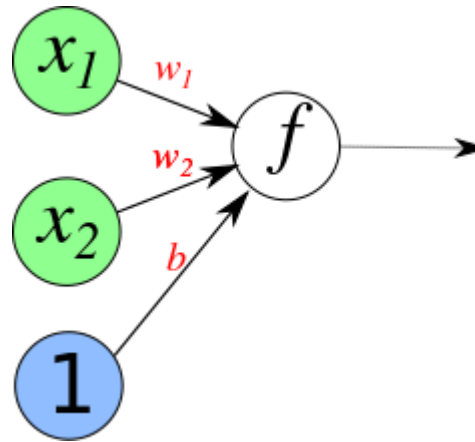


Introduction aux réseaux de neurones : La descente de gradient

Matériel de cours rédigé par Pascal Germain, 2018

out[1]: [voir/cacher le code.](#)

Reprenons notre réseau de neurone simple:



Nous avons vu que lorsque la neurone de sortie f est *linéaire*, la sortie du réseau de neurone est

$$R_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^d w_i x_i = \mathbf{w} \cdot \mathbf{x},$$

et qu'en considérant la *fonction de perte quadratique*, l'apprentissage revient à résoudre problème d'opimisation des *moindres carrés*:

$$\min_{\mathbf{w}} \left[\frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2 \right].$$

Nous désirons trouver le minimum de la **fonction objectif** $F(\mathbf{w})$ suivante:

$$F(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2.$$

Calculons la **dérivée partielle** de $F(\mathbf{w})$ selon un élément w_k du vecteur $\mathbf{w}$

$$\begin{aligned} \frac{\partial F(\mathbf{w})}{\partial w_k} &= \frac{1}{n} \sum_{i=1}^n \frac{\partial}{\partial w_k} (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2 \\ &= \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i) \frac{\partial}{\partial w_k} (\mathbf{w} \cdot \mathbf{x}_i - y_i) \\ &= \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i) \frac{\partial}{\partial w_k} (w_k x_{i,k}) \\ &= \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i) x_{i,k} . \end{aligned}$$

Le **gradient** de $F(\mathbf{w})$, noté $\nabla F(\mathbf{w})$, est le vecteur de toutes les dérivées partielles au point $\mathbf{w}$:

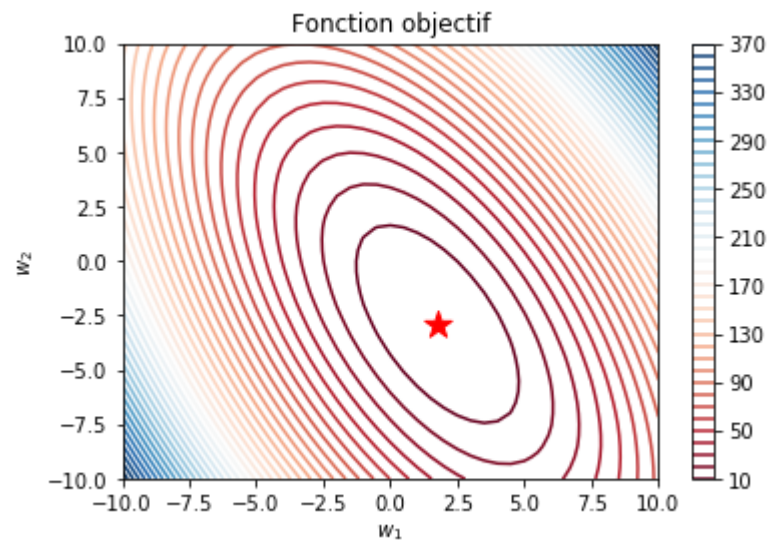
$$\nabla F(\mathbf{w}) = \begin{bmatrix} \frac{\partial}{\partial w_1} \\ \vdots \\ \frac{\partial}{\partial w_d} \end{bmatrix} = \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i) \mathbf{x}_i .$$

Le minimum d'une fonction convexe est atteint lorsque $\nabla F(\mathbf{w}) = 0$

Illustrons la fonction objectif pour un sensmble de donnés tel que:

