# Regressão Linear & Otimização Numérica


Prof. Danilo Silva

EEL410250 - Aprendizado de Máquina

PPGEEL / UFSC

# Tópicos

- Regressão linear: revisão
- Introdução à otimização numérica
- Método do gradiente
- Normalização de atributos
- Extensões

# Regressão Linear: Revisão

# Regressão Linear

- Modelo de regressão linear:

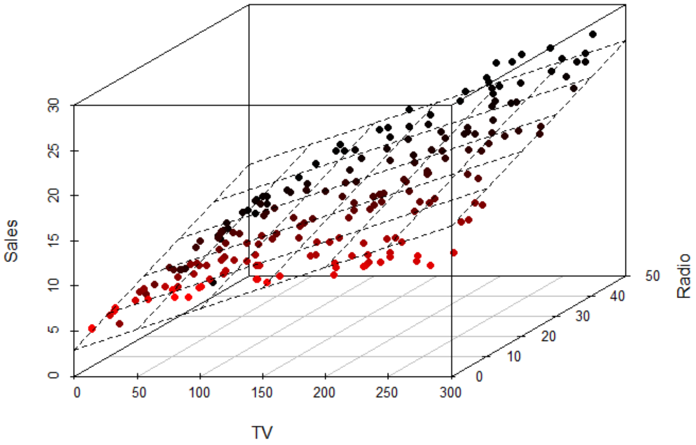
$$\hat{y} = f(\mathbf{x}) = w_0 + w_1 x_1 + \cdots + w_n x_n = \mathbf{w}^T \mathbf{x}$$

  onde
  - $y \in \mathbb{R}$ é o valor-alvo do qual $\hat{y} \in \mathbb{R}$ é uma predição
  - $\mathbf{x} = \begin{bmatrix} 1 & x_1 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^{n+1}$ é o vetor de atributos
  - $\mathbf{w} = \begin{bmatrix} w_0 & w_1 & \cdots & w_n \end{bmatrix}^T$ é o vetor de parâmetros (ou pesos)

- Conjunto de treinamento $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})\}$ organizado em uma matriz de projeto e um vetor de rótulos/alvos

$$\mathbf{X} = \begin{bmatrix} — (\mathbf{x}^{(1)})^T — \\ \vdots \\ — (\mathbf{x}^{(m)})^T — \end{bmatrix} \quad \text{e} \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

# Exemplo



Sales
0
5
10
15
20
25
30
0
50
100
150
200
250
300
TV
0
10
20
30
40
50
Radio

# Regressão Linear com Funções de Base

- De maneira geral, os atributos $x_i$ podem ser escolhidos como uma transformação não-linear de uma ou mais variáveis de entrada

$$x_i = \varphi_i(x), \qquad \text{ou} \qquad x_i = \varphi_i(u_1, \ldots, u_N)$$

  onde $\varphi_i(\cdot)$ são chamadas de funções de base

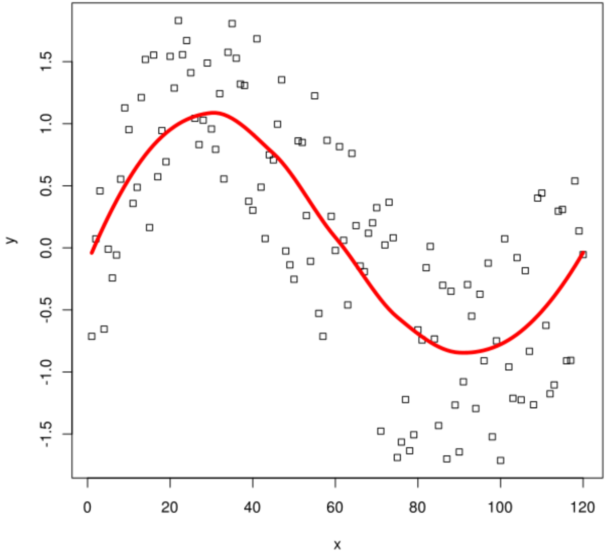
- Um exemplo é regressão polinomial de ordem $n$:

$$\hat{y} = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n$$

  onde $\varphi_i(x) = x^i$.

