

Introduction au Machine Learning

Modèles linéaires pour la régression

Emmanuel Dellandréa

emmanuel.dellandrea@ec-lyon.fr

Version du 07/01/2025

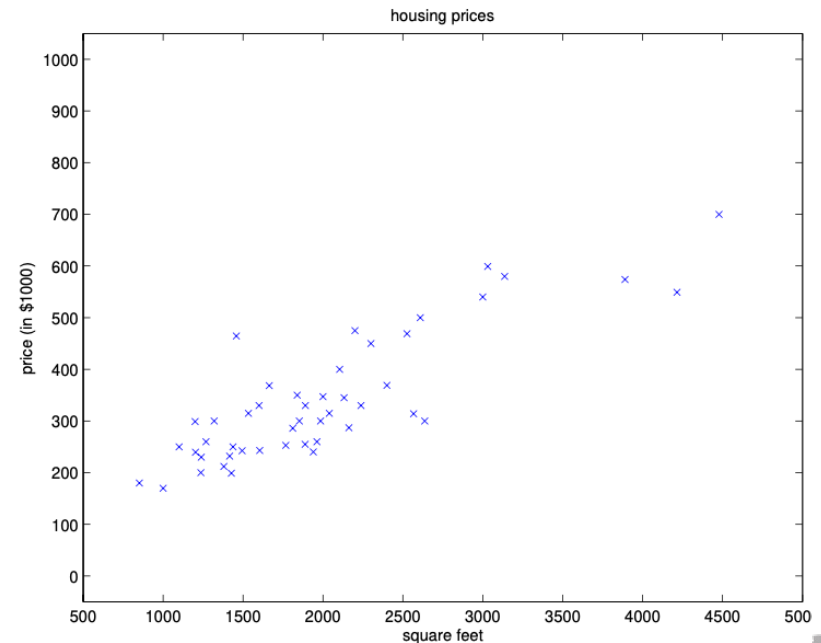


Régression linéaire

- Problématique : pour un exemple donné, prédire la valeur d'une variable cible en utilisant les valeurs connues pour la ou les variables prédictives

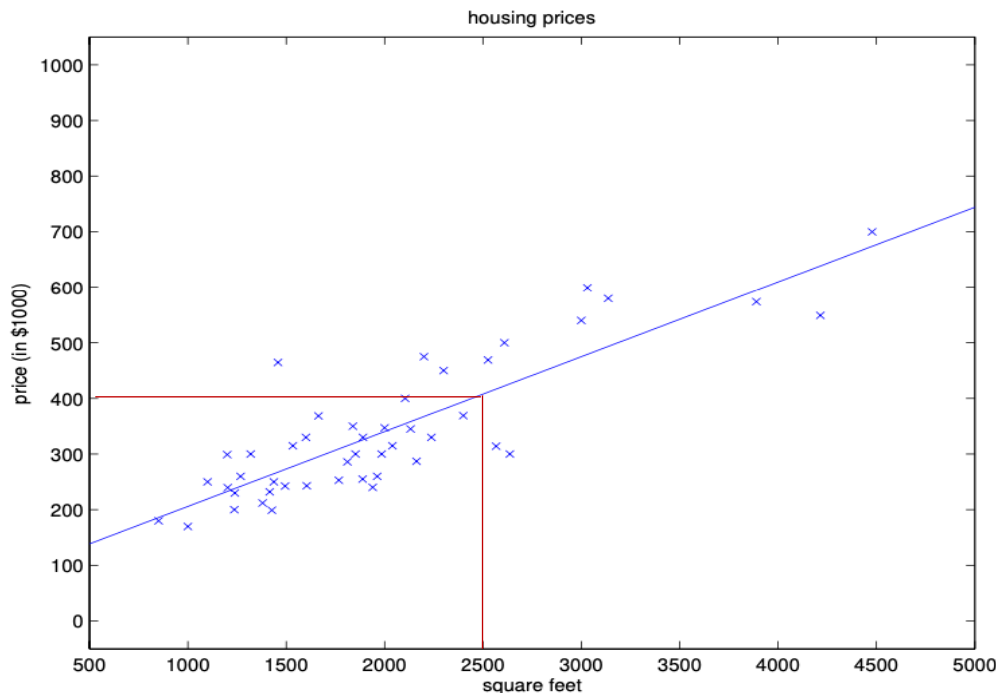
Living area (feet ²)	Price (1000\$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

$$f(2500) = ??$$



Régression linéaire

- Problématique : pour un exemple donné, prédire la valeur d'une variable cible en utilisant les valeurs connues pour la ou les variables prédictives



$$f(2500) = 400$$

Régression linéaire

- Fonction de prédiction linéaire f

Living area (feet ²)	#bedrooms	Price (1000\$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

Régression linéaire

- Fonction de prédiction linéaire f

Living area (feet ²)	#bedrooms	Price (1000\$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

Variables prédictives

Variable cible

$$f_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Expression de la fonction de prédiction