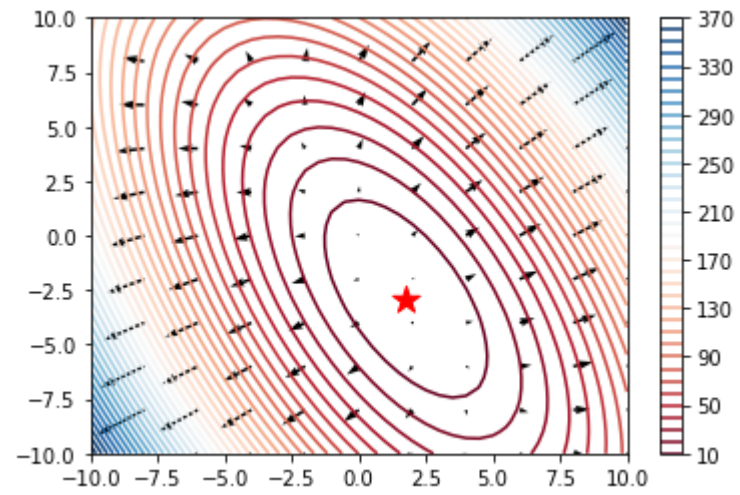
$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 0 & -1 \\ 2 & \frac{1}{2} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} -1 \\ 3 \\ 2 \end{bmatrix}$$

Out[2]: [voir/cacher le code.](#)



Illustrons maintenant les gradients de la fonction objectif.

out[3]: [voir/cacher le code.](#)



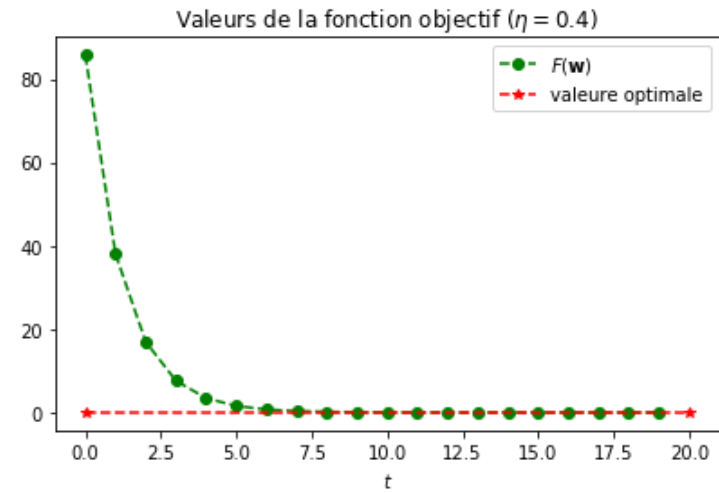
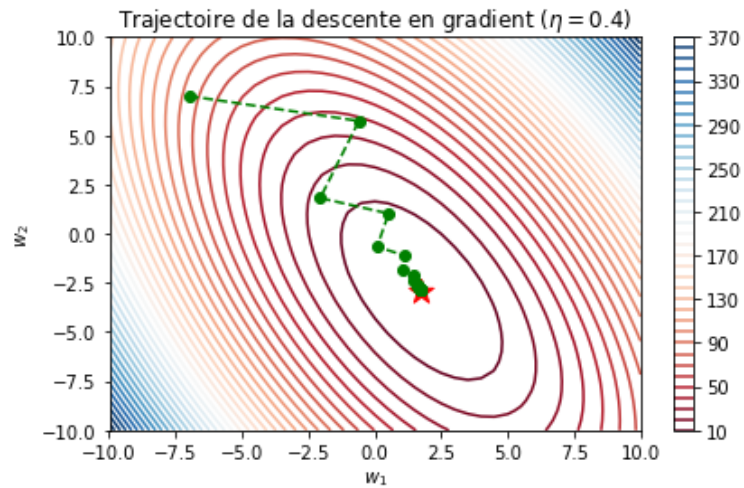
Descente en gradient

Algorithme (pour un *pas de gradient* η et un nombre d'itérations T)

- Initialiser $\mathbf{z}_0 \in \mathbb{R}^d$ aléatoirement
- Pour t de 0 à T :
 - $\mathbf{g}_t = \nabla F(\mathbf{z}_{t-1})$
 - $\mathbf{z}_t = \mathbf{z}_{t-1} - \eta \mathbf{g}_t$
- Retourner $\mathbf{z}_T$

