# ML Apprentissage: Déscente du gradient

# Apprentissage: **Gradient Descent**

- L'Apprentissage= minimiser la fonction coût (Loss Function)
- L'objectif est d'ajuster les paramètres *w* du modèle afin d'aboutir à la minimisation du loss function,

# Apprentissage: **Rappel Loss functions**

Régression Linéaire et MSE

$$L(w) = \frac{1}{l}\sum_{i=1}^{l}(w^T x_i - y_i)^2$$

$$L(w) = \frac{1}{l}\|Xw - y\|^2$$

Classification linéaire et Entropie Croisée

$$L(w) = -\sum_{i=1}^{l}\sum_{k=1}^{K}[y_i = k]log\frac{e^{w_k^T x_i}}{\sum_{j=1}^{K} e^{w_j^T x_i}}$$

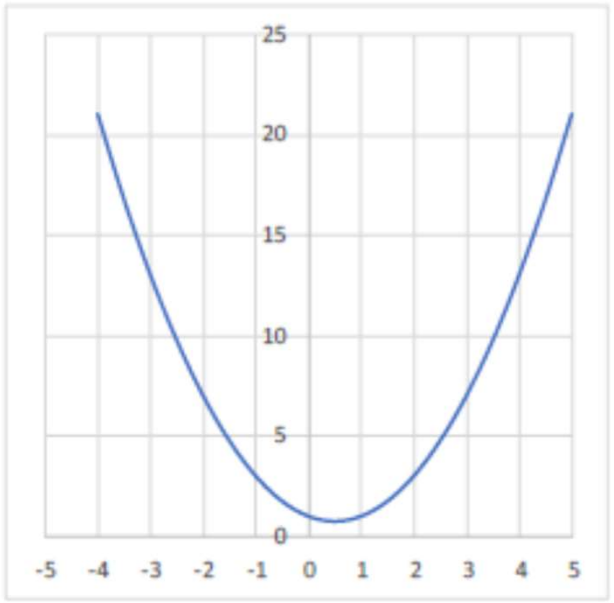
# Apprentissage: **Solution analytique**



**Rechercher le minimum d'une fonction différentiable est un problème usuel**

Ex. L(w) = $w^2$ -w+ 1 ; à minimiser par rapport au paramètre w

On recherche la valeur de w qui minimise L(w)




La solution analytique passe par :

$L'(w) = 0$

En s'assurant que $L''(w) > 0$

$L'(w) = 2w - 1 = 0 \Rightarrow \mathbf{w* = 1/2}$

Le problème est quand le nombre de paramètres est élevé :

W=(w1, w2,.....wd)

Le problème revient alors à résoudre un grand système d'équations

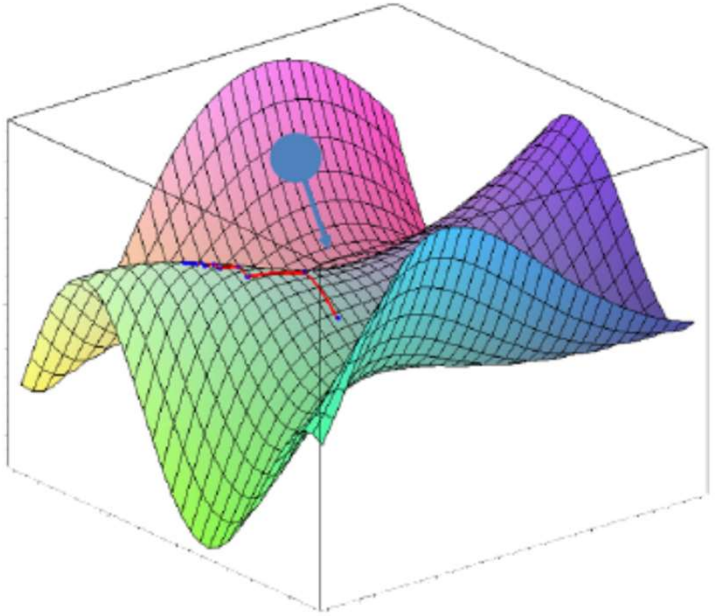
➔ Très **coûteux** en mémoire et en temps de calcul

➔ **L'optimisation** est la solution

# Apprentissage: **Problème d'optimisation**

Problème d'optimisation : $L(w) \rightarrow \min_{w} L(w)$

Supposons que nous ayons une approximation $w^0$— comment la raffiner ?



