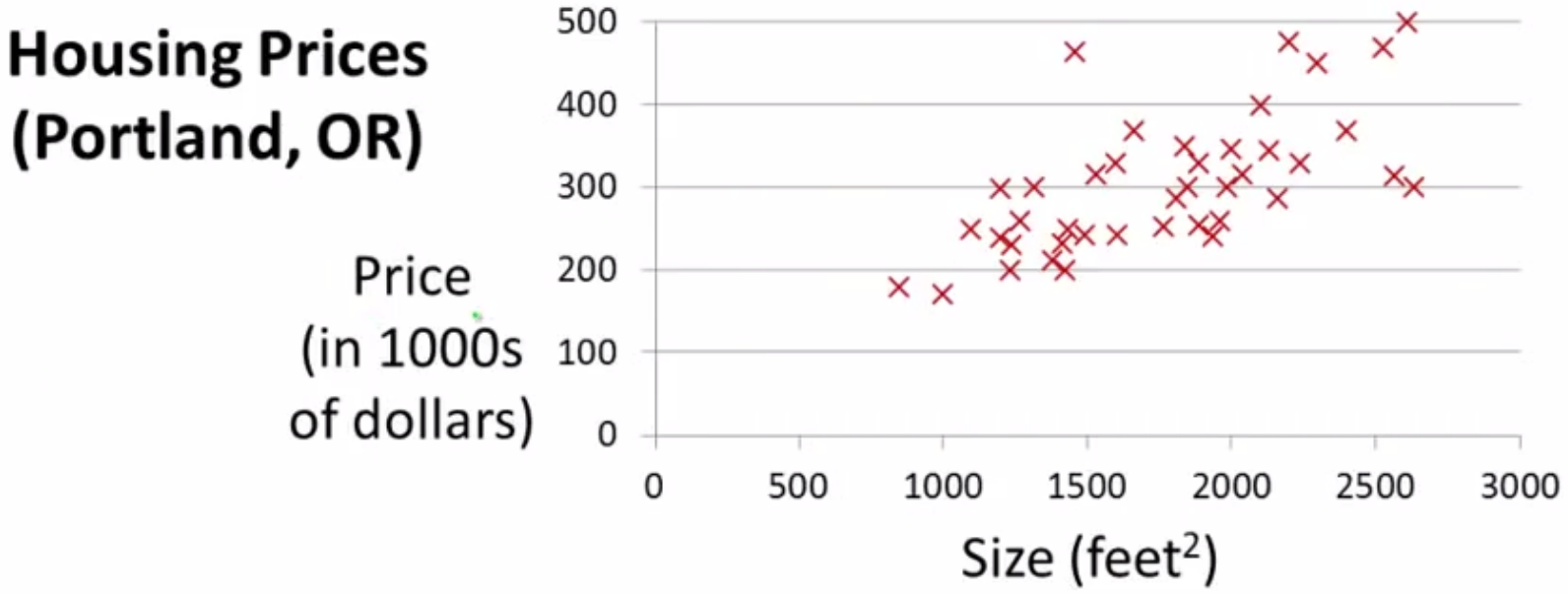
Regressão Linear

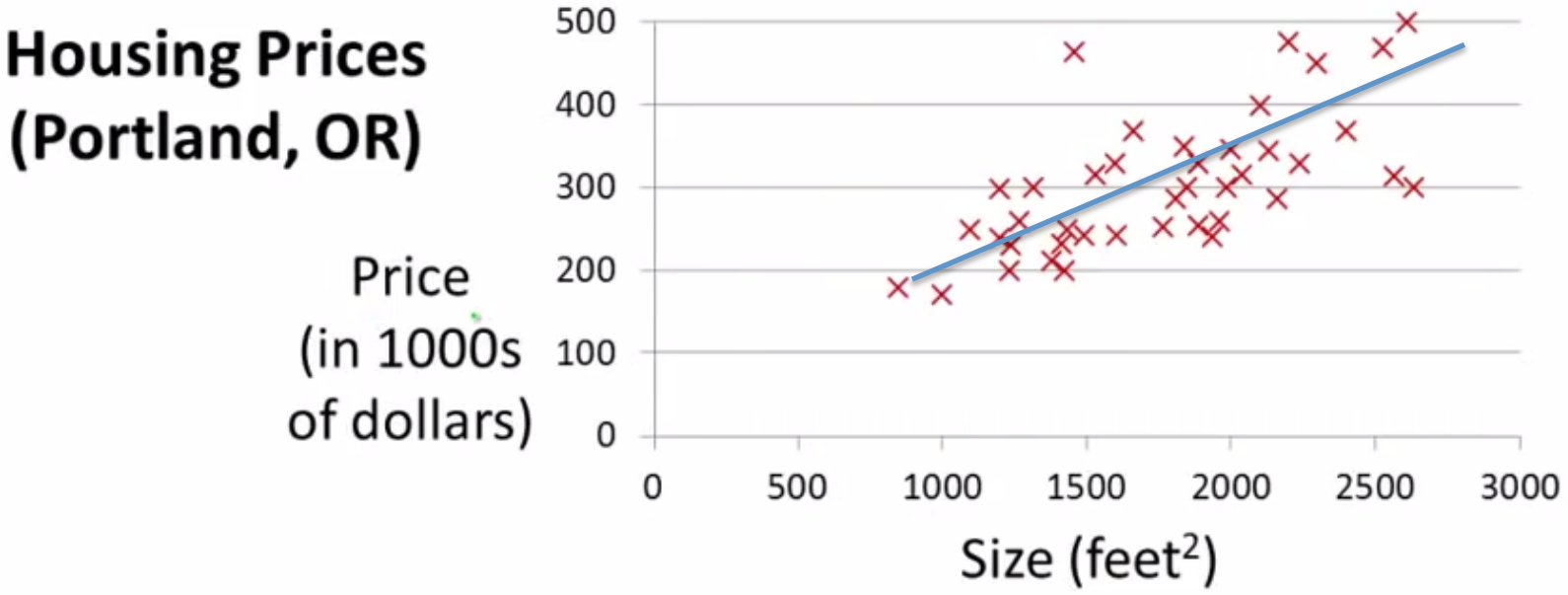
Prof.: Eric A. Antonelo

Slides baseados no curso de *Machine Learning* de Andrew Ng

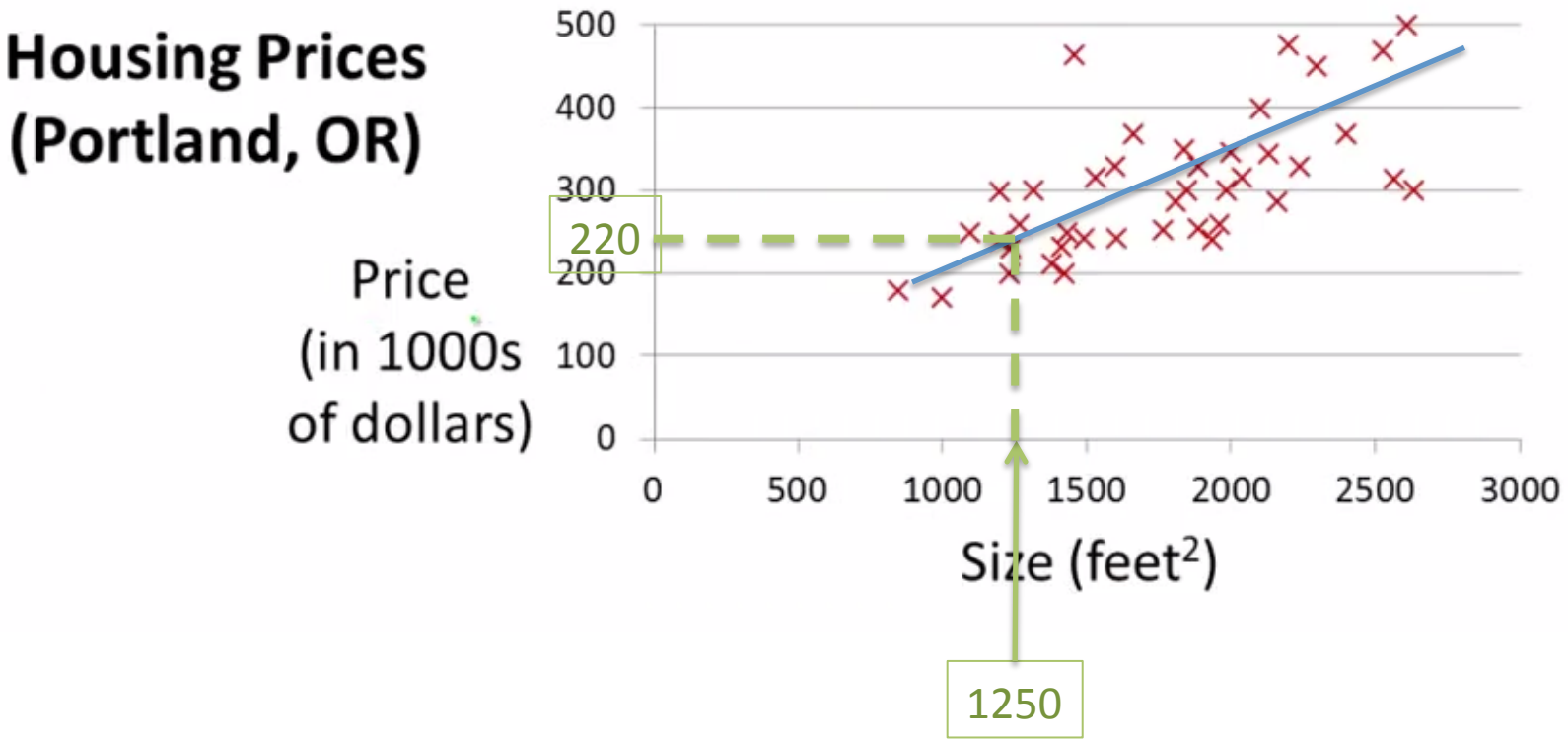
Exemplos de dados



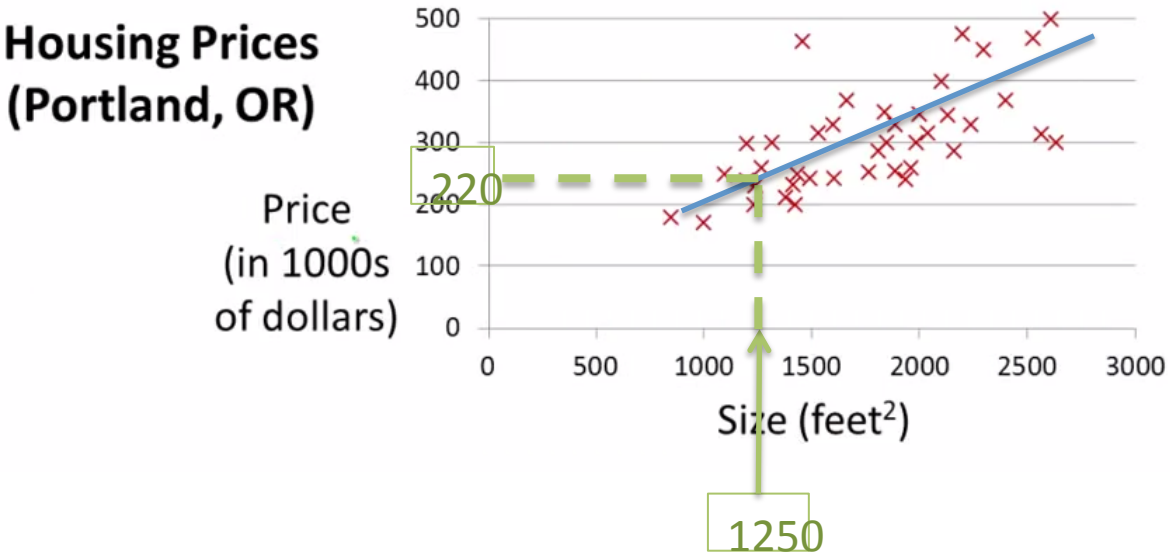
Criando um modelo/hipótese



Utilizando o modelo



Utilizando o modelo