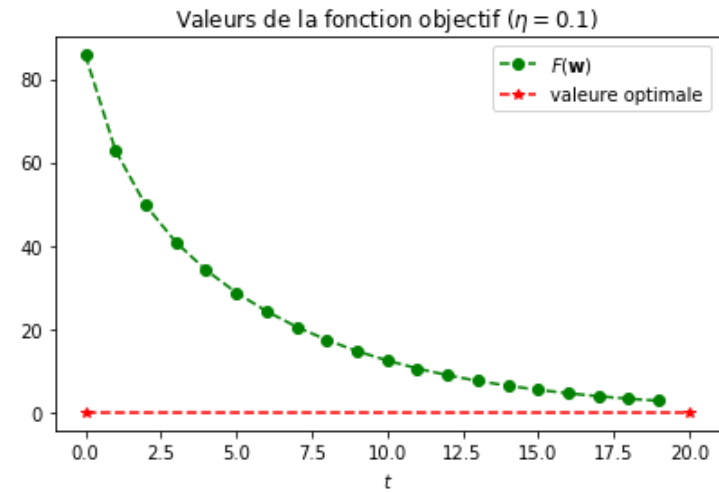
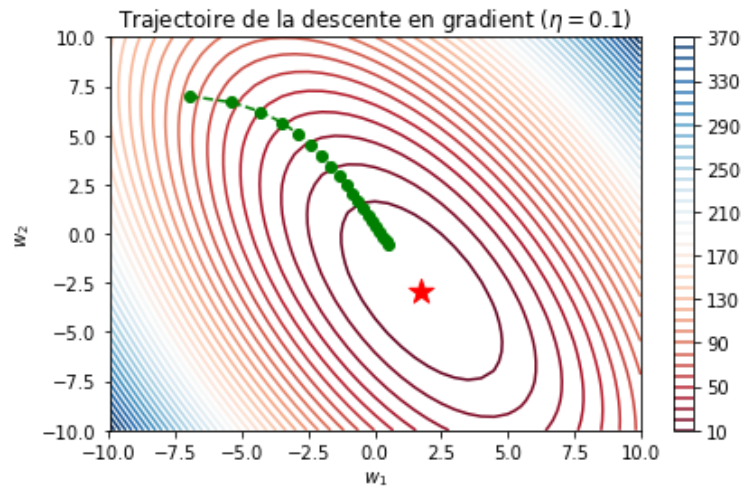
Exemple avec $\eta = 0.4$ et $T = 20$

Out[4]: [voir/cacher le code.](#)



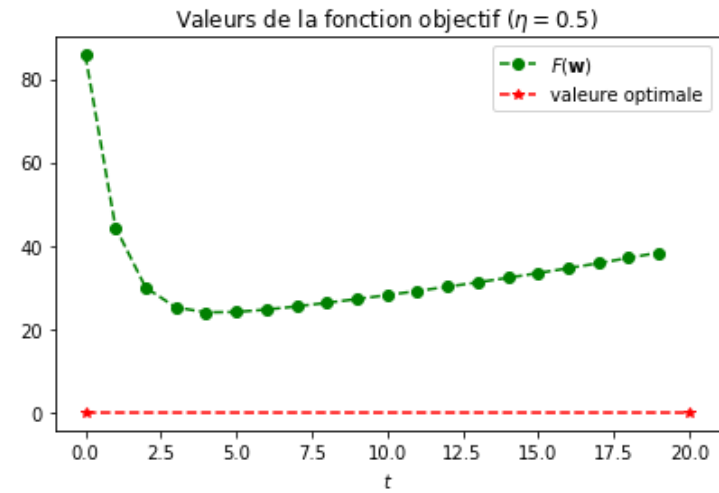
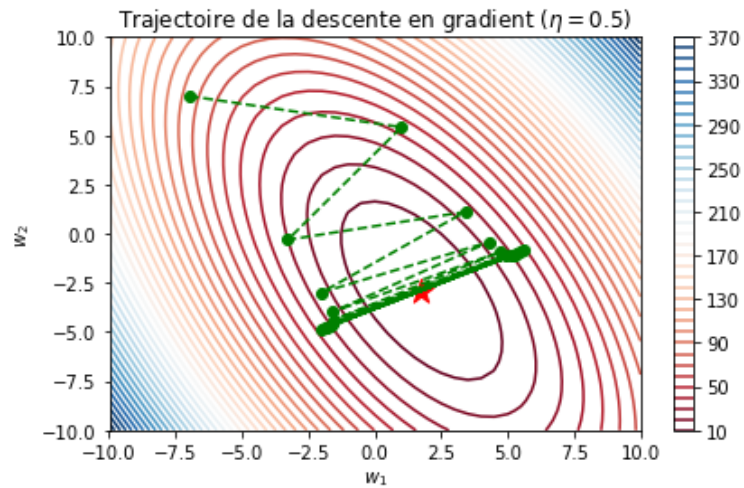
Exemple avec $\eta = 0.1$ et $T = 20$

Out[5]: [voir/cacher le code.](#)