# Descente du Gradient : **Algorithme**



Problème d'optimisation : $L(w) \rightarrow \min_{w} L(w)$

$w^0$— initialisation

$\nabla L(w^0) = (\frac{\partial L(w^0)}{\partial w_1}, \ldots, \frac{\partial L(w^0)}{\partial wd})$ Vecteur gradient

- Points dans la direction opposée de la pente à $w^0$
- La fonction décroit plus rapidement dans la direction négative du gradient,

# Descente du Gradient : **Algorithme**



Problème d'optimisation : $L(w) \rightarrow \min_{w} L(w)$

$w^0$ — initialisation

$\nabla L(w^0) = (\frac{\partial L(w^0)}{\partial w_1}, \ldots, \frac{\partial L(w^0)}{\partial wd})$ Vecteur gradient

$w^1 = w^0 - \eta \nabla L(w^0)$ gradient step// 1ère itération

# Descente du Gradient: **Algorithme**



$w^0$— initialisation

Répéter

$$w^{t+1} = w^t - \eta \times \nabla L(w^t)$$

Jusqu'à $\|w^{t+1} - w^t\| < \epsilon$

Le «gradient» ∇, généralisation multidimensionnelle de la dérivée [si un seul paramètre], au point $w^t$ . Indique la direction et l'importance de la pente au voisinage de $w^t$.

η: learning rate est un paramètre qui permet de moduler la (η trop faible, lenteur de convergence ; η trop élevé, oscillation)

—parce qu'on cherche à minimiser $L()$, on prendrait +sinon.

# Descente du Gradient: **Algorithme- Exemple1**



$$f(x) = x^2 - x + 1$$

$$\nabla f(x) = \frac{\partial f(x)}{\partial x} = f'(x) = 2x - 1$$

Il n'y a qu'un seul paramètre, la dérivée partielle est égale à la dérivée.

