Fundamentos de Redes Neurais

José Gabriel R. C. Gomes
UFRJ – EPoli/DEL e COPPE/PEE/PADS
CBIC 2011, 8 a 11 de novembro, Fortaleza

Roteiro desta Apresentação

I	Introdução
II	Perceptrons Multicamadas
	Retropropagação de Erros
	Implementação Prática e Exemplos
III	Métodos de Ordem Superior
IV	Regularização
V	Aproximador Universal
VI	Métodos Geométricos de Projeto
VII	Redes RBF
VIII	Outros Tipos de Redes Neurais
IX	Aplicações

I. Introdução

- O Neurônio Natural
- Origens: Rosenblatt, Minsky, Papert
- Perceptron e Redes Neurais Artificiais
- Backpropagation
- Situações em que devem ser usadas Redes Neurais

Neurônio Natural

- Diferenças entre Neurônios Biológicos e Artificiais

  Complexidade das conexões
  Velocidade
  Densidade
  Tipos de sinais
  Implementação de algoritmos
  Repetições de cadeias de sinais
  Vídeo (microscopia em culturas de células de ratos)

Vídeo Microscopia Stanford

- Prof. Ciro da Silva, USP – cultura de células do hipocampo de ratos

- Microscopia com reprodução em velocidades de 30 a 100 vezes o tempo real

- Destaques

  Axônio e dendritos

  Complexidade

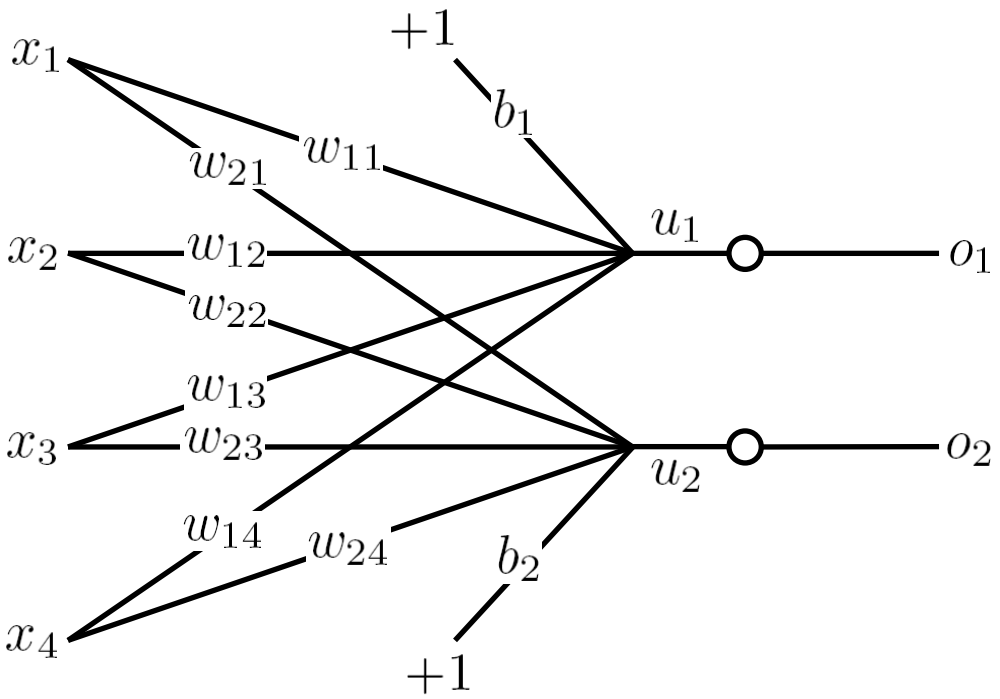
  Crescimento; formação de conexões

  Atividade

Origens

- McCulloch e Pitts (1943) – Modelo do Neurônio
- Rosenblatt (1958) – Algoritmo do *Perceptron*
- Minsky e Papert (1969) – *Perceptrons*
- Rumelhart, Williams, Hinton (1986) – *Backpropagation*
- *Proceedings IEEE, IEEE Trans. Neural Networks*

Perceptron (Single Layer)



Notação Detalhada

$$u_i = \sum_j w_{ij} x_j + b_i$$

$$o_i = f(u_i)$$

$$j = 1, \ldots, M \qquad (M = 4)$$

$$i = 1, \ldots, N \qquad (N = 2)$$

Função de Ativação

$$f(u) = \tanh(u) \qquad f(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

$$\frac{df}{du} = \frac{(e^u + e^{-u})^2 - (e^u - e^{-u})^2}{(e^u + e^{-u})^2} = 1 - \tanh^2(u)$$

$$\frac{df}{du} = 1 - f^2$$

Notação Detalhada

$$e_i = t_i - o_i$$

$$J = \frac{\sum_i e_i^2}{2} \qquad J = J(\mathbf{W}, \mathbf{b}, \mathbf{x})$$

$$\bar{J} = \frac{1}{\text{card}\{\mathbf{x}\}} \sum_x J(\mathbf{W}, \mathbf{b}, \mathbf{x})$$

$$(\mathbf{W}^*, \mathbf{b}^*) = \underset{(\mathbf{W},\mathbf{b})}{\text{argmin}}\ \bar{J}$$

Idéia Básica (atualização $\mathbf{W}$ e $\mathbf{b}$)

$$\mathbf{b} := \mathbf{b} - \eta \frac{\partial J}{\partial \mathbf{b}}$$

$$\mathbf{W} := \mathbf{W} - \eta \frac{\partial J}{\partial \mathbf{W}}$$

Regra da Cadeia e Derivadas Parciais

$$y = f(x_1, x_2)$$

$$x_1 = g(u, v) \qquad x_2 = h(u, v)$$

$$\frac{\partial y}{\partial u} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial u} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial u}$$

Cálculo do Gradiente

$$\frac{\partial J}{\partial b_i} = \frac{\partial J}{\partial e_i}\frac{\partial e_i}{\partial o_i}\frac{\partial o_i}{\partial u_i}\frac{\partial u_i}{\partial b_i} = -e_i(1-o_i^2)$$

$$e_i \qquad -1 \qquad (1-o_i^2) \qquad 1$$

$$\frac{\partial J}{\partial w_{ij}} = \frac{\partial J}{\partial e_i}\frac{\partial e_i}{\partial o_i}\frac{\partial o_i}{\partial u_i}\frac{\partial u_i}{\partial w_{ij}} = -e_i(1-o_i^2)x_j$$

$$u_i = w_{i1}x_1 + w_{i2}x_2 + \ldots + w_{ij}x_j + \ldots + w_{iM}x_M$$

$$\frac{\partial u_i}{\partial w_{ij}} = x_j$$

Notação Simplificada

$$\mathbf{u} = \mathbf{W}\mathbf{x} + \mathbf{b} \qquad \mathbf{W} \in \mathbb{R}^{N \times M}$$

$$\mathbf{o} = f(\mathbf{u})$$

$$\mathbf{e} = \mathbf{t} - \mathbf{o}$$

$$J = \frac{\mathbf{e}^T \mathbf{e}}{2}$$

Notação Simplificada

$$\frac{\partial J}{\partial \mathbf{b}} = \frac{\partial \mathbf{u}}{\partial \mathbf{b}} \frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}}$$

$$\mathbf{I} \qquad (\mathbf{I} - \mathrm{diag}^2(\mathbf{o})) \qquad -\mathbf{I} \qquad \mathbf{e} \in \mathbb{R}^{N \times 1}$$

$$\in \mathbb{R}^{N \times N} \qquad \in \mathbb{R}^{N \times N}$$

$$\frac{\partial J}{\partial \mathbf{W}} = \frac{\partial \mathbf{u}}{\partial \mathbf{W}} \odot \frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}}$$

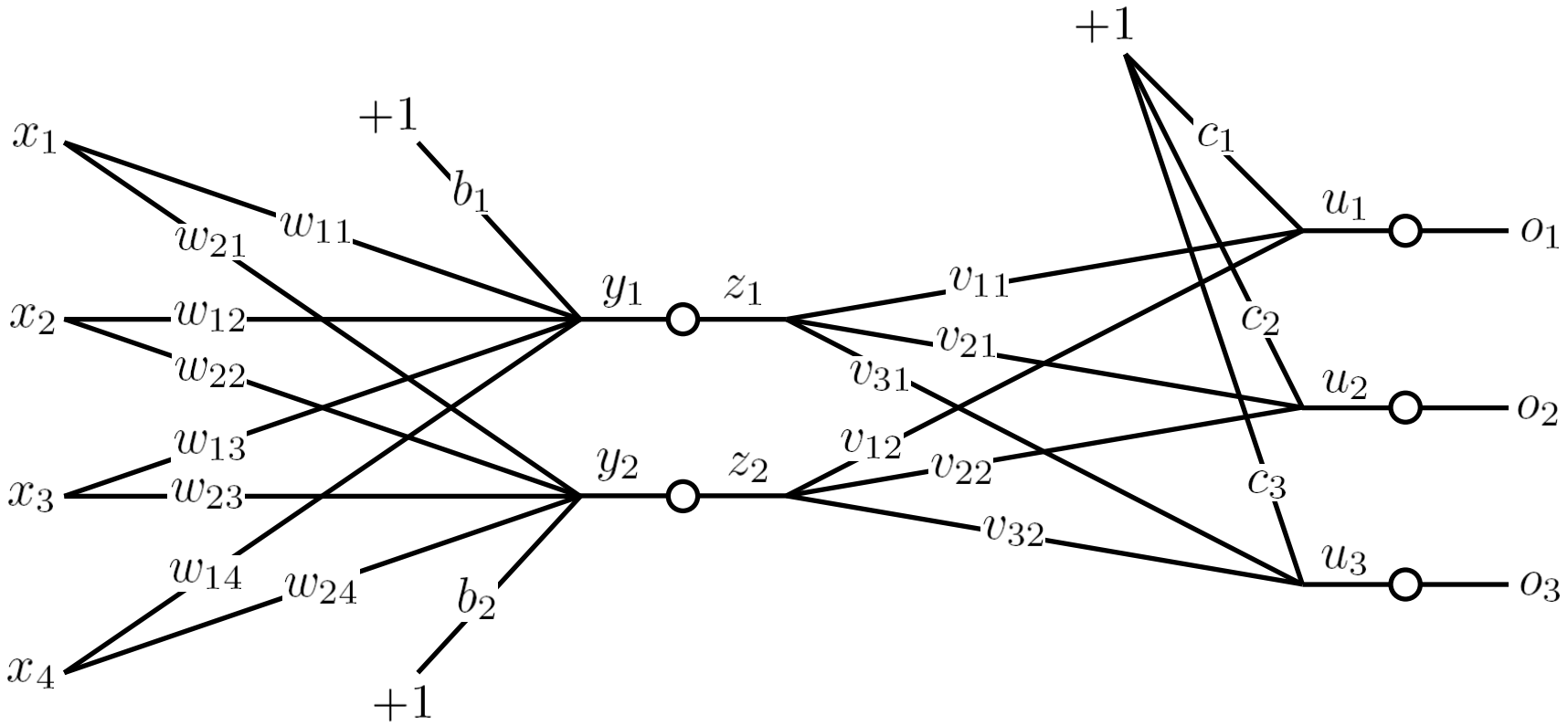
$$\begin{bmatrix} 1 & 2 \end{bmatrix} \odot \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 3 & 6 \\ 4 & 8 \end{bmatrix}$$

Notação Simplificada: d**u**/d**W**

$$\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_i \\ \vdots \\ u_N \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1j} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2j} & \cdots & w_{2M} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{i1} & w_{i2} & \cdots & w_{ij} & \cdots & w_{iM} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{Nj} & \cdots & w_{NM} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_j \\ \vdots \\ x_M \end{bmatrix}$$

$$\frac{\partial \mathbf{u}}{\partial \mathbf{W}} = \mathbf{x}^T$$

II. Multilayer Perceptron (MLP)



$$\mathbf{W} : M \times P \qquad \mathbf{V} : N \times M$$

Notação Detalhada

$$y_i = \sum_j w_{ij} x_j + b_i \qquad j = 1, \ldots, P \quad (P = 4)$$

$$i = 1, \ldots, M \quad (M = 2)$$

$$z_i = f(y_i)$$

$$u_k = \sum_l v_{kl} z_l + c_k \qquad l = 1, \ldots, M \quad (M = 2)$$

$$k = 1, \ldots, N \quad (N = 3)$$

$$o_k = f(u_k)$$

$$e_k = t_k - o_k \qquad J = \frac{\sum e_k^2}{2}$$

Gradiente na Camada de Saída

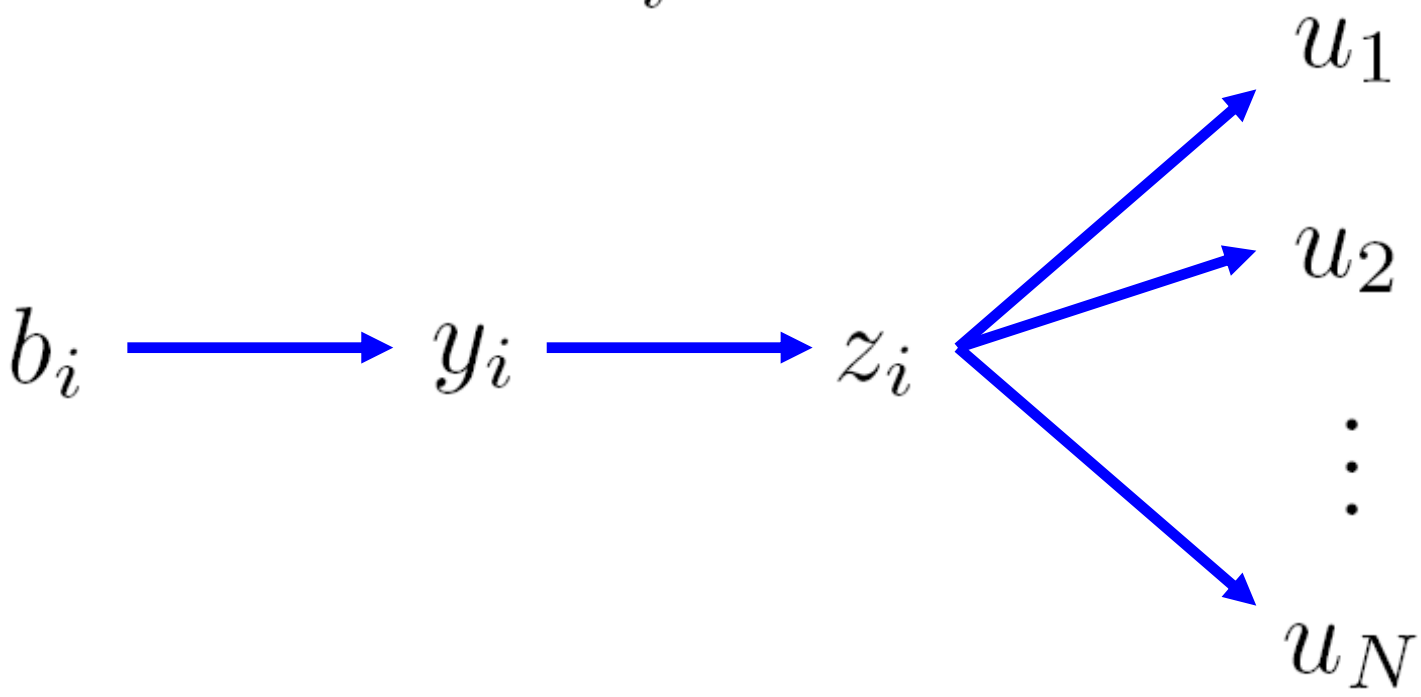
$$\frac{\partial J}{\partial c_k} = \frac{\partial J}{\partial e_k}\frac{\partial e_k}{\partial o_k}\frac{\partial o_k}{\partial u_k}\frac{\partial u_k}{\partial c_k} = -e_k(1-o_k^2)$$

$$\frac{\partial J}{\partial v_{kl}} = \frac{\partial J}{\partial e_k}\frac{\partial e_k}{\partial o_k}\frac{\partial o_k}{\partial u_k}\frac{\partial u_k}{\partial v_{kl}} = -e_k(1-o_k^2)z_l$$

$$\frac{\partial J}{\partial b_i} = ?$$

Gradiente na Camada Escondida

$$\frac{\partial J}{\partial b_i} = ?$$



Gradiente na Camada Escondida

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_i \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1j} & \cdots & w_{1P} \\ w_{21} & w_{22} & \cdots & w_{2j} & \cdots & w_{2P} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{i1} & w_{i2} & \cdots & w_{ij} & \cdots & w_{iP} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{Mj} & \cdots & w_{MP} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_j \\ \vdots \\ x_P \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_i \\ \vdots \\ b_M \end{bmatrix}$$