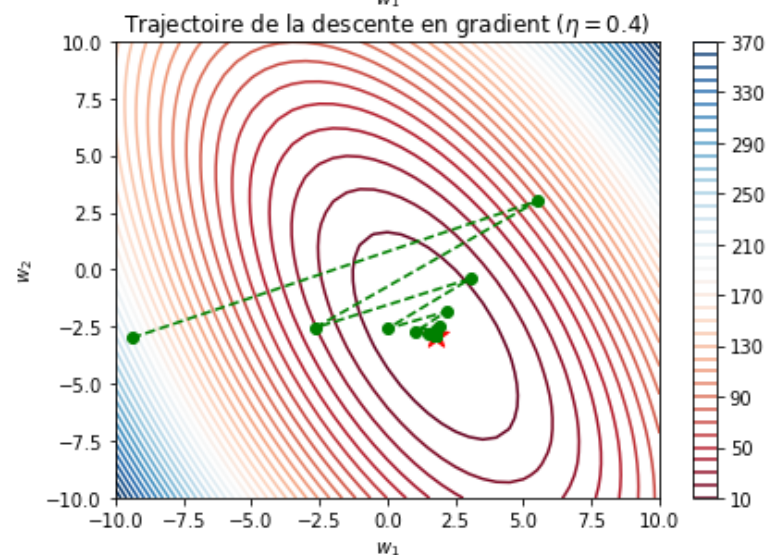
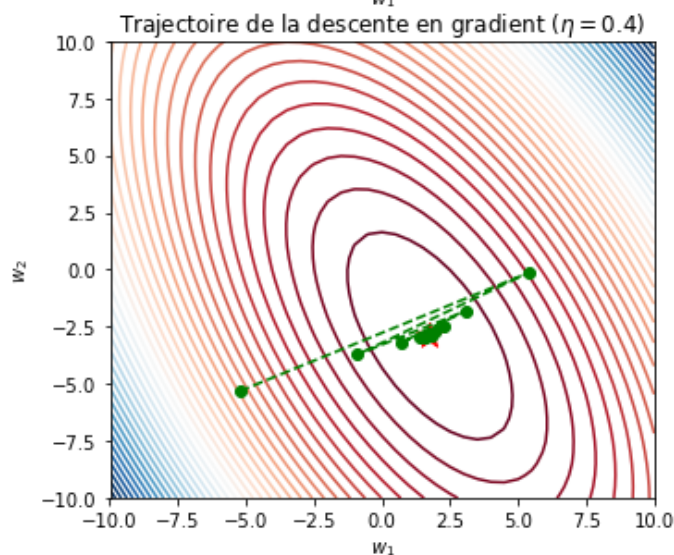
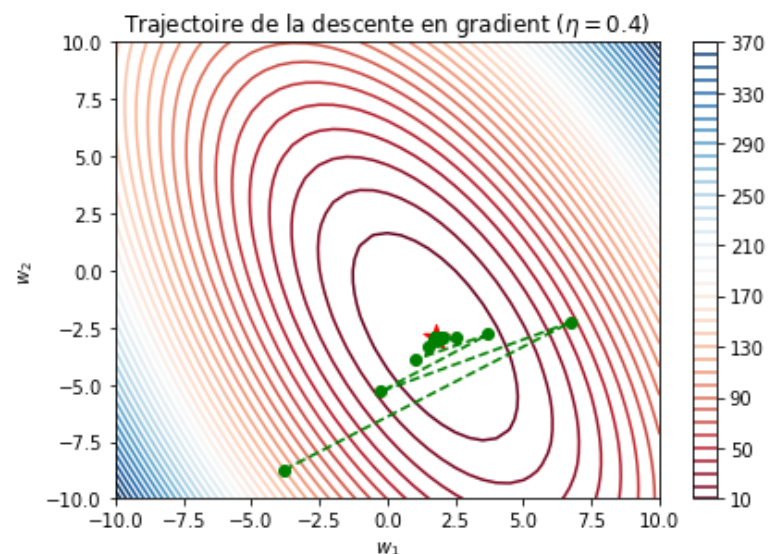
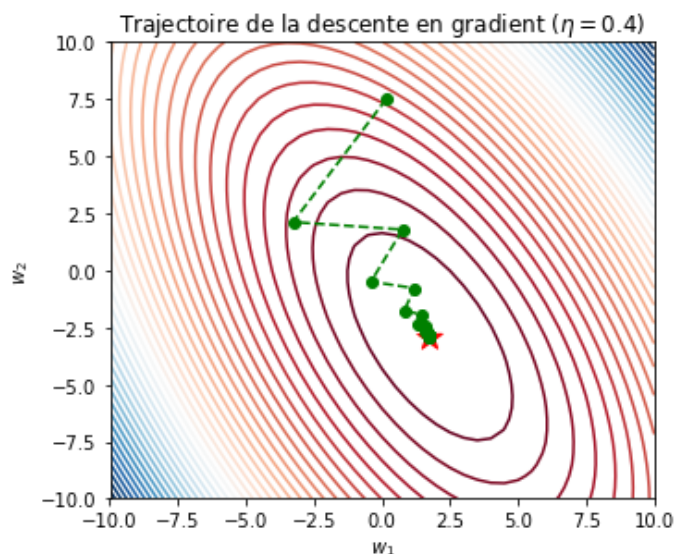
Exemple avec $\eta = 0.5$ et $T = 20$

