Mạng neuron
và ứng dụng trong xử lý tín hiệu

Giảng viên

Trần Thị Thanh Hải

International Research Institute MICA

Multimedia, Information, Communication & Applications

UMI 2954

Hanoi University of Science and Technology

1 Dai Co Viet - Hanoi - Vietnam

Bài 5:

Tối ưu hóa hàm mục tiêu

Mục tiêu

- **Câu hỏi của bài 4:**
  - Làm thế nào để xác định được ma trận trọng số khi số đầu vào lớn
  - Các giải thuật lặp cho phép giải bài toán tối ưu hàm mục tiêu
- **Mục tiêu**
  - Giới thiệu hai giải thuật học
    - Thuật toán bước giảm cực đại (Gradient Descent Learning)
    - Thuật toán L-M (Levenberg Marquardt)
  - Thực hành với một trong số các thuật toán trên

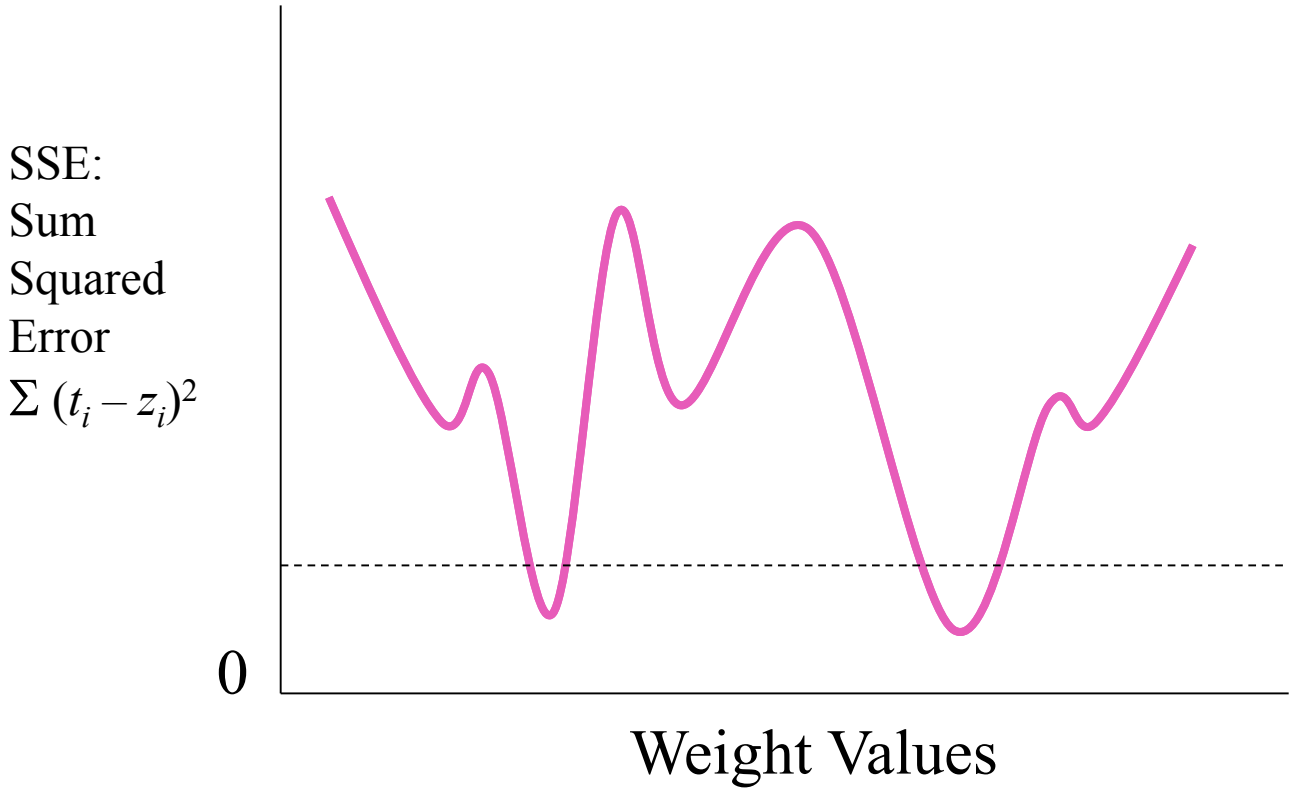
Nguyên lý chung

- Giả sử tập số liệu học có p mẫu: $\{\mathbf{x}_i, d_i\}$ với $i = 1, \ldots, p$
- Với mỗi vector đầu vào x_i, giá trị đích cần đạt $f(x_i) \sim d_i$ với mọi giá trị của I
- Nguyên lý của học: tối thiểu hóa sai số giữa các đầu ra của mạng $f(x_i)$ và giá trị đích cần đạt d_i
- Thông thường người ta gọi là tối thiểu hóa **hàm mục tiêu (objective function)**