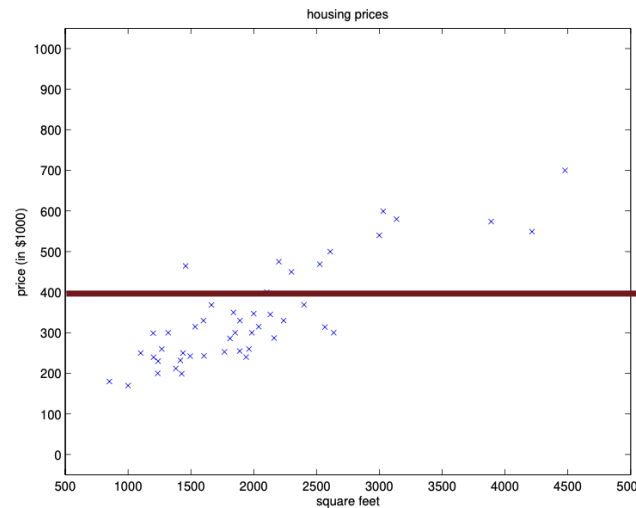
- Simplification de la notation :

$$f(x) = \sum_{i=0}^d \theta_i x_i = \theta^T x$$

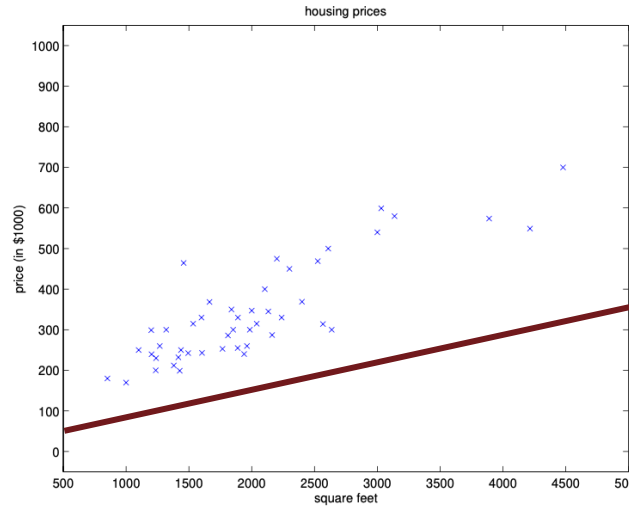
Où θ et x sont des vecteurs, d le nombre de variables prédictives (sans compter x_0) et $x_0 = 1$

Régression linéaire

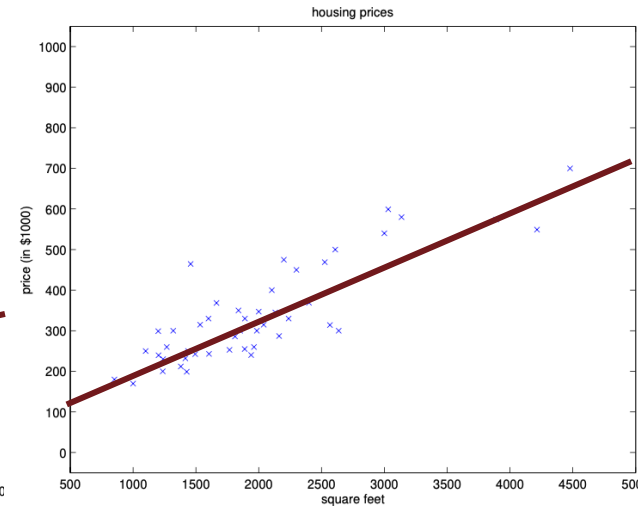
- Comment déterminer la valeur des paramètres θ ?



$$\theta_0 = 400, \theta_1 = 0$$



$$\theta_0 = 0, \theta_1 = 0.05$$



$$\theta_0 = 70, \theta_1 = 0.14$$

Régression linéaire

- Utilisation d'un ensemble d'apprentissage $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots (x^{(N)}, y^{(N)})\}$ et choisir θ tel que $f(x)$ soit aussi proche que possible de y
- Définition d'une fonction de coût (ou de perte) :

$$J(\theta) = \frac{1}{2N} \sum_{i=1}^N (f_{\theta}(x^{(i)}) - y^{(i)})^2$$

Objectif : trouver les valeurs de θ qui minimisent $J(\theta)$

Descente du gradient

- Principe

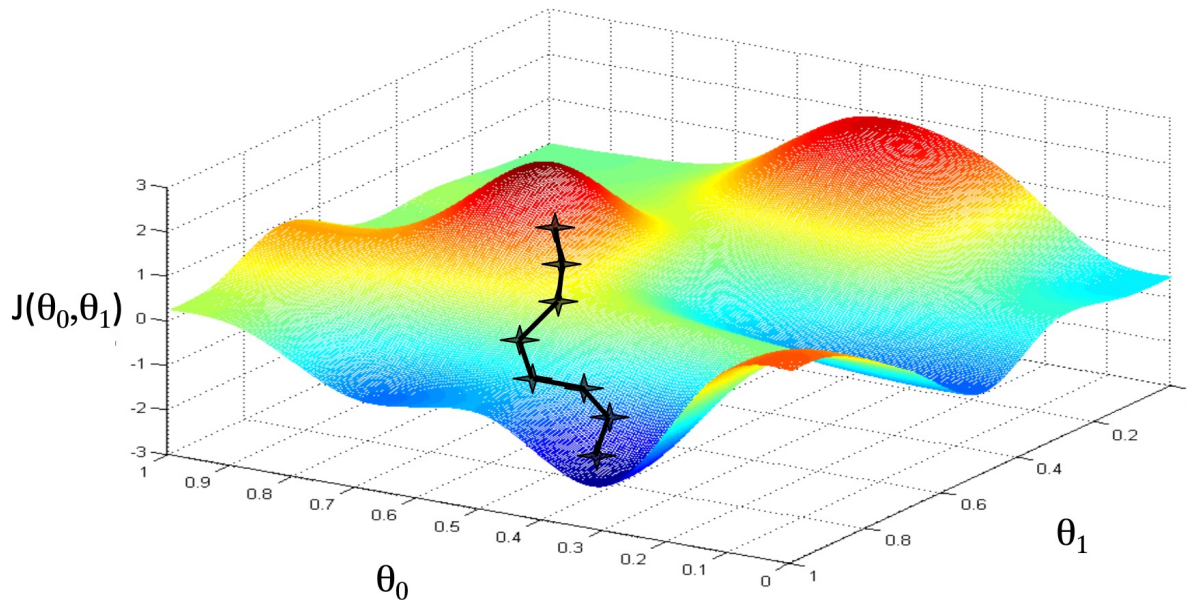
- Soit une fonction de perte $J(\theta)$
- Trouver θ qui minimise $J(\theta)$