Out[6]: [voir/cacher le code.](#)

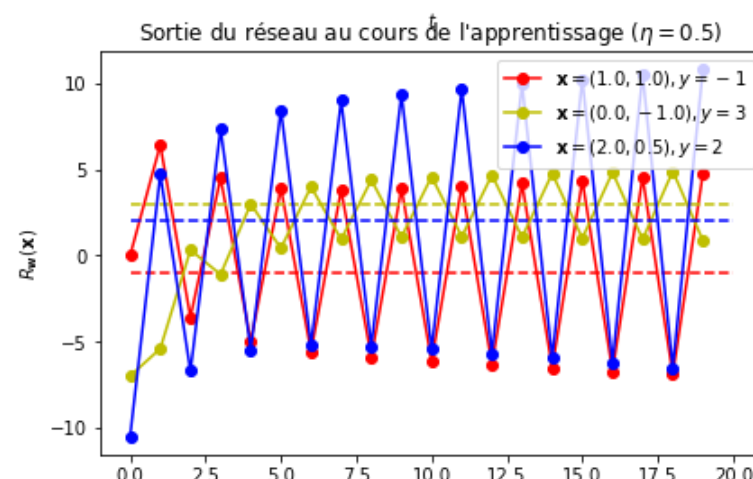
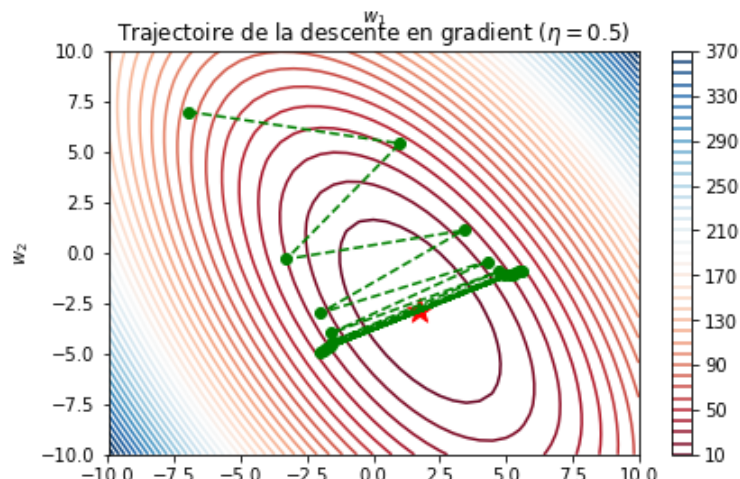
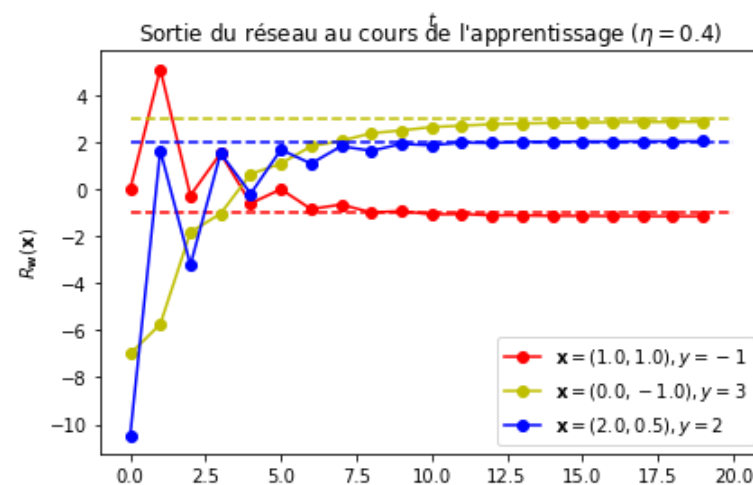