$$E = \frac{1}{2}\sum_{i=1}^{p}\left(f\left(\mathbf{x}_i\right) - d_i\right) \rightarrow \min$$

Nguyên lý chung

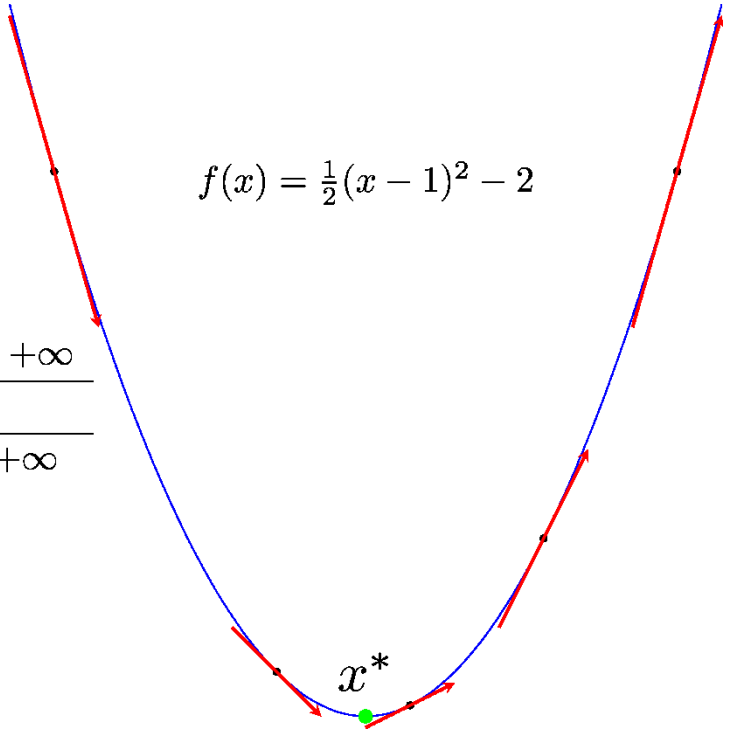
- **Tối ưu hóa hàm mục tiêu (objective function)**



Nguyên lý chung

Tiếp tuyến với đồ thị hàm số tại một điểm có hệ số góc bằng đạo hàm tại điểm đó

x	$-\infty$		1		$+\infty$
$f'(x)$		$-$	0	$+$	
$f(x)$	$+\infty$	$\searrow$	-2	$\nearrow$	$+\infty$



Điểm cực tiểu là điểm mà tại đó đạo hàm của f(x) có giá trị bằng 0

Nguyên lý chung

- **Trong đại đa số các trường hợp, tìm x_{min} sao cho f'(x) = 0 là không khả thi với f là một hàm bất kỳ bởi các nguyên nhân:**
  - Sự phức tạp của dạng của đạo hàm
  - Dữ liệu có số chiều lớn (high dimension)
  - Kích thước của tập dữ liệu lớn (large dataset)

Hướng tiếp cận

- **Nội dung của phương pháp như sau:**
  - Xuất phát từ một vị trí khởi đầu $x^{(0)}$ ta đi tìm $x^{(1)}$ sao cho $f(x^{(1)}) < f(x^{(0)})$
  - Tiếp tục tìm $x^{(2)}$ sao cho $f(x^{(2)}) < f(x^{(1)})$
  - Liên tục như vậy ta sẽ thu được kết quả là một chuỗi $\{x^{(t)}\}$ sao cho $\{f(x^{(t)})\}$ là chuỗi giảm dần và sẽ đạt tới một cực tiểu nào đó
  - Khi đó: $x^{(t)} \xrightarrow[t \to \infty]{} x_{\min}$