<u>Aprendizagem supervisionada</u>
A resposta desejada é conhecida para cada exemplo no conjunto

<u>Problema de regressão</u>
Predição de saída real (valor contínuo)

<u>Problema de classificação?</u>

Conjunto de treinamento

Training set of housing prices (Portland, OR)

Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

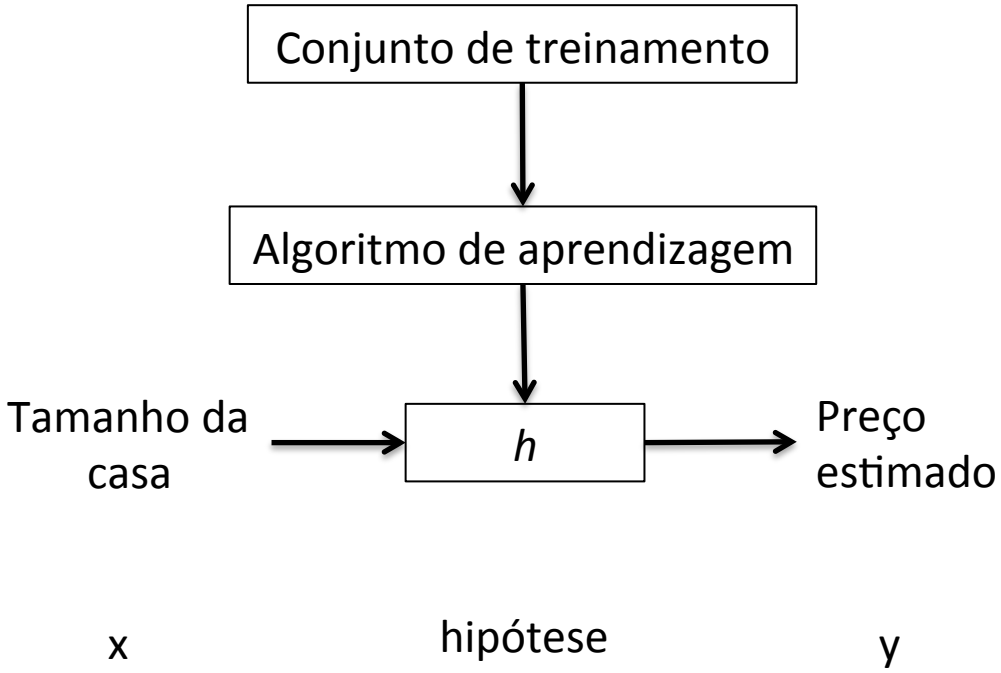
Notação:

m: número de exemplos de treinamento

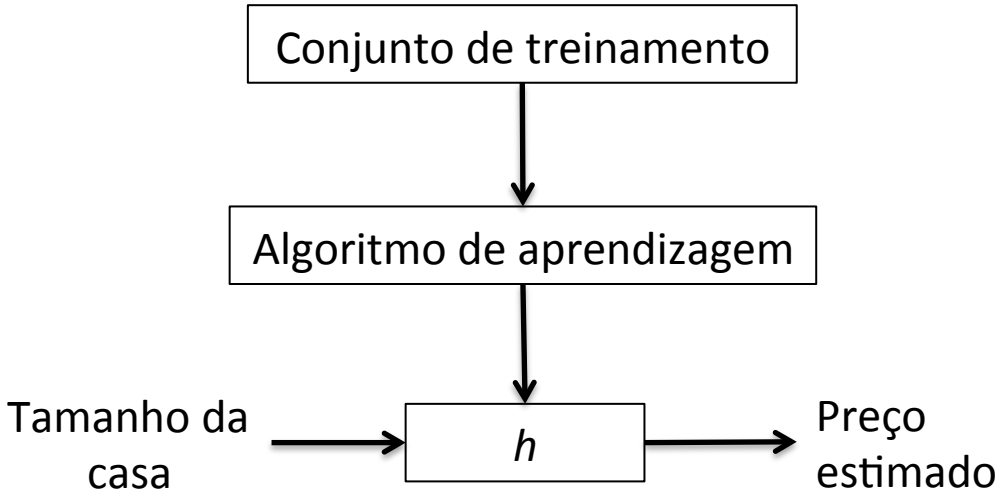
x's: variável de "entrada" / *feature*

y's: variável de "saída" / *saída desejada*

(**x**,**y**) – um exemplo


Conjunto de treinamento
Algoritmo de aprendizagem
Tamanho da casa
h
Preço estimado
x
hipótese
y

Representando a hipótese



Hipótese:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

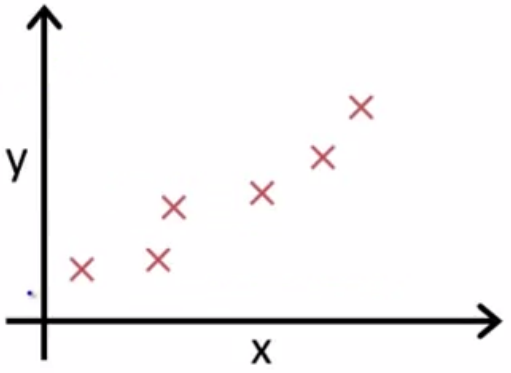
Regressão linear com uma variável

Função de custo (ou erro)

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Como escolher os parâmetros θ_i's ?

Escolher θ_i's
de maneira que a hipótese
seja tão próxima à saída desejada

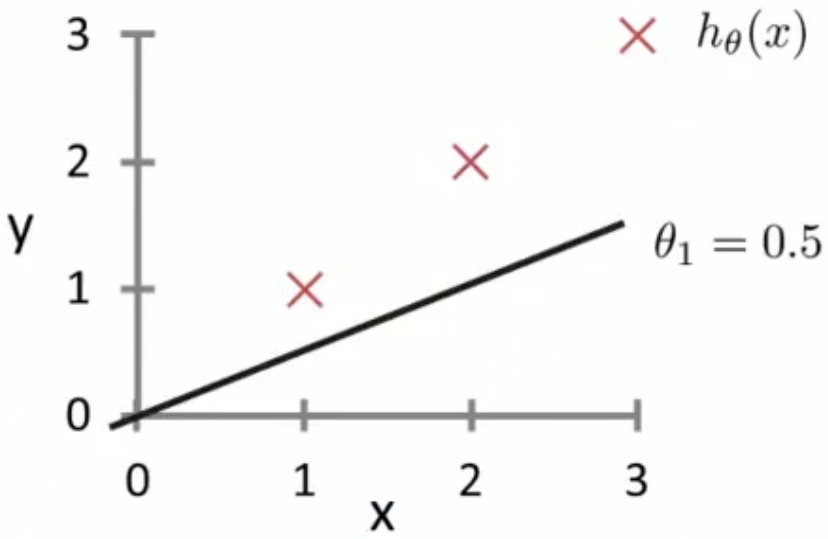


Função de custo: Exemplo 1

$h_\theta(x)$

Para θ_1 fixo, é uma função de x

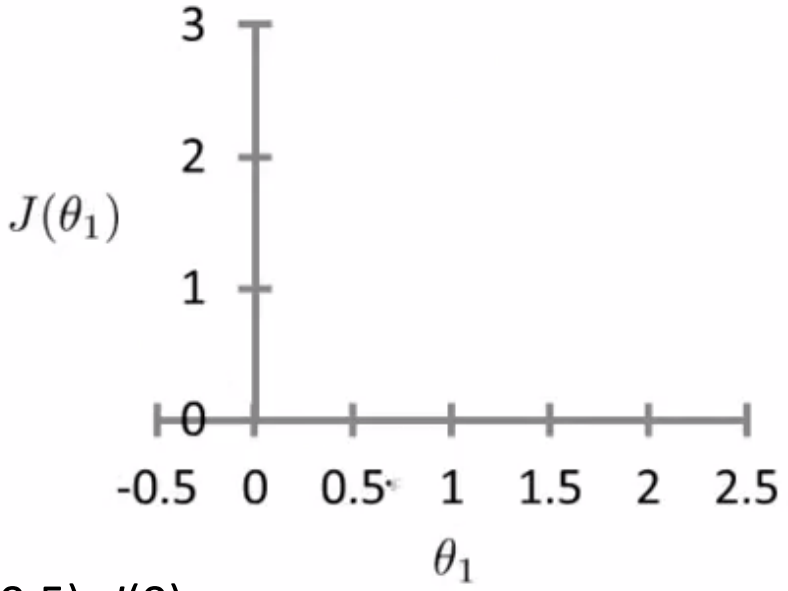
$h_\theta(x) = \theta_1 x,$



$J(\theta_1)$

É uma função de θ_1

$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

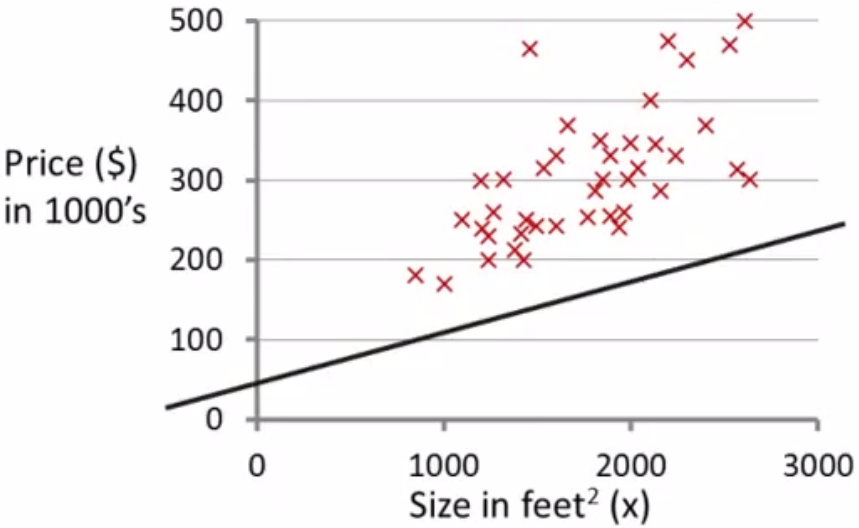


Calcular *J*(1), *J*(0.5), *J*(0)