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_i \\ \vdots \\ z_M \end{bmatrix} = \begin{bmatrix} f(y_1) \\ f(y_2) \\ \vdots \\ f(y_i) \\ \vdots \\ f(y_M) \end{bmatrix}$$

$$\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \\ \vdots \\ u_N \end{bmatrix} = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1l} & \cdots & v_{1M} \\ v_{21} & v_{22} & \cdots & v_{2l} & \cdots & v_{2M} \\ \vdots & \vdots & & \vdots & & \vdots \\ v_{k1} & v_{k2} & \cdots & v_{kl} & \cdots & v_{kM} \\ \vdots & \vdots & & \vdots & & \vdots \\ v_{N1} & v_{N2} & \cdots & v_{Nl} & \cdots & v_{NM} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_l \\ \vdots \\ z_M \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \\ \vdots \\ c_N \end{bmatrix}$$

$$\begin{bmatrix} o_1 \\ o_2 \\ \vdots \\ o_k \\ \vdots \\ o_N \end{bmatrix} = \begin{bmatrix} f(u_1) \\ f(u_2) \\ \vdots \\ f(u_k) \\ \vdots \\ f(u_N) \end{bmatrix}$$

$$\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_k \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} t_1 - o_1 \\ t_2 - o_2 \\ \vdots \\ t_k - o_k \\ \vdots \\ t_N - o_N \end{bmatrix}$$

$$J = \frac{\sum e_k^2}{2}$$

Gradiente na Camada Escondida (b)

$$\begin{matrix} e_1 & -1 & (1-o_1^2) & v_{1i} & (1-z_i^2) & 1 \end{matrix}$$

$$\frac{\partial J}{\partial b_i} = \frac{\partial J}{\partial e_1}\left[\frac{\partial e_1}{\partial o_1}\frac{\partial o_1}{\partial u_1}\frac{\partial u_1}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial b_i}\right] + \frac{\partial J}{\partial e_2}\frac{\partial e_2}{\partial o_2}\frac{\partial o_2}{\partial u_2}\frac{\partial u_2}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial b_i} + \ldots + \frac{\partial J}{\partial e_N}\frac{\partial e_N}{\partial o_N}\frac{\partial o_N}{\partial u_N}\frac{\partial u_N}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial b_i}$$

$$\frac{\partial J}{\partial b_i} = -e_1(1-o_1^2)v_{1i}(1-z_i^2) - e_2(1-o_2^2)v_{2i}(1-z_i^2) - \ldots - e_N(1-o_N^2)v_{Ni}(1-z_i^2)$$

Alternativamente (dJ/db_i):

$$\frac{\partial J}{\partial b_i} = \sum_k \frac{\partial J}{\partial e_k} \frac{\partial e_k}{\partial o_k} \frac{\partial o_k}{\partial u_k} \frac{\partial u_k}{\partial z_i} \frac{\partial z_i}{\partial y_i} \frac{\partial y_i}{\partial b_i}$$

$$\frac{\partial J}{\partial b_i} = \sum_k -e_k(1-o_k^2)v_{ki}(1-z_i^2)$$

$$\frac{\partial J}{\partial b_i} = -(1-z_i^2)\sum_k v_{ki}(1-o_k^2)\,e_k$$

Gradiente na Camada Escondida (W)

$$\frac{\partial J}{\partial w_{ij}} = \frac{\partial J}{\partial e_1}\left[\frac{\partial e_1}{\partial o_1}\frac{\partial o_1}{\partial u_1}\frac{\partial u_1}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}}\right] + \frac{\partial J}{\partial e_2}\frac{\partial e_2}{\partial o_2}\frac{\partial o_2}{\partial u_2}\frac{\partial u_2}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}} + \ldots + \frac{\partial J}{\partial e_N}\frac{\partial e_N}{\partial o_N}\frac{\partial o_N}{\partial u_N}\frac{\partial u_N}{\partial z_i}\frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}}$$

$e_1 \quad -1 \quad (1-o_1^2) \quad v_{1i} \quad (1-z_i^2) \quad x_j$

$$\frac{\partial J}{\partial w_{ij}} = -e_1(1-o_1^2)v_{1i}(1-z_i^2)x_j - e_2(1-o_2^2)v_{2i}(1-z_i^2)x_j - \ldots - e_N(1-o_N^2)v_{Ni}(1-z_i^2)x_j$$

Alternativamente (dJ/dw_{ij}):

$$\frac{\partial J}{\partial w_{ij}} = \sum_k \frac{\partial J}{\partial e_k} \frac{\partial e_k}{\partial o_k} \frac{\partial o_k}{\partial u_k} \frac{\partial u_k}{\partial z_i} \frac{\partial z_i}{\partial y_i} \frac{\partial y_i}{\partial w_{ij}}$$

$$\frac{\partial J}{\partial w_{ij}} = \sum_k -e_k(1-o_k^2)v_{ki}(1-z_i^2)x_j$$

$$\frac{\partial J}{\partial w_{ij}} = -x_j(1-z_i^2)\sum_k v_{ki}(1-o_k^2)\,e_k$$

Extensão para o Caso Geral

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b} \qquad \mathbf{W} \in \mathbb{R}^{M \times P}$$

$$\mathbf{z} = f(\mathbf{y})$$

$$\mathbf{u} = \mathbf{V}\mathbf{z} + \mathbf{c} \qquad \mathbf{V} \in \mathbb{R}^{N \times M}$$

$$\mathbf{o} = f(\mathbf{u})$$

$$\mathbf{e} = \mathbf{t} - \mathbf{o}$$

$$J = \frac{\mathbf{e}^T \mathbf{e}}{2}$$

Extensão para o Caso Geral

$$\frac{\partial J}{\partial \mathbf{c}} = \frac{\partial \mathbf{u}}{\partial \mathbf{c}} \left(\frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}} \right)$$

$$= -(\mathbf{I} - \mathrm{diag}^2(\mathbf{o}))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{V}} = \frac{\partial \mathbf{u}}{\partial \mathbf{V}} \odot \left(\frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}} \right)$$

$$= -\mathbf{z}^T \odot (\mathbf{I} - \mathrm{diag}^2(\mathbf{o}))\mathbf{e}$$

Extensão para o Caso Geral

$$\frac{\partial J}{\partial \mathbf{b}} = \frac{\partial \mathbf{y}}{\partial \mathbf{b}} \left[\frac{\partial \mathbf{z}}{\partial \mathbf{y}} \frac{\partial \mathbf{u}}{\partial \mathbf{z}} \left(\frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}} \right) \right]$$

$$= -(\mathbf{I} - \mathrm{diag}^2(\mathbf{z}))\mathbf{V}^T(\mathbf{I} - \mathrm{diag}^2(\mathbf{o}))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}} = \frac{\partial \mathbf{y}}{\partial \mathbf{W}} \odot \left[\frac{\partial \mathbf{z}}{\partial \mathbf{y}} \frac{\partial \mathbf{u}}{\partial \mathbf{z}} \left(\frac{\partial \mathbf{o}}{\partial \mathbf{u}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}} \frac{\partial J}{\partial \mathbf{e}} \right) \right]$$

$$= -\mathbf{x}^T \odot (\mathbf{I} - \mathrm{diag}^2(\mathbf{z}))\mathbf{V}^T(\mathbf{I} - \mathrm{diag}^2(\mathbf{o}))\mathbf{e}$$

$\mathbf{du/dz = V^T}$:

$$\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \\ \vdots \\ u_N \end{bmatrix} = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1l} & \cdots & v_{1M} \\ v_{21} & v_{22} & \cdots & v_{2l} & \cdots & v_{2M} \\ \vdots & \vdots & & \vdots & & \vdots \\ v_{k1} & v_{k2} & \cdots & v_{kl} & \cdots & v_{kM} \\ \vdots & \vdots & & \vdots & & \vdots \\ v_{N1} & v_{N2} & \cdots & v_{Nl} & \cdots & v_{NM} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_l \\ \vdots \\ z_M \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \\ \vdots \\ c_N \end{bmatrix}$$

$$\begin{bmatrix} \frac{\partial u_1}{\partial z_1} & \frac{\partial u_2}{\partial z_1} & \cdots & \frac{\partial u_N}{\partial z_1} \\ \frac{\partial u_1}{\partial z_2} & \frac{\partial u_2}{\partial z_2} & \cdots & \frac{\partial u_N}{\partial z_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial u_1}{\partial z_M} & \frac{\partial u_2}{\partial z_M} & \cdots & \frac{\partial u_N}{\partial z_M} \end{bmatrix} = \mathbf{V}^T$$

$$\frac{\partial \mathbf{u}}{\partial \mathbf{z}} = \mathbf{V}^T \in \mathbb{R}^{M \times N}$$

Notação Simplificada Formal

$$\mathbf{u}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1$$

$$\mathbf{o}_1 = f(\mathbf{u}_1)$$

$$\mathbf{u}_2 = \mathbf{W}_2\mathbf{o}_1 + \mathbf{b}_2$$

$$\mathbf{o}_2 = f(\mathbf{u}_2)$$

$$\mathbf{e} = \mathbf{t} - \mathbf{o}_2$$

$$J = \frac{\mathbf{e}^T\mathbf{e}}{2}$$

Notação Simplificada Formal

$$\frac{\partial J}{\partial \mathbf{b}_2} = \frac{\partial \mathbf{u}_2}{\partial \mathbf{b}_2}\left(\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_2}\frac{\partial J}{\partial \mathbf{e}}\right)$$

$$= -(\mathbf{I} - \mathrm{diag}^2(\mathbf{o}_2))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}_2} = \frac{\partial \mathbf{u}_2}{\partial \mathbf{W}_2} \odot \left(\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_2}\frac{\partial J}{\partial \mathbf{e}}\right)$$

$$= -\mathbf{o}_1^T \odot (\mathbf{I} - \mathrm{diag}^2(\mathbf{o}_2))\mathbf{e}$$

Notação Simplificada Formal

$$\frac{\partial J}{\partial \mathbf{b}_1} = \frac{\partial \mathbf{u}_1}{\partial \mathbf{b}_1}\left[\frac{\partial \mathbf{o}_1}{\partial \mathbf{u}_1}\frac{\partial \mathbf{u}_2}{\partial \mathbf{o}_1}\left(\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_2}\frac{\partial J}{\partial \mathbf{e}}\right)\right]$$

$$= -(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_1))\mathbf{W}_2^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}_1} = \frac{\partial \mathbf{u}_1}{\partial \mathbf{W}_1}\odot\left[\frac{\partial \mathbf{o}_1}{\partial \mathbf{u}_1}\frac{\partial \mathbf{u}_2}{\partial \mathbf{o}_1}\left(\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_2}\frac{\partial J}{\partial \mathbf{e}}\right)\right]$$

$$= -\mathbf{x}^T \odot (\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_1))\mathbf{W}_2^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{e}$$

Notação Simp. Formal – 3 Camadas

$$\mathbf{u}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1$$

$$\mathbf{o}_1 = f(\mathbf{u}_1)$$

$$\mathbf{u}_2 = \mathbf{W}_2\mathbf{o}_1 + \mathbf{b}_2$$

$$\mathbf{o}_2 = f(\mathbf{u}_2)$$

$$\mathbf{u}_3 = \mathbf{W}_3\mathbf{o}_2 + \mathbf{b}_3$$

$$\mathbf{o}_3 = f(\mathbf{u}_3)$$

$$\mathbf{e} = \mathbf{t} - \mathbf{o}_3$$

$$J = \frac{\mathbf{e}^T\mathbf{e}}{2}$$

Notação Simp. Formal – 3 Camadas

$$\frac{\partial J}{\partial \mathbf{b}_3} = \frac{\partial \mathbf{u}_3}{\partial \mathbf{b}_3}\left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)$$

$$= -(\mathbf{I} - \mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}_3} = \frac{\partial \mathbf{u}_3}{\partial \mathbf{W}_3} \odot \left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)$$

$$= -\mathbf{o}_2^T \odot (\mathbf{I} - \mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

Notação Simp. Formal – 3 Camadas

$$\frac{\partial J}{\partial \mathbf{b}_2} = \frac{\partial \mathbf{u}_2}{\partial \mathbf{b}_2}\left[\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_3}{\partial \mathbf{o}_2}\left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)\right]$$

$$= -(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{W}_3^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}_2} = \frac{\partial \mathbf{u}_2}{\partial \mathbf{W}_2}\odot\left[\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_3}{\partial \mathbf{o}_2}\left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)\right]$$

$$= -\mathbf{o}_1^T \odot (\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{W}_3^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

Notação Simp. Formal – 3 Camadas

$$\frac{\partial J}{\partial \mathbf{b}_1} = \frac{\partial \mathbf{u}_1}{\partial \mathbf{b}_1}\left\{\frac{\partial \mathbf{o}_1}{\partial \mathbf{u}_1}\frac{\partial \mathbf{u}_2}{\partial \mathbf{o}_1}\left[\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_3}{\partial \mathbf{o}_2}\left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)\right]\right\}$$

$$= -(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_1))\mathbf{W}_2^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{W}_3^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

$$\frac{\partial J}{\partial \mathbf{W}_1} = \frac{\partial \mathbf{u}_1}{\partial \mathbf{W}_1}\odot\left\{\frac{\partial \mathbf{o}_1}{\partial \mathbf{u}_1}\frac{\partial \mathbf{u}_2}{\partial \mathbf{o}_1}\left[\frac{\partial \mathbf{o}_2}{\partial \mathbf{u}_2}\frac{\partial \mathbf{u}_3}{\partial \mathbf{o}_2}\left(\frac{\partial \mathbf{o}_3}{\partial \mathbf{u}_3}\frac{\partial \mathbf{e}}{\partial \mathbf{o}_3}\frac{\partial J}{\partial \mathbf{e}}\right)\right]\right\}$$

$$= -\mathbf{x}^T\odot(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_1))\mathbf{W}_2^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_2))\mathbf{W}_3^T(\mathbf{I}-\mathrm{diag}^2(\mathbf{o}_3))\mathbf{e}$$

Implementação – Modo Seqüencial

fim= 0; $n = 1$; $i = 1$;
while not(fim),
% [1] *Feed-Forward*
$x = x(n)$;
$\mathbf{u}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1$; $\mathbf{o}_1 = f(\mathbf{u}_1)$;
$\mathbf{u}_2 = \mathbf{W}_2\mathbf{o}_1 + \mathbf{b}_2$; $\mathbf{o}_2 = f(\mathbf{u}_2)$;
$\mathbf{u}_3 = \mathbf{W}_3\mathbf{o}_2 + \mathbf{b}_3$; $\mathbf{o}_3 = f(\mathbf{u}_3)$;
$\mathbf{e} = \mathbf{t} - \mathbf{o}$; $J(i) = 0.5 * (\mathbf{e}^T\mathbf{e})$;
% [2] *Error Backpropagation*

$$\mathbf{M}_3 = \mathbf{I}\text{–diag}^2(\mathbf{o}_3);\quad \alpha_3 = \mathbf{M}_3\mathbf{e};\quad \Delta\mathbf{b}_3 = \alpha_3;\quad \Delta\mathbf{W}_3 = \mathbf{o}_2^T \odot \alpha_3;$$
$$\mathbf{M}_2 = \mathbf{I}\text{–diag}^2(\mathbf{o}_2);\quad \alpha_2 = \mathbf{M}_2\mathbf{W}_3^T\alpha_3;\quad \Delta\mathbf{b}_2 = \alpha_2;\quad \Delta\mathbf{W}_2 = \mathbf{o}_1^T \odot \alpha_2;$$
$$\mathbf{M}_1 = \mathbf{I}\text{–diag}^2(\mathbf{o}_1);\quad \alpha_1 = \mathbf{M}_1\mathbf{W}_2^T\alpha_2;\quad \Delta\mathbf{b}_1 = \alpha_1;\quad \Delta\mathbf{W}_1 = \mathbf{x}^T \odot \alpha_1;$$