Thuật toán Gradient Descent

- Công thức này là hệ quả của phép triển khai Taylor:

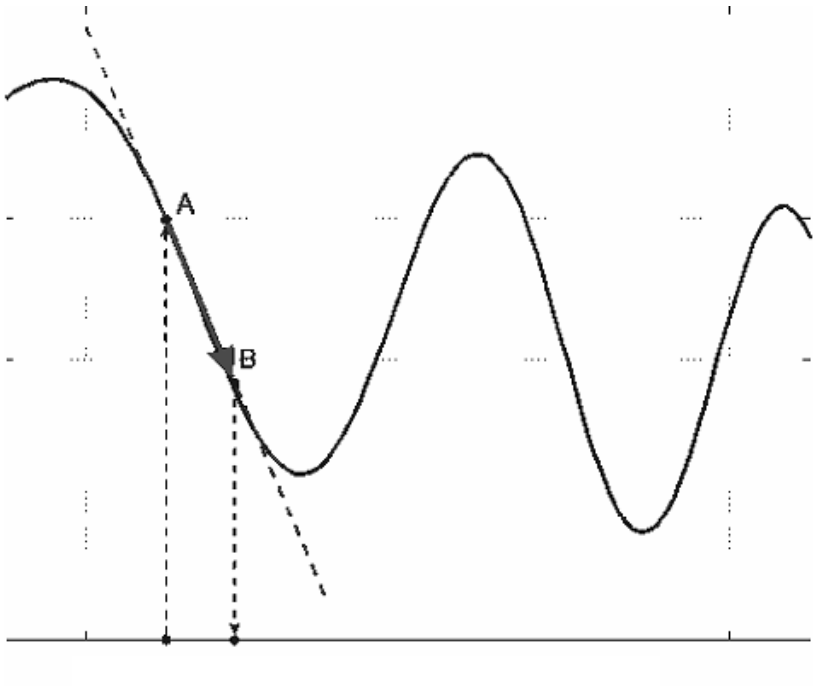
$$f\left(x^{(t)}+\Delta\right)\approx f\left(x^{(t)}\right)+\Delta\cdot f'\left(x^{(t)}\right)+\mathrm{O}\left(\Delta^2\right)$$

- Nếu chọn: $\Delta=-\eta\cdot f'\left(x^{(t)}\right)$ với eta đủ nhỏ thì:

$$f\left(x^{(t)}+\Delta\right)\approx f\left(x^{(t)}\right)-\eta\cdot\left(f'\left(x^{(t)}\right)\right)^2+\mathrm{O}\left(\Delta^2\right)<f\left(x^{(t)}\right)$$

- Nếu chọn: $x^{(t+1)}=x^{(t)}+\Delta=x^{(t)}-\eta\left.\frac{\partial f}{\partial x}\right|_{x=x^{(t)}}$ thì giá trị tại $x^{(t+1)}$ sẽ giảm