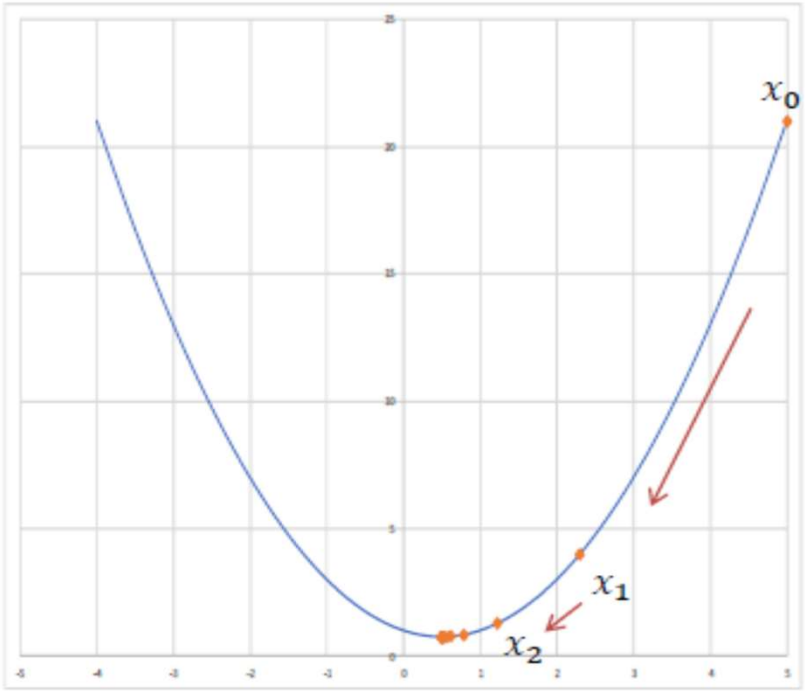
$\eta = 0.3$

| eta | 0.3 |
|---|---|

$x_0 = 5$

| x | f'(x_t) | f(x) |
|---|---|---|
| 5.0000 | | 21.0000 |
| 2.3000 | 9.0000 | 3.9900 |
| 1.2200 | 3.6000 | 1.2684 |
| 0.7880 | 1.4400 | 0.8329 |
| 0.6152 | 0.5760 | 0.7633 |
| 0.5461 | 0.2304 | 0.7521 |
| 0.5184 | 0.0922 | 0.7503 |
| 0.5074 | 0.0369 | 0.7501 |
| 0.5029 | 0.0147 | 0.7500 |
| 0.5012 | 0.0059 | 0.7500 |
| 0.5005 | 0.0024 | 0.7500 |
| 0.5002 | 0.0009 | 0.7500 |
| 0.5001 | 0.0004 | 0.7500 |
| 0.5000 | 0.0002 | 0.7500 |

$$x_{t+1} = x_t - \eta \times \nabla f(x_t)$$




$\dots x_2 < x_1 < x_0$

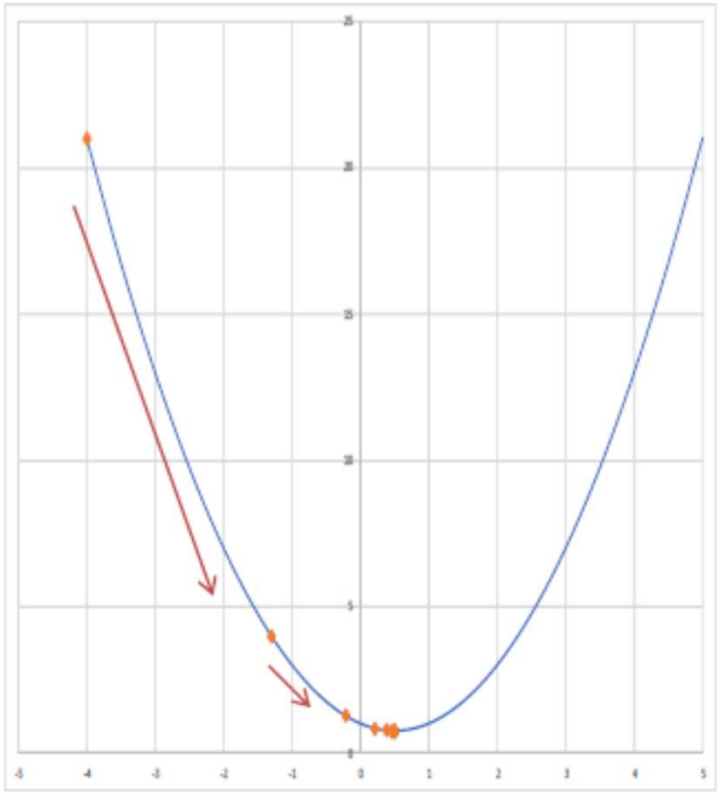
# Descente du Gradient: **Algorithme- Exemple2**



On aurait pu partir de l'autre côté…

| eta | 0.3 |
|---|---|

| x | f'(x_t) | f(x) |
|---|---|---|
| -4.0000 | | 21.0000 |
| -1.3000 | -9.0000 | 3.9900 |
| -0.2200 | -3.6000 | 1.2684 |
| 0.2120 | -1.4400 | 0.8329 |
| 0.3848 | -0.5760 | 0.7633 |
| 0.4539 | -0.2304 | 0.7521 |
| 0.4816 | -0.0922 | 0.7503 |
| 0.4926 | -0.0369 | 0.7501 |
| 0.4971 | -0.0147 | 0.7500 |
| 0.4988 | -0.0059 | 0.7500 |
| 0.4995 | -0.0024 | 0.7500 |
| 0.4998 | -0.0009 | 0.7500 |
| 0.4999 | -0.0004 | 0.7500 |
| 0.5000 | -0.0002 | 0.7500 |



$x_0 < x_1 < x_2 \ldots$

# Descente du Gradient: **Régression Linéaire multiple**



$g(x_i) = w^T x_i = w_0 + w_1 x_{i1} + \ldots + w_l x_{id}$ **avec** $w = (w_0, w_1, \ldots, w_d)$ les coefficients
**et** $x_i = (x_{i1}, \ldots x_{id})$ une donnée $i$

