\, y^{(i)}$$

$$\Rightarrow \begin{pmatrix} \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \\ \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \\ \cdots \\ \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_k \end{pmatrix} = \begin{pmatrix} \phi_0(\mathbf{x}^{(i)}), \phi_0(\mathbf{x}^{(i)}), \cdots, \phi_0(\mathbf{x}^{(i)}) \\ \phi_1(\mathbf{x}^{(i)}), \phi_1(\mathbf{x}^{(i)}), \cdots, \phi_1(\mathbf{x}^{(i)}) \\ \cdots \\ \phi_k(\mathbf{x}^{(i)}), \phi_k(\mathbf{x}^{(i)}), \cdots, \phi_k(\mathbf{x}^{(i)}) \end{pmatrix} \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \cdots \\ y^{(n)} \end{pmatrix}$$

Deriving the Normal Equation

$$\begin{pmatrix} \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_0(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \\ \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_1(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \\ \cdots \\ \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_0(\mathbf{x}^{(i)}), \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_1(\mathbf{x}^{(i)}), \cdots, \sum_{i=1}^{n} \phi_k(\mathbf{x}^{(i)})\phi_k(\mathbf{x}^{(i)}) \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_k \end{pmatrix} = \begin{pmatrix} \phi_0(\mathbf{x}^{(i)}), \phi_0(\mathbf{x}^{(i)}), \cdots, \phi_0(\mathbf{x}^{(i)}) \\ \phi_1(\mathbf{x}^{(i)}), \phi_1(\mathbf{x}^{(i)}), \cdots, \phi_1(\mathbf{x}^{(i)}) \\ \cdots \\ \phi_k(\mathbf{x}^{(i)}), \phi_k(\mathbf{x}^{(i)}), \cdots, \phi_k(\mathbf{x}^{(i)}) \end{pmatrix} \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \cdots \\ y^{(n)} \end{pmatrix}$$

$$\Rightarrow \quad \mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\mathbf{w} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{y} \quad \Rightarrow \quad \mathbf{w} = \left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{y}$$

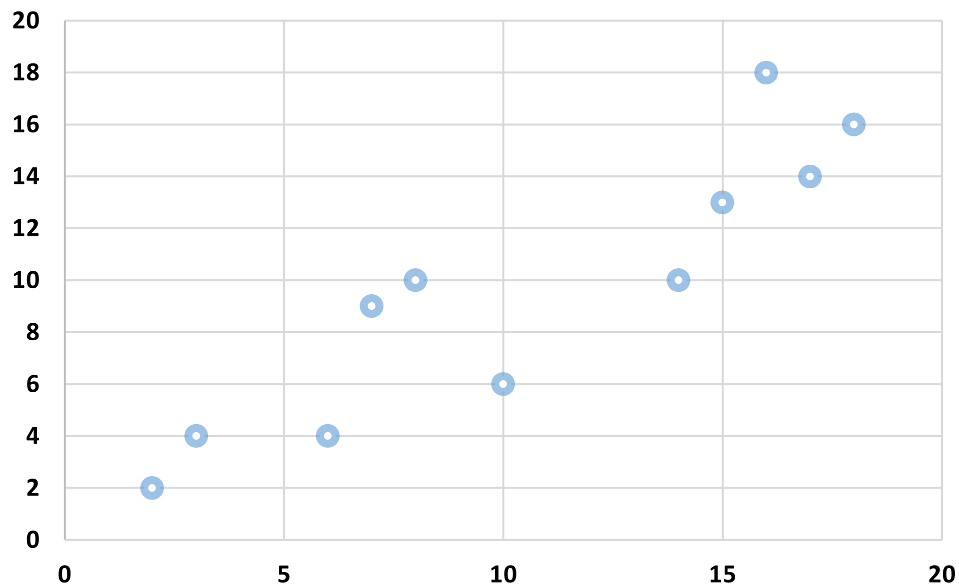
where $\mathbf{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}^{(1)}), \phi_1(\mathbf{x}^{(1)}), \cdots, \phi_k(\mathbf{x}^{(1)}) \\ \phi_0(\mathbf{x}^{(2)}), \phi_1(\mathbf{x}^{(2)}), \cdots, \phi_k(\mathbf{x}^{(2)}) \\ \cdots \\ \phi_0(\mathbf{x}^{(n)}), \phi_1(\mathbf{x}^{(n)}), \cdots, \phi_k(\mathbf{x}^{(n)}) \end{pmatrix}$ $\quad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_k \end{pmatrix}$ $\quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{pmatrix}$

Example: Generalized Linear Regression

- Finding the 3rd order polynomial which best fits the data

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$f(x) = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x)$
where $\phi_1(x) = x, \phi_2(x) = x^2, \phi_3(x) = x^3$