- Algorithme :

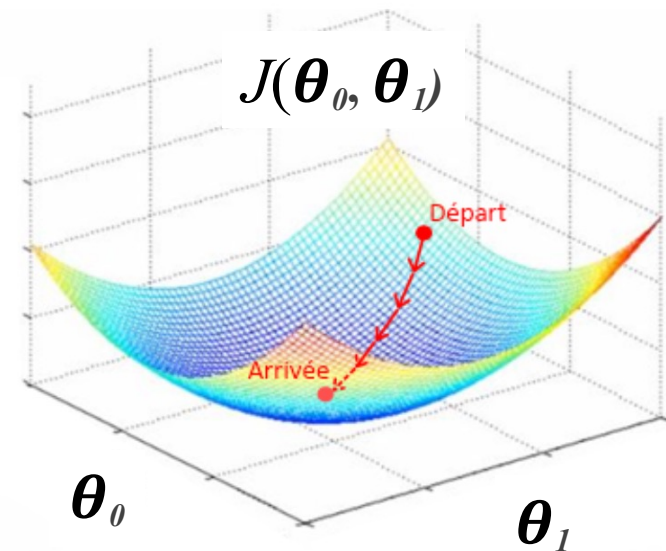
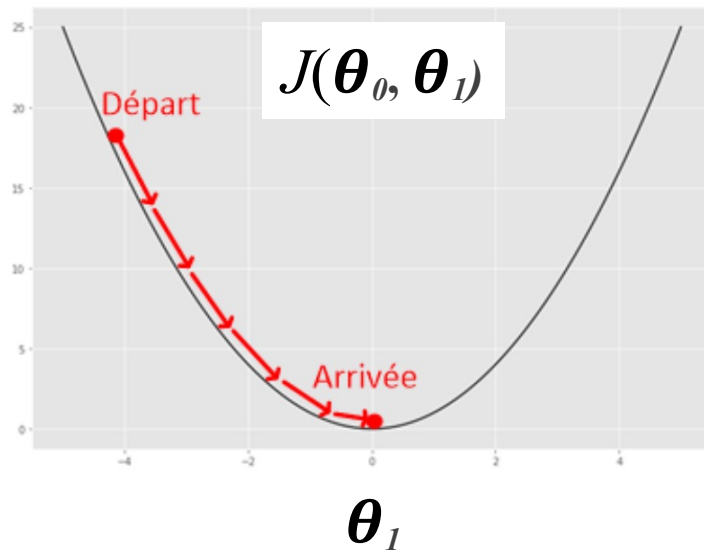
- Débuter avec une initialisation arbitraire de θ
- Modifier peu à peu θ pour diminuer la valeur $J(\theta)$

Descente du gradient

- Comment modifier θ pour diminuer la valeur $J(\theta)$?
➔ se déplacer dans la direction opposée au gradient (pente)



Descente du gradient



Descente du gradient : algorithme

- Répéter itérativement jusqu'à convergence (simultanément pour chaque j) :

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- Où α est le taux d'apprentissage (learning rate)

Calcul du gradient

- Pour un exemple (x, y)

$$\begin{aligned}\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (f_\theta(x) - y)^2 \\ &= 2 \cdot \frac{1}{2} (f_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (f_\theta(x) - y) \\ &= (f_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (\sum_{i=0}^d \theta_i x_i - y) \\ &= (f_\theta(x) - y) x_j\end{aligned}$$

Descente du gradient

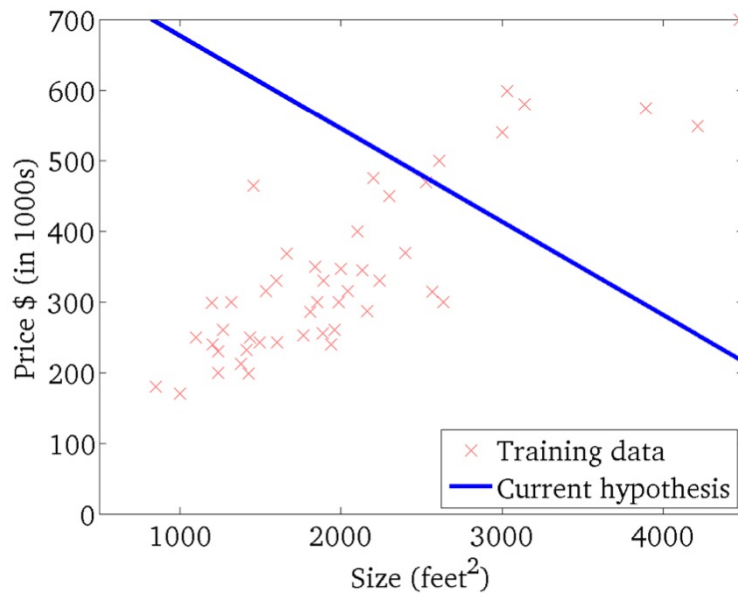
- Répéter itérativement jusqu'à convergence (simultanément pour chaque j) :

$$\theta_j = \theta_j - \alpha \frac{1}{N} \sum_{i=1}^N (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Descente du gradient : exemple