- Nesse caso, embora o modelo continue linear em relação aos atributos $x_i$ (e também em relação aos parâmetros $w_i$), um ajuste mais flexível pode ser feito com relação à variável original $x$

# Exemplo



# Mínimos quadrados

- Função custo:

$$J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)})^2 = \frac{1}{m}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

- Gradiente:

$$\nabla J(\mathbf{w}) = \frac{2}{m}\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y})$$

- Solução ótima (equação normal):

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

# Mínimos quadrados com regularização $\ell_2$

- Função custo:

$$J(\mathbf{w}) = \frac{1}{m}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda\frac{1}{m}\mathbf{w}^T\mathbf{L}\mathbf{w}$$

  onde $\mathbf{L} = \mathrm{diag}(0, 1, \ldots, 1)$

- Gradiente:

$$\nabla J(\mathbf{w}) = \frac{2}{m}\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\frac{2}{m}\mathbf{L}\mathbf{w}$$

- Solução ótima (equação normal):

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{L})^{-1}\mathbf{X}^T\mathbf{y}$$

# Limitações da solução analítica

- Nem todas as funções custo admitem solução analítica
  - Ex: regularização $\ell_1$, perda $\ell_1$ (MAE), outras perdas
- Calcular $\mathbf{X}^T\mathbf{X}$ pode ser computacionalmente custoso para $n$ muito grande: a ordem de complexidade (em número de operações) é $O(mn^2)$
- Solução: métodos iterativos de otimização

# Introdução à Otimização Numérica

# Introdução à otimização numérica

- Problema (otimização sem restrições):

$$\min_{\mathbf{w}} \; J(\mathbf{w})$$

A solução do problema (mínimo global) é denotada por $\mathbf{w}^*$

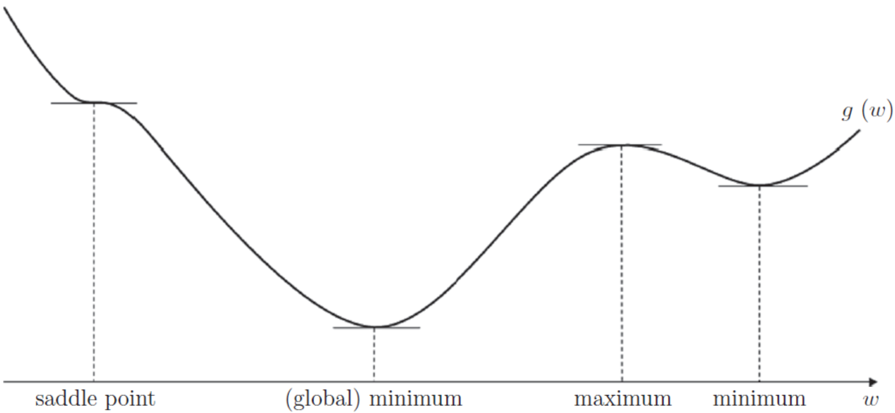
- Condição necessária de 1ª ordem para otimalidade:

$$\nabla J(\mathbf{w}) = \mathbf{0}$$

Todo ponto que satisfaz essa condição é um ponto estacionário

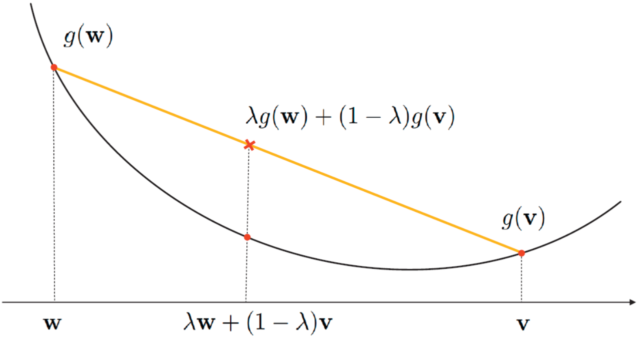
- Nem todo ponto estacionário é um mínimo local, mas a maioria dos métodos de otimização satisfaz-se em encontrar um ponto estacionário

# Exemplo



# Convexidade

- Se $J(\mathbf{w})$ é uma função convexa, então todos os pontos estacionários são mínimos globais (i.e., a condição de 1ª ordem é suficiente)




- Uma função é convexa se todo segmento de reta conectando dois pontos no gráfico da função situa-se inteiramente acima da função