% [3] Atualizações

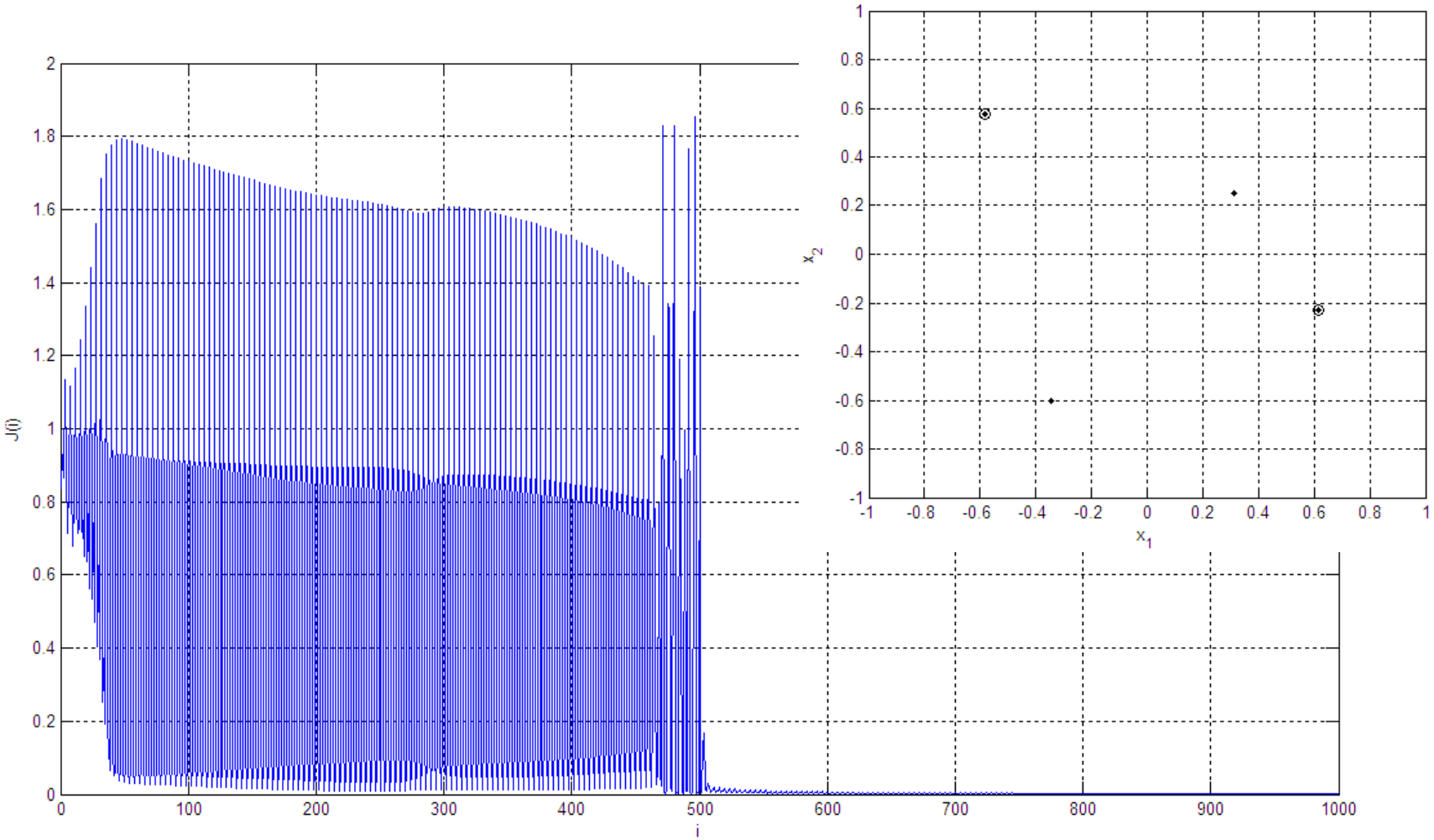
$$\mathbf{b}_3 = \mathbf{b}_3 + \eta\Delta\mathbf{b}_3;\quad \mathbf{W}_3 = \mathbf{W}_3 + \eta\Delta\mathbf{W}_3;$$
$$\mathbf{b}_2 = \mathbf{b}_2 + \eta\Delta\mathbf{b}_2;\quad \mathbf{W}_2 = \mathbf{W}_2 + \eta\Delta\mathbf{W}_2;$$
$$\mathbf{b}_1 = \mathbf{b}_1 + \eta\Delta\mathbf{b}_1;\quad \mathbf{W}_1 = \mathbf{W}_1 + \eta\Delta\mathbf{W}_1;$$

% [4] Critério de Parada
if $(i > 1)$ & $(J(i) - J(i-1))/J(i) < \delta$, fim= 1;
else $\eta = \alpha\eta$; $n = n + 1$; if $n > N$, $n = 1$; end if; $i = i + 1$; end if;
end;

Implementação – Modo Seqüencial

```
% 070626 gabriel@pads.ufrj.br [2] – Exemplo1Sequential.m

close all; clear all;

% [A] Data

C = [0 0 1 1 ; 0 1 0 1];
rand('state',0)
X = []; P = 1;
for k = 1:size(C,2),
   X = [X 0.7*rand(2,P)+repmat(C(:,k),1,P)];
end;
X = X - repmat(mean(X,2),1,size(X,2));
plot(X(1,:),X(2,:),'k.');
t = []; T = [1 -1 -1 1];
for k = 1:size(C,2),
   t = [t repmat(T(k),1,P)];
end;

% [A1] Randomize Data Order

randn('state',0);
X = [X ; t ; randn(1,size(X,2))]';
X = sortrows(X,4)';
t = X(3,:); X = X(1:2,:);

% [B] Network Init

% [B1] Parameters

K = 2; % Number of Layers
eta = 0.5; % Learning Rate Initial Value
Delta = 1e-4; % Stop Criterion
N = size(X,2); % Number of Input Vectors
E = 4*P; % Number of Feed-Forward Iterations per Epoch
alpha = 0.999; % Learning Rate Decay Factor

eta = 0.5;
alpha = 1;

% [B2] Layers

L(1).W = rand(2,2)-0.5;
L(1).b = rand(2,1)-0.5;
L(2).W = rand(1,2)-0.5;
L(2).b = rand(1,1)-0.5;

% [C] Sequential Error Backpropagation Training

n=1; i=1; fim=0;
while not(fim),

   % [C1] Feed-Forward

   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   e = t(n) - L(K).o;
   J(i) = (e'*e)/2;

   % [C2] Error Backpropagation

   L(K+1).alpha = e; L(K+1).W = eye(length(e));
   for k = fliplr(1:K),
      L(k).M = eye(length(L(k).o)) - diag(L(k).o)^2;
      L(k).alpha = L(k).M*L(k+1).W'*L(k+1).alpha;
      L(k).db = L(k).alpha;
      L(k).dW = kron(L(k).x',L(k).alpha);
   end;

   % [C3] Updates

   for k = 1:K,
      L(k).b = L(k).b + eta*L(k).db;
      L(k).W = L(k).W + eta*L(k).dW;
   end;

   % [C4] Stop criterion

   if (i>1),
      if (abs(J(i)-J(i-1))/J(i) < Delta)|(i>1000),
         fim = 1;
      end;
   end;
   if not(fim)
      i = i+1; n = n+1; if n>N, n=1; end; eta = eta*alpha;
   end;

end;

% [D] Test

hold on;
for n = 1:size(X,2),
   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   if L(K).o < 0, plot(X(1,n),X(2,n),'ko'); end;
end;
figure; plot(J);
```

Exemplo – Modo Seqüencial



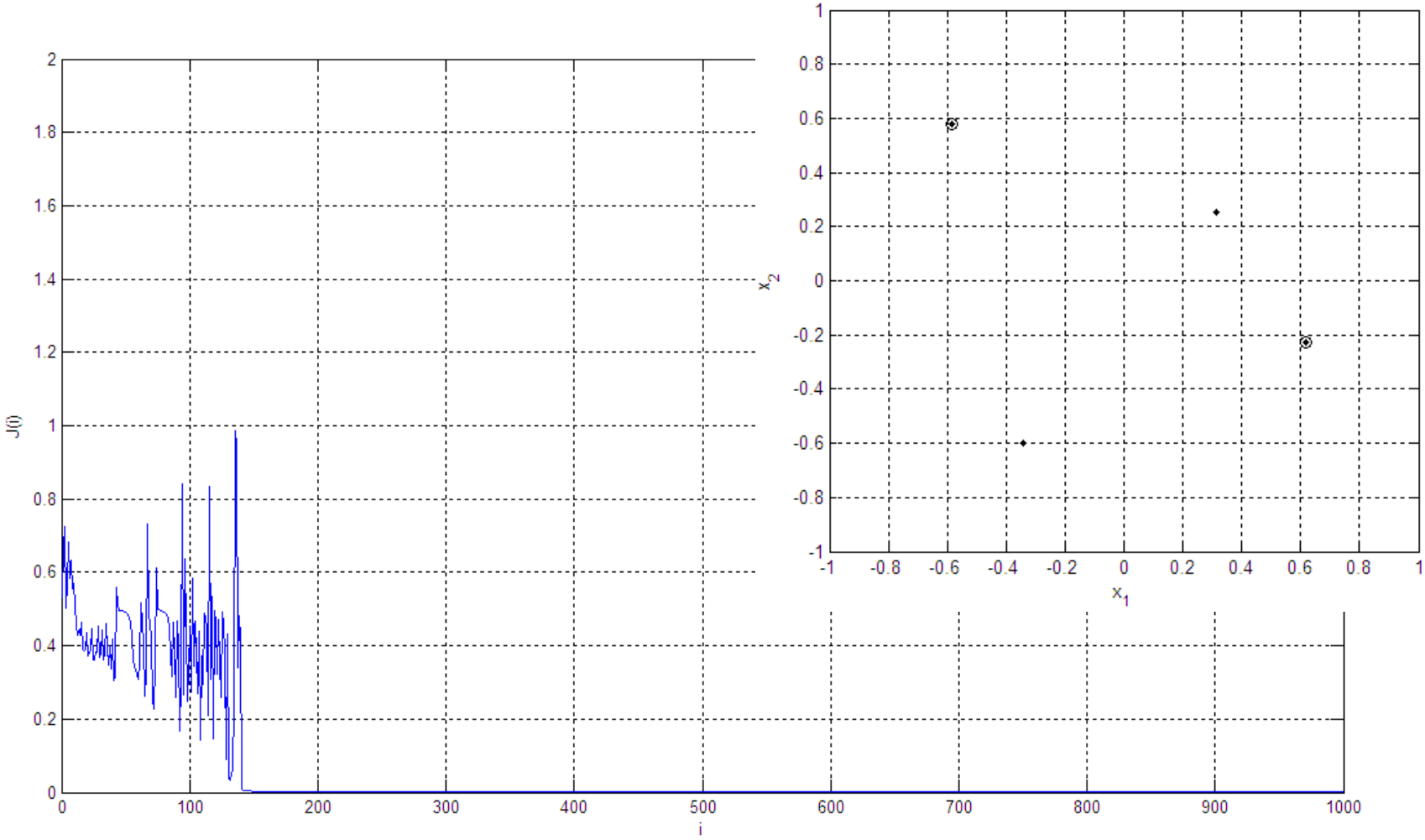
Implementação – Modo *Batch*

fim= 0; $n = 1$; $i = 1$;
while not(fim),
$\Delta\mathbf{b}_1 = \mathbf{0}$; $\Delta\mathbf{W}_1 = \mathbf{0}$; $\Delta\mathbf{b}_2 = \mathbf{0}$; $\Delta\mathbf{W}_2 = \mathbf{0}$; $\Delta\mathbf{b}_3 = \mathbf{0}$; $\Delta\mathbf{W}_3 = \mathbf{0}$; $J(i) = 0$;
for e = 1:E,
% [1] *Feed-Forward*
$x = x(n)$;
$\mathbf{u}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1$; $\mathbf{o}_1 = f(\mathbf{u}_1)$;
$\mathbf{u}_2 = \mathbf{W}_2\mathbf{o}_1 + \mathbf{b}_2$; $\mathbf{o}_2 = f(\mathbf{u}_2)$;
$\mathbf{u}_3 = \mathbf{W}_3\mathbf{o}_2 + \mathbf{b}_3$; $\mathbf{o}_3 = f(\mathbf{u}_3)$;
$\mathbf{e} = \mathbf{t} - \mathbf{o}$; $J(i) = J(i) + 0.5 * (\mathbf{e}^T\mathbf{e})$;
% [2] *Error Backpropagation*
$\mathbf{M}_3 = \mathbf{I}-\text{diag}^2(\mathbf{o}_3)$; $\alpha_3 = \mathbf{M}_3\mathbf{e}$; $\Delta\mathbf{b}_3 = \Delta\mathbf{b}_3 + \alpha_3$; $\Delta\mathbf{W}_3 = \Delta\mathbf{W}_3 + \mathbf{o}_2^T \odot \alpha_3$;
$\mathbf{M}_2 = \mathbf{I}-\text{diag}^2(\mathbf{o}_2)$; $\alpha_2 = \mathbf{M}_2\mathbf{W}_3^T\alpha_3$; $\Delta\mathbf{b}_2 = \Delta\mathbf{b}_2 + \alpha_2$; $\Delta\mathbf{W}_2 = \Delta\mathbf{W}_2 + \mathbf{o}_1^T \odot \alpha_2$;
$\mathbf{M}_1 = \mathbf{I}-\text{diag}^2(\mathbf{o}_1)$; $\alpha_1 = \mathbf{M}_1\mathbf{W}_2^T\alpha_2$; $\Delta\mathbf{b}_1 = \Delta\mathbf{b}_1 + \alpha_1$; $\Delta\mathbf{W}_1 = \Delta\mathbf{W}_1 + \mathbf{x}^T \odot \alpha_1$;
$n = n + 1$; if $n > N$, $n = 1$; end if;
end;
% [3] Atualizações
$\mathbf{b}_3 = \mathbf{b}_3 + \eta\Delta\mathbf{b}_3$; $\mathbf{W}_3 = \mathbf{W}_3 + \eta\Delta\mathbf{W}_3$;
$\mathbf{b}_2 = \mathbf{b}_2 + \eta\Delta\mathbf{b}_2$; $\mathbf{W}_2 = \mathbf{W}_2 + \eta\Delta\mathbf{W}_2$;
$\mathbf{b}_1 = \mathbf{b}_1 + \eta\Delta\mathbf{b}_1$; $\mathbf{W}_1 = \mathbf{W}_1 + \eta\Delta\mathbf{W}_1$;
% [4] Critério de Parada
$J(i) = J(i)/E$;
if $(i > 1)$ & $(J(i) - J(i-1))/J(i) < \delta$, fim= 1;
else $\eta = \alpha\eta$; $n = n + 1$; if $n > N$, $n = 1$; end if; $i = i + 1$; end if;
end;

Implementação – Modo *Batch*

```
% 070626 gabriel@pads.ufrj.br (2D) – Exemplo1Batch.m

close all; clear all;

% [A] Data

C = [0 0 1 1 ; 0 1 0 1];
rand('state',0)
X = []; P = 1;
for k = 1:size(C,2),
   X = [X 0.7*rand(2,P)+repmat(C(:,k),1,P)];
end;
X = X - repmat(mean(X,2),1,size(X,2));
plot(X(1,:),X(2,:),'k.');
t = []; T = [1 -1 -1 1];
for k = 1:size(C,2),
   t = [t repmat(T(k),1,P)];
end;

% [A1] Randomize Data Order

randn('state',0);
X = [X ; t ; randn(1,size(X,2))]';
X = sortrows(X,4)';
t = X(3,:); X = X(1:2,:);

% [B] Network Init

% [B1] Parameters

K = 2; % Number of Layers
eta = 1; % Learning Rate Initial Value
Delta = 1e-6; % Stop Criterion
N = size(X,2); % Number of Input Vectors
E = 4*P; % Number of Feed-Forward Iterations per Epoch
alpha = 0.99; % Learning Rate Decay Factor

eta = 0.7;
alpha = 1;

% [B2] Layers

L(1).W = rand(2,2)-0.5;
L(1).b = rand(2,1)-0.5;
L(2).W = rand(1,2)-0.5;
L(2).b = rand(1,1)-0.5;

% [C] Batch Error Backpropagation Training

n=1; i=1; fim=0;
while not(fim),

   for k=1:K,
      L(k).db = zeros(size(L(k).b));
      L(k).dW = zeros(size(L(k).W));
   end;
   J(i) = 0;
   for ep=1:E,

      % [C1] Feed-Forward

      L(1).x = X(:,n);
      for k = 1:K,
         L(k).u = L(k).W*L(k).x + L(k).b;
         L(k).o = tanh(L(k).u);
         L(k+1).x = L(k).o;
      end;
      e = t(n) - L(K).o;
      J(i) = J(i) + (e'*e)/2;

      % [C2] Error Backpropagation

      L(K+1).alpha = e; L(K+1).W = eye(length(e));
      for k = fliplr(1:K),
         L(k).M = eye(length(L(k).o)) - diag(L(k).o)^2;
         L(k).alpha = L(k).M*L(k+1).W'*L(k+1).alpha;
         L(k).db = L(k).db + L(k).alpha;
         L(k).dW = L(k).dW + kron(L(k).x',L(k).alpha);
      end;
      n = n+1; if n>N, n=1; end;

   end;

   % [C3] Updates

   for k = 1:K,
      L(k).b = L(k).b + eta*L(k).db;
      L(k).W = L(k).W + eta*L(k).dW;
   end;
   J(i) = J(i)/E;

   % [C4] Stop criterion

   if (i>1),
      if (abs(J(i)-J(i-1))/J(i) < Delta)|(i>1000),
         fim = 1;
      end;
   end;
   if not(fim)
      i = i+1; if n>N, n=1; end; eta = eta*alpha;
   end;

end;

% [D] Test

hold on;
for n = 1:size(X,2),
   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   if L(K).o < 0, plot(X(1,n),X(2,n),'ko'); end;
end;
figure; plot(J);
```

Exemplo – Modo *Batch*



Implementação – Detalhes Práticos

- Pré-processamento ; redução de dimensão
- Faixa dinâmica adequada (escalamento e normalização); saída
- Conjuntos 60/20/20 (*N* grande); dimensionamento da rede
- Inicialização
- Usar taxa de aprendizado baixa
- Generalização e *overtraining*; ruído; *outliers*
- Linguagens de programação
- Extensões: validação cruzada, confusão, outros tipos de *J*, adaptação

Momento

- Até agora:

$$\mathbf{b}_k = \mathbf{b}_k + \eta \Delta \mathbf{b}_k$$

$$\mathbf{W}_k = \mathbf{W}_k + \eta \Delta \mathbf{W}_k$$

$$\eta = \alpha \eta$$

- traingd.m (alpha = 0)
- traingda.m (alpha <> 0)

Momento

$$(\mathbf{b}, \mathbf{W})_{i+1} = (\mathbf{b}, \mathbf{W})_i + \eta_i (\Delta \mathbf{b}, \Delta \mathbf{W})_i$$

$$\eta_{i+1} = \alpha \eta_i$$

$$\mathbf{v}_i = \eta_i \Delta p_i$$

$\mu = 0$:

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \eta_i \Delta p_i$$

$$\eta_{i+1} = \alpha \eta_i$$

$$\mathbf{v}_i = \eta_i \Delta p_i + \mu \mathbf{v}_{i-1}$$

$0 < \mu < 1$:

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \eta_i \Delta p_i + \mu \mathbf{v}_{i-1}$$

$$\eta_{i+1} = \alpha \eta_i$$

Momento – Implementação

$$
\begin{aligned}
&\text{for } k = 1{:}K, \\
&\quad L(k).\mathbf{v}_b = \eta * L(k).\Delta\mathbf{b} + \mu * L(k).\mathbf{v}_b; \\
&\quad L(k).\mathbf{b} = L(k).\mathbf{b} + L(k).\mathbf{v}_b \\
&\quad L(k).\mathbf{v}_W = \eta * L(k).\Delta\mathbf{W} + \mu * L(k).\mathbf{v}_W; \\
&\quad L(k).\mathbf{W} = L(k).\mathbf{W} + L(k).\mathbf{v}_W \\
&\text{end;}
\end{aligned}
$$

- Inicialização:

$$\text{for } k = 1{:}K,\ L(k).\mathbf{v}_b = \mathbf{0};\ L(k).\mathbf{v}_W = \mathbf{0};\ \text{end;}$$

Momento – Implementação

- Regra “delta” generalizada:

$$L(k).\mathbf{v}_b = \eta * L(k).\Delta\mathbf{b} + \mu * L(k).\mathbf{v}_b;$$
$$L(k).\mathbf{v}_W = \eta * L(k).\Delta\mathbf{W} + \mu * L(k).\mathbf{v}_W;$$

Se $\mu = 0$, tem-se a regra “delta” basica.