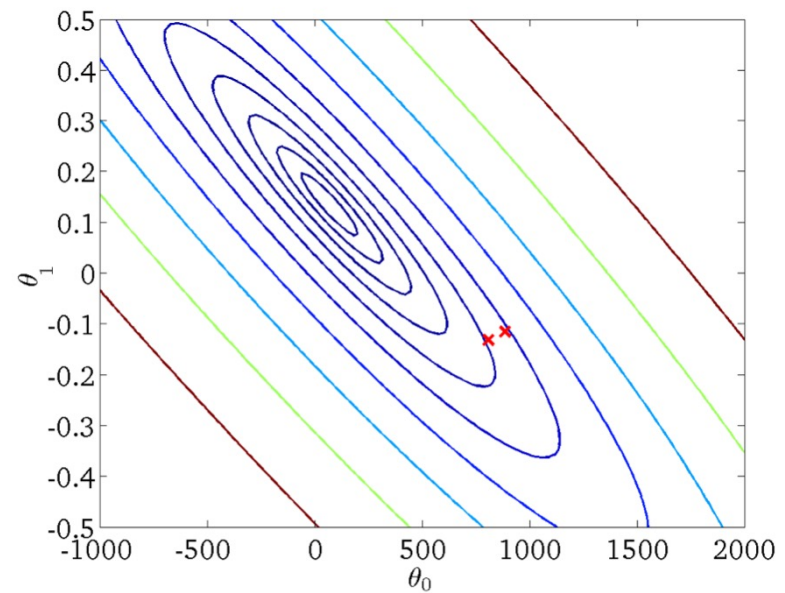
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

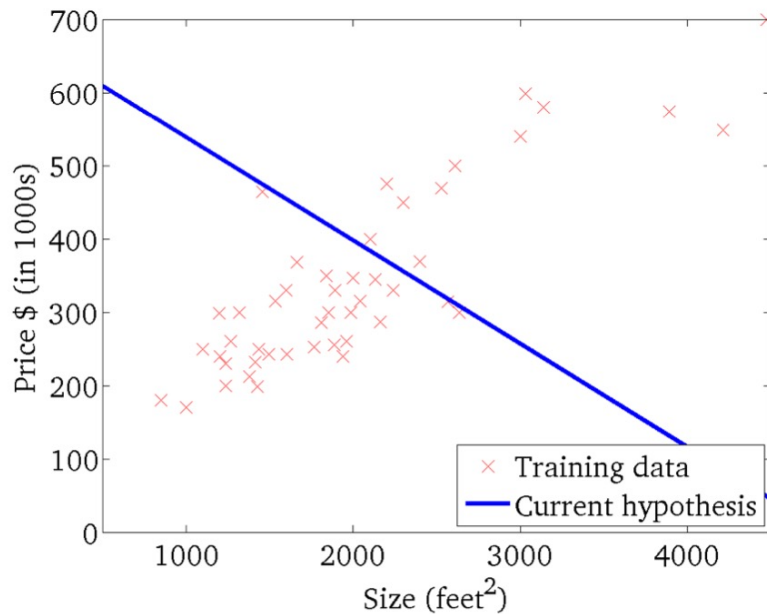
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

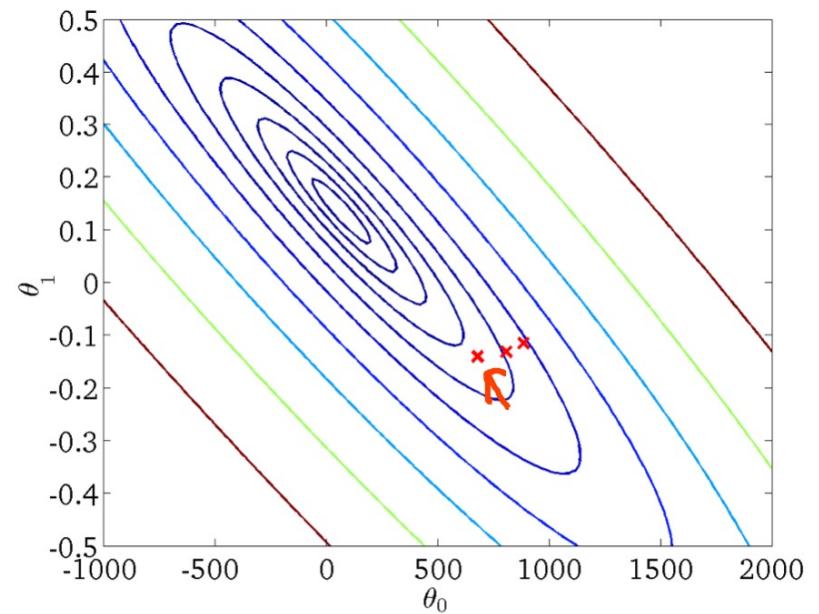
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

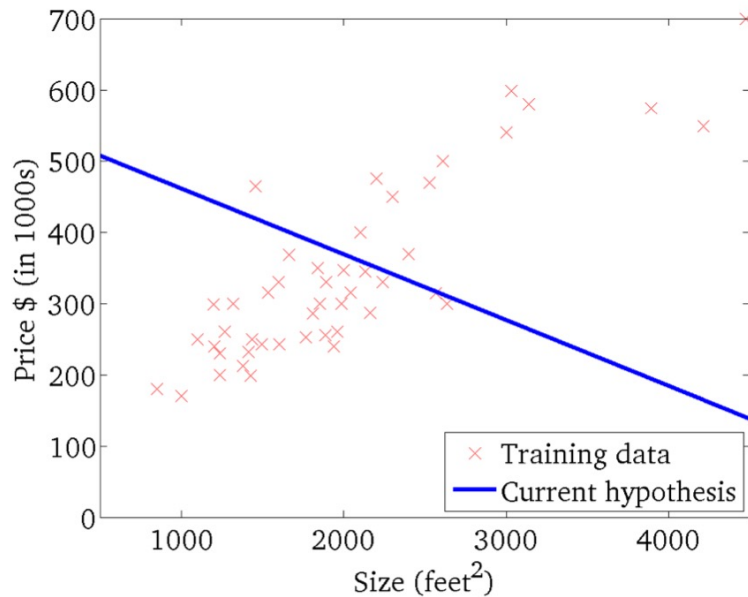
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