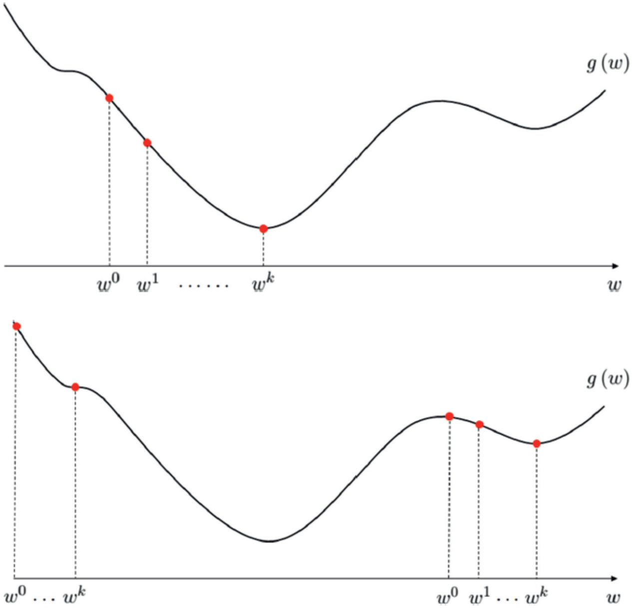
# Métodos de otimização

- Em geral, produzem uma sequência de pontos $\mathbf{w}^{[0]}, \mathbf{w}^{[1]}, \ldots, \mathbf{w}^{[t]}$ que reduzem o valor da função $J(\mathbf{w})$ a cada iteração
- Algoritmo genérico:

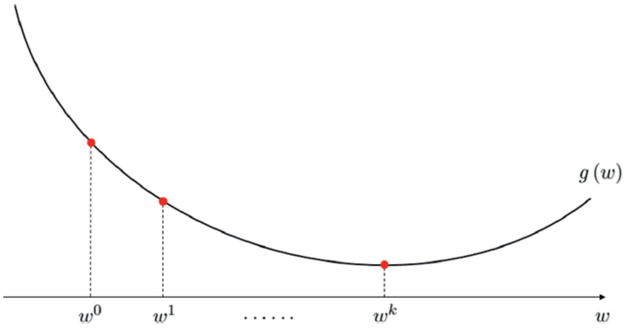
  `initialize` $\mathbf{w}^{[0]}$
  
  `for` $t = 1, \ldots,$ `max_iter:`
  
      `update` $\mathbf{w}^{[t]}$
  
      `if` $\|\nabla J(\mathbf{w}^{[t]})\| <$ `tol:  break`

- A diferença entre os métodos está na atualização de $\mathbf{w}^{[t]}$
- Se a função é não-convexa, o ponto estacionário encontrado depende do ponto inicial $\mathbf{w}^{[0]}$

# Exemplo



# Exemplo



# Métodos de otimização

- Constroem uma aproximação local para $J(\mathbf{w}^{[t]})$ em torno de $\mathbf{w}^{[t-1]}$
- Métodos de 1ª ordem:

$$J(\mathbf{w}) \approx J(\mathbf{w}^{[0]}) + \nabla J(\mathbf{w}^{[0]})^T(\mathbf{w} - \mathbf{w}^{[0]})$$

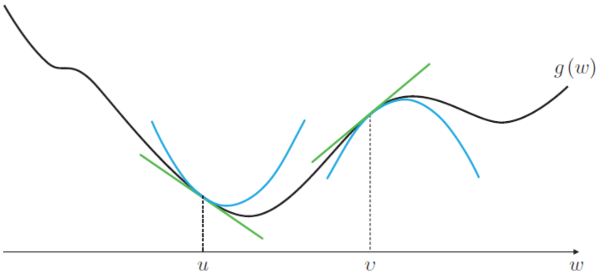
- Métodos de 2ª ordem:

$$\begin{aligned} J(\mathbf{w}) \approx\ & J(\mathbf{w}^{[0]}) + \nabla J(\mathbf{w}^{[0]})^T(\mathbf{w} - \mathbf{w}^{[0]}) \\ & + \tfrac{1}{2}(\mathbf{w} - \mathbf{w}^{[0]})^T \nabla^2 J(\mathbf{w}^{[0]})(\mathbf{w} - \mathbf{w}^{[0]}) \end{aligned}$$