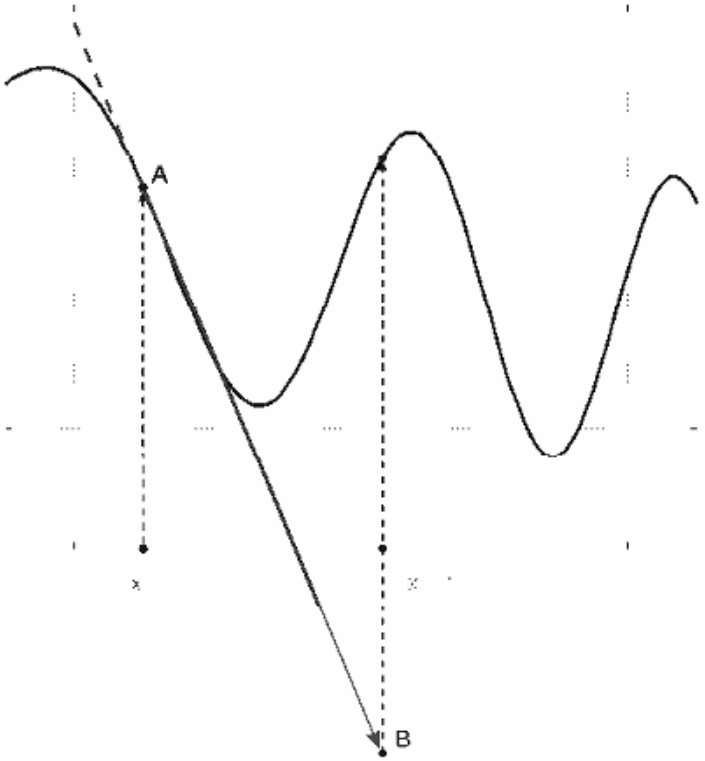
Thuật toán học



$$x^{(t+1)} = x^{(t)} - \eta \left. \frac{\partial f}{\partial x} \right|_{x=x^{(t)}}$$

Trong đó eta là tốc độ học của mạng

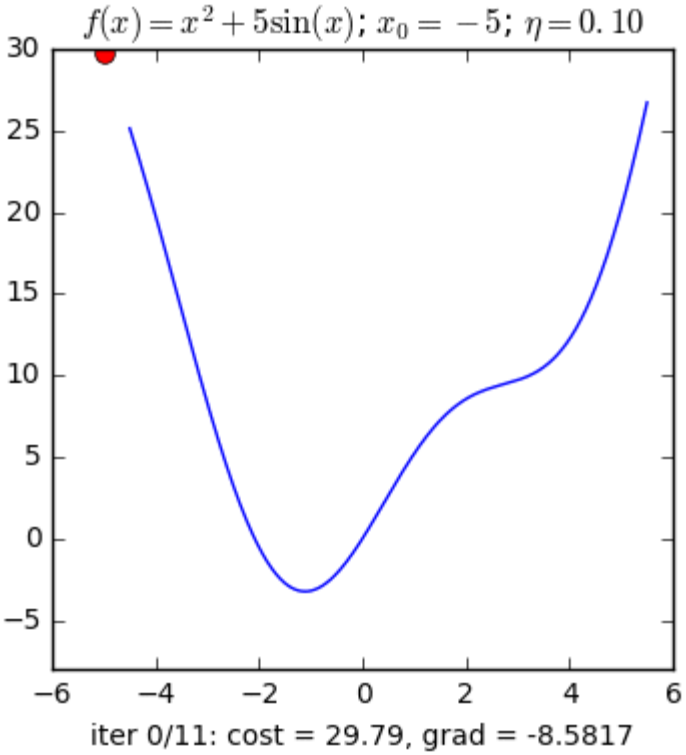
Thuật toán học

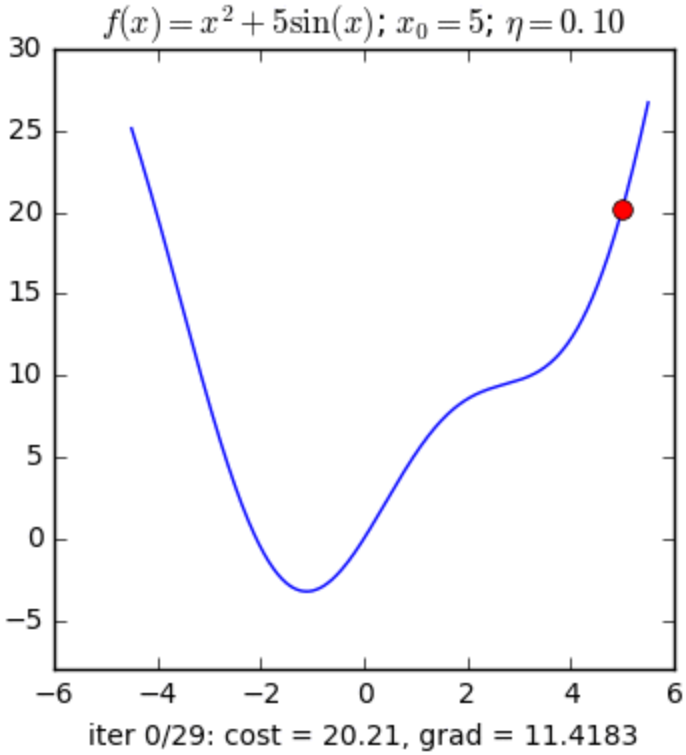


$$x^{(t+1)} = x^{(t)} - \eta \left.\frac{\partial f}{\partial x}\right|_{x=x^{(t)}}$$