Momento – Implement. Alternativa

for $k = 1{:}K$,

$$L(k).\mathbf{v}_b = \eta * (1 - \mu) * L(k).\Delta\mathbf{b} + \mu * L(k).\mathbf{v}_b;$$
$$L(k).\mathbf{b} = L(k).\mathbf{b} + L(k).\mathbf{v}_b$$
$$L(k).\mathbf{v}_W = \eta * (1 - \mu) * L(k).\Delta\mathbf{W} + \mu * L(k).\mathbf{v}_W;$$
$$L(k).\mathbf{W} = L(k).\mathbf{W} + L(k).\mathbf{v}_W$$

end;

- traingdm.m

Momento – Implement. Alternativa

$$\eta_i = \eta \qquad 0 < \mu < 1$$

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \eta\Delta\mathbf{p}_i + \mu\mathbf{v}_{i-1} \qquad \mathbf{v}_i = \eta\Delta\mathbf{p}_i + \mu\mathbf{v}_{i-1}$$

$$\mathbf{v}_1 = \eta\Delta\mathbf{p}_1$$

$$\mathbf{v}_2 = \eta\Delta\mathbf{p}_2 + \mu\mathbf{v}_1 = \eta\Delta\mathbf{p}_2 + \mu\eta\Delta\mathbf{p}_1$$

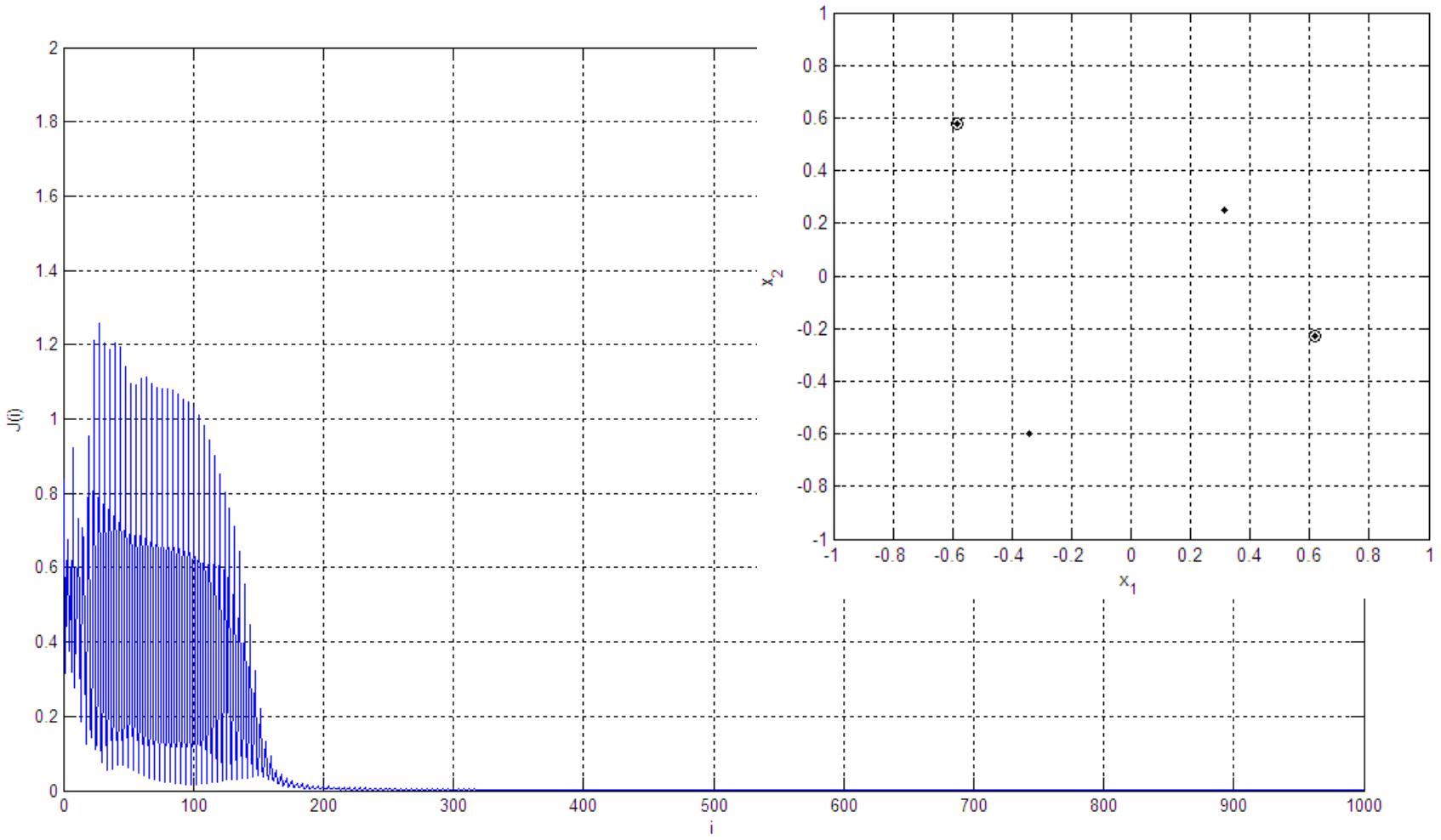
$$\mathbf{v}_3 = \eta\Delta\mathbf{p}_3 + \mu\mathbf{v}_2 = \eta\Delta\mathbf{p}_3 + \mu\eta\Delta\mathbf{p}_2 + \mu^2\eta\Delta\mathbf{p}_1$$

$$\mathbf{v}_i = \eta\sum_{j=1}^{i}\Delta\mathbf{p}_i\mu^{i-j}$$

Momento – Exemplo Seqüencial

```
% 070627 gabriel@pads.ufrj.br [2]
% Exemplo1SequentialMomentum.m

close all; clear all;

% [A] Data

C = [0 0 1 1 ; 0 1 0 1];
rand('state',0)
X = []; P = 1;
for k = 1:size(C,2),
   X = [X 0.7*rand(2,P)+repmat(C(:,k),1,P)];
end;
X = X - repmat(mean(X,2),1,size(X,2));
plot(X(1,:),X(2,:),'k.');
t = []; T = [1 -1 -1 1];
for k = 1:size(C,2),
   t = [t repmat(T(k),1,P)];
end;

% [A1] Randomize Data Order

randn('state',0);
X = [X ; t ; randn(1,size(X,2))]';
X = sortrows(X,4)';
t = X(3,:); X = X(1:2,:);

% [B] Network Init

% [B1] Parameters

K = 2; % Number of Layers
eta = 0.5; % Learning Rate Initial Value
Delta = 1e-4; % Stop Criterion
N = size(X,2); % Number of Input Vectors
E = 4*P; % Number of Feed-Forward Iterations per Epoch
alpha = 0.999; % Learning Rate Decay Factor
mu = 0.9; % Momentum constant

mu = 0.9;
eta = 0.1;
alpha = 1;

% [B2] Layers

L(1).W = rand(2,2)-0.5;
L(1).b = rand(2,1)-0.5;
L(2).W = rand(1,2)-0.5;
L(2).b = rand(1,1)-0.5;

for k=1:K,
   L(k).vb = zeros(size(L(k).b));
   L(k).vW = zeros(size(L(k).W));
end;

% [C] Sequential Error Backpropagation Training

n=1; i=1; fim=0;
while not(fim),

   % [C1] Feed-Forward

   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   e = t(n) - L(K).o;
   J(i) = (e'*e)/2;

   % [C2] Error Backpropagation

   L(K+1).alpha = e; L(K+1).W = eye(length(e));
   for k = fliplr(1:K),
      L(k).M = eye(length(L(k).o)) - diag(L(k).o)^2;
      L(k).alpha = L(k).M*L(k+1).W'*L(k+1).alpha;
      L(k).db = L(k).alpha;
      L(k).dW = kron(L(k).x',L(k).alpha);
   end;

   % [C3] Updates

   for k = 1:K,
      L(k).vb = eta*L(k).db + mu*L(k).vb;
      L(k).b = L(k).b + L(k).vb;
      L(k).vW = eta*L(k).dW + mu*L(k).vW;
      L(k).W = L(k).W + L(k).vW;
   end;

   % [C4] Stop criterion

   if (i>1),
      if (abs(J(i)-J(i-1))/J(i) < Delta)|(i>1000),
         fim = 1;
      end;
   end;
   if not(fim)
      i = i+1; n = n+1; if n>N, n=1; end; eta = eta*alpha;
   end;

end;

% [D] Test

hold on;
for n = 1:size(X,2),
   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   if L(K).o < 0, plot(X(1,n),X(2,n),'ko'); end;
end;
figure; plot(J);
```

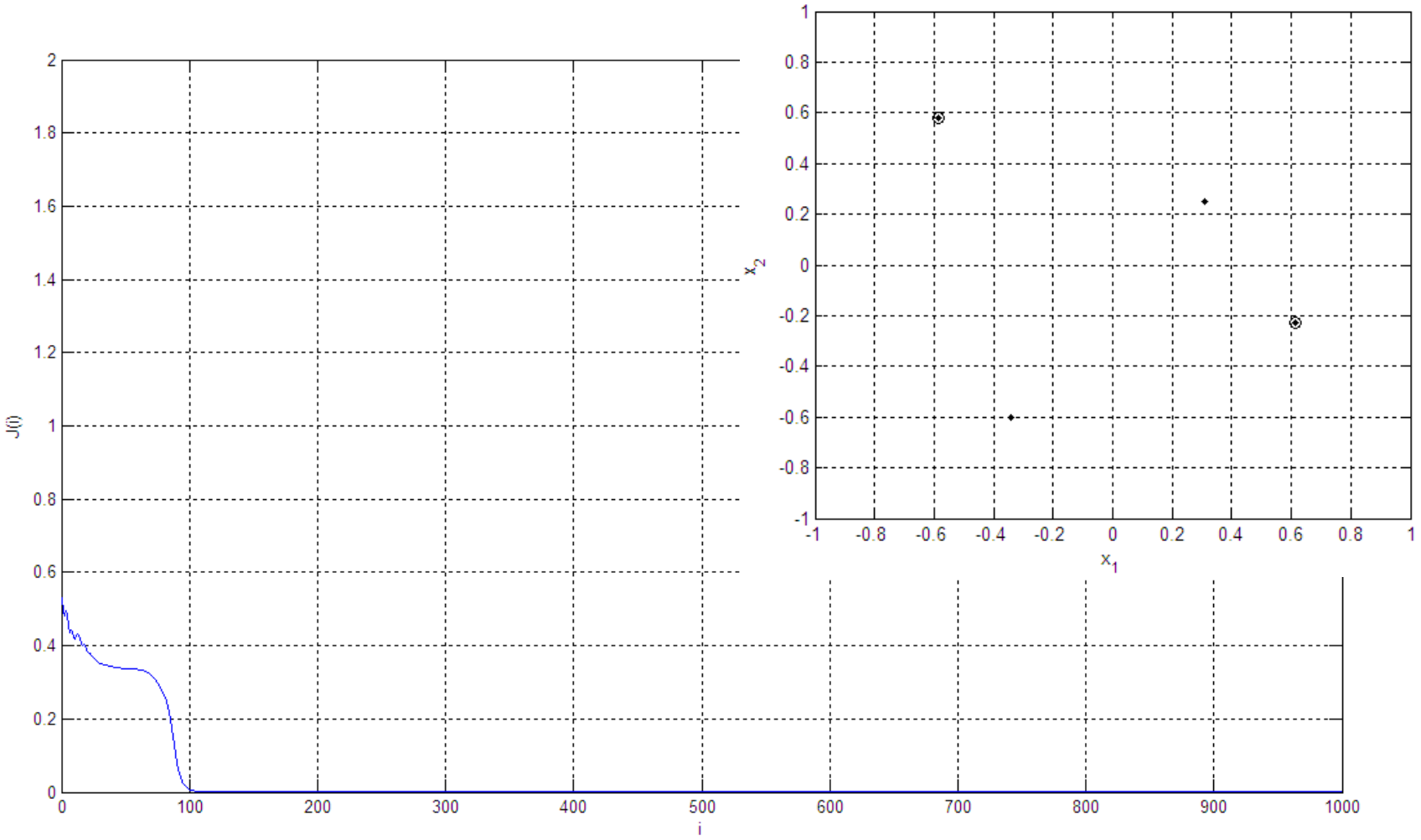
Momento – Exemplo Seqüencial



Momento – Exemplo *Batch*

```
% 070628 gabriel@pads.ufrj.br (2D)
% Exemplo1BatchMomentum.m

close all; clear all;

% [A] Data

C = [0 0 1 1 ; 0 1 0 1];
rand('state',0)
X = []; P = 1;
for k = 1:size(C,2),
   X = [X 0.7*rand(2,P)+repmat(C(:,k),1,P)];
end;
X = X - repmat(mean(X,2),1,size(X,2));
plot(X(1,:),X(2,:),'k.');
t = []; T = [1 -1 -1 1];
for k = 1:size(C,2),
   t = [t repmat(T(k),1,P)];
end;

% [A1] Randomize Data Order

randn('state',0);
X = [X ; t ; randn(1,size(X,2))]';
X = sortrows(X,4)';
t = X(3,:); X = X(1:2,:);

% [B] Network Init

% [B1] Parameters

K = 2; % Number of Layers
eta = 1; % Learning Rate Initial Value
Delta = 1e-6; % Stop Criterion
N = size(X,2); % Number of Input Vectors
E = 4*P; % Number of Feed-Forward Iterations per Epoch
alpha = 0.99; % Learning Rate Decay Factor
mu = 0.65; % Momentum constant

mu = 0.9;
eta = 0.1;
alpha = 1;

% [B2] Layers

L(1).W = rand(2,2)-0.5;
L(1).b = rand(2,1)-0.5;
L(2).W = rand(1,2)-0.5;
L(2).b = rand(1,1)-0.5;

for k=1:K,
   L(k).vb = zeros(size(L(k).b));
   L(k).vW = zeros(size(L(k).W));
end;

% [C] Batch Error Backpropagation Training

n=1; i=1; fim=0;
while not(fim),

   for k=1:K,
      L(k).db = zeros(size(L(k).b));
      L(k).dW = zeros(size(L(k).W));
   end;
   J(i) = 0;
   for ep=1:E,

      % [C1] Feed-Forward

      L(1).x = X(:,n);
      for k = 1:K,
         L(k).u = L(k).W*L(k).x + L(k).b;
         L(k).o = tanh(L(k).u);
         L(k+1).x = L(k).o;
      end;
      e = t(n) - L(K).o;
      J(i) = J(i) + (e'*e)/2;

      % [C2] Error Backpropagation

      L(K+1).alpha = e; L(K+1).W = eye(length(e));
      for k = fliplr(1:K),
         L(k).M = eye(length(L(k).o)) - diag(L(k).o)^2;
         L(k).alpha = L(k).M*L(k+1).W'*L(k+1).alpha;
         L(k).db = L(k).db + L(k).alpha;
         L(k).dW = L(k).dW + kron(L(k).x',L(k).alpha);
      end;
      n = n+1; if n>N, n=1; end;

   end;

   % [C3] Updates

   for k = 1:K,
      L(k).vb = eta*L(k).db + mu*L(k).vb;
      L(k).b = L(k).b + L(k).vb;
      L(k).vW = eta*L(k).dW + mu*L(k).vW;
      L(k).W = L(k).W + L(k).vW;
   end;
   J(i) = J(i)/E;

   % [C4] Stop criterion

   if (i>1),
      if (abs(J(i)-J(i-1))/J(i) < Delta)|(i>1000),
         fim = 1;
      end;
   end;
   if not(fim)
      i = i+1; if n>N, n=1; end; eta = eta*alpha;
   end;

end;

% [D] Test

hold on;
for n = 1:size(X,2),
   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   if L(K).o < 0, plot(X(1,n),X(2,n),'ko'); end;
end;
figure; plot(J);
```

Momento – Exemplo *Batch*



III. Métodos de Segunda Ordem

- 1ª ordem (gradiente – steepest descent) (Haykin p. 161-175):

$$\mathbf{v}_i = -\eta \frac{\partial J}{\partial \mathbf{p}_i}$$

- Ordem superior:

Hessiana (Haykin p. 204)

$$J(\mathbf{p} + \Delta\mathbf{p}) = J(\mathbf{p}) + \left(\frac{\partial J}{\partial \mathbf{p}} \right)^T \Delta\mathbf{p} + \frac{1}{2} \Delta\mathbf{p}^T \mathbf{H} \Delta\mathbf{p} + \ldots$$

- Momento (Haykin p. 169-171):

$$\mathbf{v}_i = -\eta \frac{\partial J}{\partial \mathbf{p}_i} + \mu \mathbf{v}_{i-1}$$

Métodos de Segunda Ordem

- Método de Newton (2ª ordem) (Haykin p. 235):

$$\mathbf{v}_i = \mathbf{H}^{-1} \frac{\partial J}{\partial \mathbf{p}_i}$$

- Exemplo: $f(x) = x^2$ $x_0 = 3$ $f(x_0) = 9$ $f(0) = \min_x f(x)$

$$\left.\frac{df}{dx}\right|_{x=x_0} = 6 \quad \frac{\partial J}{\partial \mathbf{p}} \qquad \frac{d^2 f}{dx^2} = 2 \quad \mathbf{H} \qquad x_0 - \mathbf{H}^{-1} \frac{\partial J}{\partial \mathbf{p}} = 0$$

- Acelera muito a convergência (1 iteração, se a superfície de erro $J(\mathbf{p})$ for quadrática).
- Problema: cálculo de $\mathbf{H}^{-1}$ (Haykin p. 224)

Métodos de Segunda Ordem

- Método Gradiente Conjugado (2ª ordem) (Haykin p. 236-243):

$$\mathbf{p}_0, \mathbf{s}_0 = -\left.\frac{\partial J}{\partial \mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}_0} \quad (\mathbf{r}_0 = \mathbf{s}_0)$$

$$\mathbf{p}_1 = \mathbf{p}_0 + \eta_0 \mathbf{s}_0 \text{ (usar valor ótimo de } \eta_0)$$

$$\mathbf{r}_1 = -\left.\frac{\partial J}{\partial \mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}_1}$$

$$\beta_1 = \max\left(\frac{\mathbf{r}_1^T(\mathbf{r}_1 - \mathbf{r}_0)}{\mathbf{r}_0^T \mathbf{r}_0}, 0\right)$$

$$\mathbf{s}_1 = \mathbf{r}_1 + \beta_1 \mathbf{s}_0$$

$||\mathbf{r}_1 < \epsilon||\mathbf{r}_0||$? — Não → $i = i + 1$ $(1 \longrightarrow 2)$ $(0 \longrightarrow 1)$

Sim → Fim

- traincgp.m

Métodos de Segunda Ordem

- Método Quasi-Newton (2ª ordem) (Haykin p. 242-244):

$$\mathbf{v}_i = -\eta \mathbf{S}_i \frac{\partial J}{\partial \mathbf{p}_i}$$

Positiva Definida

  - trainbfg.m

- Levenberg-Marquardt (LM) (2ª ordem):

$$\mathbf{H} := \mathbf{H} + \epsilon \mathbf{I}, \text{ “}\epsilon \approx 0\text{”}$$

Evitar mau-condicionamento de H.

  - trainlm.m

- Conclusão: algoritmo lento quando número de parâmetros é elevado. Problema de velocidade não resolvido por causa da complexidade computacional dos métodos de 2ª ordem. Usar momento.

IV. Regularização – Exemplo $\lambda = 0$

$f(x) = \sin(2\pi x)$ $\qquad$ $n_n \sim N(0, \sigma^2)$ $\qquad$ $\sigma = 0.2$

n	x_n	$f(x)$	t_n	v_n
1	0.0	0.00	−0.09	−0.04
2	0.1	0.59	0.25	0.73
3	0.2	0.95	0.98	0.83
4	0.3	0.95	1.01	1.39
5	0.4	0.59	0.36	0.56
6	0.5	0.00	0.24	0.02
7	0.6	−0.59	−0.35	−0.37
8	0.7	−0.95	−0.96	−0.94
9	0.8	−0.95	−0.89	−0.97
$N = 10$	0.9	−0.59	−0.55	−0.75

$\mathcal{D} = \{(x_1, t_1), (x_2, t_2), \ldots, (x_N, t_N)\}$

Regularização – Exemplo λ = 0

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

$$\mathbf{w} = \begin{bmatrix} w_0 \\ \vdots \\ w_M \end{bmatrix}$$

$$E_n = \frac{(y(x_n, \mathbf{w}) - t_n)^2}{2}$$

$$E(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} E_n =$$

$$\frac{1}{2N} \sum_{n=1}^{N} (w_0 + w_1 x_n + w_2 x_n^2 + \ldots + w_M x_n^M - t_n)^2$$

Regularização – Exemplo $\lambda = 0$

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} E(\mathbf{w})$$

$$\left. \frac{dE}{d\mathbf{w}} \right|_{\mathbf{w}=\mathbf{w}^*} = \mathbf{0}$$

$$\begin{bmatrix} \frac{\partial E}{\partial w_1} \\ \frac{\partial E}{\partial w_2} \\ \vdots \\ \frac{\partial E}{\partial w_M} \end{bmatrix} = \mathbf{0}$$

Regularização – Exemplo λ = 0

$$E = \frac{1}{N}(E_1 + E_2 + \ldots + E_N)$$

$$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial E_1}\frac{\partial E_1}{\partial w_j} + \frac{\partial E}{\partial E_2}\frac{\partial E_2}{\partial w_j} + \ldots + \frac{\partial E}{\partial E_N}\frac{\partial E_N}{\partial w_j}$$

$$\frac{\partial E}{\partial w_j} = \frac{1}{N}\sum_{n=1}^{N}\frac{\partial E_n}{\partial w_j}$$

$$E_n = \frac{1}{2}(w_0 + w_1 x_n + w_2 x_n^2 + \ldots + w_M x^M - t_n)^2$$

$$\frac{\partial E_n}{\partial w_j} = x_n^j(w_0 + w_1 x_n + w_2 x_n^2 + \ldots + w_M x^M - t_n)$$

Regularização – Exemplo $\lambda = 0$

$$\left.\frac{\partial E}{\partial w_j}\right|_{w=w^*} = \frac{1}{N}\sum_{n=1}^{N} x_n^j(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0$$

$$\begin{aligned}
j = 0 \qquad & \sum_{n=1}^{N}(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0 \\
j = 1 \qquad & \sum_{n=1}^{N} x_n(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0 \\
j = 2 \qquad & \sum_{n=1}^{N} x_n^2(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0 \\
& \vdots \\
j \qquad & \sum_{n=1}^{N} x_n^j(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0 \\
& \vdots \\
j = M \qquad & \sum_{n=1}^{N} x_n^M(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) = 0
\end{aligned}$$

Regularização – Exemplo $\lambda = 0$

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^M \\ 1 & x_2 & x_2^2 & \cdots & x_2^M \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^M \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}$$

$$\mathbf{e} = \mathbf{X}\mathbf{w}^* - \mathbf{t}$$

$$\mathbf{X}^T(\mathbf{X}\mathbf{w}^* - \mathbf{t}) = \mathbf{0}$$

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t}$$

Regularização – Exemplo $\lambda = 0$

```
clear all;
x = ((0:9)/10)'; y = sin(2*pi*x); randn('state',0); t = y + 0.2*randn(10,1);
a = 0; b = 1; pts = 5000; stp = (b-a)/pts; h = a:stp:(b-stp); plot(h,sin(2*pi*h),'g-');
X = [ones(size(x)) x]; w = inv(X'*X)*X'*t; H = [ones(size(h))' h']; hold on; plot(h,H*w,'k-');
axis([0 1 -2 2]); grid on; v = y + 0.2*randn(10,1); [mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^2]; w = inv(X'*X)*X'*t; H = [H (h').^2]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^3]; w = inv(X'*X)*X'*t; H = [H (h').^3]; hold on; plot(h,H*w,'b-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^4]; w = inv(X'*X)*X'*t; H = [H (h').^4]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^5]; w = inv(X'*X)*X'*t; H = [H (h').^5]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^6]; w = inv(X'*X)*X'*t; H = [H (h').^6]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^7]; w = inv(X'*X)*X'*t; H = [H (h').^7]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^8]; w = inv(X'*X)*X'*t; H = [H (h').^8]; hold on; plot(h,H*w,'k-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
X = [X x.^9]; w = inv(X'*X)*X'*t; H = [H (h').^9]; hold on; plot(h,H*w,'r-');
[mean((X*w-t).^2)/2 mean((X*w-v).^2)/2]
xlabel('x'); plot(x,t,'r.'); plot(x,v,'b.');
```

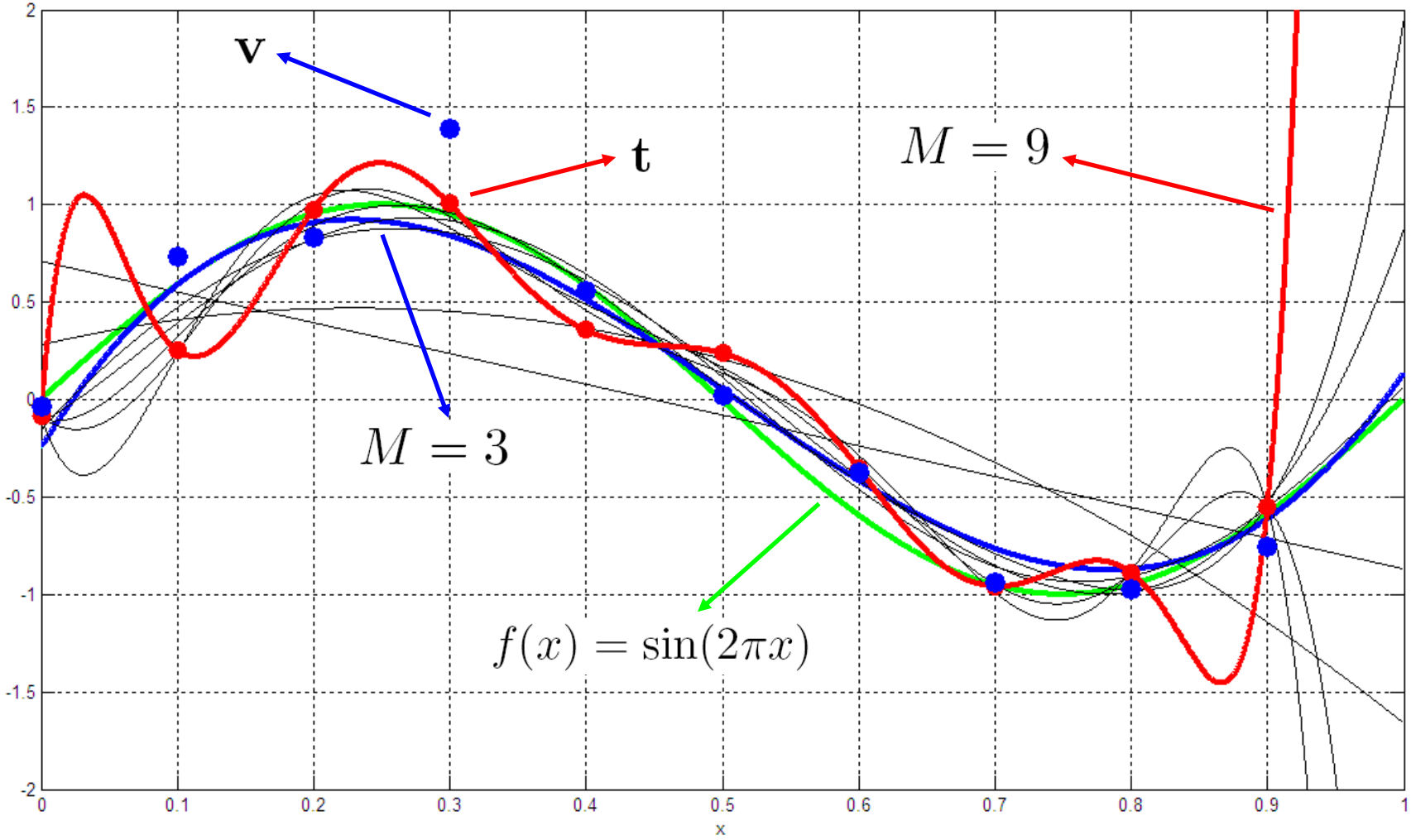
Regularização – Exemplo λ = 0

M	1	2	3	4	5	6	7	8	9
w_0^*	0.71	-0.28	-0.25	-0.16	-0.12	-0.10	-0.09	-0.09	-0.09
w_1^*	-1.58	1.67	11.35	7.42	3.98	-2.06	-21.92	-5.68	87.72
w_2^*		-3.61	-31.96	-9.89	21.98	104.75	468.77	104.94	-2292.16
w_3^*			21.01	-18.33	-118.44	-514.26	-2912.83	136.35	24126.90
w_4^*				21.85	149.35	999.25	8534.91	-4246.01	-129472.60
w_5^*					-56.67	-896.67	-13106.88	16458.29	395572.50
w_6^*						311.11	10156.70	-28168.52	-717835.60
w_7^*							-3125.58	22945.99	766203.60
w_8^*								-7242.10	-444171.10
w_9^*									107883.40
E_R	0.1151	0.0811	0.0136	0.0098	0.0085	0.0070	0.0022	0.0016	0.0000
E_V	0.1421	0.1008	0.0217	0.0223	0.0208	0.0219	0.0313	0.0275	0.0266

- **A complexidade do modelo pode ser controlada através de restrição na norma do vetor w:**

$$\sum w_j^2$$

Regularização – Exemplo λ = 0



Regularização – Exemplo $\lambda \neq 0$

- **A complexidade do modelo pode ser controlada através de restrição na norma do vetor w:**

$$\sum w_j^2$$

- Considere então:

$$\mathbf{w}^T\mathbf{w}$$

$$\tilde{E}(\mathbf{w}) = \sum_{n=1}^{N} E_n + \frac{\lambda}{2}||\mathbf{w}||^2$$

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(y(x_n, \mathbf{w}) - t_n)^2 + \frac{\lambda}{2}\sum_{j=0}^{M} w_j^2$$

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}}\tilde{E}(\mathbf{w})$$

Regularização – Exemplo $\lambda \neq 0$

$$\left.\frac{\partial \tilde{E}}{\partial w_j}\right|_{w=w^*} = \frac{1}{N}\sum_{n=1}^{N} x_n^j(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_j^* = 0$$

$$\begin{aligned}
j = 0 \qquad & \sum_{n=1}^{N}(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_0^* = 0 \\
j = 1 \qquad & \sum_{n=1}^{N} x_n(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_1^* = 0 \\
j = 2 \qquad & \sum_{n=1}^{N} x_n^2(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_2^* = 0 \\
& \vdots \\
j \qquad & \sum_{n=1}^{N} x_n^j(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_j^* = 0 \\
& \vdots \\
j = M \qquad & \sum_{n=1}^{N} x_n^M(w_0^* + w_1^* x_n + w_2^* x_n^2 + \ldots + w_M^* x^M - t_n) + \lambda w_M^* = 0
\end{aligned}$$

Regularização – Exemplo $\lambda \neq 0$

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^M \\ 1 & x_2 & x_2^2 & \cdots & x_2^M \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^M \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}$$

$$\mathbf{e} = \mathbf{X}\mathbf{w}^* - \mathbf{t}$$

$$\mathbf{X}^T(\mathbf{X}\mathbf{w}^* - \mathbf{t}) + \lambda\mathbf{w}^* = \mathbf{0}$$