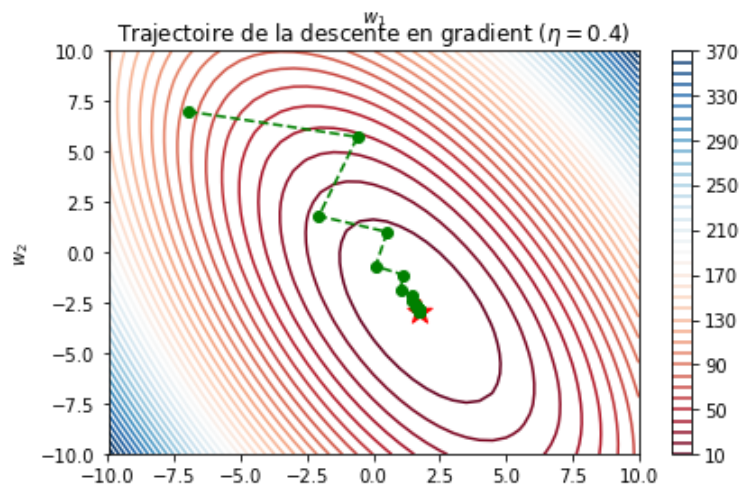
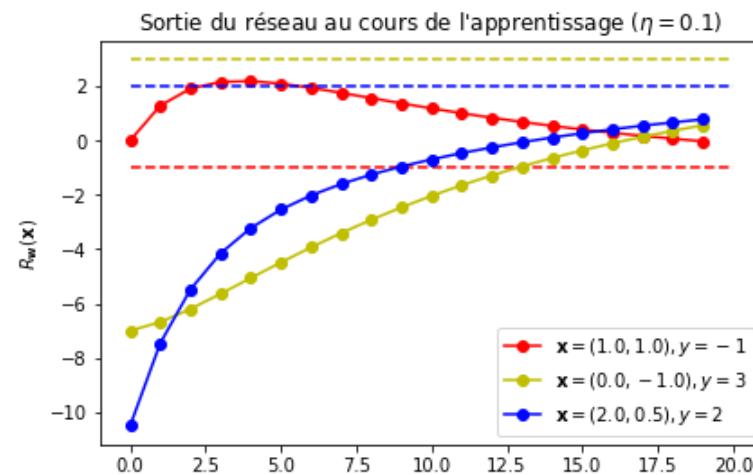
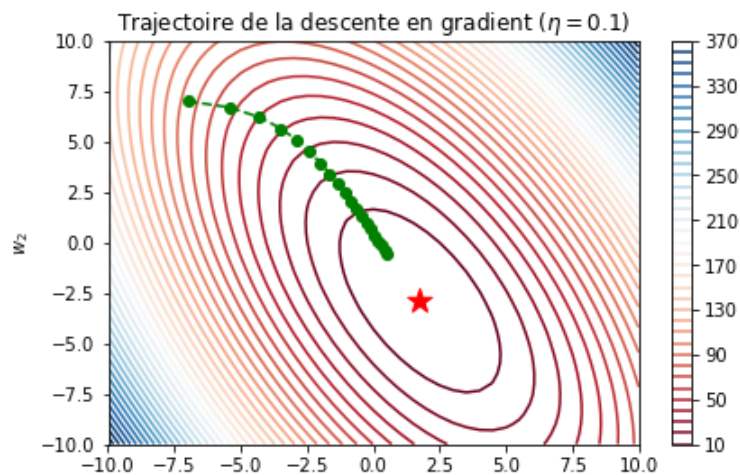


