# 4. Linear Regression with multiple variables

## 4.1 Multiple features

**Multiple features (variables).**

| Size (feet²) $x_1$ | Number of bedrooms $x_2$ | Number of floors $x_3$ | Age of home (years) $x_4$ | Price ($1000) $y$ |
|---|---|---|---|---|
| 2104 | 5 | 1 | 45 | 460 |
| 1416 | 3 | 2 | 40 | 232 |
| 1534 | 3 | 2 | 30 | 315 |
| 852 | 2 | 1 | 36 | 178 |
| … | … | … | … | … |

$m = 47$ example 수

Notation

$n$ = feature 수

$x^{(i)}$ = i 번째 training example

$x_j^{(i)}$ = i 번째 training example의 j 번째 feature

ex) $x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$, $x_3^{(2)} = 2$

hypothesis

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \quad (x_0^{(i)} = 1)$

$= \boxed{\theta^T \cdot x}$

$\hookrightarrow \quad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$

$\Rightarrow \underbrace{[\theta_0\ \theta_1\ \cdots\ \theta_n]}_{\theta^T} \cdot \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}}_{x} = \theta^T \cdot x$

# 4.2 Gradient descent for multiple variables

Cost function:

$$J(\theta_0, \theta_1, \cdots, \theta_n) = J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Gradient descent:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial\theta_j} J(\theta) \quad \text{(simultaneously update)}$$

$$= \theta_j - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}}$$

$$\begin{cases} \theta_0 := \theta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right) x_0^{(i)} \\ \theta_1 := \theta_1 - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right) x_1^{(i)} \end{cases}$$

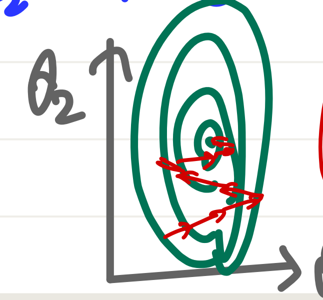
# 4.3 Gradient descent in practice I: Feature Scaling

make sure features are on a similar scale.

ex) $x_1$ : size (0~2000 $feet^2$) $\rightarrow$ $x_1 : \frac{size(feet^2)}{2000}$

$x_2$ : # bedrooms (1~5) $\rightarrow$ $x_2 = \frac{\# bedrooms}{5}$

$\boxed{-1 \leq x_i \leq 1}$



뾰족, 좁은 타원모양
오랜 시간 동안
앞뒤로 진동 후 → 최솟값 도달

더 균형적인 원모양
빠르게 수렴,
안정적인 경로

# 4. Linear Regression with multiple variables

Feature scaling 함으로써 Gradient descent 더 빨라지고 적은 수의 반복으로도 수렴!

- 정확하게 같은 값 or 범위가 1이 아니라도 괜찮! (충분히 근접하면 ok)

ex) $-3 \le x_1 \le 3$ (ok)   $-100 \le x_3 \le 100$ (X)

$-\frac{1}{3} \le x_2 \le \frac{1}{3}$ (ok)   $-0.0001 \le x_4 \le 0.0001$ (X)

· Mean normalization

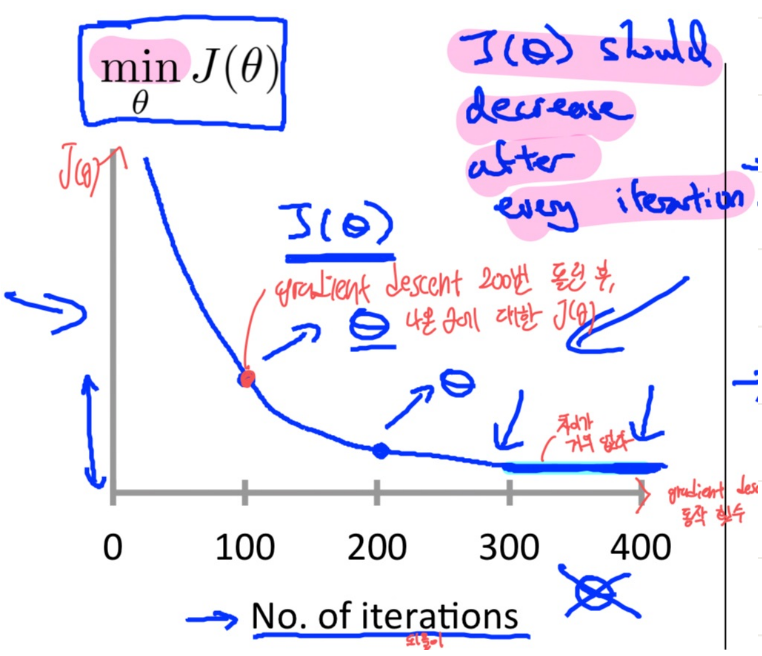
ex) $x_1 = \frac{size - 1000}{2000 - 0}$   average

