Νευρο-Ασαφής Υπολογιστική
Neuro-Fuzzy Computing

Διδάσκων –

Δημήτριος Κατσαρός

@ Τμ. ΗΜΜΥ

Πανεπιστήμιο Θεσσαλίας

Steepest descent

The algorithm (1/3)

Αλγόριθμος 4.2. Μέθοδος της μεγαλύτερης αλλαγής, Cauchy

Εστω μία πραγματική συνάρτηση $f \in C^1$ στο $E \subseteq \Re^n$. Για την εύρεση ενός σημείου $\mathbf{x}^*$ το οποίο δίνει τοπικό ελάχιστο μέσα στο E επιλέγουμε μία αρχική προσέγγιση $\mathbf{x}_0 \in E$ και δημιουργούμε μία ακολουθία σημείων $\{\mathbf{x}_k\}$ η οποία συγκλίνει στο βέλτιστο σημείο. Για να πάμε από το σημείο $\mathbf{x}_k$ στο $\mathbf{x}_{k+1}$ ακολουθούμε την εξής διαδικασία:

Βήμα 1: Υπολογίζουμε τη κλίση $\nabla f(\mathbf{x}_k)$

Βήμα 2: Υπολογίζουμε τη διεύθυνση μετάβασης $\mathbf{s}_k = -\dfrac{\nabla f(\mathbf{x}_k)}{\|\nabla f(\mathbf{x}_k)\|}$

The algorithm (23)

<u>**Βήμα 3:**</u> Λύουμε το πρόβλημα $\underset{\lambda_k \geq 0}{\text{Minimize}}\ f(\mathbf{x}_k + \lambda_k \mathbf{s}_k)$ για να βρούμε το βήμα λ_k. Η λύση μπορεί να βρεθεί

α) Από την $$\frac{d f\left(\mathbf{x}_k - \lambda_k \dfrac{\nabla f(\mathbf{x}_k)}{\left\|\nabla f(\mathbf{x}_k)\right\|}\right)}{d\lambda_k} = 0$$

β) Από την εφαρμογή κάποιας μονοδιάστατης μεθόδου (π.χ. χρυσών τομών)

The algorithm (3/3)

Βήμα 4: Εντοπίζουμε το επόμενο σημείο $\mathbf{x}_{k+1} := \mathbf{x}_k + \lambda_k \mathbf{s}_k$

Βήμα 5: Ελέγχουμε κριτήριο σύγκλισης. Αν είναι αληθές τέλος, διαφορετικά ανακυκλώνουμε. ◊

Summary of gradient descent

- The gradient of a function is a vector perpendicular to the contour of f(x) that passes from x_0, and defines the direction of maximum local increase of f(x) at this point (steepest ascent direction)
- Thus looking for the optimal x*, while having an approximation x_k, imposes us to search along the direction which is opposite to the direction of $-\nabla f(x_k)$
- Unit vector along this direction is: $\dfrac{-\nabla f(x_k)}{||\nabla f(x_k)||}$
- Therefore, each step is: $x_{k+1} = x_k + \lambda_k s_k$, i.e.,

$$x_{k+1} = x_k + \lambda_k\Big(-\frac{\nabla f(x_k)}{||\nabla f(x_k)||}\Big) = x_k - \lambda_k\Big(\frac{\nabla f(x_k)}{||\nabla f(x_k)||}\Big)$$

An example of gradient descent

- We wish to find the minimum of $f(x)= x_1^2+5x_2^2+x_3^2 - 4$
- Let us, start with the initial guess $x_0=(2,2,2)^T$
- The gradient is: $\nabla f(x) = \begin{pmatrix} 2x_1 \\ 10x_2 \\ 2x_3 \end{pmatrix}$
- Thus, $||\nabla f(x)|| = \sqrt{\nabla^T f(x) \nabla f(x)} = 2\sqrt{x_1^2 + 25x_2^2 + x_3^2}$
- At $x_0=(2,2,2)^T$, the above equation yields: $2\sqrt{108}$
- Direction of descending: $\dfrac{-\nabla f(x_0)}{||\nabla f(x_0)||} = -\dfrac{1}{2\sqrt{108}} \begin{pmatrix} 4 \\ 20 \\ 4 \end{pmatrix} = -\dfrac{1}{\sqrt{108}} \begin{pmatrix} 2 \\ 10 \\ 2 \end{pmatrix}$
- Thus, $x_1 = (2,2,2)^T - \lambda_0 \dfrac{1}{\sqrt{108}} (2,10,2)^T$

$$x_1 = \left(2 - \frac{2\lambda_0}{\sqrt{108}}, 2 - \frac{10\lambda_0}{\sqrt{108}}, 2 - \frac{2\lambda_0}{\sqrt{108}}\right)$$