Nếu eta lớn quá, có thể bỏ qua cực trị và không tốt cho quá trình update mạng

Hướng tiếp cận





Thuật toán Gradient Descent Learning

- **Mục đích:** giảm thiểu lỗi của mạng mỗi khi hệ số mạng thay đổi
- **Hàm mục tiêu:** $E = \Sigma (t_i - z_i)^2$
- Tìm một giải thuật thay đổi trọng số sao cho đạo hàm của hàm mục tiêu theo trọng số là âm

$$\frac{\partial E}{\partial w_{ij}} < 0$$

- **Chú ý do phương pháp này sử dụng đạo hàm nên hàm ngưỡng không phù hợp để sử dụng pp này**

Sử dụng Gradient Descent cho 1 neuron

- **Bộ số liệu:** p mẫu $\{\mathbf{x}_i, d_i\}, i = 1, \ldots p,$
- **Đầu ra tương ứng với từng đầu vào**

$$y_i = f\left(\sum_{j=0}^{N} W_j x_{ij}\right)$$

- **Sai số trên bộ dữ liệu học** $E = \frac{1}{2}\sum_{i=1}^{p}(y_i - d_i)^2$
- **Ta cần tìm các trọng số:** $W_j \ (j = 0,1,\ldots,N)$
- **Với các giá trị khởi tạo**

$$W_j^{(t+1)} = W_j^{(t)} - \eta \left.\frac{\partial E}{\partial W_j}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(t)}}$$