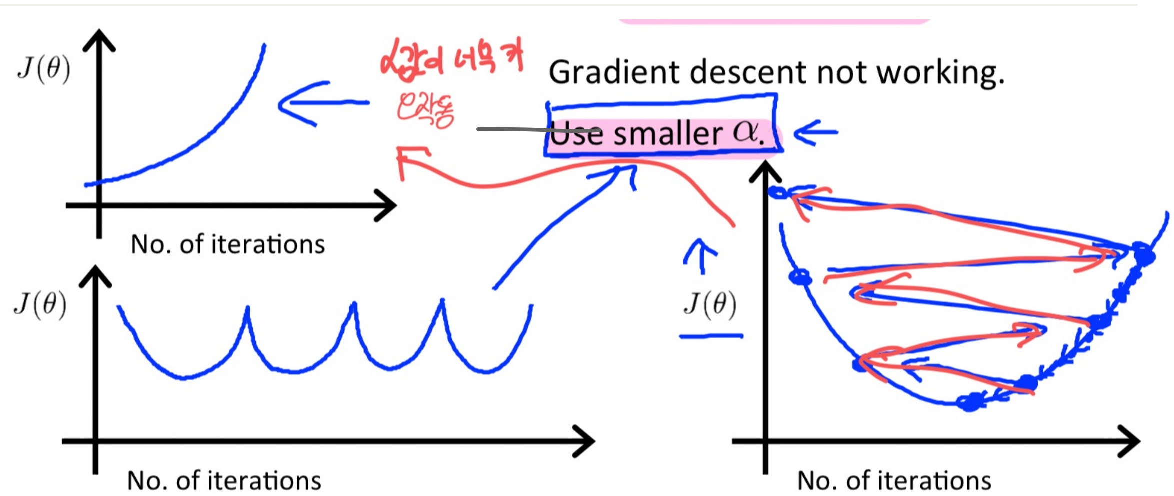
range (max-min) or $s$ (표준편차)

$x_2 : \frac{\#bedrooms - 2}{5 - 1}$

$$\boxed{\frac{x_1 - \mu_1}{s_1}}$$

## 4.4 Gradient descent in practice II: Learning rate (α)




ex) 0.001 → 0.003 → 0.01 → 0.03 → 0.1 → 0.3 → 1 ...

α가 너무 작으면 → slow convergence

α가 너무 크면 → not decrease on every iteration, may not converge

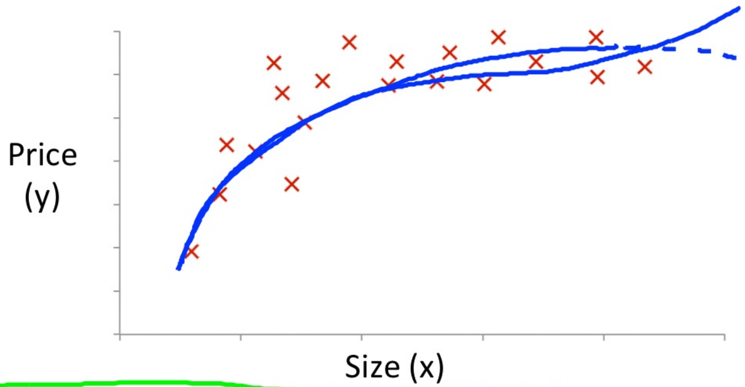
교수님 → α 3배씩 늘려서 J(θ) 그래프 관찰 → 적절한 α 찾기

# 4. Linear Regression with multiple variables

## 4.5 Features and polynomial regression

(다항 회귀, 비선형)