An example of gradient descent

- Recall, $f(x)= x_1^2+5x_2^2+x_3^2-4$
- Thus $$f(x_1) = \left(2 - \frac{2\lambda_0}{\sqrt{108}}\right)^2 + 5\left(2 - \frac{10\lambda_0}{\sqrt{108}}\right)^2 + \left(2 - \frac{2\lambda_0}{\sqrt{108}}\right)^2 - 4$$
- i.e., $$f(x_1) = 2\left(2 - \frac{2\lambda_0}{\sqrt{108}}\right)^2 + 5\left(2 - \frac{10\lambda_0}{\sqrt{108}}\right)^2 - 4$$
- What is the optimal λ_0;
  - Take the derivative of the previous equation relative to λ_0 and set to 0

$$\frac{df(x_1)}{d\lambda_0} = 4\left(2 - \frac{2\lambda_0}{\sqrt{108}}\right)\left(-\frac{2}{\sqrt{108}}\right) + 10\left(2 - \frac{10\lambda_0}{\sqrt{108}}\right)\left(-\frac{10}{\sqrt{108}}\right) = 0$$

  - which leads to: $\frac{254}{\sqrt{108}}\lambda_0 = 54 \implies \lambda_0 \approx 2.21$
  - Therefore: $x_1 = (1.575, -0.127, 1.575)^T$