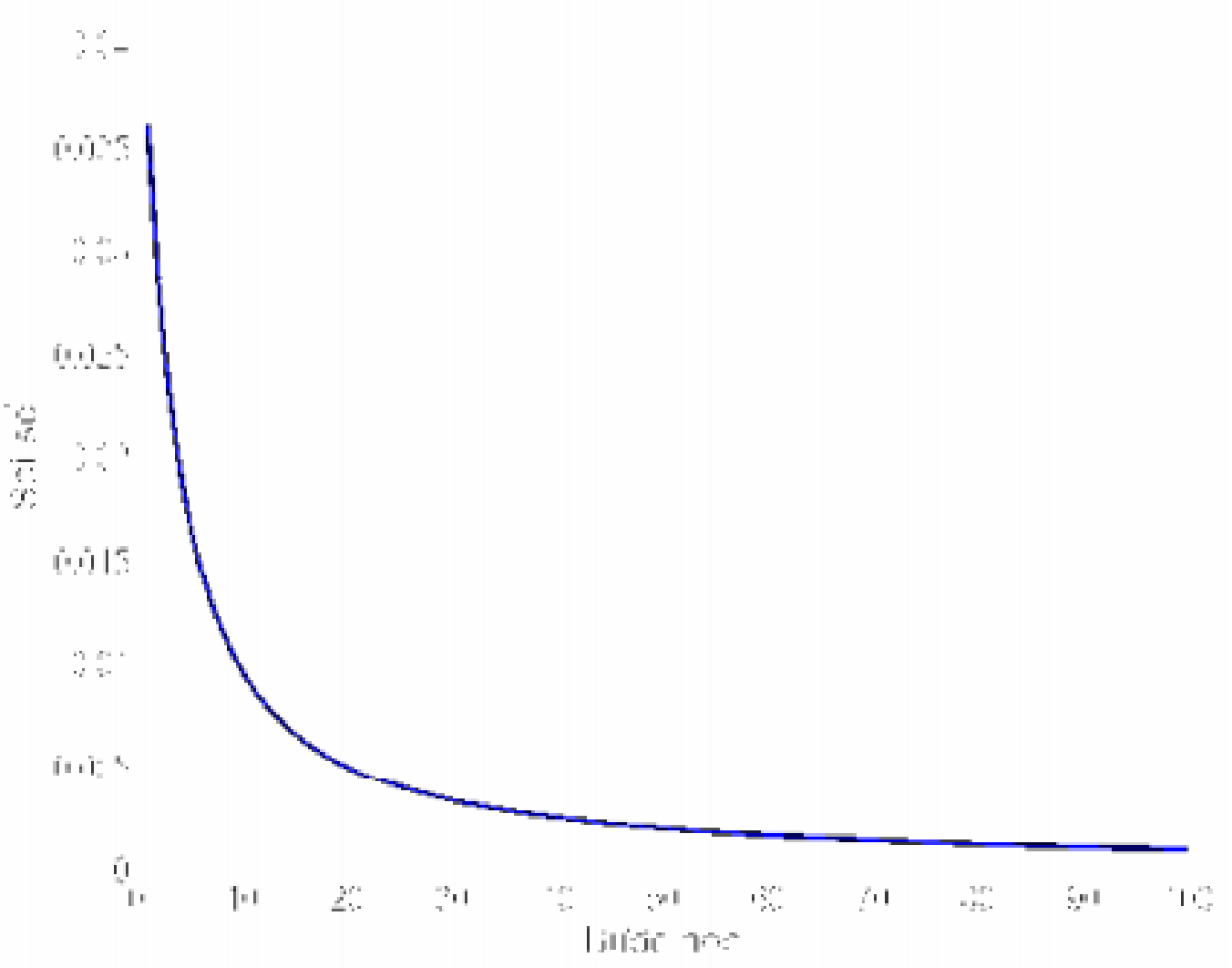
Sử dụng Gradient Descent cho 1 neuron

$$\frac{\partial E}{\partial W_\alpha} = \sum_{i=1}^{p} (y_i - d_i) \frac{\partial y_i}{\partial W_\alpha}$$

$$= \sum_{i=1}^{p} (y_i - d_i) f' \left(\sum_{j=0}^{N} W_j x_{ij} \right) \frac{\partial \left(\sum_{j=0}^{N} W_j x_{ij} \right)}{\partial W_\alpha}$$

$$= \sum_{i=1}^{p} (y_i - d_i) f' \left(\sum_{j=0}^{N} W_j x_{ij} \right) x_{i\alpha}$$

Đồ thị hàm sai số theo bước lặp



Thuật toán L-M

- Thuật toán Levenberg-Marquardt (L-M) dựa trên triển khai bậc 2 của khai triển Taylor

$$E(\mathbf{W}+\mathbf{p}) = E(\mathbf{W}) + [\mathbf{g}(\mathbf{W})]^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \mathbf{H}(\mathbf{W})\mathbf{p} + O(\mathbf{p}^3)$$

- Với p: là khoảng lân cận trong khai triển
- g(W): là vector gradient của E theo W

$$\mathbf{g}(\mathbf{W}) = \nabla \mathbf{E} = \left[\frac{\partial E}{\partial W_1}, \frac{\partial E}{\partial W_2}, \ldots, \frac{\partial E}{\partial W_n}\right]^T$$

- H(W) là ma trận đạo hàm bậc 2:

$$\mathbf{H}(\mathbf{W}) = [H_{ij}] = \begin{bmatrix} \frac{\partial^2 E}{\partial W_1 \partial W_1} & \cdots & \frac{\partial^2 E}{\partial W_n \partial W_1} \\ \vdots & & \vdots \\ \frac{\partial^2 E}{\partial W_1 \partial W_n} & \cdots & \frac{\partial^2 E}{\partial W_n \partial W_n} \end{bmatrix}$$

Thuật toán L-M

- Tại điểm cần tìm g(W) = 0 và H(W) xác định dương
- Giả sử: $\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \mathbf{p}$ $\frac{\partial E\left(\mathbf{W}^{(t+1)}\right)}{\partial \mathbf{p}} = \frac{\partial E\left(\mathbf{W}^{(t)} + \mathbf{p}\right)}{\partial \mathbf{p}} \approx 0$

$$\mathbf{g}\left(\mathbf{W}^{(t)}\right) + \mathbf{H}\left(\mathbf{W}^{(t)}\right)\mathbf{p}^{(t)} = 0$$