Exemple avec $\eta = 0.4$ et $T = 20$, différentes initialisation aléatoires

out[7]: [voir/cacher le code.](#)



out[8]: [voir/cacher le code.](#)



Descente en gradient stochastique

Nous désirons trouver le minimum de la **fonction objectif** $F(\mathbf{w})$ suivante:

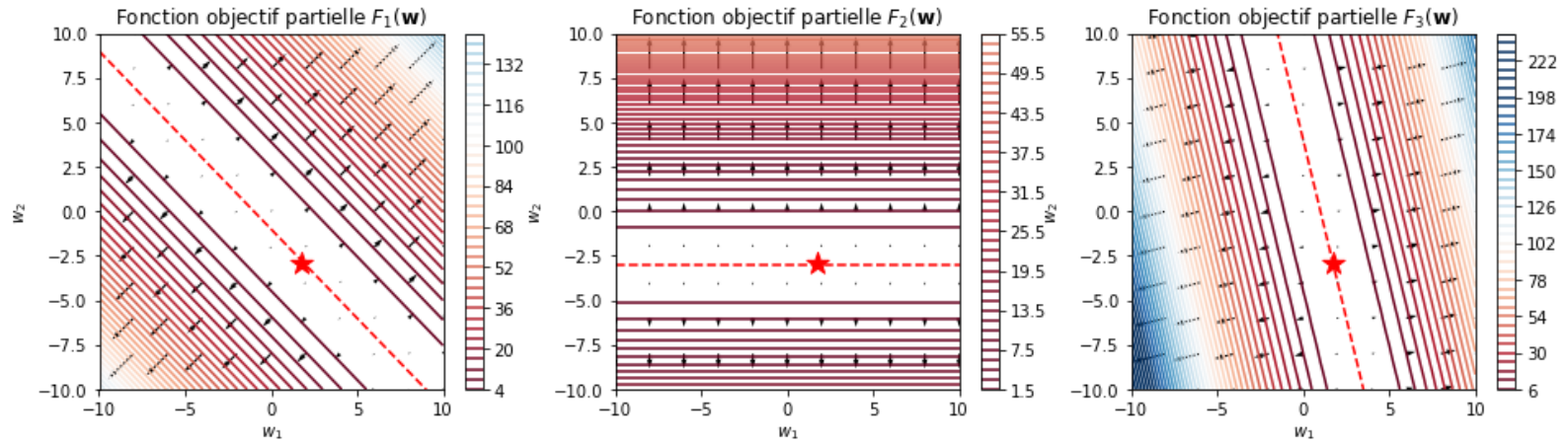
$$F(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^n F_i(\mathbf{w})$$

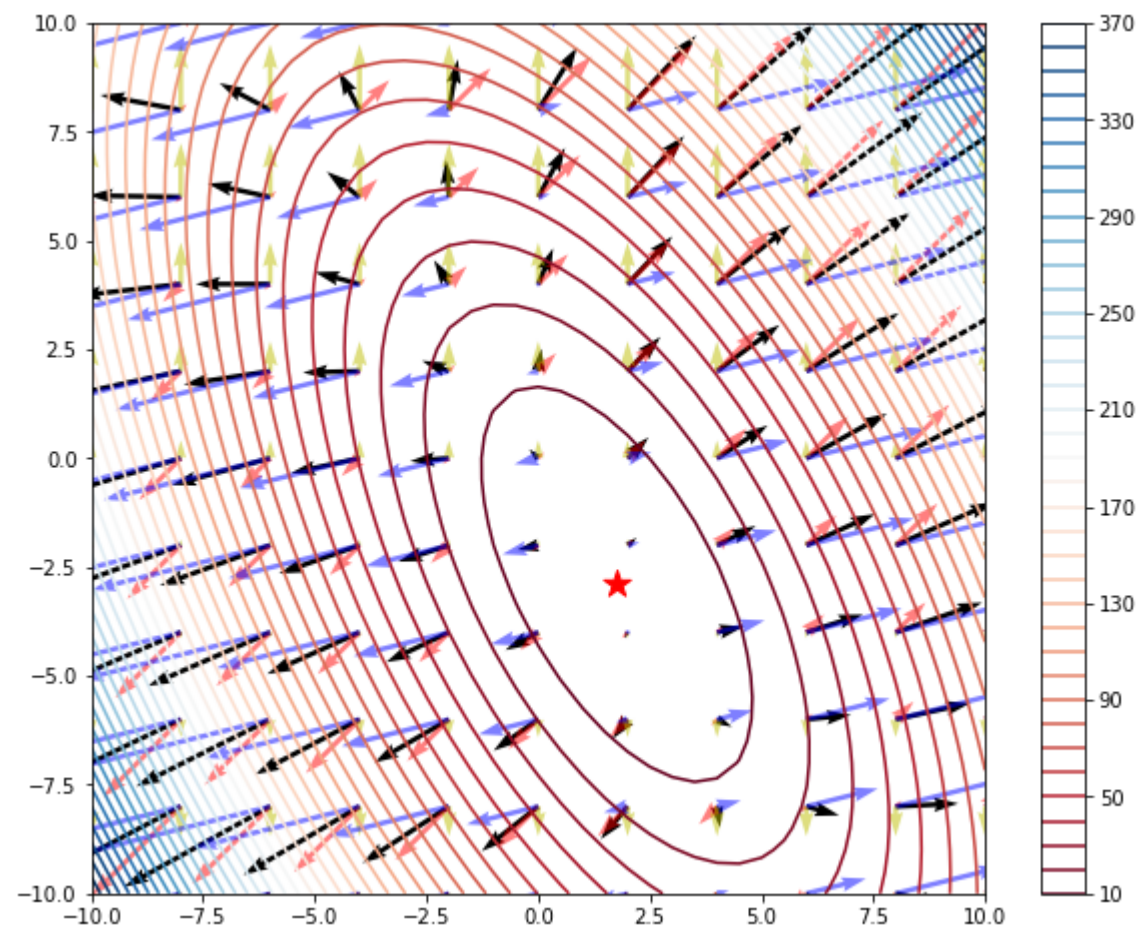
avec $F_i(\mathbf{w}) = (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2$ et donc $\nabla F_i(\mathbf{w}) = (\mathbf{w} \cdot \mathbf{x}_i - y_i) \mathbf{x}_i$.