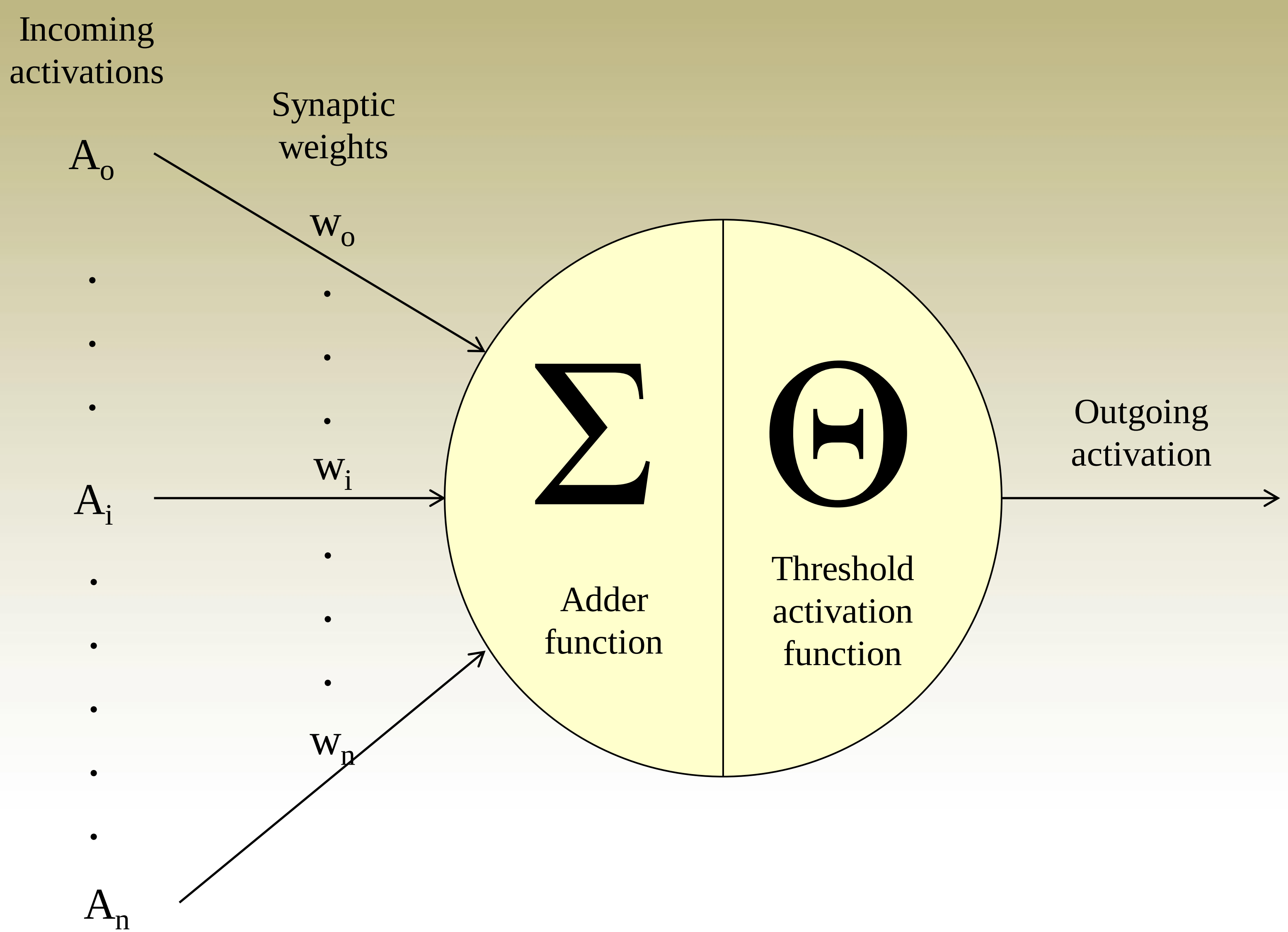
An example of gradient descent

k	x_k^T	λ_k	$\nabla^T f(x_k)$	$f(x_k)$
0	(2,2,2)	2.2100	(4, 20, 4)	24
1	(1,575,-0.127, 1.575)	1.7807	(3.15, -1.27, 3.15)	1.042
2	(0.364, 0.361, 0.364)	0.5100	(0.728, 3.61, 0.728)	-3.08
3	(0.265, -0.13, 0.265)	0.1824	(0.53, -1.3, 0.53)	-3.775
4	(0.190, 0.032, 0.190)	0.1550	(0.38, 0.32, 0.38)	-3.923
5	(0.090, -0.04, 0.090)	0.0610	(0.18, -0.4, 0.18)	-3.976
6	(0.060, 0.010, 0.060)		(0.12, 0.1, 0.12)	-3.999

- It holds that $x^*=(0,0,0)^T$ and
- opt value $f(x^*)= -4$

The Perceptron network

Back to single-node perceptron



The reward-punishment concept

- Given two training sets belong to two classes ω_1 and ω_2, respectively
- let w(1) represent the initial weight vector, which may be arbitrary chosen
- c, is a <u>positive</u> constant **(should it be?)**
- Then, at the k-th training step, we execute:

- If $x(k) \in \omega_1$ and $w'(k)x(k) \leq 0$, replace w(k) by

  $$w(k+1)= w(k) + cx(k)$$
- If $x(k) \in \omega_2$ and $w'(k)x(k) \geq 0$, replace w(k) by

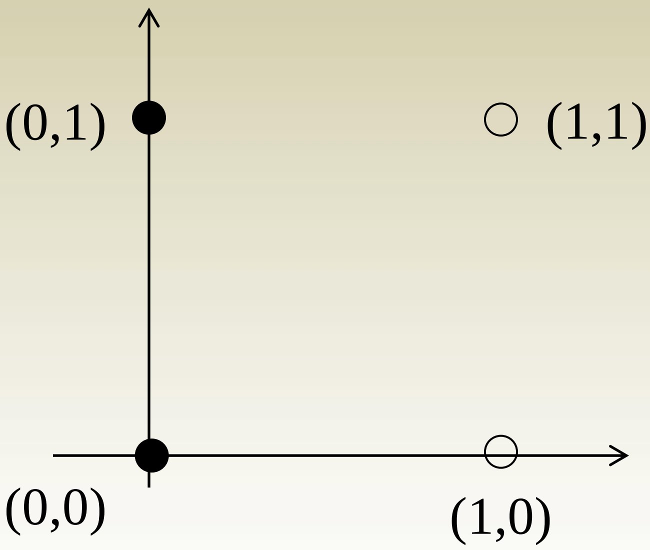
  $$w(k+1)= w(k) - cx(k)$$
- Otherwise,

  $$w(k+1)= w(k)$$

- Convergence occurs when perceptron produces correct output for each and every input

Workout training example

- Train a single-node perceptron to learn the following two-class case:
  - learning rate c=1, threshold=0, and w(1)=(0 0 0)’