- Khi đó:

$$\mathbf{p}^{(t)} = -\left[\mathbf{H}\left(\mathbf{W}^{(t)}\right)\right]^{-1}\mathbf{g}\left(\mathbf{W}^{(t)}\right)$$

- Để tránh bước dịch chuyển quá lớn:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \eta\left[\mathbf{H}\left(\mathbf{W}^{(t)}\right)\right]^{-1}\mathbf{g}\left(\mathbf{W}^{(t)}\right)$$

Thuật toán L-M

- Về cơ bản: nếu ta sử dụng triển khai Taylor ở bậc cao hơn thì sẽ có tốc độ hội tụ nhanh hơn
- Tuy nhiên, các công thức sẽ phức tạp và cồng kềnh
- Vì vậy trong thực tế, ta sẽ chỉ sử dụng đến các triển khai bậc 2

Ví dụ về toán tử OR

- $x_1 = 1; x_2 = 1$ then $y = 1$

$$[\mathbf{W}]^{(0)} = \left[W_0^{(0)} \quad W_1^{(0)} \quad W_2^{(0)} \right] = [-0,2 \quad 0,7 \quad 0,5]$$

- Nếu neuron có hàm truyền logsig

$$y^{(0)} = f\left(W_0^{(0)} x_0 + W_1^{(0)} x_1 + W_2^{(0)} x_2\right) = f(1,0) = 0,731$$

- Sai số ban đầu:

$$E^{(0)} = \frac{1}{2}\left(y^{(0)} - d\right)^2 = \frac{1}{2}(0,731 - 1)^2 = 0,0362$$

Ví dụ về toán tử OR

- **Các giá trị gradient**

$$\left.\frac{\partial E}{\partial W_0}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = \left(y^{(0)} - d\right) f'\left(W_0^{(0)} + W_1^{(0)} x_1 + W_2^{(0)} x_2\right) x_0 = \left(y^{(0)} - d\right) y^{(0)} (1 - y^{(0)}) x_0$$

$$= (0,731 - 1) \cdot 0,731 \cdot (1 - 0,731) \cdot 1 = -0,0529$$

$$\left.\frac{\partial E}{\partial W_1}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = \left(y^{(0)} - d\right) f'\left(W_0^{(0)} + W_1^{(0)} x_1 + W_2^{(0)} x_2\right) x_1 = \left(y^{(0)} - d\right) y^{(0)} (1 - y^{(0)}) x_1$$

$$= -0,0529$$

$$\left.\frac{\partial E}{\partial W_2}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = \left(y^{(0)} - d\right) f'\left(W_0^{(0)} + W_1^{(0)} x_1 + W_2^{(0)} x_2\right) x_2 = \left(y^{(0)} - d\right) y^{(0)} (1 - y^{(0)}) x_2$$

$$= -0,0529$$

Ví dụ về toán tử OR

- Chọn hệ số học: $\eta = 1$
- Ta có các trọng số kết nối mới

$$W_0^{(1)} = W_0^{(0)} - \eta \left.\frac{\partial E}{\partial W_0}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = -0,2 + 1 \cdot 0,0529 = -0,147$$

$$W_1^{(1)} = W_1^{(0)} - \eta \left.\frac{\partial E}{\partial W_1}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = 0,7 + 1 \cdot 0,0529 = 0,753$$

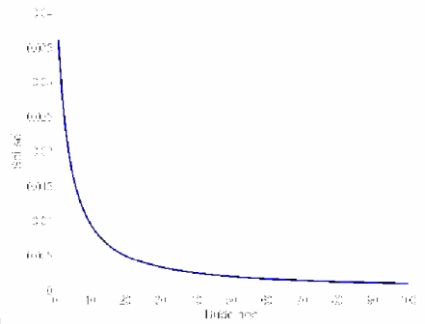
$$W_2^{(1)} = W_2^{(0)} - \eta \left.\frac{\partial E}{\partial W_2}\right|_{[\mathbf{w}]=[\mathbf{w}]^{(0)}} = 0,5 + 1 \cdot 0,0529 = 0,553$$