**Polynomial regression**




→ $\theta_0 + \theta_1 x + \theta_2 x^2$ 2차식

→ $\boxed{\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3}$ 3차식

자동으로 feature 선택해주는 알고리즘 존재

↓

알맞은 feature 선택하는 다양한 방법이 있음

$$\boxed{h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3}$$
$$= \theta_0 + \theta_1(size) + \theta_2(size)^2 + \theta_3(size)^3$$

→ $x_1 = (size)$
→ $x_2 = (size)^2$
→ $x_3 = (size)^3$

이렇게 적용하면
feature scailing의 적용은 더 중요해짐
(매우 다른 범위를 가지기 때문에 → 비슷한 범위로 바꿔주는건 필요)

Size: 1 ~ 1000 (range)
$size^2$: 1 ~ 1,000,000
$size^3$: 1 ~ $10^9$

Andrew Ng

## 4.6 Normal equation (정규 방정식)

Gradient Descent에서 θ의 최적값 (반복없이) 한번에 구할 수 있다!

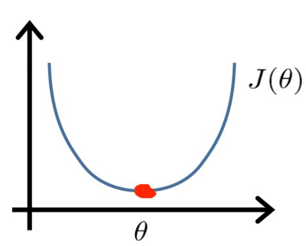
Intuition: If 1D $(\theta \in \mathbb{R})$ Scalar (not vector)

→ $J(\theta) = a\theta^2 + b\theta + c$

$\frac{d}{d\theta} J(\theta) = \cdots \overset{set}{\underset{change}{=}} 0$

Solve for θ

θ가 실수인 경우 가능




$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

↑ cost ftn 최소화 하는 값

* θ가 vector 인 경우

$\theta \in \mathbb{R}^{n+1}$ $\quad J(\theta_0, \theta_1, \ldots, \theta_m) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

$\frac{\partial}{\partial \theta_j} J(\theta) = \cdots \overset{set}{=} 0$ (for every $j$) 편미분

Solve for $\theta_0, \theta_1, \ldots, \theta_n$

Examples: $m = 4$.

| $x_0$ | Size (feet²) $x_1$ | Number of bedrooms $x_2$ | Number of floors $x_3$ | Age of home (years) $x_4$ | Price ($1000) $y$ |
|---|---|---|---|---|---|
| 1 | 2104 | 5 | 1 | 45 | 460 |
| 1 | 1416 | 3 | 2 | 40 | 232 |
| 1 | 1534 | 3 | 2 | 30 | 315 |
| 1 | 852 | 2 | 1 | 36 | 178 |

m=4

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

m × (n+1) $\qquad$ m-dimensional vector

$\boxed{\theta = (X^T X)^{-1} X^T y}$ ← cost ftn 최소화 하는 값

Andrew Ng

# 4. Linear Regression with multiple variables

$m$ **examples** $(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$ ; $n$ **features.**

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$\underbrace{X}_{\text{(design matrix)}} = \underbrace{\begin{bmatrix} \text{---} & (x^{(1)})^T & \text{---} \\ \text{---} & (x^{(2)})^T & \text{---} \\ & \vdots & \\ \text{---} & (x^{(m)})^T & \text{---} \end{bmatrix}}_{m \times (n+1)}$$

E.g. If $x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$

$$X = \underbrace{\begin{bmatrix} 1 & x_1^{(1)} \\ 1 & x_1^{(2)} \\ \vdots & \vdots \\ 1 & x_1^{(m)} \end{bmatrix}}_{m \times 2} \qquad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

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

※ feature 수 1000보다 작은 경우 효과적!

| Gradient Descent | Normal Equation |
| --- | --- |
| · $\alpha$ 선택해야 함 | · $\alpha$ 선택할 필요X |
| · 반복 필요 | · 반복 필요X |
| · n이 커도 잘동작 $O(kn^2)$ | · 계산 필요 |
| | · n이 커지면 느려짐 $O(n^3)$ |

# 4.7 Normal equation and non-invertibility

$X^T X$ : 역행렬 없는 경우 (매우 드물다)

$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

① Redundant features → 1st 불필요한 feature 있는지 확인 후, delete

② Too many features → 2nd Delete some features

→ or 3rd regularization