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{t}$$

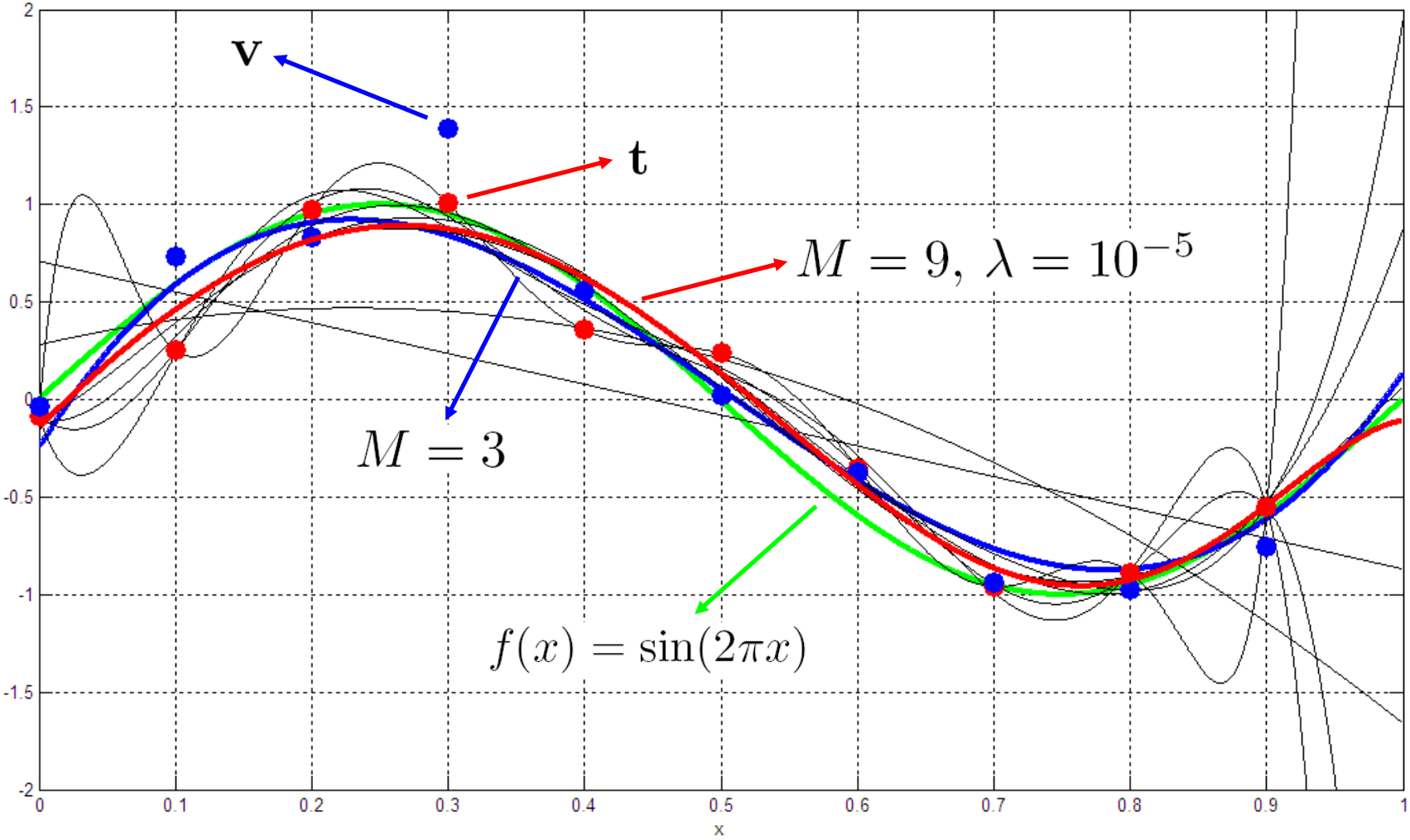
Regularização – Exemplo $\lambda \neq 0$

M=[]; lambda = 1e-5; w = inv(X'*X + lambda*eye(10))*X'*t;
plot(h,H*w,'r-'); M = [M [[mean((X*w-t).^2)/2 ; mean((X*w-v).^2)/2] ; w]]

$-\log_{10}\lambda$	1	2	3	4	5	6	7	8	9	10
w_0^*	0.48	0.24	-0.01	-0.13	-0.14	-0.13	-0.12	-0.10	-0.09	-0.08
w_1^*	-0.00	2.47	5.52	7.38	7.02	5.77	3.33	-2.30	-10.99	-12.79
w_2^*	-1.11	-3.65	-8.55	-12.18	-8.17	3.60	31.71	115.41	247.92	271.51
w_3^*	-0.90	-2.94	-4.39	-6.46	-15.81	-51.08	-160.61	-580.87	-1257.67	-1341.98
w_4^*	-0.47	-1.24	0.18	2.70	5.66	38.27	201.63	1050.37	2427.26	2422.42
w_5^*	-0.10	0.12	2.36	6.11	10.74	22.39	-13.10	-446.14	-1118.88	-587.02
w_6^*	0.16	1.01	2.75	5.32	7.14	-8.17	-110.92	-705.25	-1756.10	-2577.72
w_7^*	0.34	1.53	2.20	2.59	2.26	-12.73	-2.03	308.85	827.99	769.11
w_8^*	0.44	1.78	1.25	-0.76	-2.18	-2.10	84.46	781.14	2025.70	2905.18
w_9^*	0.50	1.87	0.22	-4.10	-6.61	4.46	-33.59	-524.73	-1396.79	-1863.17
E_R	0.0797	0.0309	0.0137	0.0098	0.0094	0.0089	0.0080	0.0047	0.0019	0.0017
E_V	0.1050	0.0516	0.0304	0.0224	0.0213	0.0209	0.0207	0.0221	0.0270	0.0288

- **Problema passa a ser a escolha do valor ideal para λ, com base em métodos estatísticos (p(w|D) = ...)**

Regularização – Exemplo $\lambda \neq 0$



Teorema de Bayes

$$p(x) = \sum_y p(x, y) \qquad p(x, y) = p(y|x)p(x)$$

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$p(x) = \sum_y p(x|y)p(y)$$

Teorema de Bayes

$$\mathcal{D} = \{t_1, t_2, \ldots, t_N\}$$

$$p(\mathbf{w}|\mathcal{D}) = ?$$

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

Likelihood

Prior

Posterior

$$\int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})d\mathbf{w}$$

- ***Likelihood***
  Estimador "freqüentista" (diversos conjuntos D)
  Estimador Bayesiano (um só conjunto D)

- *Bayesian Linear Regression* (Bishop, p. 152-161)

- trainbr.m

V. MLP como Aproximador Universal

- Mapeamento contínuo qualquer: $\mathbf{y} = f(\mathbf{x})$ $(\mathbf{x} \in \mathbb{R}^M$ e $\mathbf{y} \in \mathbb{R}^N)$

- Qual é o número mínimo de camadas escondidas que o MLP deve ter, para que ele possa realizar uma aproximação desta função contínua ?

- Teorema da Aproximação Universal

Teorema da Aproximação Universal

- Seja ϕ uma função limitada, monotonicamente crescente, e contínua. Considere o hipercubo unitário com M dimensões: $[0,1]^M$. O espaço das funções contínuas em $[0,1]^M$ é chamado de $C([0,1]^M)$. Considere uma função f qualquer, pertencente ao $C([0,1]^M)$ e considere um número real arbitrariamente pequeno, $\epsilon > 0$.

Teorema da Aproximação Universal

- Então: existem um número inteiro P , e existem conjuntos de constantes reais w_{ij}, b_i e α_i (com $i = 1, \ldots, P$ e $j = 1, \ldots, M$ tais que:

$$F(x_1, \ldots, x_M) = \sum_{i=1}^{P} \alpha_i \phi \left(\sum_{j=1}^{M} w_{ij} x_j + b_i \right)$$

Aproxima f com erro absoluto menor que ϵ, ou seja:

$$|F(x_1, \ldots, x_M) - f(x_1, \ldots, x_M)| < \epsilon$$

Para qualquer vetor $(x_1, \ldots, x_M) \in C([0,1]^M)$.

Observações

1. Haykin, Seção 4.13.

2. Teorema de Weierstrass (1885): “Qualquer função contínua sobre um intervalo fechado no eixo real pode ser expressa (neste intervalo) como uma série de polinômios absoluta e uniformemente convergente.” ($f \longrightarrow$ série polinomial ; $\phi \longrightarrow$ série polinomial)

3. Demonstração rigorosa específica para MLPs:
   - G. Cybenko. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems, 2: pp. 304-314, 1989.
   - Funahashi (1989).
   - Hornik, Stinchcombe, White (1989).
   - Light (1992) (desenvolvimentos posteriores).

Observações

4. Correspondência com um MLP tendo uma só camada escondida:

Entradas do MLP: x_1 até x_M

Pesos da camada escondida: w_{ij}

Biases da camada escondida: b_i

Neurônios da camada escondida: $i = 1, \ldots, P$

Saídas da camada escondida: $\phi_i = \phi\left(\sum_{j=1}^{M} w_{ij} x_j + b_i\right)$

Camada de saída: $F = \sum_{i=1}^{P} \alpha_i \phi_i$ (*)

(*) Um só neurônio – se a saída for vetorial ($\mathbb{R}^N$), pode-se repetir o procedimento acima N vezes e colocar os N MLPs em paralelo.

Observações

Logo, um MLP com topologia M–P–N (uma só camada escondida, arbitrariamente grande – tamanho P) aproxima qualquer f em $C([0,1]^M)$.

5. Teorema de existência – podemos tentar resolver um problema, porque a solução por MLP existe. Teorema não-construtivo: não especifica um procedimento para obter o MLP.

6. Outros detalhes não considerados no Teor. Aproximação Universal:

- Tempo de treinamento (interação entre w_{ij})
- Facilidade de implementação (P muito elevado)
- Generalização (*overfitting* ocorre quando P muito elevado)
- Qualidade de mínimos locais (interação entre w_{ij})

Observações

Com respeito a estes quatro detalhes, pode ser vantajoso usar mais do que uma camada escondida. Considerações empíricas sobre mais de uma camada escondida:

- Camada Escondida #1 – *Local Features*

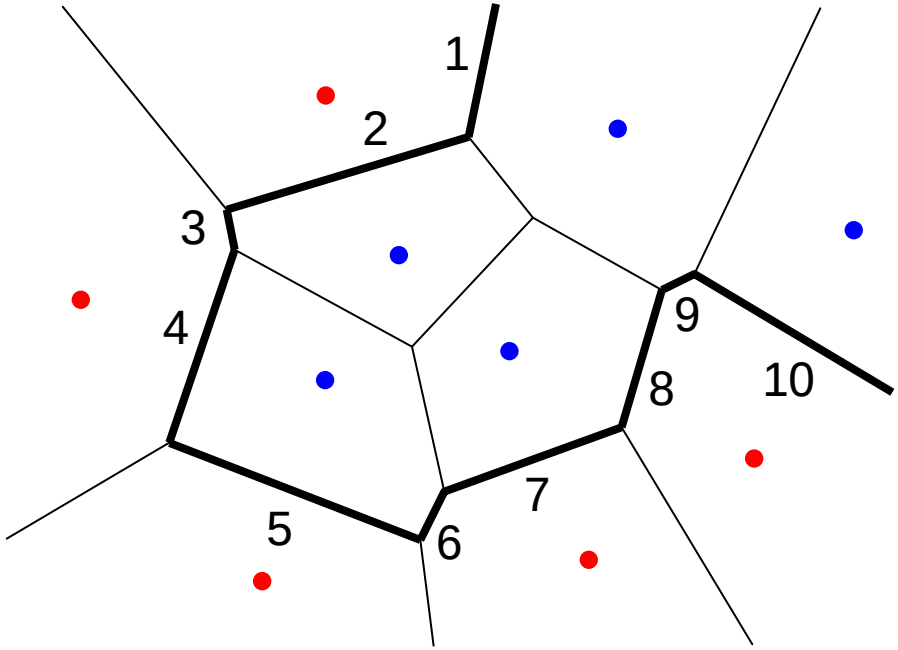
- Camada Escondida #2 – *Global Features*

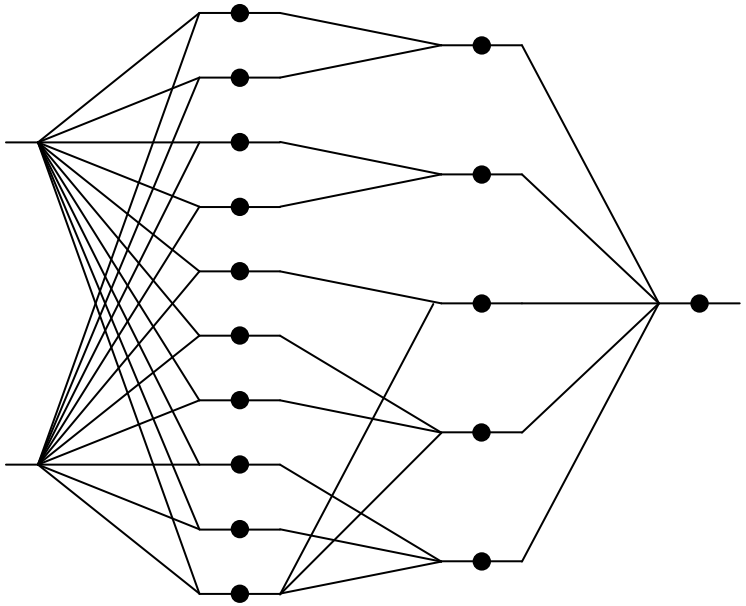
VI. Métodos Geométricos

- Diagramas de Voronoi (Bose, 1992)
- Kernel PCA (Schölkopf, 1998)

Diagramas de Voronoi

- "Utilização da estrutura geométrica inerente a um dado problema".



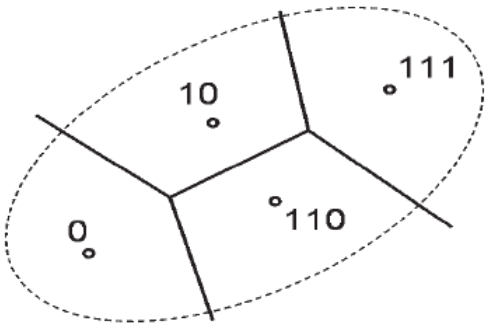


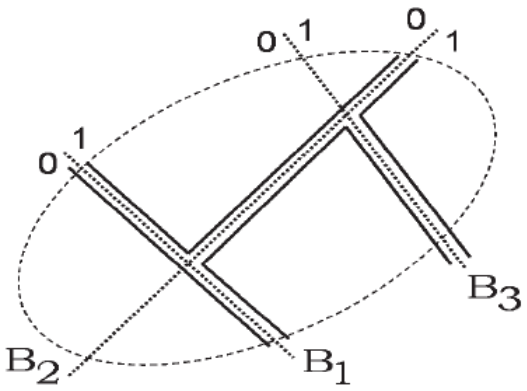
- Síntese não-única.

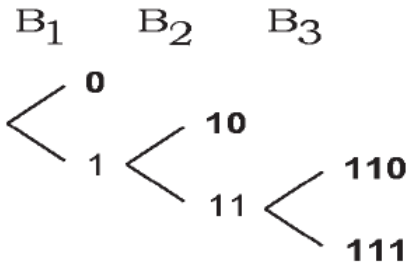
- Algoritmo de difícil extensão para dimensões mais elevadas.

Diagramas de Voronoi

- Redução de complexidade







Kernel PCA

- “Solução linear em um espaço de dimensão muito elevada”.

- Definições:

$$\phi : \mathbb{R}^M \longrightarrow \mathbb{R}^I$$

$$\mathbf{q}(\mathbf{x}(n)) = \phi(\mathbf{x}(n)) \ (\text{ou } \mathbf{q}(n))$$

$$\mathbf{Q} = [\mathbf{q}(1) \ \mathbf{q}(2) \ \ldots \ \mathbf{q}(N)]$$

$$\mathbf{R}' = \frac{\mathbf{Q}\mathbf{Q}^T}{N}$$

Kernel PCA

Autovetor

$$\mathbf{R}'\mathbf{v}_j = \rho_j \mathbf{v}_j$$

$$\mathbf{v}_j = \sum_i \mathbf{a}_{ij}\mathbf{q}(i) = \mathbf{Q}\mathbf{a}_j$$

$$\left[\begin{array}{c} \mathbf{q}^T 1 \dfrac{\mathbf{Q}\mathbf{Q}^T}{N}\mathbf{Q}\mathbf{a}_j = \rho_j \mathbf{q}^T(1)\mathbf{Q}\mathbf{a}_j \\ \vdots \\ \mathbf{q}^T N \dfrac{\mathbf{Q}\mathbf{Q}^T}{N}\mathbf{Q}\mathbf{a}_j = \rho_j \mathbf{q}^T(N)\mathbf{Q}\mathbf{a}_j \end{array} \right]$$

$$\mathbf{K} = \mathbf{Q}^T\mathbf{Q}$$

Kernel PCA

$$k(\mathbf{x},\mathbf{y}) = \tanh(g\mathbf{x}^T\mathbf{y} + b)$$

$$k(\mathbf{x},\mathbf{y}) = \exp\left(\frac{-||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}\right)$$

$$[\mathbf{K}]_{ij} = \mathbf{q}^T(i)\mathbf{q}(j) = k(\mathbf{x}(i),\mathbf{x}(j))$$

Normalização

$$\mathbf{K}^2\mathbf{a}_j = N\rho_j\mathbf{K}\mathbf{a}_j \qquad ||\mathbf{a}_j|| = 1$$

$$\mathbf{A} = \begin{bmatrix} | & | & & | \\ \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_N \\ | & | & & | \end{bmatrix}$$

Kernel PCA

N' Dados Fixos: $\mathbf{l}_j \qquad j = 1, \ldots, N'$

$$\left[\begin{array}{cccc} \mathbf{l}_1 & \mathbf{l}_2 & \ldots & \mathbf{l}_{N'} \end{array} \right] = \left[\begin{array}{cccc} \mathbf{0} & \mathbf{e}_1 & \mathbf{e}_2 & \ldots & \mathbf{e}_{N'-1} \end{array} \right]$$

$$\mathbf{L} = \left[\begin{array}{cccc} \mathbf{l}_1 & \mathbf{l}_2 & \ldots & \mathbf{l}_{M+1} \end{array} \right] = \left[\begin{array}{cccc} \mathbf{0} & \mathbf{e}_1 & \mathbf{e}_2 & \ldots & \mathbf{e}_M \end{array} \right]$$

$$\mathbf{x}(n) \longrightarrow \mathbf{q}(\mathbf{x}(n))$$

$$\mathbf{z}(n) = \left[\begin{array}{c} k(\mathbf{l}_1, \mathbf{x}(n)) \\ \vdots \\ k(\mathbf{l}_{N'}, \mathbf{x}(n)) \end{array} \right] \begin{array}{l} \longleftarrow \mathbf{q}^T(1)\mathbf{q}(\mathbf{x}(n)) \\ \\ \longleftarrow \mathbf{q}^T(N')\mathbf{q}(\mathbf{x}(n)) \end{array}$$

$$\mathbf{f} = \mathbf{A}^T \mathbf{z}(n)$$

Kernel PCA

$$\mathbf{f} = \mathbf{A}^T \mathbf{z}(n)$$

$$\begin{bmatrix} \mathbf{f}_1 = \mathbf{v}_1^T \mathbf{q} = \mathbf{a}_1^T \mathbf{Q}^T \mathbf{q} \\ \mathbf{f}_2 = \mathbf{v}_2^T \mathbf{q} = \mathbf{a}_2^T \mathbf{Q}^T \mathbf{q} \\ \vdots \\ \mathbf{f}_{N'} = \mathbf{v}_{N'}^T \mathbf{q} = \mathbf{a}_{N'}^T \mathbf{Q}^T \mathbf{q} \end{bmatrix}$$