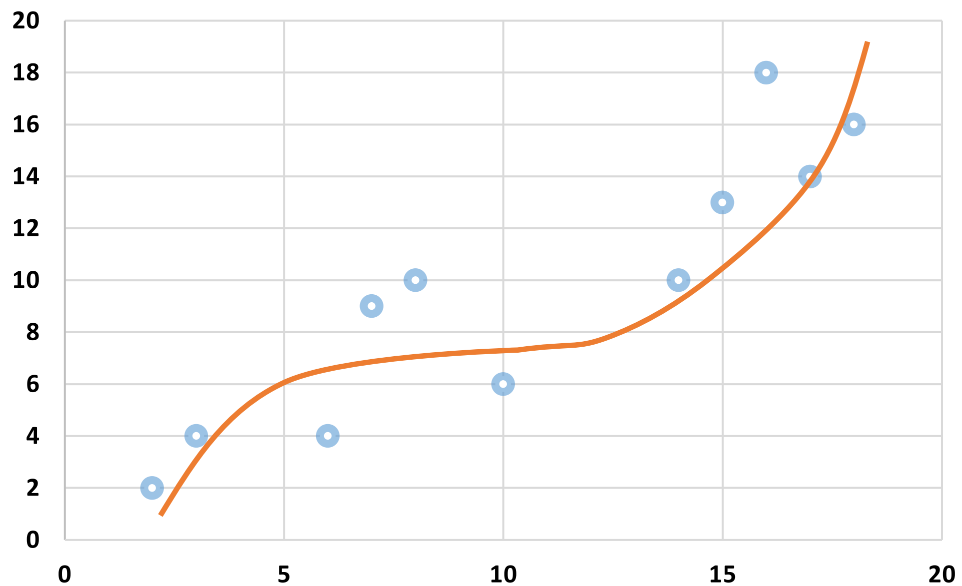


Example: Generalized Linear Regression

- Finding the 3rd order polynomial which best fits the data

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = -0.63143704 + 1.713506327 \times x - 0.121906349 \times x^2 + 0.004481823 \times x^3$$

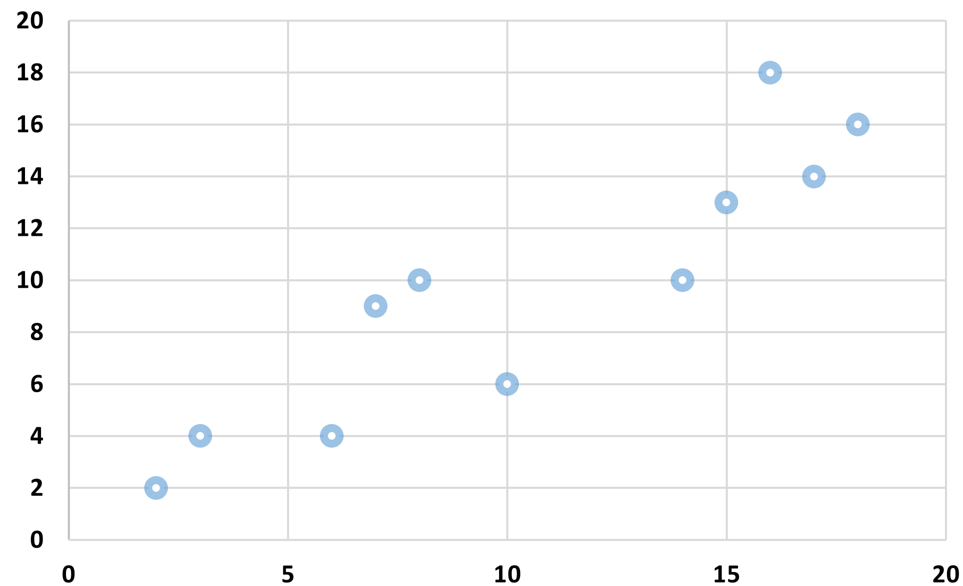


Example: Generalized Linear Regression

- **Finding the periodic curve which best fits the data**

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = w_0 + w_1\phi_1(x) + w_2\phi_2(x)$$
where $\phi_1(x) = x, \phi_2(x) = \sin(x)$

