

# Last session

- ▶ GD - SGD.
- ▶ MLP.
- ▶ Backprop.
- ▶ Multiple outputs.
- ▶ Activation functions.

# Today's outline

- ▶ Code revision.
- ▶ Paper 1 discussion.
- ▶ Paper 2 discussion.
- ▶ Optimization functions.
- ▶ New code.

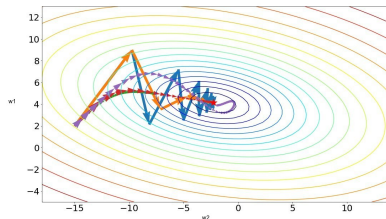
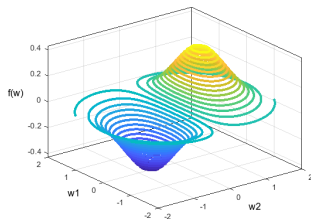
# Outline

Contour plots

Optimization functions

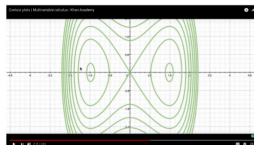
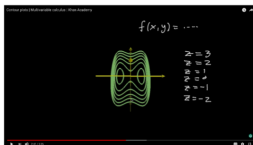
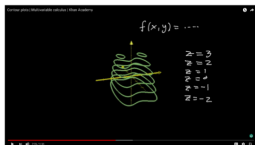
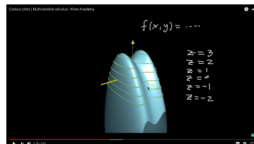
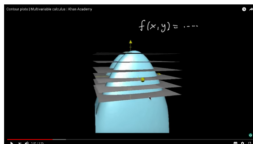
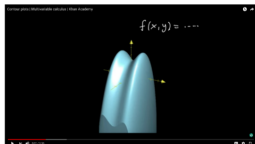
# Parameter space

Example of different loss functions with 2 parameters.  
Seen in 3D, and from above.



Colored arrows represent different possible paths to reach the local minimum, as followed by different optimization strategies.

# Several local minima



Images from Video: Contour plots - Multivariable calculus - Khan Academy  
<https://www.youtube.com/watch?v=WsZj5Rb6do8>

# Example for SGD



- Batch gradient descent
- Mini-batch gradient Descent
- Stochastic gradient descent

# SGD Limitation

$$\omega_i = \omega_i - \nabla_{\omega_i} \mathcal{L}(y, \hat{y}).$$

- ▶ Slows down around ravines.
- ▶ Oscillates across the slopes of the ravine.
- ▶ Limited progress towards the local minimum.
- ▶ Might never escape from saddle points.

Qian, 1999. “On the momentum term in gradient descent learning algorithms”.

# Outline

Contour plots

Optimization functions

## Parámetros

Parámetros:  $w$ , aprendidos automáticamente durante entrenamiento

Hyperparámetros: ajustados manualmente mediante validación.

Ej: tasa aprendizaje  $\eta$   
optimizador  
función de costo

# capas y neuronas

# épocas

activaciones no lineales

tamaño del lote



# Notation

In this section, we will use subscripts to index time, e.g.,  $\omega_t$  refers to the value of parameter  $\omega$  at time  $t$ .

Also, we omit the position index ( $i$ -th element) commonly seen as subscript, i.e,  $\omega_i$ .

# Momentum

$$\omega_t = \omega_{t-1} - v_t,$$

$$v_t = \gamma v_{t-1} + \eta \nabla_{\omega} \mathcal{L}(y, \hat{y}), \quad v_0 = 0, \quad \gamma = 0.9, \eta \approx 0.001.$$

- ▶ Accelerates SGD in the relevant direction.
- ▶ Dampens oscillations.
- ▶ Includes a fraction of the historic direction.
- ▶ Momentum accelerates for gradients pointing in the same direction, and reduces for those in changing direction.

Sutskever et al., 2013. “On the importance of initialization and momentum in deep learning”.

# Nesterov Accelerated Gradient (NAG)

$$\omega_t = \omega_{t-1} - v_t,$$

$$v_t = \gamma v_{t-1} + \eta \nabla_{(\omega - \gamma v_{t-1})} \mathcal{L}(y, \hat{y}),$$

- ▶  $\nabla_{(\omega - \gamma v_{t-1})}$  approximates the next position of  $\omega$ .
- ▶ Looks ahead by calculating the gradient w.r.t. future positions.
- ▶ Anticipates changes in the direction of the gradient.

Nesterov, 1983. "A method for unconstrained convex minimization problem with the rate of convergence  $\mathcal{O}(1/k^2)$ ".

# Adaptive Gradient (AdaGrad)

$$\omega_t = \omega_{t-1} - \eta \frac{1}{\sqrt{G_t + \epsilon}} g_t, \quad g_t = \nabla_{\omega} \mathcal{L}(y, \hat{y}),$$

$$G_t = \sum_{k=0}^t g_k^2, \quad \epsilon \approx 1e^{-8}.$$

- ▶  $G_t$ : sum of gradients<sup>2</sup> up to time  $t$  (heavy memory loads).
- ▶ Adapts  $\eta$  at each time step (always decreasing).
- ▶ Works well on sparse data and large models.

Duchi et al., 2011. “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”.

# Adadelta

$$\omega_t = \omega_{t-1} - \eta \frac{1}{\sqrt{\mathbb{E}[g^2]_t}} g_t, \quad g_t = \nabla_{\omega} \mathcal{L}(y, \hat{y}),$$
$$\mathbb{E}[g^2]_t = \gamma \mathbb{E}[g^2]_{t-1} + (1 - \gamma) g_t^2, \quad \gamma = 0.9.$$

- ▶ Addresses the issue of monotonically decreasing  $\eta$ .
- ▶ Restricts the past to a moving window.
- ▶ Recursively computes the sum of past gradients using exponential smoothing.

Zeiler, 2012. “ADADELTA: An Adaptive Learning Rate Method”.  
RMSprop: a variant by Hinton (unpublished).

## Adaptive Momentum (Adam)

Adadelta + Momentum.

$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$ , first moment estimate (mean),  
 $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ , second moment estimate (stddv).

Correcting for bias towards zero:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}; \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

$$\omega_t = \omega_{t-1} - \eta \frac{1}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t,$$

Kingma & Ba, 2015. "Adam: a Method for Stochastic Optimization".

Other variants: AdaMax, Nadam, AMSGrad.

## To know more

Ruder, 2016. "An overview of gradient descent optimization algorithms". <https://arxiv.org/abs/1609.04747>

Recette:

• Intender con Adam, SGD con momentum