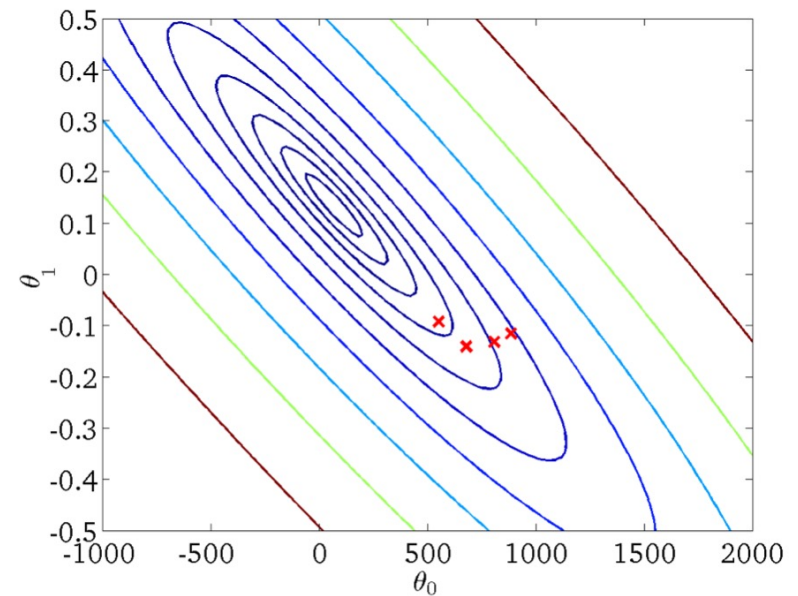
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

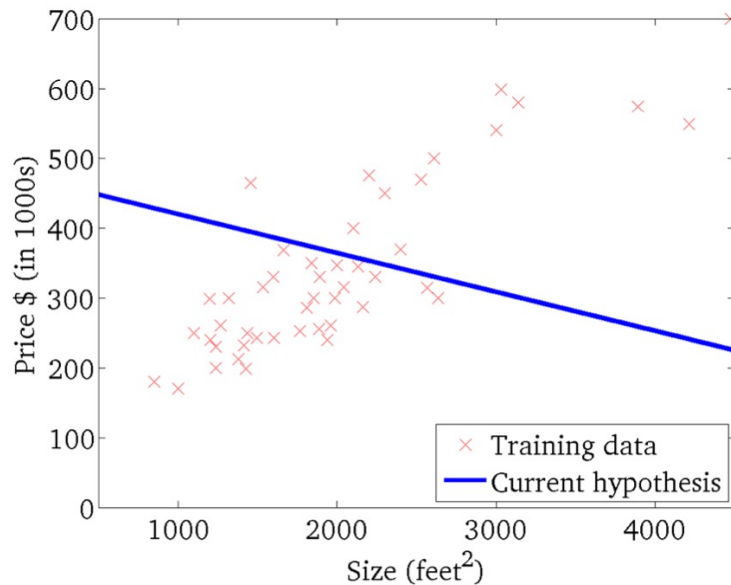
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

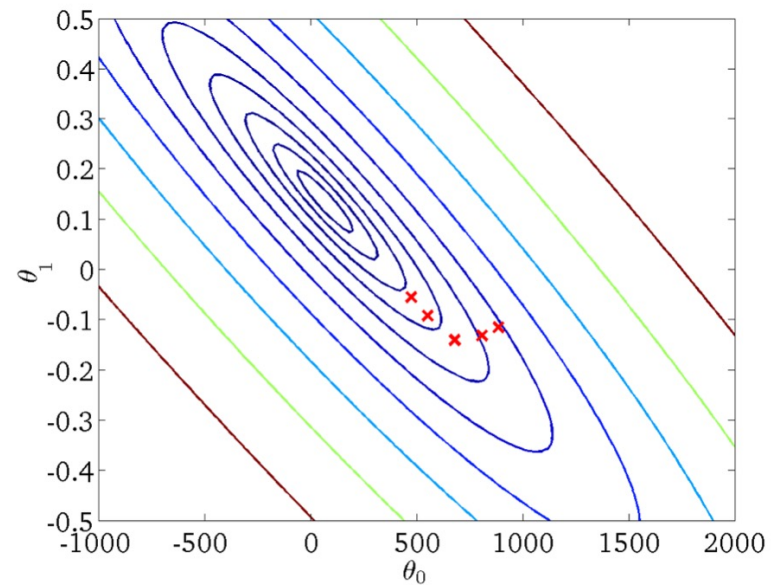
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

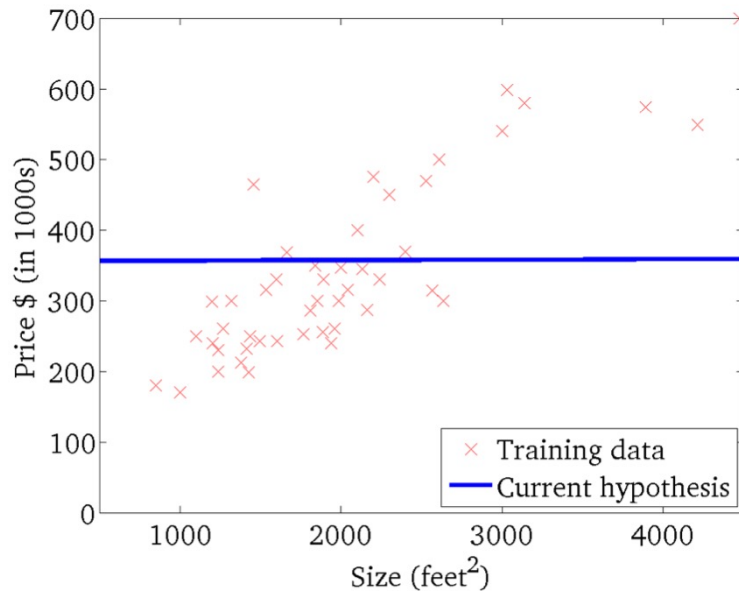
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

