# 梯度下降法的優化

# Optimization of Gradient Descent Method

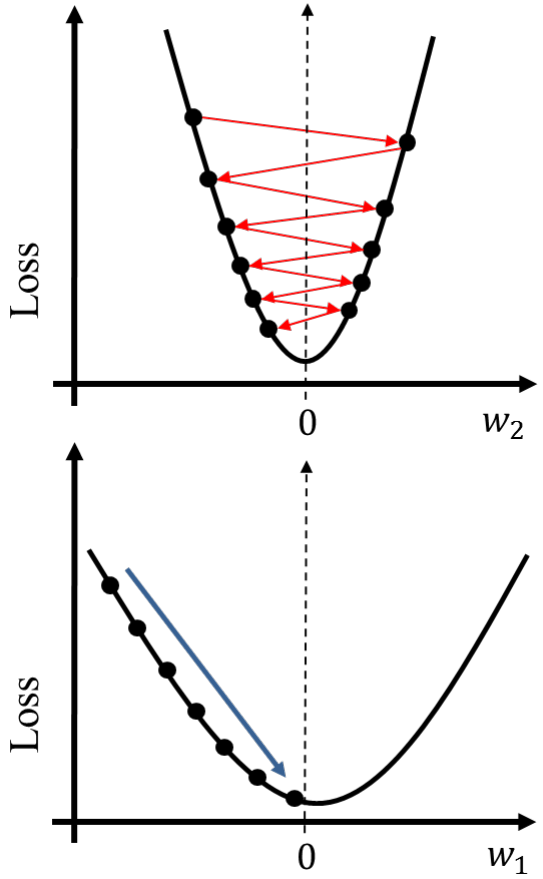
國立東華大學電機工程學系 楊哲旻

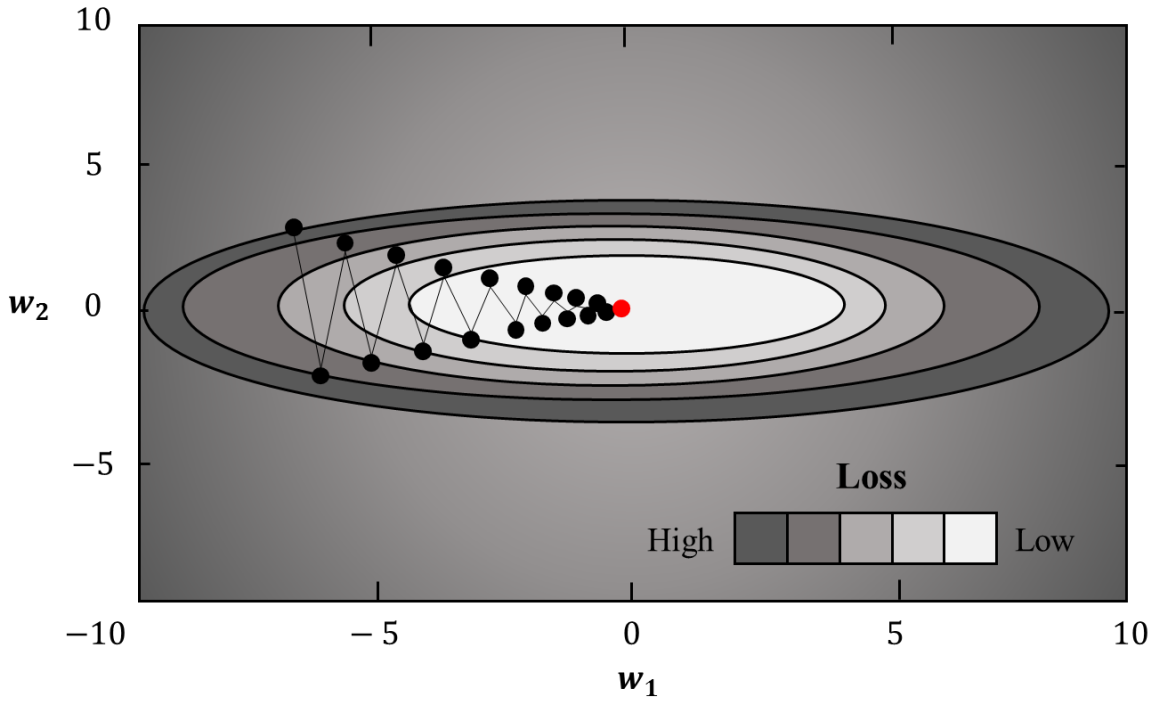
# Outline

1. 梯度下降法
2. 動量的梯度下降法
3. AdaGrad
4. RMSProp
5. Adam
6. 演算法的比較

# 梯度下降法的優化 01. 梯度下降法

## 梯度下降法(Stochastic Gradient descent)




$$w^t = w^{t-1} - \eta \frac{\partial L}{\partial w^{t-1}}$$