onde a matriz hessiana é dada por

$$\nabla^2 J(\mathbf{w}) = \begin{bmatrix} \frac{\partial^2}{\partial w_0 \partial w_0} J(\mathbf{w}) & \frac{\partial^2}{\partial w_0 \partial w_1} J(\mathbf{w}) & \cdots & \frac{\partial^2}{\partial w_0 \partial w_n} J(\mathbf{w}) \\ \frac{\partial^2}{\partial w_1 \partial w_0} J(\mathbf{w}) & \frac{\partial^2}{\partial w_1 \partial w_1} J(\mathbf{w}) & \cdots & \frac{\partial^2}{\partial w_1 \partial w_n} J(\mathbf{w}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial w_n \partial w_0} J(\mathbf{w}) & \frac{\partial^2}{\partial w_n \partial w_1} J(\mathbf{w}) & \cdots & \frac{\partial^2}{\partial w_n \partial w_n} J(\mathbf{w}) \end{bmatrix}$$

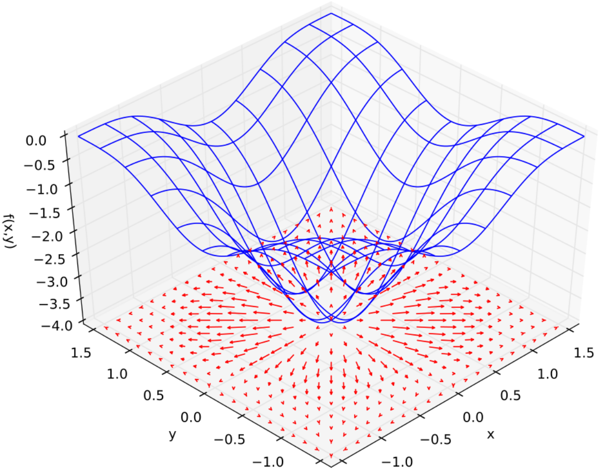
# Exemplo




- O gradiente indica a tangente, enquanto a hessiana indica a curvatura
- Ex:

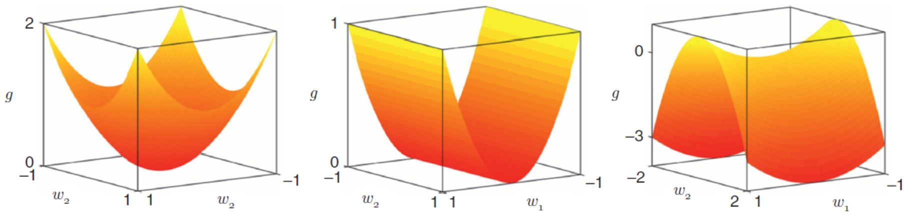
$$\begin{aligned} J(\mathbf{w}) &= \frac{1}{m}\|\mathbf{X}\mathbf{w}-\mathbf{y}\|^2 + \lambda\frac{1}{m}\|\mathbf{w}\|^2 \\ \nabla J(\mathbf{w}) &= \frac{2}{m}\mathbf{X}^T(\mathbf{X}\mathbf{w}-\mathbf{y}) + \lambda\frac{2}{m}\mathbf{w} \\ \nabla^2 J(\mathbf{w}) &= \frac{2}{m}\mathbf{X}^T\mathbf{X} + \lambda\frac{2}{m}\mathbf{I} \end{aligned}$$

# Exemplo




- O gradiente fornece direção e taxa de maior subida:
  - A direção é a direção de subida mais rápida
  - A magnitude é a taxa de subida nessa direção

# Exemplo




$$\mathbf{Q} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} 2 & 0 \\ 0 & -2 \end{bmatrix}$$

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{Q}\mathbf{w}, \quad \mathbf{Q} = \mathbf{Q}^T \quad \Longrightarrow \quad \nabla^2 J(\mathbf{w}) = \mathbf{Q}$$

- Os autovalores da hessiana estão associados ao grau de curvatura

# Método do Gradiente

# Método do gradiente (*gradient descent* / *steepest descent*)

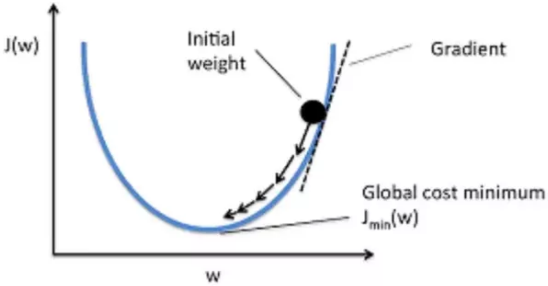
- Método de 1ª ordem
- Percorre o espaço de busca escolhendo sempre a direção de maior declive na função objetivo
- Atualização de pesos:

$$\mathbf{w}^{[t]} = \mathbf{w}^{[t-1]} - \alpha^{[t]} \nabla J(\mathbf{w}^{[t-1]})$$