Zero-centering

$$\mathbf{K}_c = \mathbf{K} - \mathbf{U}_0\mathbf{K} - \mathbf{K}\mathbf{U}_0 + \mathbf{U}_0\mathbf{K}\mathbf{U}_0$$

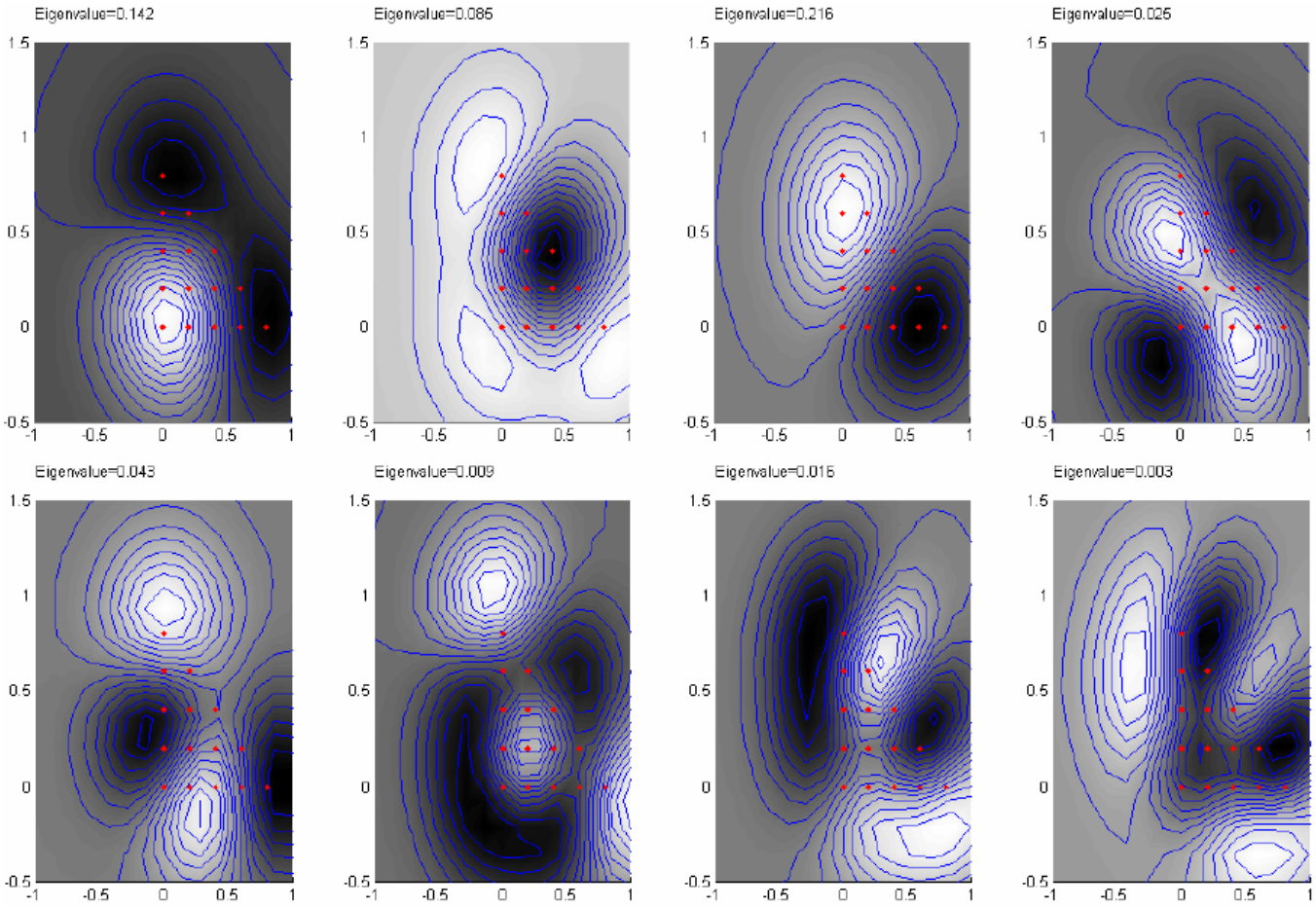
Camada Escondida

$$\mathbf{z}_c = (\mathbf{I} - \mathbf{U}_0)\mathbf{z} + \mathbf{U}_0\mathbf{K}^T\mathbf{U}_0 - \mathbf{K}^T\mathbf{U}_0$$

Camada de Saída

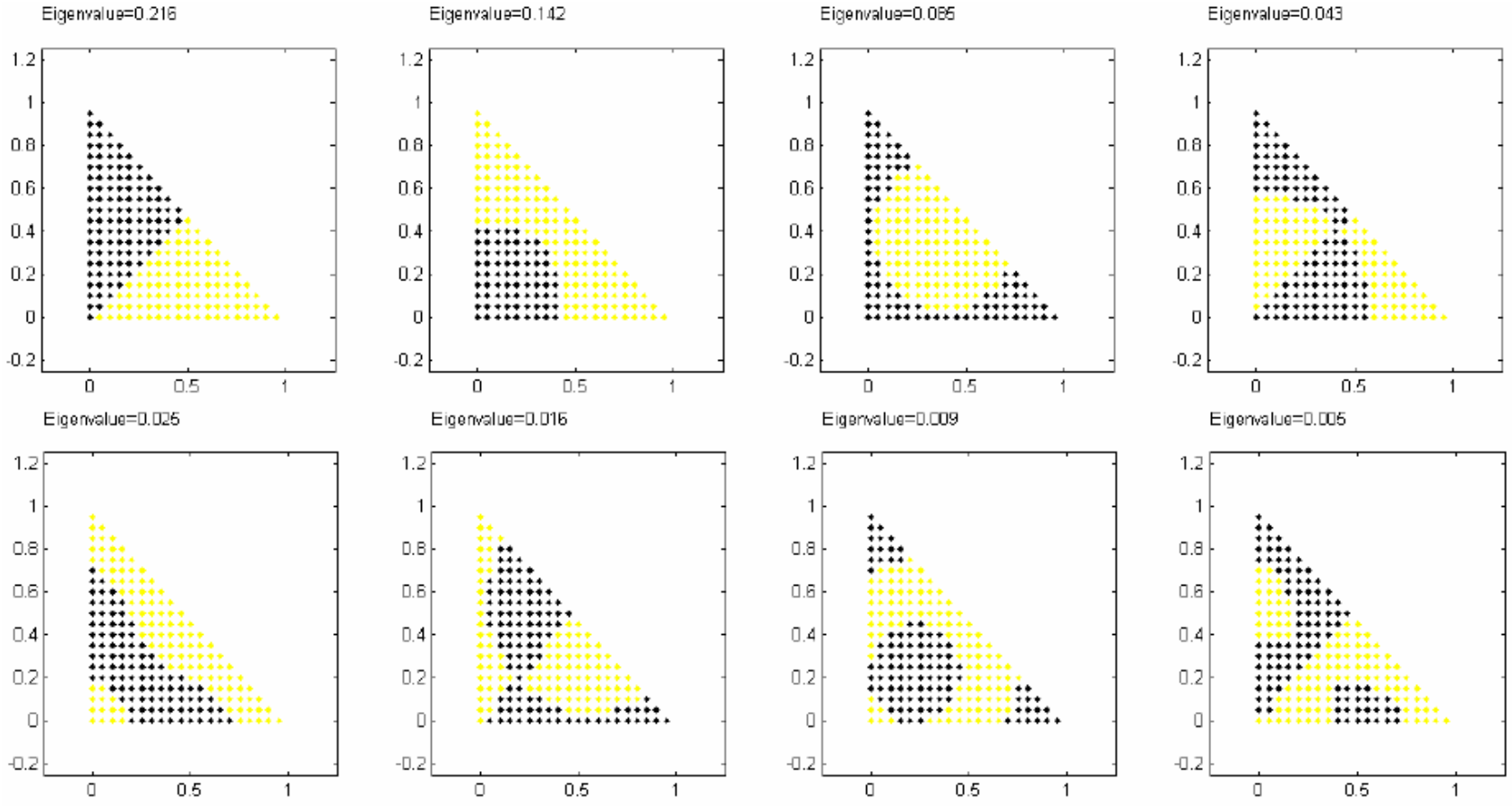
$$\mathbf{f}(n) = \mathbf{A}^T \mathbf{z}_c(n) = \mathbf{W}\mathbf{z}(n) + \mathbf{b}$$

Kernel PCA



[Figura gerada com código cedido por Bernhard Schölkopf]

Kernel PCA



[Figura gerada com código cedido por Bernhard Schölkopf]

VII. RBF – Radial-Basis Functions

- Teorema de Cover (1965) sobre a Separabilidade de Padrões:

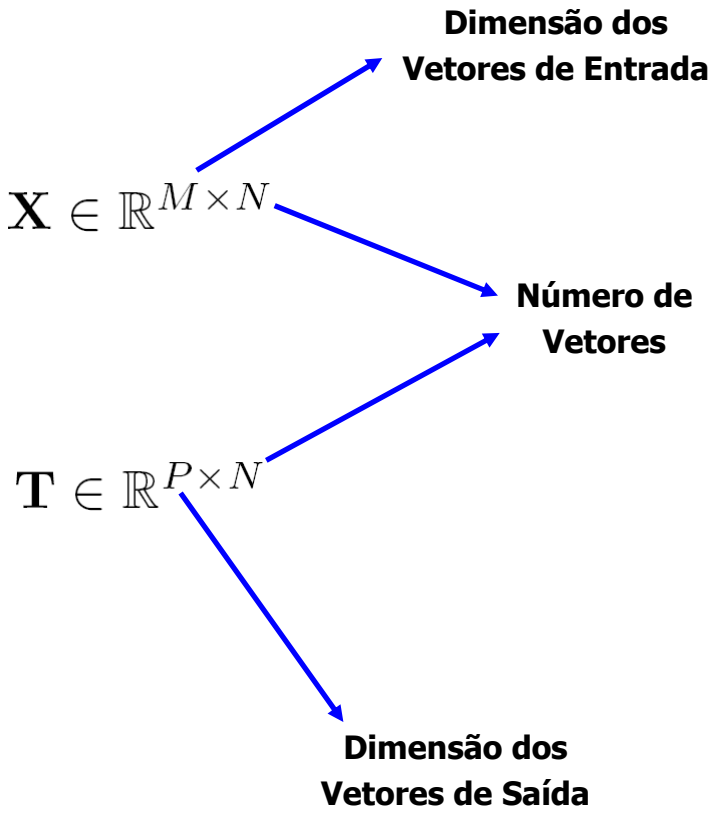
  "Um problema complexo de classificação de padrões colocado em um espaço de dimensão elevada tem mais chance de ser resolvido linearmente (ou seja, de ser um problema descrito por classes linearmente separáveis) do que quando colocado em um espaço de dimensão baixa."

RBF – Radial-Basis Functions

- Exemplo: dicotomia (partição binária do espaço)

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \\ | & | & & | \end{bmatrix}$$

$$\mathbf{T} = \begin{bmatrix} | & | & & | \\ \mathbf{t}_1 & \mathbf{t}_2 & \cdots & \mathbf{t}_N \\ | & | & & | \end{bmatrix}$$



Classe $\mathcal{C}_0$: $\mathcal{C}_0 = \{\mathbf{x}_n | t_n = 0\}$

Classe $\mathcal{C}_1$: $\mathcal{C}_1 = \{\mathbf{x}_n | t_n = 1\}$

RBF – Radial-Basis Functions

- Seja: $\varphi_i(x) : \mathbb{R}^M \longrightarrow \mathbb{R},\ i = 1, \ldots, I$

- $\varphi_i(x)$ recebe o nome de “função escondida”.

$$\varphi(\mathbf{x}_n) = \begin{bmatrix} \varphi_1(\mathbf{x}_n) \\ \varphi_2(\mathbf{x}_n) \\ \vdots \\ \varphi_I(\mathbf{x}_n) \end{bmatrix}$$

- Espaço gerado pelo conjunto de vetores $\mathbf{x}_n \longrightarrow \varphi(\mathbf{x}_n)$: “espaço escondido” ou “espaço de propriedades” (*feature space*).

RBF – Radial-Basis Functions

- Dizemos que as classes $\mathcal{C}_0$ e $\mathcal{C}_1$ são linearmente separáveis em φ se existe $\mathbf{w} \in \mathbb{R}^I$ tal que:

$$\mathbf{w}^T \varphi(\mathbf{x}) < 0 \longrightarrow \mathbf{x} \in \mathcal{C}_0$$

$$\mathbf{w}^T \varphi(\mathbf{x}) > 0 \longrightarrow \mathbf{x} \in \mathcal{C}_1$$

- No espaço escondido $\mathbf{w}^T \varphi(\mathbf{x}) = 0$ é um (hiper)plano. No espaço $\mathbb{R}^M$ dos vetores $\mathbf{x}$ (espaço de entrada), a superfície não-linear $\mathbf{x} : \mathbf{w}^T \varphi(\mathbf{x}) = 0$ é chamada superfície de separação.

Exemplo 1 – RBF

$$\mathbf{X} = \left[\begin{array}{cccc|cccc|cccc|cccc} 0.46 & 0.51 & 0.39 & 0.62 & 0.53 & 0.48 & 0.44 & 0.49 & -0.39 & -0.51 & -0.47 & -0.43 & -0.57 & -0.37 & -0.64 & -0.54 \\ -0.67 & -0.47 & -0.38 & -0.50 & 0.52 & 0.57 & 0.72 & 0.51 & -0.49 & -0.58 & -0.63 & -0.34 & 0.59 & 0.34 & 0.56 & 0.57 \end{array}\right]$$

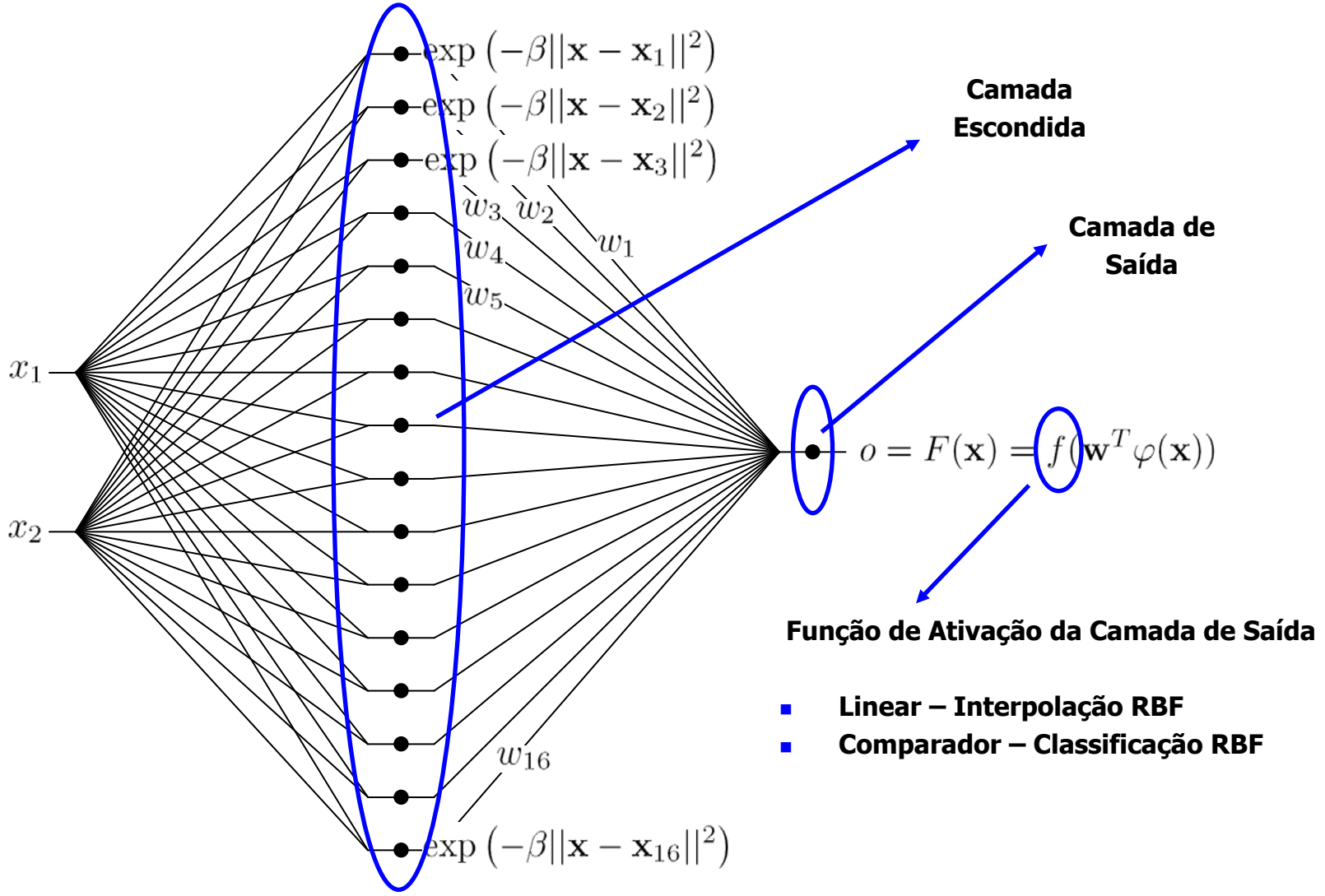
$$\mathbf{T} = \left[\begin{array}{cccc|cccc|cccc|cccc} -1.00 & -1.00 & -1.00 & -1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & -1.00 & -1.00 & -1.00 & -1.00 \end{array}\right]$$

$$\varphi_i(\mathbf{x}) = \exp\left(-\beta||\mathbf{x} - \mathbf{x}_i||^2\right),\ i = 1, \ldots, N \qquad \beta = \frac{1}{2\sigma^2}$$

$$I = N$$

$$N = 16$$

Exemplo 1 – RBF



RBF – Radial-Basis Functions

- Interpolação RBF: $f(\mathbf{w}^T\varphi(\mathbf{x})) = \mathbf{w}^T\varphi(\mathbf{x})$