Recall the Gradient (Steepest) Descent for finding (a/the) minimum of a function

- Recall that the gradient of a function f with respect to a vector $y=(y_1,y_2,\ldots,y_n)'$ is defined as

$$grad\ f(y) = \frac{df(y)}{dy} = \begin{pmatrix} \frac{\partial f}{\partial y_1} \\ \frac{\partial f}{\partial y_2} \\ \vdots \\ \frac{\partial f}{\partial y_n} \end{pmatrix}$$

- It points to the direction of the maximum rate of increase of the function f, when the argument increases

Recall the Gradient (Steepest) Descent for finding (a/the) minimum of a function

- We will consider functions with a unique minimum
- Consider the criterion function

$$J(w,x) = (|w'x| - w'x)$$

- Apparently, its minimum is J(w,x)=0
- Thus, we increment w in the direction of negative gradient of J(w,x)
- If we let w(k) represent the value of w at the k-th step, the gradient descent can be written:

$$w(k+1) = w(k) - c\left\{\frac{\partial J(w,x)}{\partial w}\right\}_{w=w(k)}$$

Perceptron algorithm as a case of Gradient Descent-based optimization

- Let the criterion function be

$$J(w,x) = \frac{1}{2}(|w'x| - w'x)$$

- It holds that $\frac{\partial J}{\partial w} = \frac{1}{2}[(x\ sgn(w'x) - x)]$

- Substituting the last equation into the last equation of the previous slide, we get:

$$w(k+1) = w(k) + \frac{c}{2}\{x(k) - x(k)\ sgn(w'(k)x(k))\}$$

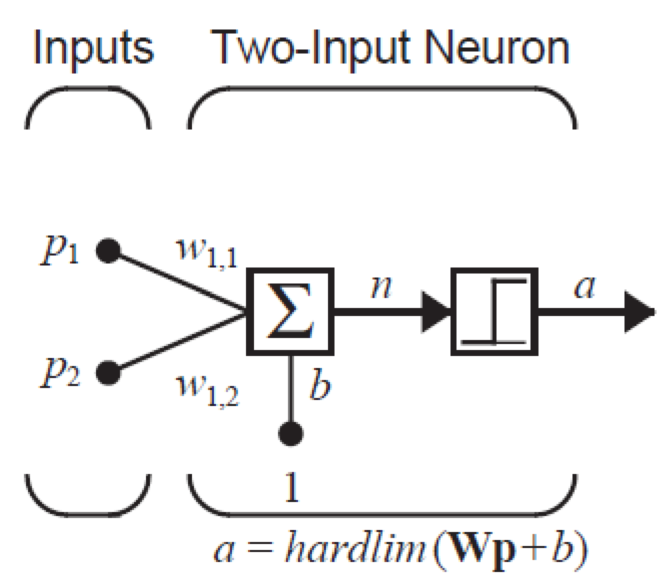
- From the definition of the sign (sgn) function:

$$w(k+1) = w(k) + c\begin{cases} 0, & \text{if } w^{'}(k)x(k) > 0 \\ x(k), & \text{if } w^{'}(k)x(k) \leq 0 \end{cases}$$

Decision boundary and the perpendicular vector

Decision boundary and vectors

- Consider the following neural network:



- So, a = hardlim(n) = hardlim(**Wp**+b) = hardlim($w_{1,1}$ p_1 + $w_{1,2}$ p_2 + b)
- The decision boundary is determined by those net inputs that satisfy: $w_{1,1}$ p_1 + $w_{1,2}$ p_2 + b = 0
- The weights define a decision boundary, and vector **W** is perpendicular to that boundary

Basic geometric concepts

- A line L is defined: $Ax + By + \Gamma = 0, \ |A| + |B| \neq 0$

It holds that:

  - This line is parallel to vector $\delta_1(B, -A)$
  - This line is perpendicular to vector $\delta_2(A, B)$

PROOF

- Condition for parallelism
  - If $B = 0$, then L // y'y axis, and $\delta_1(0, -A)$ // y'y axis. Thus, L // δ_1
  - If $B \neq 0$, then $slope_{\delta 1} = -A/B$ and $slope_L = -A/B$. Thus, L // δ_1
- Condition for perpendicularity
  - We know that if dot product$(\delta_1, \delta_2)=0$, then $\delta_1 \perp \delta_2$
    - Indeed, $\delta_1 \bullet \delta_2 = B * A + (-A) * B = 0$

Basic geometric concepts: Example

- Consider the line: 3x - 4y + 12 = 0
  - So, A= 3, B= -4, and Γ= 12