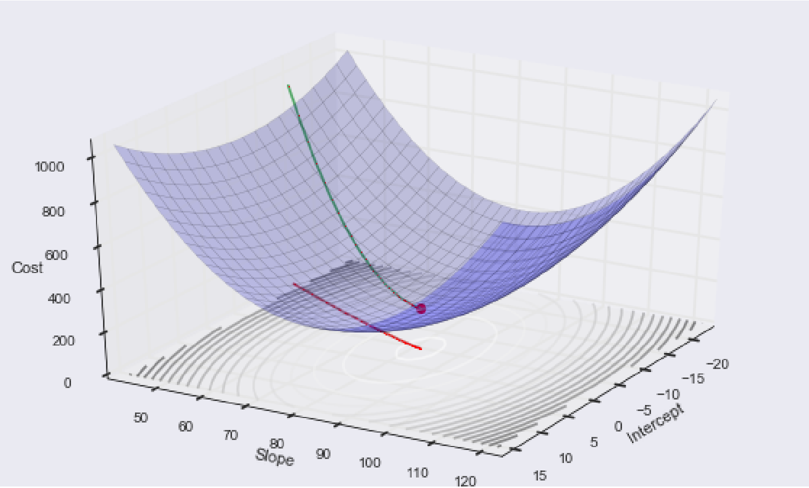
  onde $\alpha^{[t]}$ é o tamanho do passo (*step length*), também chamado de taxa de aprendizado (*learning rate*)

- A taxa de aprendizado pode ser escolhida fixa ($\alpha^{[t]} = \alpha$) ou adaptativamente

# Exemplo

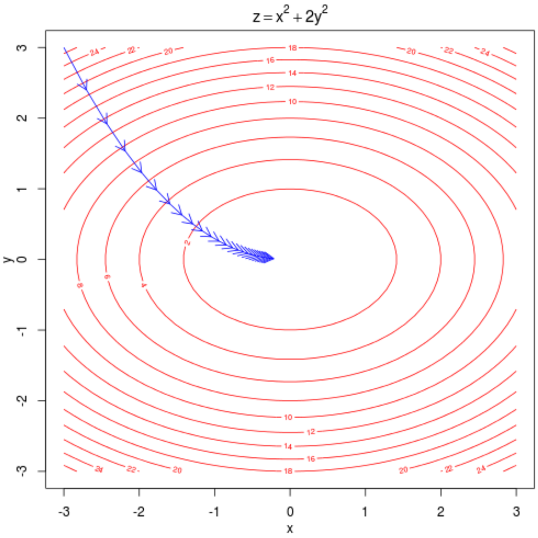


# Exemplo



Cost
1000
800
600
400
200
0
50
60
70
80
90
100
110
120
Slope
15
10
5
0
−5
−10
−15
−20
Intercept

# Exemplo



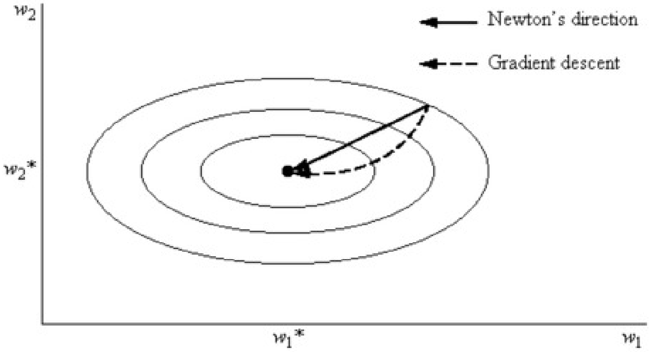
# Método de Newton

- Método de 2ª ordem
- Encontra o ponto de mínimo da aproximação quadrática
- Atualização de pesos:

$$\mathbf{w}^{[t]} = \mathbf{w}^{[t-1]} - \left[\nabla^2 J(\mathbf{w}^{[t-1]})\right]^{-1} \nabla J(\mathbf{w}^{[t-1]})$$