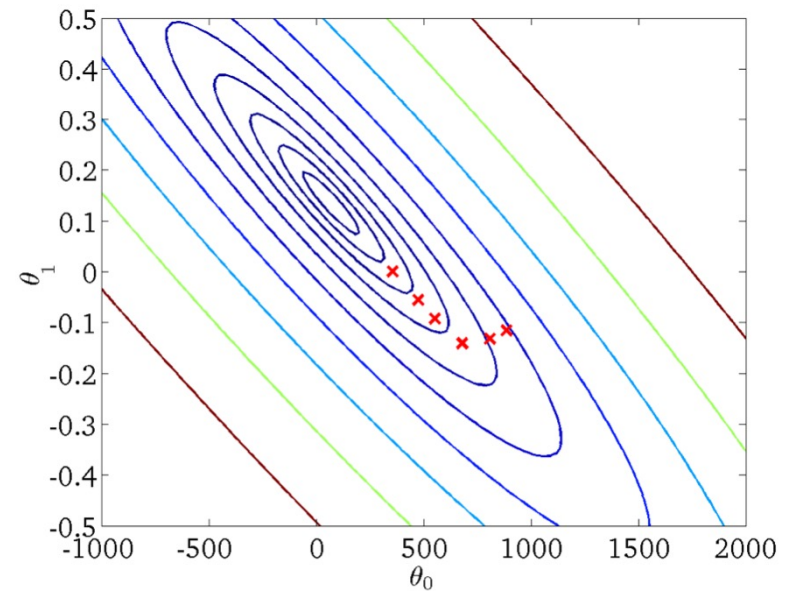
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

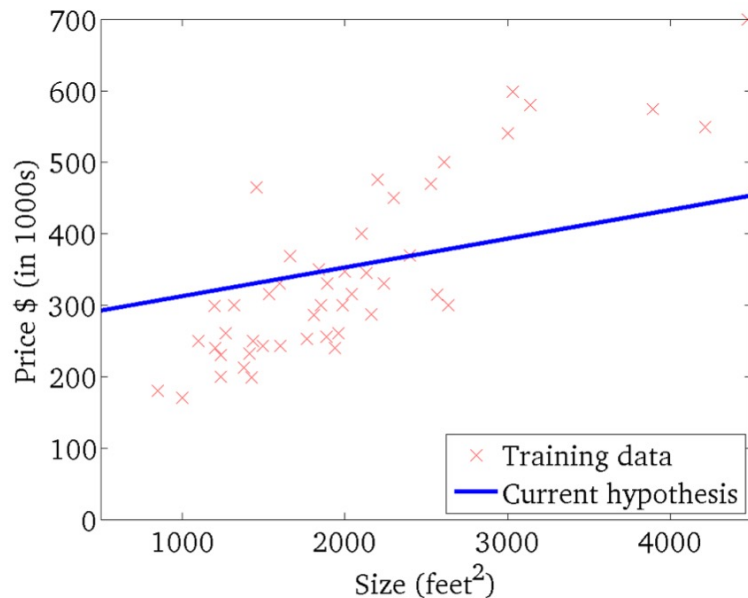
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

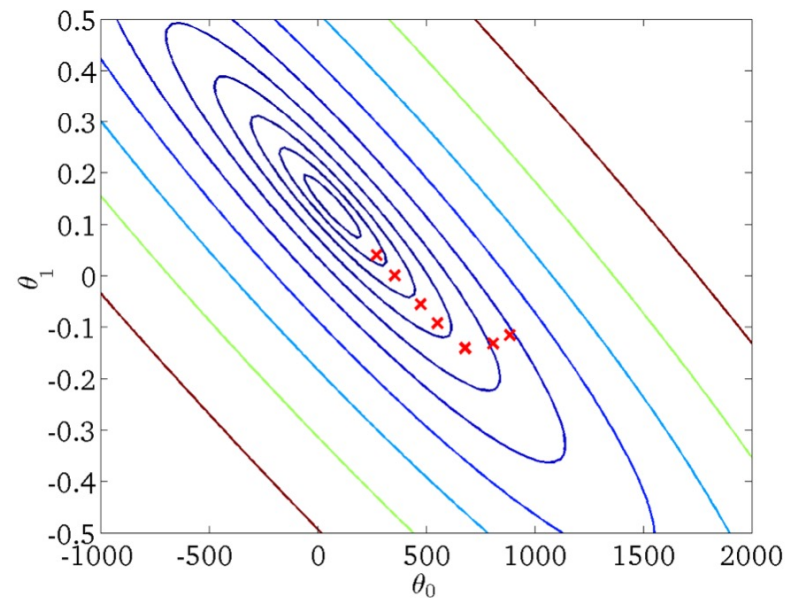
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