Função de custo: Exemplo 2

$h_\theta(x)$

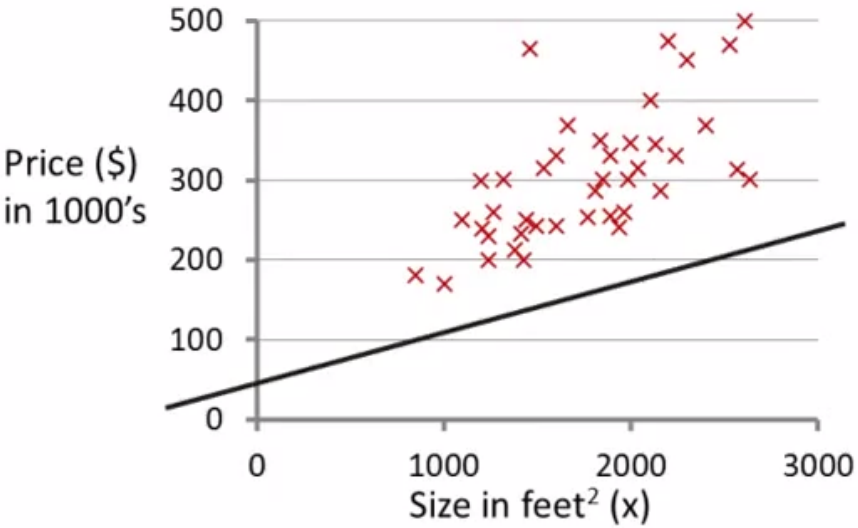
$J(\theta_0, \theta_1)$



$h_\theta(x) = 50 + 0.06x$

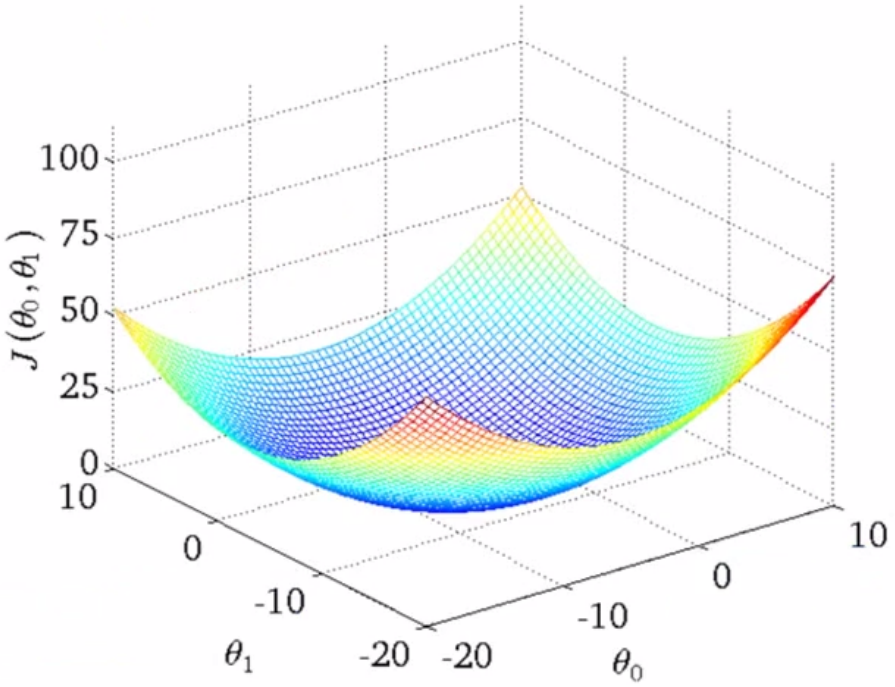
Função de custo: Exemplo 2

$h_\theta(x)$