- Converge mais rapidamente que o método do gradiente, mas tem a desvantagem de exigir o cálculo da hessiana
- Ex: se $J(\mathbf{w})$ é uma função quadrática, o método de Newton converge em um único passo—a solução é exatamente a equação normal

# Exemplo




- Newton's direction : pointing to a local minimum
- Gradient direction : pointing to maximum direction of change

# Método do Gradiente para Regressão Linear

- Complexidade reduzida de $O(mn^2)$ para $O(mn \cdot \texttt{max\_iter})$

- Sem regularização:

$$\mathbf{w}^{[t]} = \mathbf{w}^{[t-1]} - \alpha \frac{2}{m} \mathbf{X}^T (\mathbf{X}\mathbf{w}^{[t-1]} - \mathbf{y})$$

- Com regularização $\ell_2$ (sobre todo o vetor $\mathbf{w}$):

$$\mathbf{w}^{[t]} = \left(1 - \lambda\alpha \frac{2}{m}\right) \mathbf{w}^{[t-1]} - \alpha \frac{2}{m} \mathbf{X}^T (\mathbf{X}\mathbf{w}^{[t-1]} - \mathbf{y})$$

  - Por isso a regularização $\ell_2$ também é chamada de *weight decay*

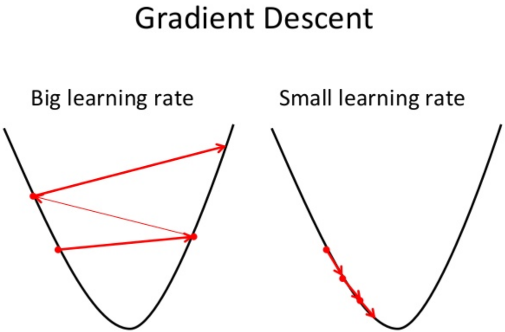
- Com regularização $\ell_2$ (sem regularizar $w_0$):

$$\mathbf{w}^{[t]} = \left(\mathbf{I} - \lambda\alpha \frac{2}{m}\mathbf{L}\right) \mathbf{w}^{[t-1]} - \alpha \frac{2}{m} \mathbf{X}^T (\mathbf{X}\mathbf{w}^{[t-1]} - \mathbf{y})$$

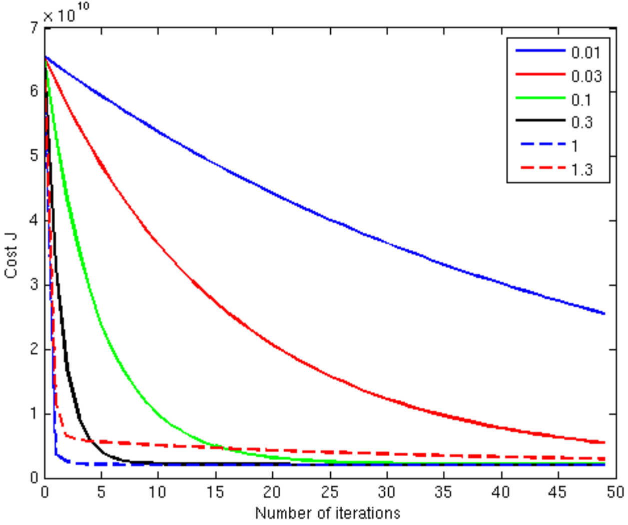
# Método do Gradiente: Escolha da Taxa de Aprendizado

- Uma das desvantagens do método do gradiente é ter de escolher a taxa de aprendizado
- Se $\alpha$ é muito pequeno, a convergência pode ser lenta
- Se $\alpha$ é muito grande, pode ocorrer overshoot. Nesse caso, o método pode não convergir ou até mesmo divergir
- Sempre é possível encontrar um valor de $\alpha$ (suficientemente pequeno) que garante convergência. No entanto, para acelerar a convergência na prática, também é possível usar um valor de $\alpha$ selecionado adaptativamente de acordo com a iteração, $\alpha^{[t]}$.