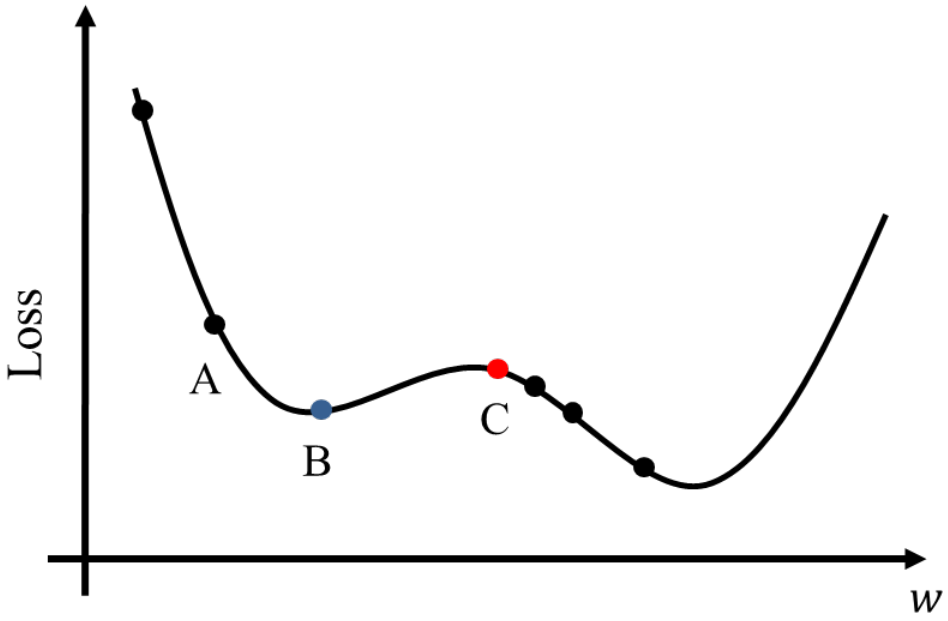
缺點：不同的權重，卻共用同樣的學習速率，造成梯度下降時的震盪

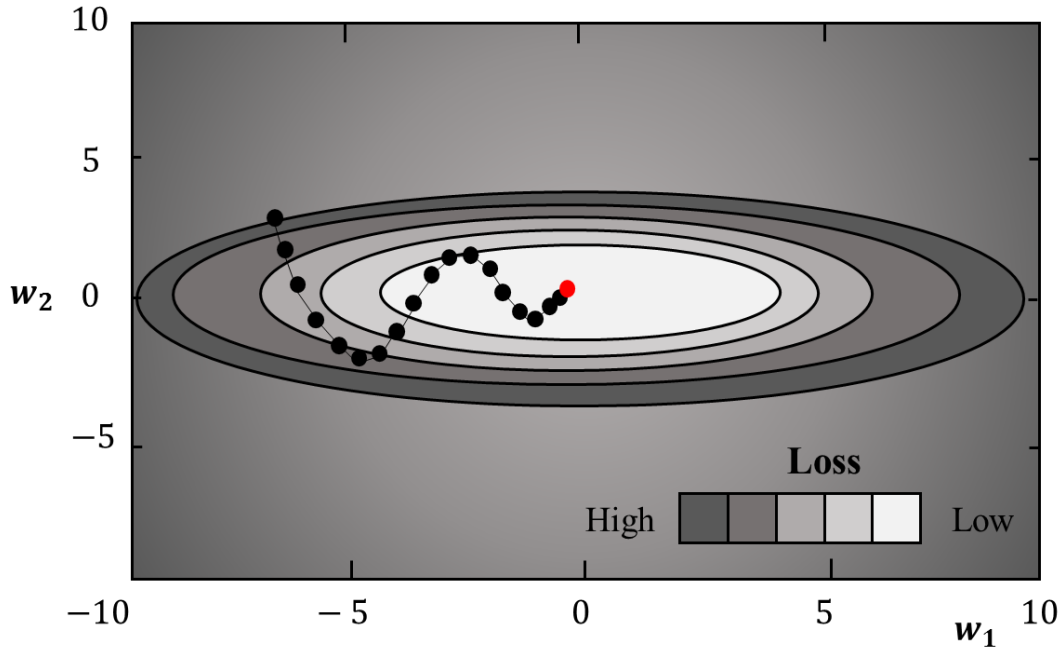
# 梯度下降法的優化　02. 動量的梯度下降法

## 動量的梯度下降法 (Momentum)

$$w^t = w^{t-1} + v^t \qquad v^t = \alpha v^{t-1} - \eta \frac{\partial L}{\partial w^{t-1}}$$

梯度下降法增加物體加速度的物理概念，即使在B點梯度為0，即$\frac{\partial L}{\partial w_B}$因為前個$\alpha v_A$造成繼續往前的動力，使下次權重到達C點，有機率可以越過局部最低點



# 梯度下降法的優化 03. AdaGrad



## AdaGrad

$$w^t = w^{t-1} - \frac{\eta^{t-1}}{\sigma^{t-1}} \frac{\partial L}{\partial w^{t-1}}$$

其中 $\eta^{t-1} = \frac{\eta}{\sqrt{t}}$ $\sigma^{t-1} = \sqrt{\frac{1}{t}\sum_{i=0}^{t-1}(\frac{\partial L}{\partial w_{t-1}})^2}$

每個權重使用同個學習速率會造成震盪，所以為了解決這方法，AdaGrad 方法是將學習速率逐漸降低，稱為「學習速率衰減(Learning rate decay)」，並針對每個參數準備客製值