$h_\theta(x) = 50 + 0.06x$

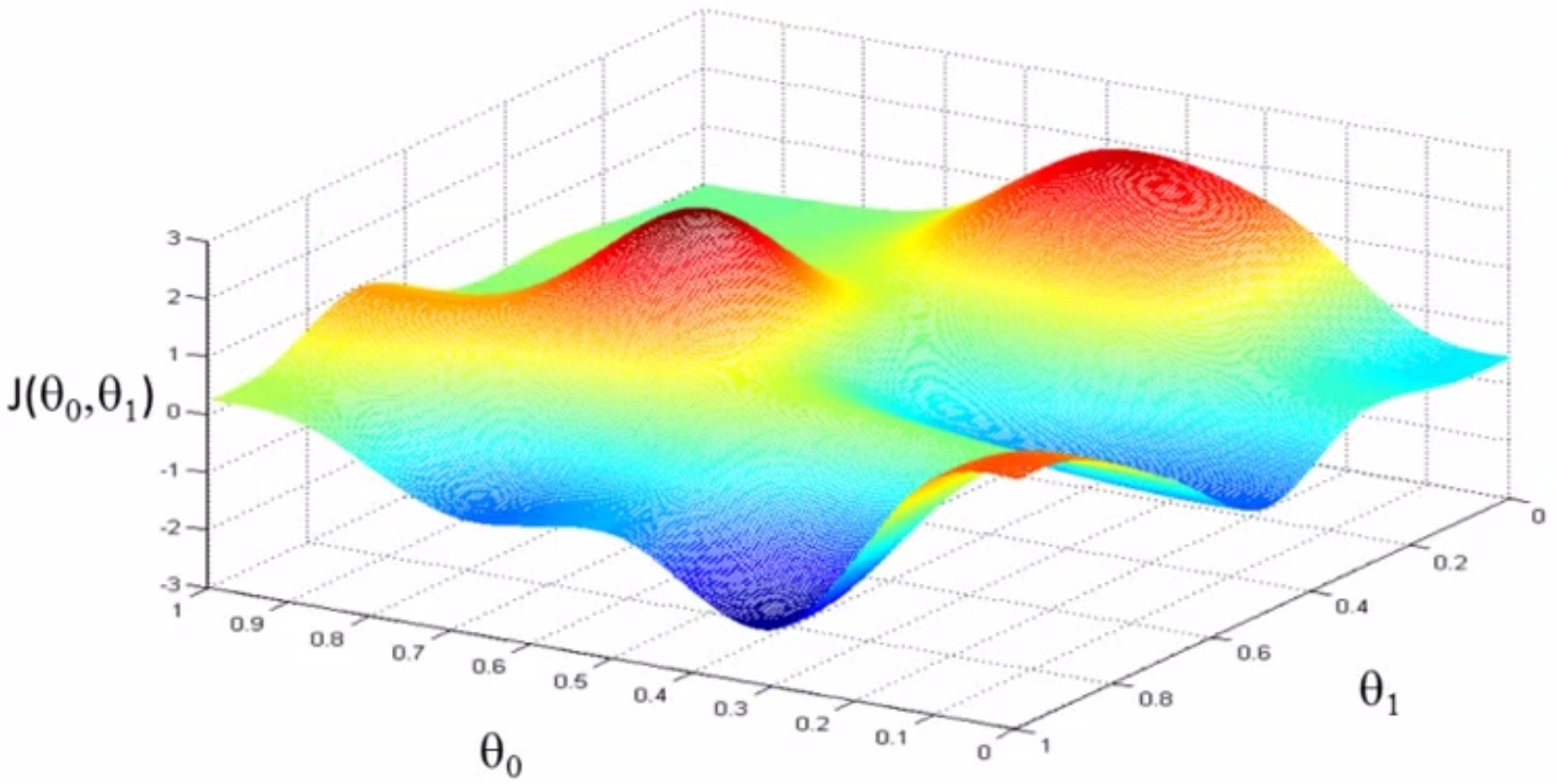
$J(\theta_0, \theta_1)$

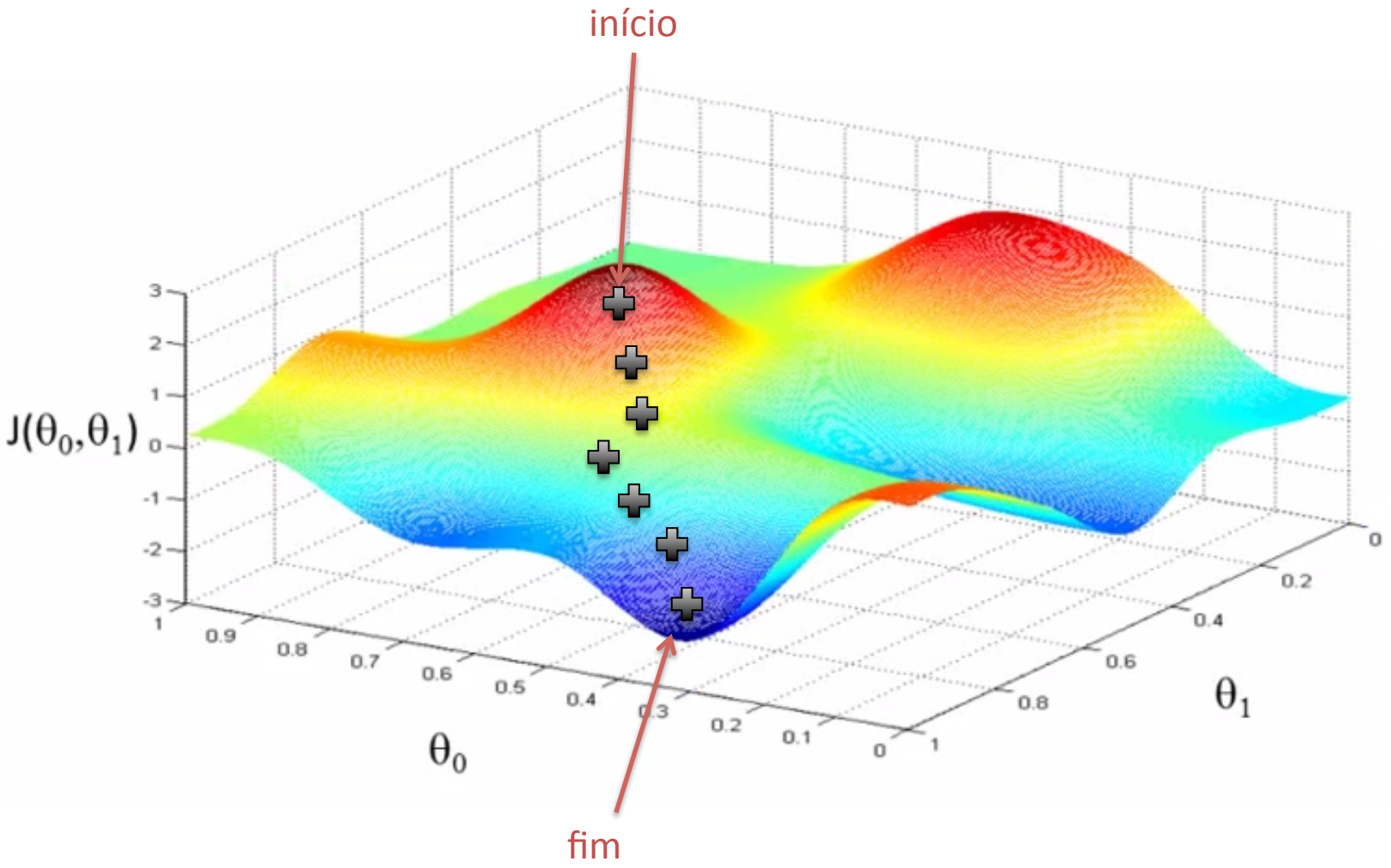


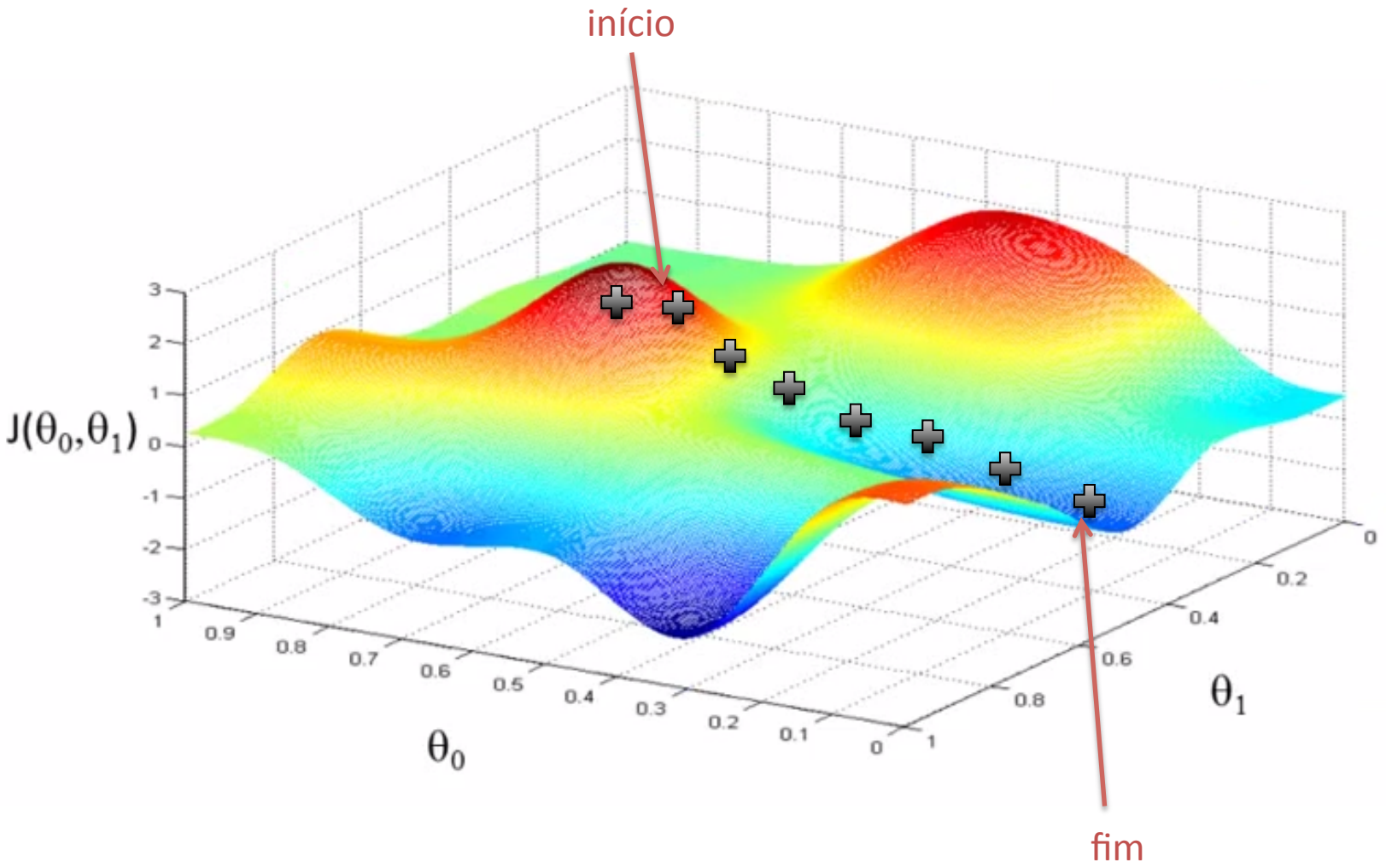
Método do Descenso do Gradiente
Gradient Descent