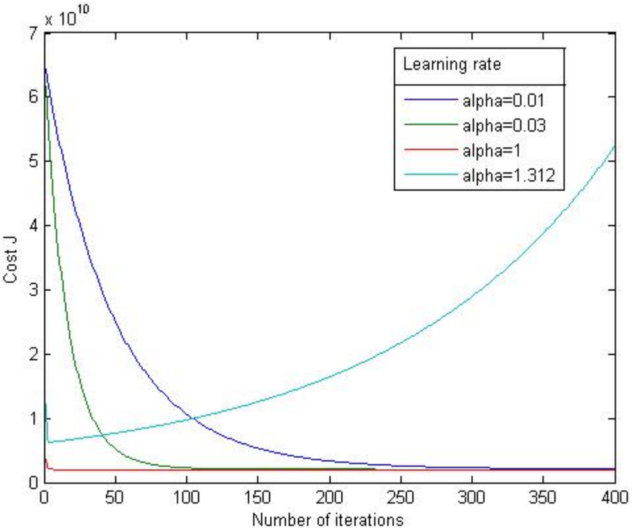
# Exemplo



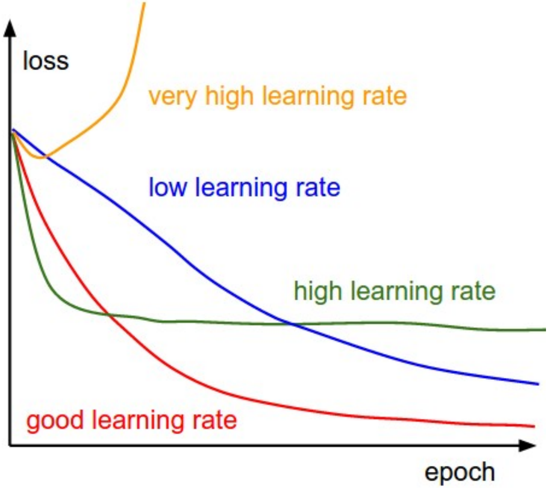
# Exemplo: Custo em função da iteração



# Exemplo: Custo em função da iteração



# Exemplo: Custo em função da iteração



# Escolha da taxa de aprendizado

- Importante: para analisar a convergência e escolher a taxa de aprendizado, deve ser usada a função objetivo da otimização, $J(\mathbf{w})$, mesmo que regularizada—ao invés de usar o erro de treinamento, $J_{\text{train}}(\mathbf{w})$
- Afinal, dependendo de $\lambda$, da função objetivo, e do ponto inicial, o erro $J_{\text{train}}(\mathbf{w})$ pode até aumentar em algumas iterações, mas $J(\mathbf{w})$ deve sempre diminuir se a taxa de aprendizado for escolhida adequadamente

# Normalização de Atributos

# Convergência do Método do Gradiente

- Se a matriz hessiana $\nabla^2 J(\mathbf{w}) = \frac{2}{m}\mathbf{X}^T\mathbf{X}$ for mal condicionada (isto é, com valor elevado da razão entre os autovalores máximo e mínimo), então o método do gradiente apresentará dificuldades de convergir (comportamento em “zig-zag”)

- Exemplo:

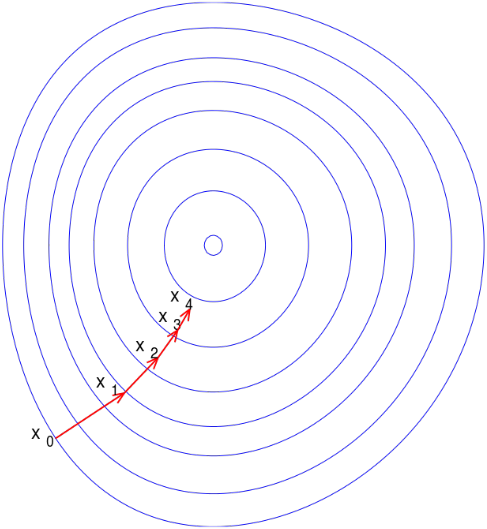
$$J(\mathbf{w}) = \frac{1}{2}\lambda_0 w_0^2 + \frac{1}{2}\lambda_1 w_1^2$$

$$\nabla J(\mathbf{w}) = \begin{bmatrix} \lambda_0 w_0 \\ \lambda_1 w_1 \end{bmatrix}, \quad \nabla^2 J(\mathbf{w}) = \begin{bmatrix} \lambda_0 & \\ & \lambda_1 \end{bmatrix}$$