Algorithme (pour un *pas de gradient* η et un nombre d'itérations T)

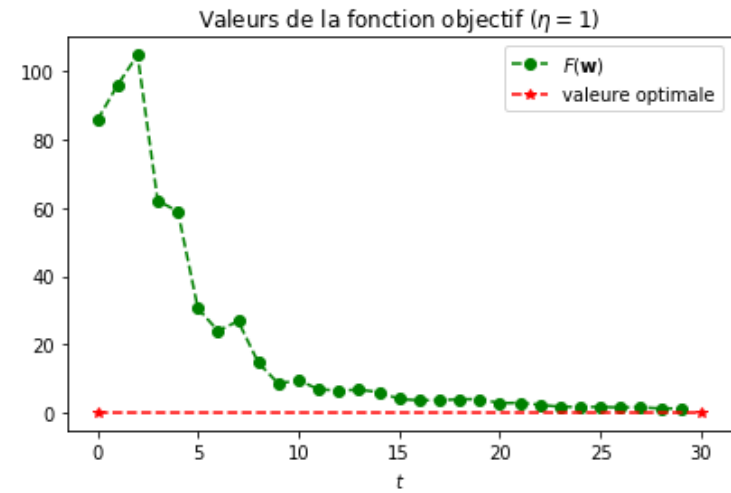
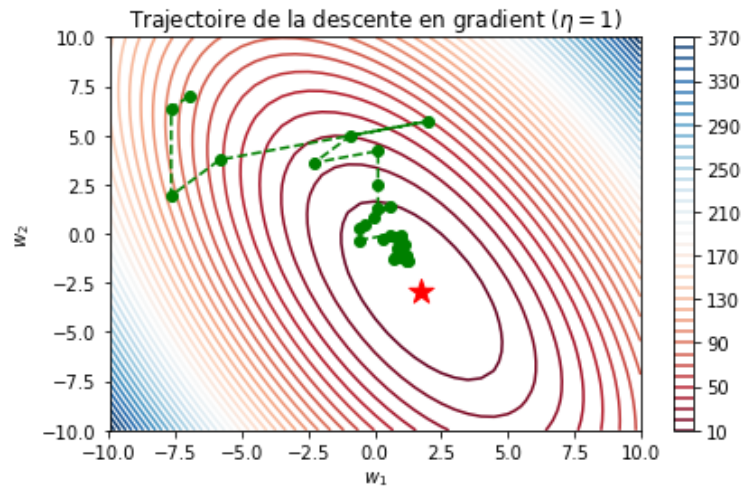
- Initialiser $\mathbf{z}_0 \in \mathbb{R}^d$ aléatoirement
- Pour t de 0 à T :
 - Choisir aléatoirement $i \in \{1, \dots, d\}$
 - $\mathbf{g}_t = \nabla F_i(\mathbf{z}_{t-1})$
 - $\mathbf{z}_t = \mathbf{z}_{t-1} - \frac{\eta}{\sqrt{t}} \mathbf{g}_t$
- Retourner $\mathbf{z}_T$

out[9]: voir/cacher le code.





out[11]: [voir/cacher le code.](#)



Descente en gradient stochastique avec momentum

Algorithme (pour un *pas de gradient* η , une *vélocité* α et un nombre d'itérations T)

- Initialiser $\mathbf{z}_0 \in \mathbb{R}^d$ aléatoirement
- $\mathbf{v}_0 = 0$
- Pour t de 0 à T :
 - Choisir aléatoirement $i \in \{1, \dots, d\}$
 - $\mathbf{g}_t = \nabla F_i(\mathbf{z}_{t-1})$
 - $\mathbf{v}_t = \alpha \mathbf{v}_{t-1} - \frac{\eta}{\sqrt{t}} \mathbf{g}_t$
 - $\mathbf{z}_t = \mathbf{z}_{t-1} + \mathbf{v}_t$
- Retourner $\mathbf{z}_T$

out[12]: [voir/cacher le code.](#)