Temos uma função $J(\theta_0, \theta_1)$ que desejamos minimizar para θ_0, θ_1

- Começar com um valor inicial θ_0, θ_1
- Continuar alterando θ_0, θ_1 para reduzir o custo $J(\theta_0, \theta_1)$ até atingir um mínimo


$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
1
0.8
0.6
0.4
0.2
0
θ_1


início
fim
$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1


início
fim
J(θ0,θ1)
θ0
θ1
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1

Algoritmo: método do descenso

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(for } j = 0 \text{ and } j = 1\text{)}$$

}

Atualização simultânea:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\theta_1 := \text{temp1}$$

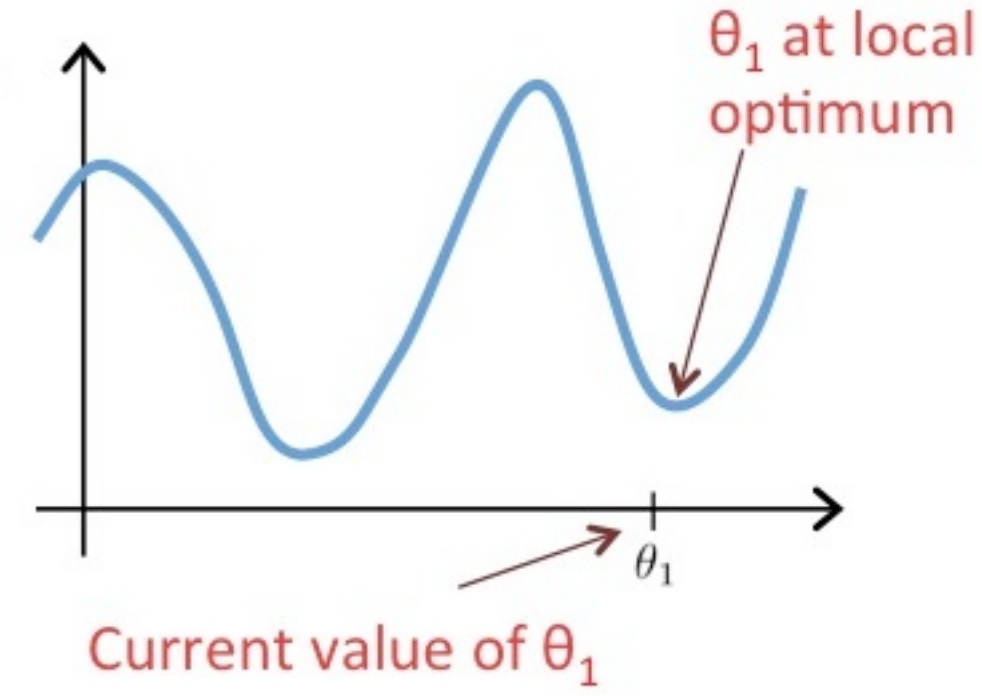
Taxa de aprendizagem

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

Se α é muito pequeno, o método do descenso pode ser muito lento.

Se α é muito grande, o método do descenso pode não convergir ou divergir .

O que um passo do método faz nesse caso?



Convergência do método

O método do descenso pode convergir para um mínimo local mesmo que a taxa de aprendizagem α esteja fixa.

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

A medida que nos aproximamos do mínimo local, o método vai "dando passos" automaticamente menores.
Não há necessidade de diminuir α ao longo do tempo.

Método do descenso para regressão linear

Método do descenso do gradiente

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(for $j = 1$ and $j = 0$)

}

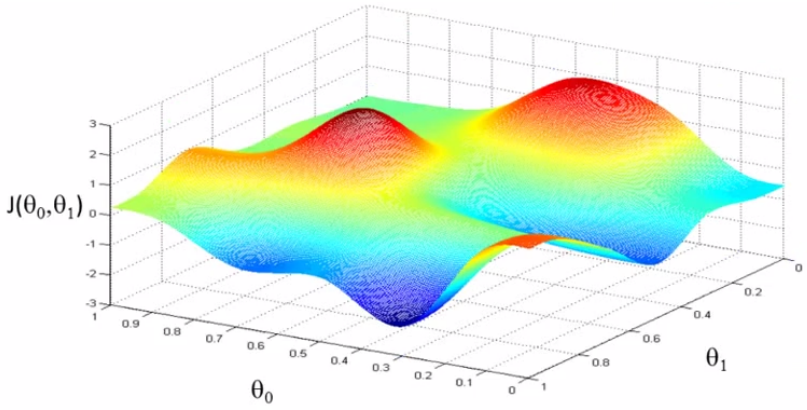
Modelo de regressão linear

$$h_\theta(x) = \theta_0 + \theta_1 x$$

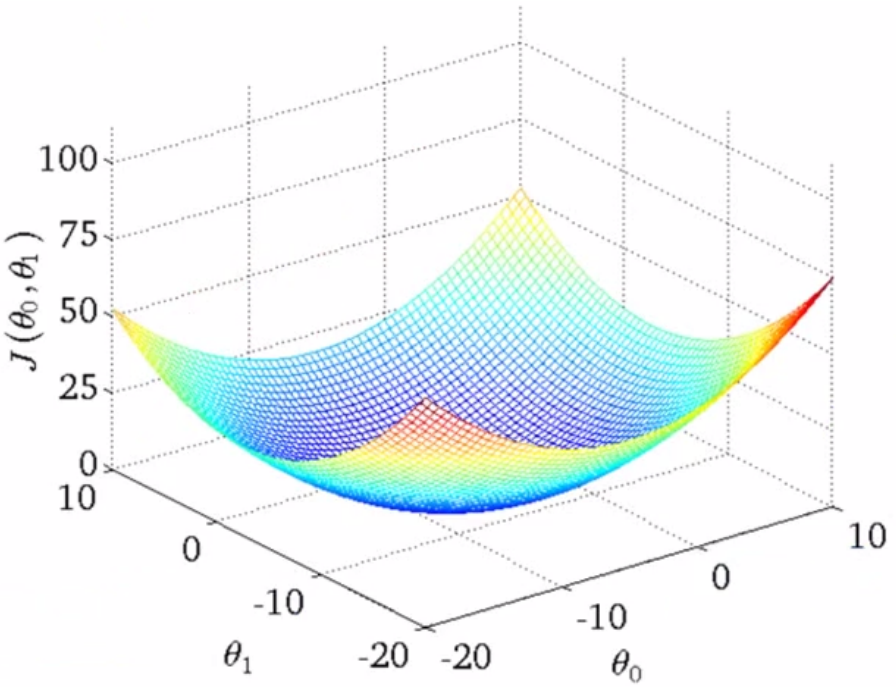
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