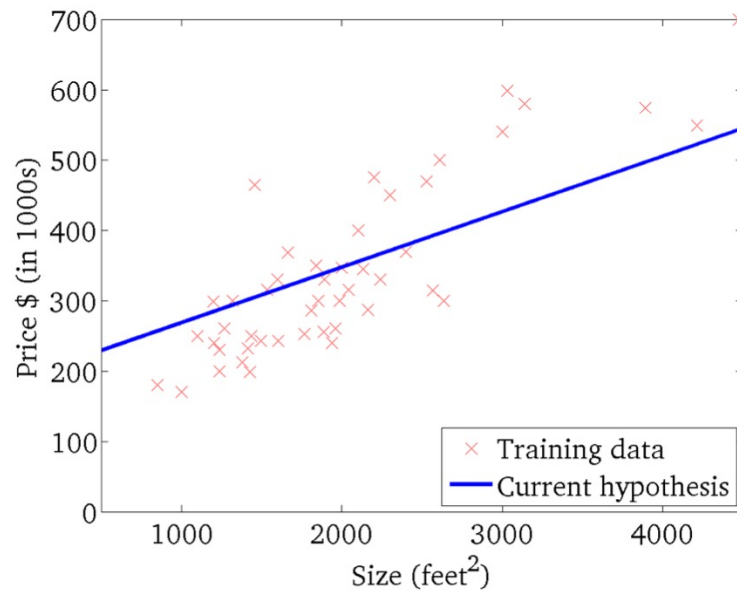
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

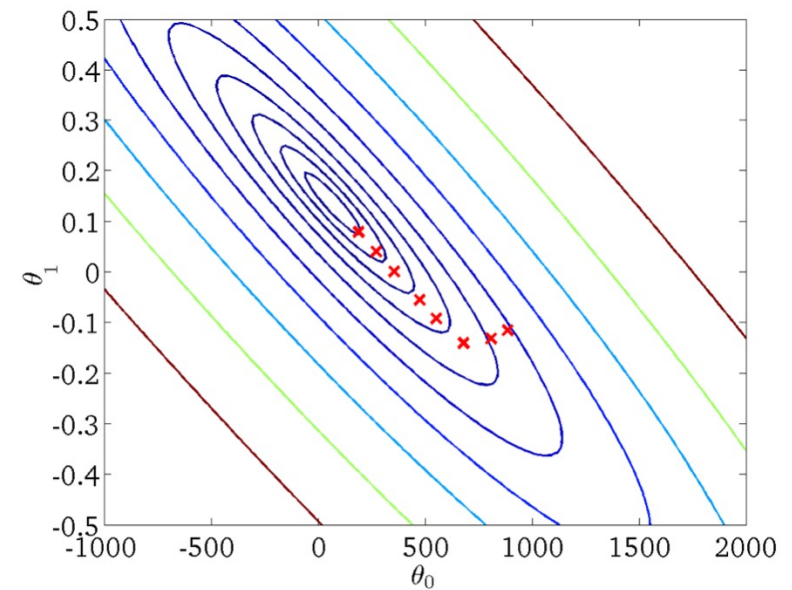
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

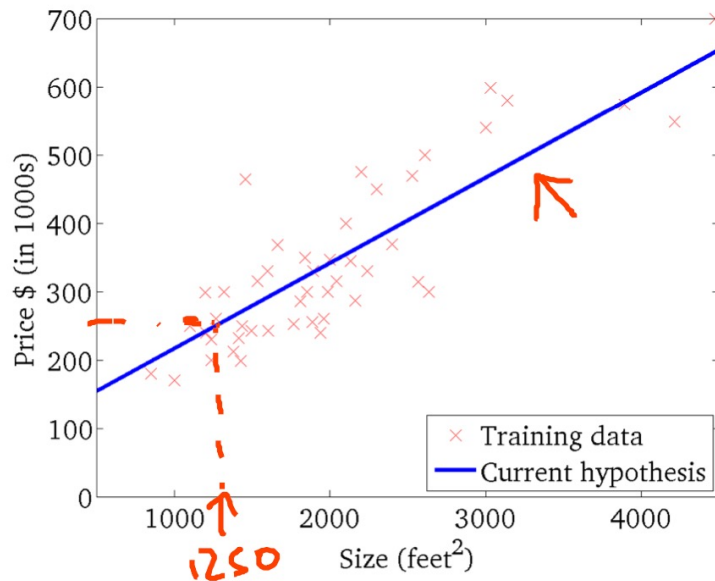
(function of the parameters θ_0, θ_1)



Descente du gradient : exemple

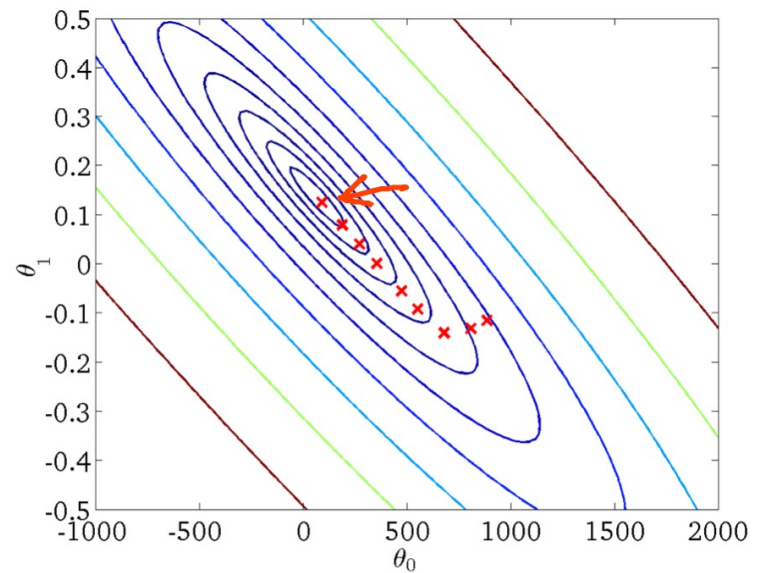
$$f_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



Descente du gradient

- **Recommandations :**
 - Normaliser les variables prédictives
 - Choisir la stratégie adaptée d'utilisation des exemples d'apprentissage pour la mise à jour du gradient
 - Choisir le taux d'apprentissage avec précaution