**loss function MSE**: $L(w) = \frac{1}{2n}\sum_{i=1}^{n}(w^T x_i - y_i)^2$ → $\min_{w_0,..,w_l} L(w)$

**Descente du gradient** $w^{t+1} = w^t - \eta \times \nabla L(w^t)$

Où $\nabla L(w^t) =$

$$\frac{\partial L(w^t)}{\partial w_0} = \frac{1}{n}\sum_{i=1}^{n} 1 \times (w^T x_i - y_i)$$

$$\frac{\partial L(w^t)}{\partial w_1} = \frac{1}{n}\sum_{i=1}^{n} x_{i1} \times (w^T x_i - y_i)$$

....

$$\frac{\partial L(w^t)}{\partial w_l} = \frac{1}{n}\sum_{i=1}^{n} x_{il} \times (w^T x_i - y_i)$$

# Régression linéaire multiple – Exemple

| x0 | x1 | x2 | y |
|---|---|---|---|
| 1 | 0.72 | 0.32 | 6.93 |
| 1 | 0.75 | 0.12 | 5.99 |
| 1 | 0.53 | 0.65 | 1.46 |
| 1 | 0.27 | 0.82 | 1.44 |
| 1 | 0.49 | 0.15 | 4.51 |
| 1 | 0.02 | 0.19 | 1.25 |
| 1 | 0.35 | 0.87 | 2.53 |
| 1 | 0.99 | 0.71 | 6.88 |
| 1 | 0.98 | 0.92 | 6.25 |
| 1 | 0.73 | 0.19 | 6.36 |

Il est préférable d'harmoniser les données (normalisation, standardisation) pour éviter les problèmes d'échelles.

$$a^0 = (0.1, 0.1, 0.1)$$

$$a^{30} = (1.658, 6.618, -2.314)$$

$$a^{solution} = (1.424, 7.173, -2.523)$$

S est loss function

a est le vecteur des paramètres



| | eta | 1.05 | | | | | |
|---|---|---|---|---|---|---|---|
| t | a0 | a1 | a2 | S | dS/d_ao | dS/d_a1 | dS/d_a2 |
| 0 | 0.100 | 0.100 | 0.100 | 224.72 | -4.152 | -3.003 | -1.889 |
| 1 | 4.460 | 3.253 | 2.083 | 127.71 | 3.026 | 1.483 | 1.892 |
| 2 | 1.283 | 1.697 | 0.097 | 77.53 | -2.040 | -1.633 | -0.825 |
| 3 | 3.425 | 3.411 | 0.963 | 50.95 | 1.529 | 0.609 | 1.045 |
| 4 | 1.819 | 2.771 | -0.135 | 36.36 | -0.992 | -0.931 | -0.315 |
| 5 | 2.860 | 3.749 | 0.196 | 27.95 | 0.783 | 0.193 | 0.606 |
| 6 | 2.039 | 3.546 | -0.440 | 22.79 | -0.472 | -0.564 | -0.078 |
| 7 | 2.534 | 4.138 | -0.358 | 19.39 | 0.410 | 0.002 | 0.373 |
| 8 | 2.104 | 4.135 | -0.750 | 17.00 | -0.216 | -0.367 | 0.026 |
| 9 | 2.330 | 4.521 | -0.778 | 15.22 | 0.222 | -0.079 | 0.245 |
| 10 | 2.097 | 4.604 | -1.035 | 13.83 | -0.090 | -0.257 | 0.067 |
| 11 | 2.191 | 4.874 | -1.105 | 12.72 | 0.127 | -0.108 | 0.171 |
| 12 | 2.058 | 4.987 | -1.284 | 11.80 | -0.029 | -0.191 | 0.078 |
| 13 | 2.088 | 5.188 | -1.365 | 11.04 | 0.078 | -0.112 | 0.125 |
| 14 | 2.006 | 5.306 | -1.497 | 10.41 | 0.000 | -0.149 | 0.075 |
| 15 | 2.006 | 5.463 | -1.576 | 9.87 | 0.052 | -0.106 | 0.096 |
| 16 | 1.951 | 5.574 | -1.676 | 9.42 | 0.013 | -0.121 | 0.068 |
| 17 | 1.938 | 5.701 | -1.747 | 9.03 | 0.038 | -0.096 | 0.075 |
| 18 | 1.898 | 5.802 | -1.826 | 8.71 | 0.018 | -0.100 | 0.059 |
| 19 | 1.879 | 5.907 | -1.888 | 8.43 | 0.030 | -0.085 | 0.060 |
| 20 | 1.847 | 5.996 | -1.951 | 8.20 | 0.019 | -0.084 | 0.050 |
| 21 | 1.827 | 6.084 | -2.003 | 8.00 | 0.025 | -0.074 | 0.048 |
| 22 | 1.801 | 6.161 | -2.054 | 7.83 | 0.019 | -0.071 | 0.042 |
| 23 | 1.782 | 6.236 | -2.098 | 7.68 | 0.021 | -0.064 | 0.039 |
| 24 | 1.759 | 6.303 | -2.139 | 7.56 | 0.018 | -0.061 | 0.035 |
| 25 | 1.741 | 6.367 | -2.175 | 7.46 | 0.018 | -0.055 | 0.032 |
| 26 | 1.722 | 6.425 | -2.209 | 7.37 | 0.016 | -0.052 | 0.029 |
| 27 | 1.705 | 6.479 | -2.239 | 7.29 | 0.016 | -0.047 | 0.026 |
| 28 | 1.688 | 6.529 | -2.267 | 7.23 | 0.015 | -0.044 | 0.024 |
| 29 | 1.673 | 6.575 | -2.292 | 7.17 | 0.014 | -0.041 | 0.022 |
| 30 | 1.658 | 6.618 | -2.314 | 7.12 | 0.013 | -0.038 | 0.019 |