O método do descenso do gradiente

Função de custo arbitrária



Função de custo para regressão linear: mínimo global



“Batch” Gradient Descent

- “Batch”: Cada passo do método usa todos os exemplos de treinamento.

$$\sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)$$

Regressão linear multi-variável

Hipótese: $$h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Parâmetros: $$\theta_0, \theta_1, \ldots, \theta_n$$

Função de custo: $$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Método do descenso revisto: caso multi-variável

(n=1) 1 variável

Repeat {

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial\theta_0}J(\theta)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

(simultaneously update θ_0, θ_1)

}

Método do descenso revisto: caso multi-variável

(n=1) 1 variável

Repeat {

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial\theta_0}J(\theta)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

(simultaneously update θ_0, θ_1)

}

$(n \geq 1)$: ***n*** variáveis

Repeat {

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

(simultaneously update θ_j for $j = 0, \ldots, n$)

}

Normalização das variáveis de entrada

Ideia: Fazer as entradas ficarem numa escala similar. ($0 < x_i < 1$, $-1 < x_i < 1$, $-3 < x_i < 3$)

x_1: área (0-300m2)

x_2: número de quartos (1-5)

$x_1 = x_1 / 300$

$x_2 = x_2 / 5$

Ou, normalização de modo que a variável tenha média igual a 0

$x_i = [x_i - \text{mean}(x_i)] / \text{std}(x_i)$

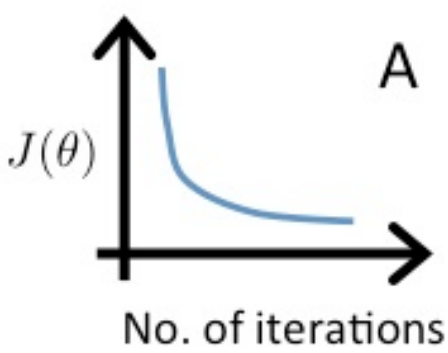
Escolhendo a taxa de aprendizagem

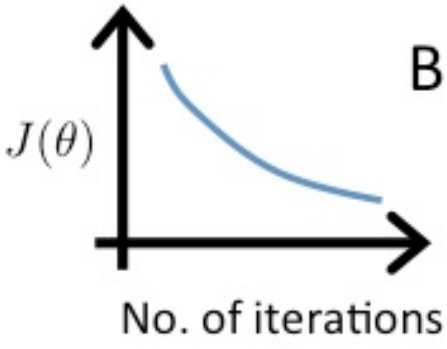
Para α suficientemente pequeno, $J(\theta)$ deve diminuir para cada iteração.

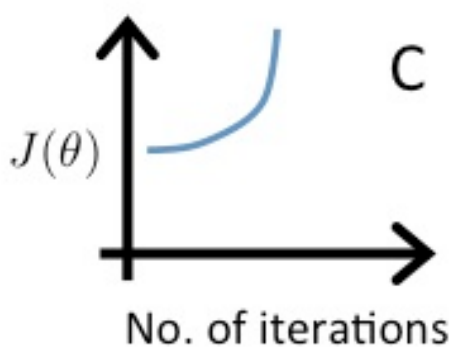
Visualizar graficamente $J(\theta)$ ao longo das iterações.

Quais gráficos correspondem às taxas de aprendizagem abaixo?

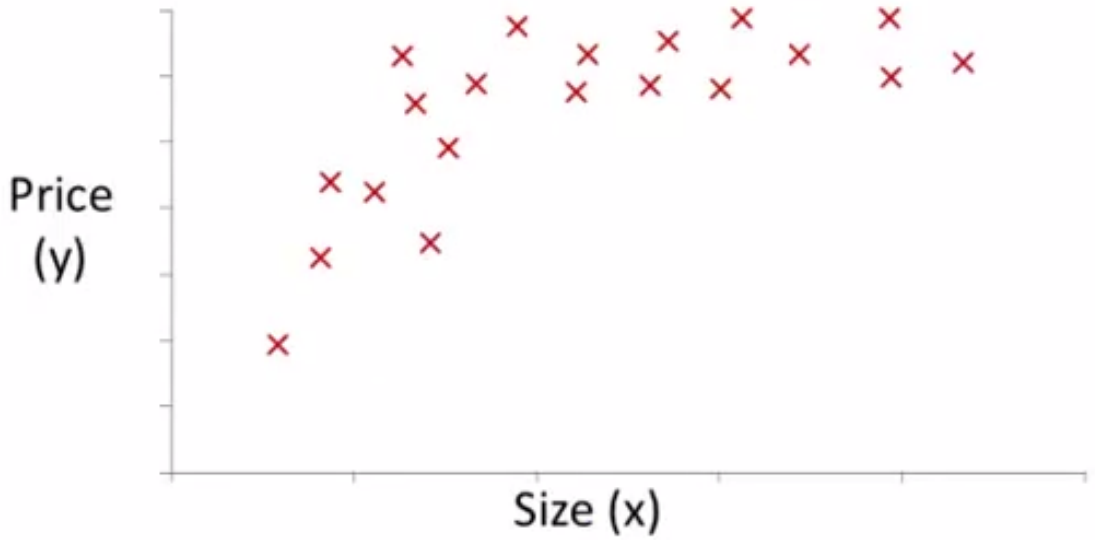
$$\alpha = 0.01,\ \alpha = 0.1, \text{ and } \alpha = 1,$$