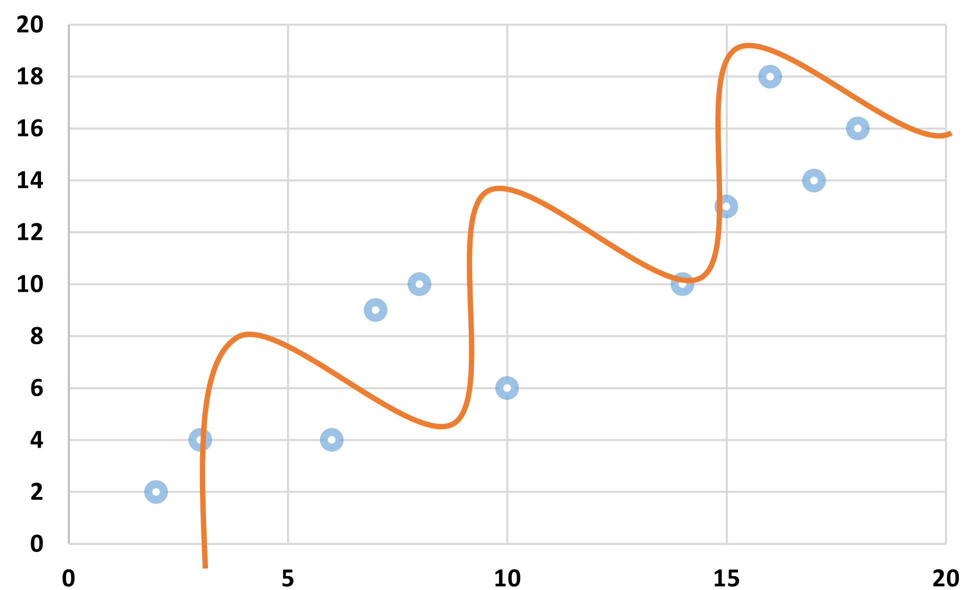


Example: Generalized Linear Regression

- Finding the **periodic curve** which best fits the data

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = 0.328770262 + 0.873739195 * x + 0.680191174 * sin(x)$$



Example: Generalized Linear Regression

➢ Kernel regression

- Possible to find a linear combination of given **kernel functions**

x	y
2	2
3	4
6	4
7	9
8	10
10	6
14	10
15	13
16	18
17	14
18	16

$$f(x) = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x)$$

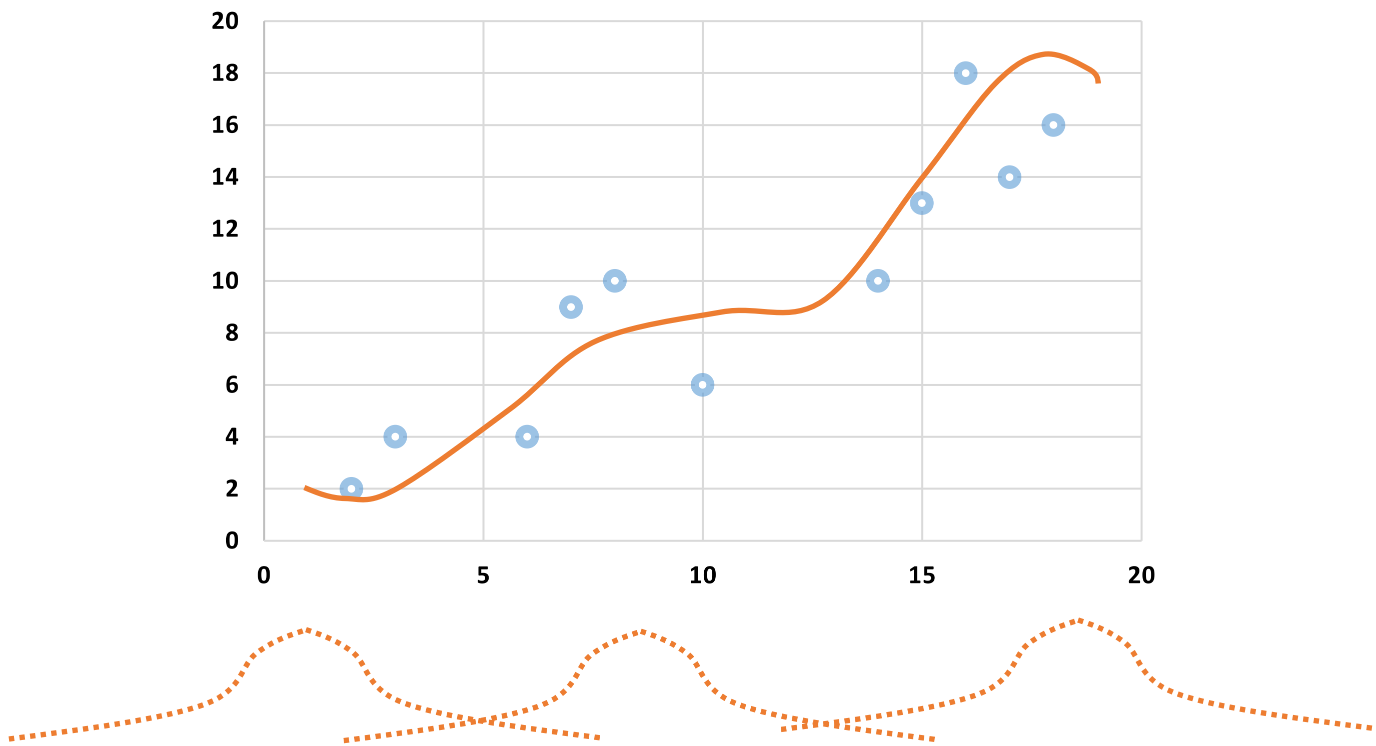
$$\text{where } \phi_1(x) = \exp\left(\frac{-(x-1)^2}{18}\right),$$

$$\phi_2(x) = \exp\left(\frac{-(x-9)^2}{18}\right),$$

$$\phi_3(x) = \exp\left(\frac{-(x-18)^2}{18}\right)$$

Example: Generalized Linear Regression

$$f(x) = 5.76 - 3.64\exp\left(\frac{-(x-1)^2}{18}\right) + 2.40\exp\left(\frac{-(x-9)^2}{18}\right) + 10.82\exp\left(\frac{-(x-18)^2}{18}\right)$$



Q&A