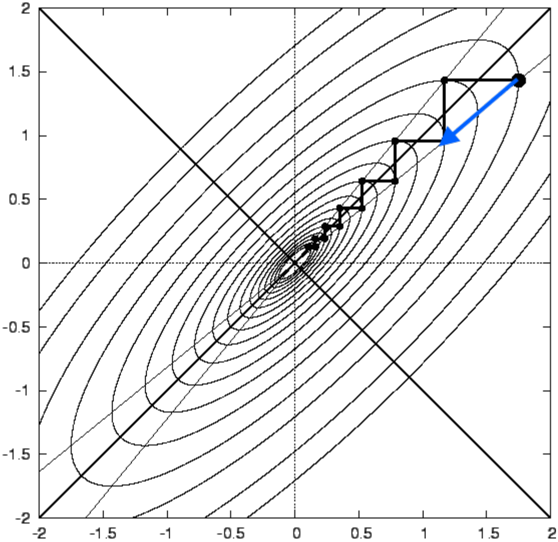
$$\mathbf{w}^{[t]} = \mathbf{w}^{[t-1]} - \alpha \begin{bmatrix} \lambda_0 w_0^{[t-1]} \\ \lambda_1 w_1^{[t-1]} \end{bmatrix}$$

  - Se $|\lambda_0/\lambda_1| \gg 1$ ou $|\lambda_1/\lambda_0| \gg 1$, então não existe uma taxa de aprendizado igualmente boa para os dois parâmetros

# Exemplo (bem condicionado)



# Exemplo (mal condicionado)



# Normalização de Atributos

- O melhor condicionamento ocorre quando $\mathbf{X}^T\mathbf{X} \propto \mathbf{I}$
- Uma solução é normalizar todos os atributos (exceto $x_0 = 1$) para que tenham média nula e variância unitária:

$$x'_j = \frac{x_j - \bar{x}_j}{\sigma_{x_j}}$$

onde $\bar{x}_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$ e $\sigma_{x_j}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)} - \bar{x}_j)^2$

- Exemplo:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} \\ 1 & \vdots \\ 1 & x_1^{(m)} \end{bmatrix} \implies \frac{1}{m}\mathbf{X}^T\mathbf{X} = \begin{bmatrix} 1 & \bar{x}_1 \\ \bar{x}_1 & \sigma_{x_1}^2 + \bar{x}_1^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

se $\bar{x}_1 = 0$ e $\sigma_{x_1} = 1$

# Normalização de Atributos

- A normalização de atributos resulta no modelo linear:

$$\hat{y} = f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}' = w_0 + w_1\left(\frac{x_1 - \bar{x}_1}{\sigma_{x_1}}\right) + \cdots + w_n\left(\frac{x_n - \bar{x}_n}{\sigma_{x_n}}\right)$$

- Os parâmetros $\bar{x}_j$ e $\sigma_{x_j}$ devem ser estimados exclusivamente a partir do conjunto de treinamento e guardados para serem usados na predição

- Alternativamente, o modelo pode ser reexpresso como:

$$\hat{y} = f(\mathbf{x}) = \mathbf{w}'^T\mathbf{x}$$

  onde $w'_j = w_j/\sigma_{x_j}$ e $w'_0 = w_0 - \sum_j w_j\bar{x}_j/\sigma_{x_j}$

- Obs: normalização é essencial quando os atributos possuem faixas de valores bastante diferentes (ex: regressão polinomial)
  - Também auxilia na interpretação do modelo (importância de cada atributo)

# Extensões

# Regressão Não-Linear

- Modelo:
$$\hat{y} = f(\mathbf{x}) = g(\mathbf{w}^T\mathbf{x})$$
onde $g(z)$ é uma função não-linear

- Função custo (perda quadrática):
$$J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}(g(\mathbf{w}^T\mathbf{x}^{(i)}) - y^{(i)})^2$$

- Gradiente:
$$\nabla J(\mathbf{w}) = \frac{2}{m}\sum_{i=1}^{m}(g(\mathbf{w}^T\mathbf{x}^{(i)}) - y^{(i)})g'(\mathbf{w}^T\mathbf{x}^{(i)})\mathbf{x}^{(i)}$$
onde $g'(z) = \frac{d}{dz}g(z)$