# 梯度下降法的優化 03. AdaGrad

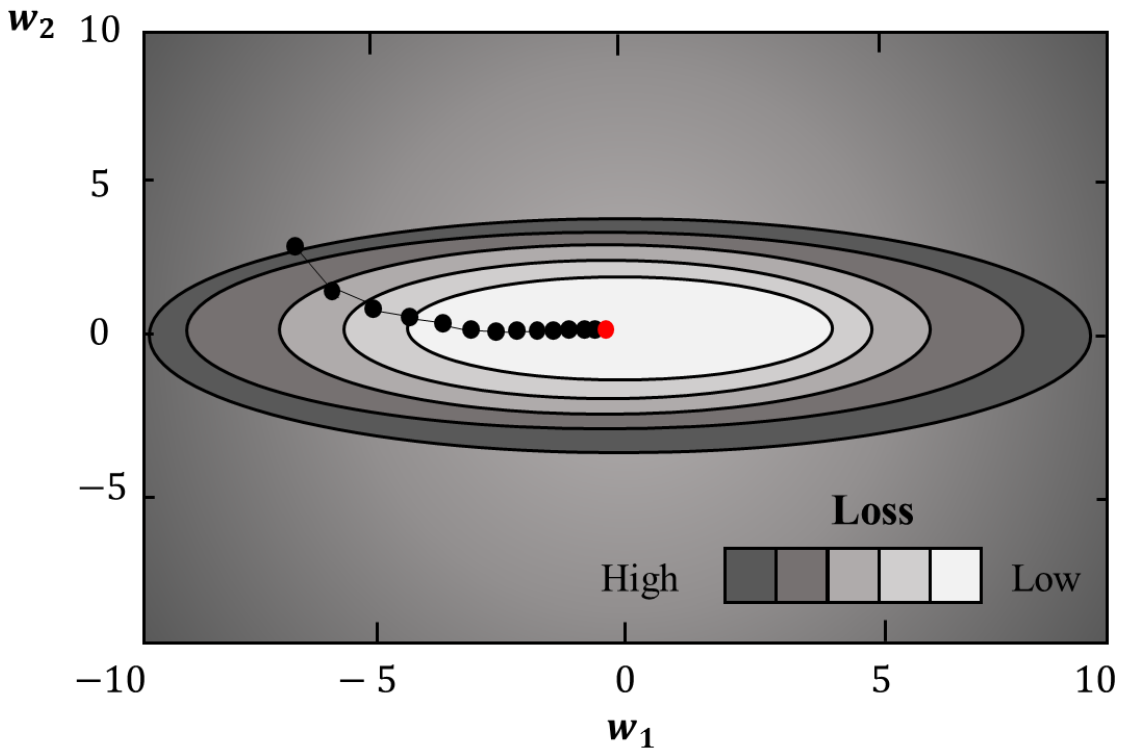
## AdaGrad

$$w^t = w^{t-1} - \frac{\eta^{t-1}}{\sigma^{t-1}} \frac{\partial L}{\partial w^{t-1}} \qquad \eta^{t-1} = \frac{\eta}{\sqrt{t}} \qquad \sigma^{t-1} = \sqrt{\frac{1}{t}\sum_{i=0}^{t-1}(\frac{\partial L}{\partial w_{t-1}})^2}$$

$$\Rightarrow w^t = w^{t-1} - \frac{\eta}{\sqrt{\sum_{i=0}^{t-1}(\frac{\partial L}{\partial w_{t-1}})^2}} \frac{\partial L}{\partial w^{t-1}}$$

一開始的初始點離最低點較遠，可讓學習率比較大，之後學習率隨著訓練時間t的增加而不斷變小，加速訓練時間

缺點：$\sigma$過大會整個梯度會被拘束趨近於0，導至訓練提前結束



# 梯度下降法的優化 04. RMSProp



## RMSProp

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sigma^t} g^t \qquad \sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1-\alpha)(g^t)^2}$$

AdaGrad算法中的$\sigma$為過去所有累加平方梯度，容易造成提前訓練結束。

RMSProp算法是加了一個衰減係數來控制前一個梯度信息的獲取多少，

並可以調控$\alpha$控制舊的梯度佔比資訊

# 梯度下降法的優化 05. Adam

## Adam (Adaptive Moment Estimation)

Adam 為加入了動量概念的 RMSprop，且在更新梯度過程中考慮了偏差校正 ( bias-correction )

$$w_i^{t+1} \leftarrow w_i^t - \eta \cdot \frac{\hat{m}^t}{\sqrt{\hat{v}^t} + \epsilon}$$

其中 $m^{t+1} = \beta_1 \cdot m^t + (1 - \beta_1) \cdot g^t$

$$v^{t+1} = \beta_2 \cdot v^t + (1 - \beta_2) \cdot (g^t)^2$$

$$\hat{m}^t = \frac{m^t}{1 - \beta_1} \qquad \hat{v}^t = \frac{v^t}{1 - \beta_2} \qquad g^t = \frac{\partial L}{\partial w_i}$$

# 梯度下降法的優化 06. 演算法的比較



## 梯度下降演算法的範例比較