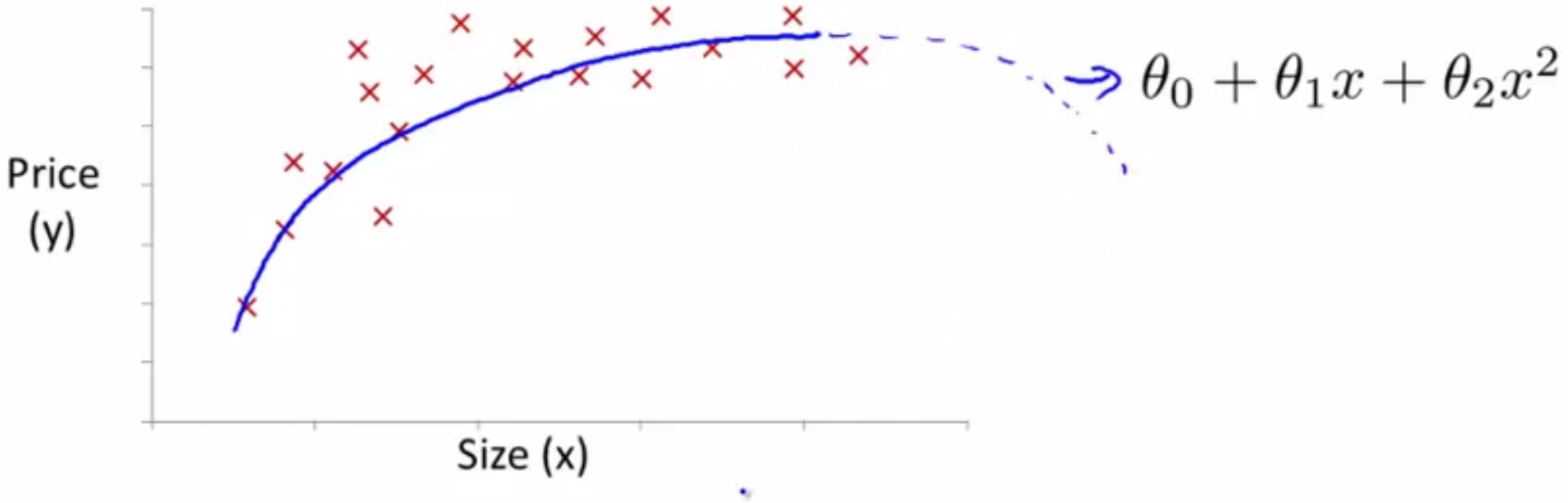


Regressão Polinomial

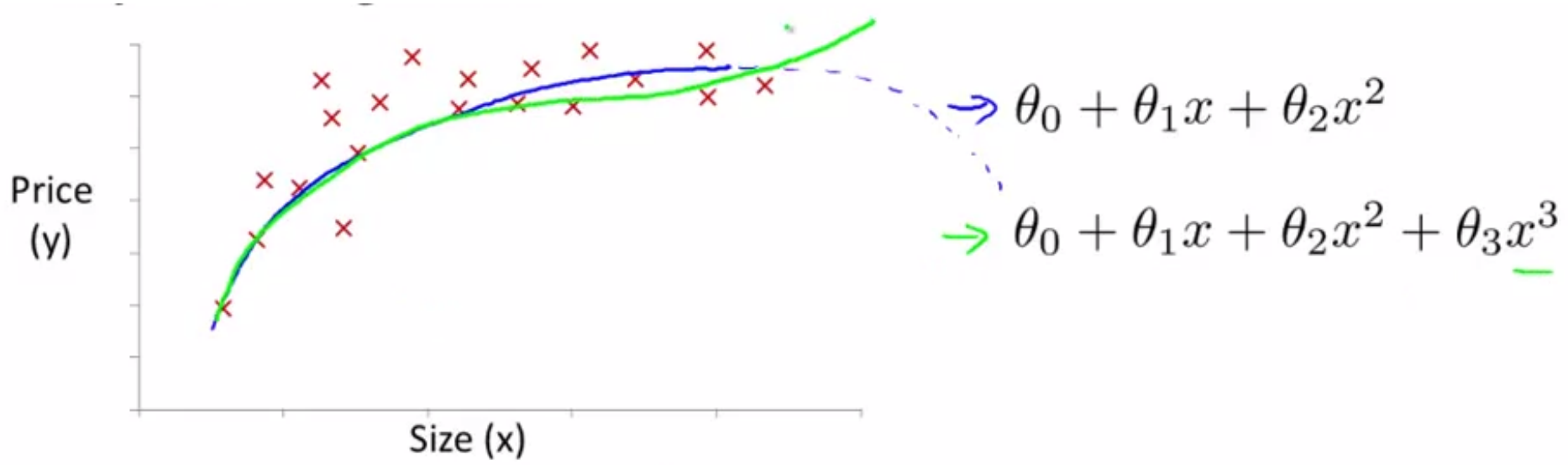


$$\theta_0 + \theta_1 x + \theta_2 x^2$$

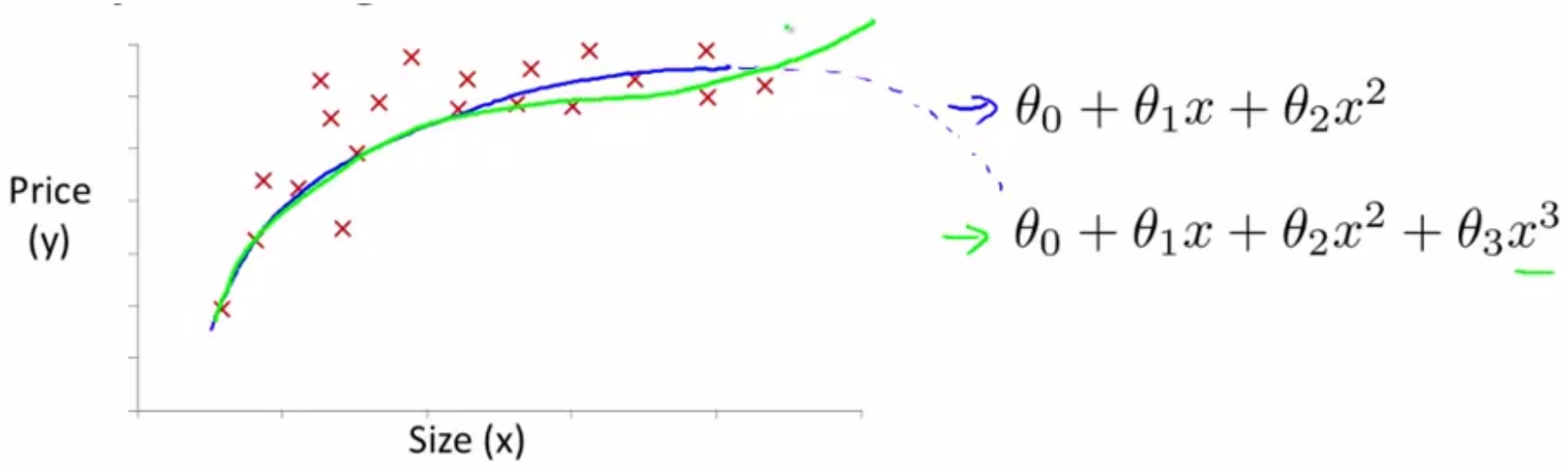
Regressão Polinomial



Regressão Polinomial



Regressão Polinomial

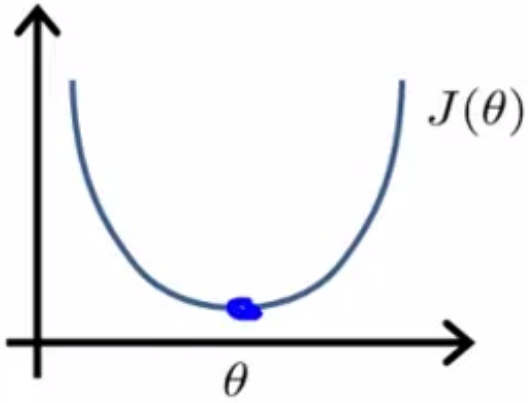


$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$
$$= \theta_0 + \theta_1 (size) + \theta_2 (size)^2 + \theta_3 (size)^3$$

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

Equação Normal

Solucionar para θ analiticamente



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

$$\frac{\partial}{\partial \theta_j} J(\theta) = \cdots = 0 \qquad \text{(for every } j\text{)}$$

Equação Normal

$m = 4.$

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
x_0	x_1	x_2	x_3	x_4	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

Equação Normal

$m = 4.$

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

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

$$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

Equação Normal

$m = 4.$

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

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

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

"One-shot learning"

Equação Normal

Octave: `pinv(X'*X)*X'*y`

(ou Matlab)

m exemplos de treinamento, *n* variáveis de entrada

Método do descenso do gradiente

- Precisa escolher a taxa de aprendizagem
- Precisa muitas iterações
- Funciona bem para *n* grande (*n* >= 10^6)

Equação Normal

- **Não** precisa escolher a taxa de aprendizagem
- **Não** precisa de iterações
- Precisa calcular $(X^T X)^{-1}$
- Lento se *n* grande.