Normalisation des variables prédictives

- Objectif : s'assurer que l'ordre des grandeurs de toutes les variables est similaire
- ➔ Centrage réduction

$$x_i^k = \frac{x_i^k - \mu_i}{\sigma_i}$$

- Où μ_i est la moyenne de la variable prédictive x_i sur l'ensemble d'apprentissage
et σ_i est son écart-type

Stratégie de descente du gradient

- **Par lot (batch)** : Calcul du gradient en utilisant tous les exemples d'apprentissage

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{N} \sum_{i=1}^N (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

- **Stochastique** : Calcul du gradient à partir d'un unique exemple d'apprentissage (à répéter pour l'ensemble des exemples d'apprentissage ordonnés aléatoirement)

$$\frac{\partial}{\partial \theta_j} J(\theta) = (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

- **Par mini lots (mini-batch)** : On considère b exemples par itération pour le calcul du gradient où $b < N$

Stratégie de descente du gradient

→ Quel choix ?

- Le traitement par lot est très coûteux si les données d'apprentissage sont nombreuses
- L'algorithme stochastique converge souvent plus rapidement que celui par lot (mais reste lent car il faut boucler sur toutes les données d'apprentissage)
- Le traitement par mini lots est souvent une alternative intéressante car efficace et plus rapide que les deux autres
 - Peut utiliser des méthodes de calcul vectoriel
 - Convergence plus lisse que la stochastique (car considère plus d'exemples)

Choix du taux d'apprentissage

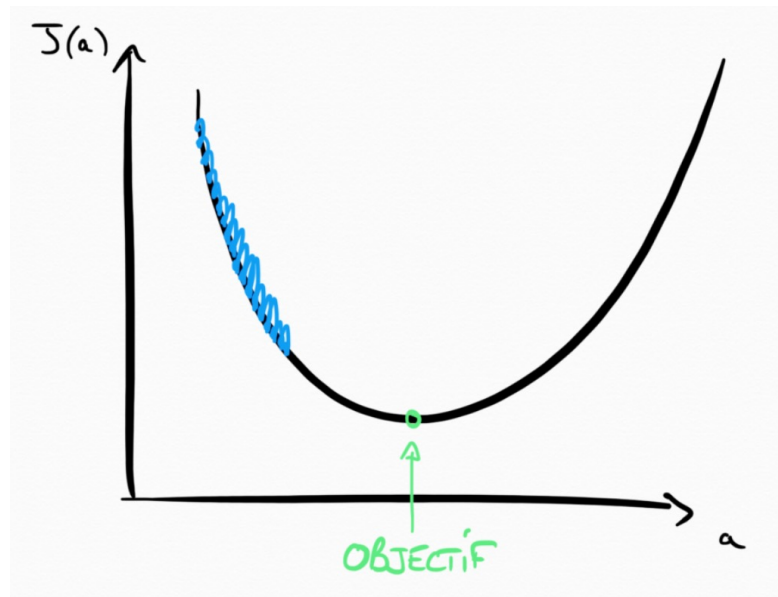
- Rappel : mise à jour des paramètres par :

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

➔ Comment choisir le taux d'apprentissage α ?

Choix du taux d'apprentissage

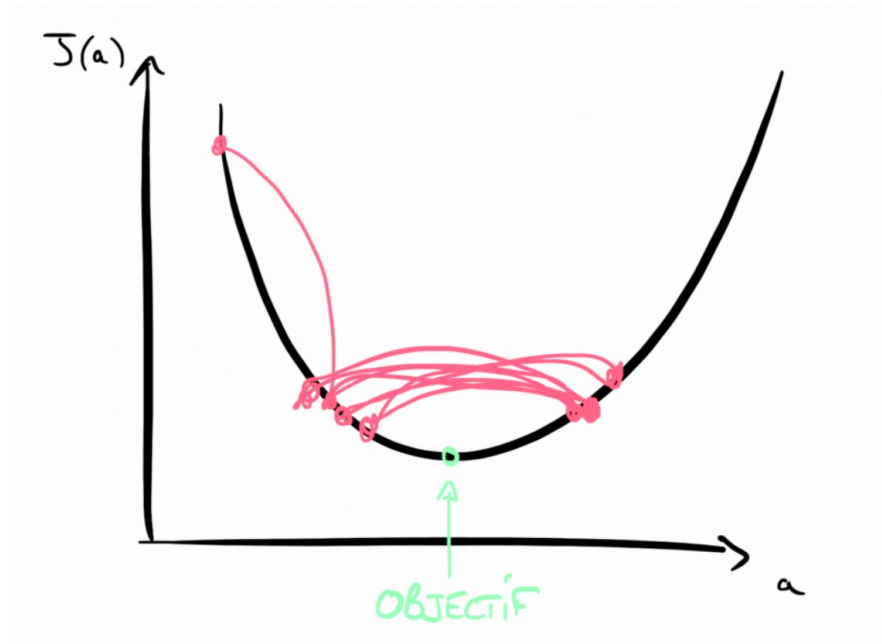
- α trop petit :



→ Convergence très lente

Choix du taux d'apprentissage

- α trop grand :



➔ La fonction de perte peut ne pas diminuer à chaque itération. L'algorithme risque alors de ne pas converger

Choix du taux d'apprentissage

- Il s'agit d'un hyper-paramètre que l'on doit fixer de manière empirique en réalisant des essais
- Pour choisir le taux d'apprentissage, essayer (x 3 à chaque étape) :
..., 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, ...

Il existe des stratégies pour changer la valeur du taux d'apprentissage progressivement au cours de la descente du gradient (*voir plus tard*)



ÉCOLE
CENTRALE LYON

36, avenue Guy de Collongue 69130 Écully - France
+33 (0)4 72 18 60 00

www.ec-lyon.fr