- $\varphi(\mathbf{x})$ – função de base radial: $\varphi(||\mathbf{x} - \mathbf{y}||)$

- Então: $F(\mathbf{x}) = \mathbf{w}^T\varphi(\mathbf{x}) = \sum_{i=1}^{N} w_i\varphi(||\mathbf{x} - \mathbf{x}_i||)$

RBF – Radial-Basis Functions

$$\mathbf{x}_1 : \varphi(||\mathbf{x}_1 - \mathbf{x}_1||)w_1 + \varphi(||\mathbf{x}_1 - \mathbf{x}_2||)w_2 + \ldots + \varphi(||\mathbf{x}_1 - \mathbf{x}_N||)w_N = t_1$$

$$\mathbf{x}_2 : \varphi(||\mathbf{x}_2 - \mathbf{x}_1||)w_1 + \varphi(||\mathbf{x}_2 - \mathbf{x}_2||)w_2 + \ldots + \varphi(||\mathbf{x}_2 - \mathbf{x}_N||)w_N = t_2$$

$$\vdots$$

$$\mathbf{x}_N : \varphi(||\mathbf{x}_N - \mathbf{x}_1||)w_1 + \varphi(||\mathbf{x}_N - \mathbf{x}_2||)w_2 + \ldots + \varphi(||\mathbf{x}_N - \mathbf{x}_N||)w_N = t_N$$

$$\varphi_{ij} = (||\mathbf{x}_i - \mathbf{x}_j||) \ (i, j = 1, \ldots, N)$$

RBF – Radial-Basis Functions

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}$$

$$\mathbf{w} = \varphi^{-1}\mathbf{t}^T$$

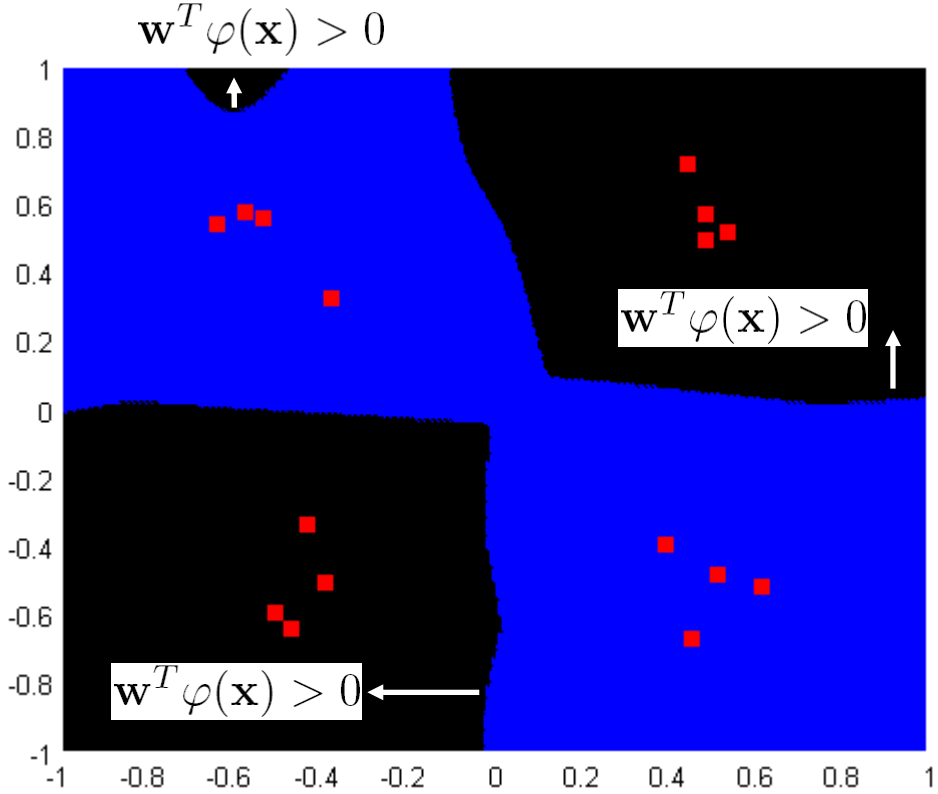
Exemplo 1 – RBF

```
% 070712 gabriel@pads.ufrj.br

close all; clear all;
C = [0.5 0.5 -0.5 -0.5 ; -0.5 0.5 -0.5 0.5];
t = [-1 1 1 -1];
randn('state',0);
E = repmat(C(:,1),1,4) + 0.1*randn(2,4);
F = repmat(C(:,2),1,4) + 0.1*randn(2,4);
G = repmat(C(:,3),1,4) + 0.1*randn(2,4);
H = repmat(C(:,4),1,4) + 0.1*randn(2,4);
X = [E F G H]; t = [-1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1];
% P = 1;
P = 100; for i = 1:16, for j = 1:16, Phi(i,j) = exp(-P*norm(X(:,i)-X(:,j))^2); end; end;
w = inv(Phi)*t'
X = zeros(2,10000);
for i = 1:100,
   for j = 1:100,
      X(:,100*(i-1)+j) = [(i-50.5)/50 ; (j-50.5)/50];
   end;
end;
Y = [E F G H];
figure; hold on;
for n=1:size(X,2),
o = exp(-P*sum((Y-repmat(X(:,n),1,16)).^2,1))*w;
if o < 0,
plot(X(1,n),X(2,n),'b.');
else
plot(X(1,n),X(2,n),'k.');
end;
end;
plot(Y(1,:),Y(2,:),'r.');
```

Exemplo 1 – RBF

$$\beta = 100$$

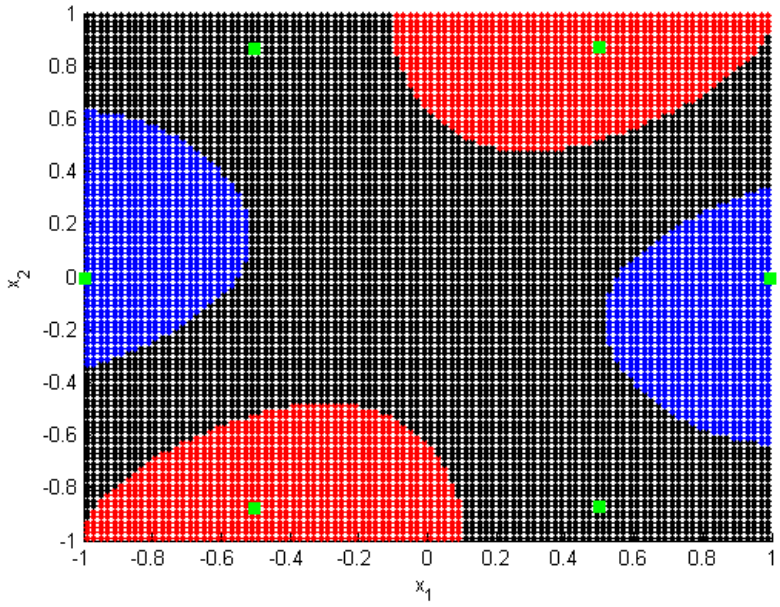
$$\mathbf{w} = \begin{bmatrix} -0.99 \\ -0.67 \\ -0.94 \\ -0.80 \\ 0.54 \\ 0.41 \\ 0.95 \\ 0.27 \\ 0.82 \\ 0.50 \\ 0.60 \\ 0.94 \\ 0.44 \\ -1.00 \\ -0.86 \\ -1.10 \end{bmatrix}$$

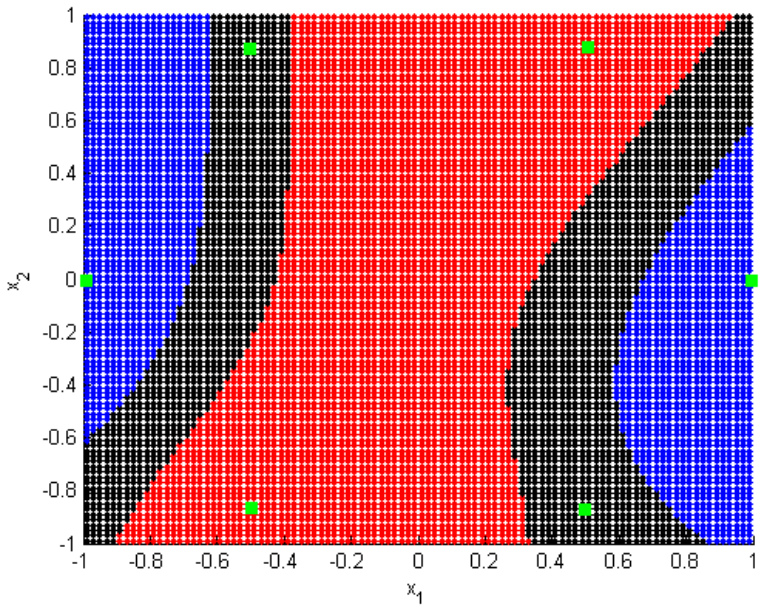


Exemplo 2 – RBF

```
% 070712 gabriel@pads.ufrj.br – ExemploRBF2.m

% RBF

close all; clear all;
X = [ 1.0000   0.5000  -0.5000  -1.0000  -0.5000   0.5000 ; ...
           0  -0.8660  -0.8660  -0.0000   0.8660   0.8660 ];
t = [-1 0 1 -1 0 1];
P = 2; for i = 1:6, for j = 1:6, Phi(i,j) = exp(-P*norm(X(:,i)- ...
X(:,j))^2); end; end;
w = inv(Phi)*t'
Y = X; X = zeros(2,10000);
for i = 1:100,
   for j = 1:100,
      X(:,100*(i-1)+j) = [(i-50.5)/50 ; (j-50.5)/50];
   end;
end;
figure; hold on;
for n=1:size(X,2),
   o = exp(-P*sum((Y-repmat(X(:,n),1,6)).^2,1))*w;
   if o < -0.5,
      plot(X(1,n),X(2,n),'b.');
   elseif o < 0.5,
      plot(X(1,n),X(2,n),'k.');
   else
      plot(X(1,n),X(2,n),'r.');
   end;
end;
plot(Y(1,:),Y(2,:),'g.');

% MLP

% Parameters

X = Y; rand('state',0); K = 2; Delta = 1e-5; N = size(X,2); E = 6;
eta = 0.02; alpha = 1; mu = 0.65; MaxIter = 400;
L(1).W = rand(6,2)-0.5; L(1).b = rand(6,1)-0.5;
L(2).W = rand(1,6)-0.5; L(2).b = rand(1,1)-0.5;

for k=1:K,
   L(k).vb = zeros(size(L(k).b));
   L(k).vW = zeros(size(L(k).W));
end;

% Batch Error Backpropagation Training

n=1; i=1; fim=0;

while not(fim),

   for k=1:K,
      L(k).db = zeros(size(L(k).b));
      L(k).dW = zeros(size(L(k).W));
   end;
   J(i) = 0;
   for ep=1:E,

      % Feed-Forward

      L(1).x = X(:,n);
      for k = 1:K,
         L(k).u = L(k).W*L(k).x + L(k).b;
         L(k).o = tanh(L(k).u);
         L(k+1).x = L(k).o;
      end;
      e = t(n) - L(K).o;
      J(i) = J(i) + (e'*e)/2;

      % Error Backpropagation

      L(K+1).alpha = e; L(K+1).W = eye(length(e));
      for k = fliplr(1:K),
         L(k).M = eye(length(L(k).o)) - diag(L(k).o)^2;
         L(k).alpha = L(k).M*L(k+1).W'*L(k+1).alpha;
         L(k).db = L(k).db + L(k).alpha;
         L(k).dW = L(k).dW + kron(L(k).x',L(k).alpha);
      end;
      n = n+1; if n>N, n=1; end;

   end;

   % Updates

   for k = 1:K,
      L(k).vb = eta*L(k).db + mu*L(k).vb;
      L(k).b = L(k).b + L(k).vb;
      L(k).vW = eta*L(k).dW + mu*L(k).vW;
      L(k).W = L(k).W + L(k).vW;
   end;
   J(i) = J(i)/E;

   if rem(i,100)==0,
      EMQ = J(i)
   end;

   % Stop criterion

      if (i>1),
         if (abs(J(i)-J(i-1))/J(i) < Delta)|(i>MaxIter),
            fim = 1;
         end;
      end;
      if not(fim)
         i = i+1; if n>N, n=1; end; eta = eta*alpha;
      end;

end;

% [D] Test

X = zeros(2,10000);
for i = 1:100,
   for j = 1:100,
      X(:,100*(i-1)+j) = [(i-50.5)/50 ; (j-50.5)/50];
   end;
end;
figure; hold on;
for n = 1:size(X,2),
   L(1).x = X(:,n);
   for k = 1:K,
      L(k).u = L(k).W*L(k).x + L(k).b;
      L(k).o = tanh(L(k).u);
      L(k+1).x = L(k).o;
   end;
   if L(K).o < -0.5,
      plot(X(1,n),X(2,n),'b.');
   elseif L(K).o < 0.5,
      plot(X(1,n),X(2,n),'k.');
   else
      plot(X(1,n),X(2,n),'r.');
   end;
end;
plot(Y(1,:),Y(2,:),'g.');
```

Exemplo 2 – RBF





VIII. Outros Tipos de Rede Neural

- Redes Neurais Recorrentes (Realimentadas)
- *Kohonen Self-Organizing Maps* (VQ/ECVQ + Vizinhança)
- Hopfield (Memória Associativa)
- *Support Vector Machines*
- Modelos de Grossberg , ART (Padrões)
- Redes sem Pesos
- Árvores de Decisão

IX. Áreas de Aplicações

- Modelos ARMA Não-Lineares
- Predição de Séries Temporais
  Residuais ; Escolha de Variáveis Relevantes
  Correlação ; Qualidade de Dados ; Tendência
- Visão Computacional / Reconhecimento
- Compressão de Dados
- Inteligência Artificial etc.

IX. Exemplos de Aplicações

- Compressão de Dados com Baixa Complexidade
- Produção Mensal de Bebida
- Análise de Dados Geográficos
- Consumo de Gasolina de Veículo Específico
- Classificação (Partículas, Sonar, Potência, TME)
- www.20q.org

Referências

- Simon Haykin, Neural Networks and Learning Machines, 3rd Edition, Prentice-Hall, 2008.

- Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

- Ivan N. da Silva, Danilo H. Spatti e Rogério A. Flauzino, Redes Neurais Artificiais: para Engenharia e Ciências Aplicadas, Ed. Artliber, 2010.

- Russel D. Reed e Robert J. Marks, Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks, MIT Press, 1999.

Referências

- Christopher M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
- Nirmal K. Bose and Ping Liang, Neural Network Fundamentals with Graphs, Algorithms, and Applications, McGraw-Hill, 1996.
- Teuvo Kohonen, Self-Organizing Maps, 3rd Edition, Springer-Verlag, 2001.
- Vladimir N. Vapnik, Statistical Learning Theory, Wiley, 1998.
- David B. Fogel e Charles J. Robinson, Computational Intelligence – The Experts Speak, IEEE / Wiley-Interscience, 2003.

Referências

- MATLAB Help Documentation.
- P. Wasserman, Neural Computing, Van Nostrand Reinhold, 1989, Capítulos 1 a 6.
- W. S. Sarle - ftp://ftp.sas.com/pub/neural/FAQ.html.
- IEEE Computational Intelligence Society; IJCNN.
- IEEE Transactions on Neural Networks.

Referências

- Sociedade Brasileira de Redes Neurais, SBRN, www.sbrn.org.br.
- SNNS (Stuttgart Neural Network Simulator) ftp.informatik.uni-stuttgart.de
- NeuroLab (UFRJ e CEPEL) www.lps.ufrj.br/~caloba
- B. Wandell, A. El Gamal, B. Girod, Common Principles of Image Acquisition Systems and Biological Vision, Proceedings of the IEEE, vol. 90, no. 1, pp. 5-17, Janeiro 2002.

Agradecimentos

- Prof. Guilherme de Alencar Barreto (UFC/DETI)

- 

- 

Perguntas e Comentários ?

gabriel@pads.ufrj.br