Ví dụ về toán tử OR

- **Bộ trọng số mới** $[\mathbf{W}]^{(1)} = \left[W_0^{(1)} \quad W_1^{(1)} \quad W_2^{(1)} \right] = [-0,147 \quad 0,753 \quad 0,553]$
- **Đầu ra:** $y^{(1)} = f\left(W_0^{(1)} x_0 + W_1^{(1)} x_1 + W_2^{(1)} x_2\right) = f\left(1,159\right) = 0,761\,.$
- **Sai số ở bước 1:**

$$E^{(1)} = \frac{1}{2}\left(y^{(1)} - d\right)^2 = \frac{1}{2}\left(0,761 - 1\right)^2 = 0,0286 < 0,0362 = E^{(0)}$$

- **Như vậy sau một bước, sai số đã giảm đi, tiếp tục thực hiện sau 100 bước lặp $E^{(100)}$= 0.00947**



Hình 2.11: Đồ thị giá trị hàm sai số giảm trong quá trình học

Đối với thuật toán L-M

- **Với 3 mẫu học, sau 100 bước lặp**
  - Sai số khi sử dụng Gradient Descent: E = 0.211
  - Sai số khi sử dụng L-M: E = 0.0182
- **L-M hội tụ nhanh hơn**

Một số phương pháp nâng cao chất lượng quá trình học

- **Học với hệ số thích nghi**
  - Eta: ảnh hưởng đến quá trình học
  - Cách lựa chọn đơn giản nhất: lấy giá trị cố định
  - Tuy nhiên cách này thường cho giải thuật rơi vào cực trị địa phương hoặc cần số bước lặp lớn khi eta nhỏ hoặc bị giao động xung quanh cực trị nếu như eta quá lớn
- **Khắc phục: sử dụng phương pháp hệ số thích nghi.**
  - Nếu $E^{(k)} < E^{(k-1)}$ thì tăng hệ số eta: $\eta^{(k+1)} = \eta^{(k)} \cdot \varepsilon_{inc}$
  - Nếu $E^{(k)} > E^{(k-1)}$ thì giảm hệ số eta: $\eta^{(k+1)} = \eta^{(k)} \cdot \varepsilon_{dec}$

Một số phương pháp nâng cao chất lượng quá trình học

- **Học với quán tính**

$$\Delta W^{(t)} = -\eta \nabla E^{(t)} + \alpha^{(t)} \Delta W^{(t-1)}$$

- $\alpha^{(t)}$ **là hệ số quán tính có giá trị thuộc [0, 1]**

Xấp xỉ thuật toán L-M

- Thuật toán L-M khá phức tạp do phải tính nghịch đảo của ma trận H(W)
- [Hagan94] đề xuất thay thế H(W) bởi G(W)

$$E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{M} e_i^2(\mathbf{W}) \qquad e_i(\mathbf{W}) = y_i(\mathbf{W}) - d_i$$

$$\mathbf{e}(\mathbf{W}) = \begin{bmatrix} e_1(\mathbf{W}) \\ e_2(\mathbf{W}) \\ \cdots \\ e_M(\mathbf{W}) \end{bmatrix}, \quad \mathbf{J}(\mathbf{W}) = \begin{bmatrix} \frac{\partial e_1}{\partial W_1} & \frac{\partial e_1}{\partial W_2} & \cdots & \frac{\partial e_1}{\partial W_n} \\ \frac{\partial e_2}{\partial W_1} & \frac{\partial e_2}{\partial W_2} & \cdots & \frac{\partial e_2}{\partial W_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_n}{\partial W_1} & \frac{\partial e_n}{\partial W_2} & \cdots & \frac{\partial e_n}{\partial W_n} \end{bmatrix}$$

$$\mathbf{g}(\mathbf{W}) = [\mathbf{J}(\mathbf{W})]^T \mathbf{e}(\mathbf{W})$$

$$\mathbf{G}(\mathbf{W}) = [\mathbf{J}(\mathbf{W})]^T \mathbf{J}(\mathbf{W}) + \mathbf{R}(\mathbf{W})$$

Bài tập về nhà

- Sử dụng matlab, viết hàm huấn luyện neuron cho toán tử OR, AND với giải thuật Gradient Descent hoặc L-M
- Hiển thị sai số theo số bước lặp của hai giải thuật trên cùng một hình vẽ
- Nhận xét và so sánh mức độ hội tụ của hai giải thuật