Convergence lente parce petit effectifs (n=10).

# Gradient Descent: **Méthodes**

**Batch Gradient Descent (BGD)** utilise les échantillons (m au total) dans tous les ensembles de données pour mettre à jour les paramètres de poids en fonction de la valeur du gradient au point actuel.

$$w_{k+1} = w_k - \eta \frac{1}{m} \sum_{i=1}^{m} \nabla f_{w_k}(x^i)$$

**La descente de gradient stochastique (SGD)** sélectionne au hasard un échantillon dans un ensemble de données pour mettre à jour le paramètre de poids en fonction de la valeur du gradient au point actuel.

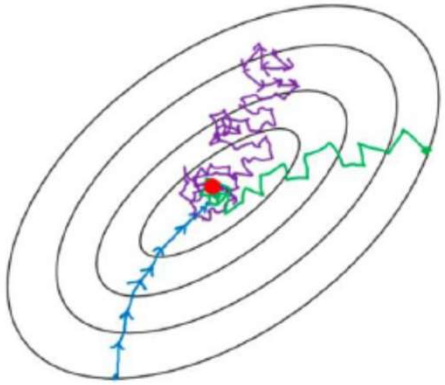
$$w_{k+1} = w_k - \eta \nabla f_{w_k}(x^i)$$

**Mini-Batch Gradient Descent (MBGD)** combine les fonctionnalités de BGD et SGD et sélectionne les gradients de n échantillons dans un ensemble de données pour mettre à jour le poids paramètre.

$$w_{k+1} = w_k - \eta \frac{1}{n} \sum_{i=\mathrm{t}}^{t+n-1} \nabla f_{w_k}(x^i)$$

# Gradient Descent: **Méthodes-Comparaison**

- Comparison of three gradient descent methods
  - In the SGD, samples selected for each training are stochastic. Such instability causes the loss function to be unstable or even causes reverse displacement when the loss function decreases to the lowest point.
  - BGD has the highest stability but consumes too many computing resources. MBGD is a method that balances SGD and BGD.



**BGD**
Uses **all** training samples for training each time.

**SGD**
Uses **one** training sample for training each time.

**MBGD**
Uses a certain number of training samples for training each time.