第7章 典型神经网络

Chapter 7 Typical Neural Networks

内容提要

7.1 单神经元网络

7.2 BP神经网络

7.3 RBF神经网络

7.1 单神经元网络

图7-1中 u_i为神经元的内部状态，θ_i 为阈值，x_j为输入信号，$j=1,\cdots,n$，w_{ij}为表示从单元 u_i 到单元u_j的连接权系数，s_i 为外部输入信号。

$$Net_i = \sum_j w_{ij} x_j + s_i - \theta_i$$

$$u_i = f(Net_i)$$

$$y_i = g(u_i) = h(Net_i)$$

通常情况下，取 $g(u_i) = u_i$

即 $y_i = f(Net_i)$

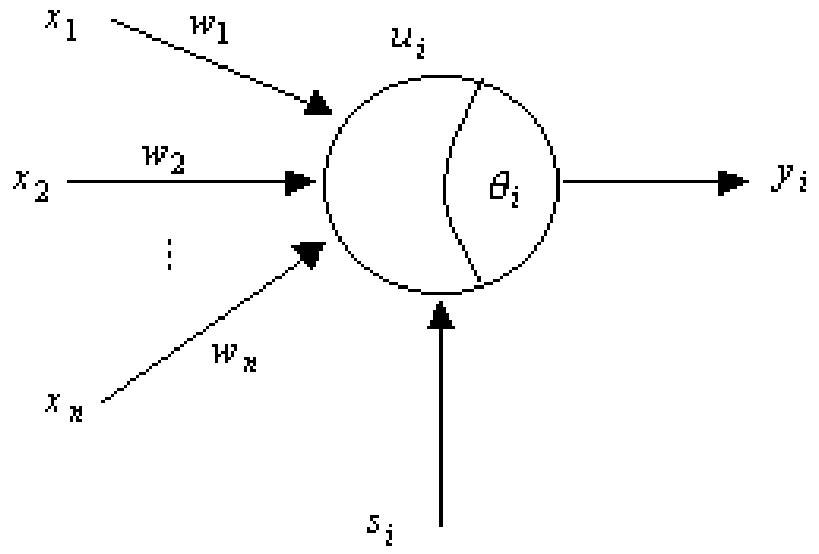
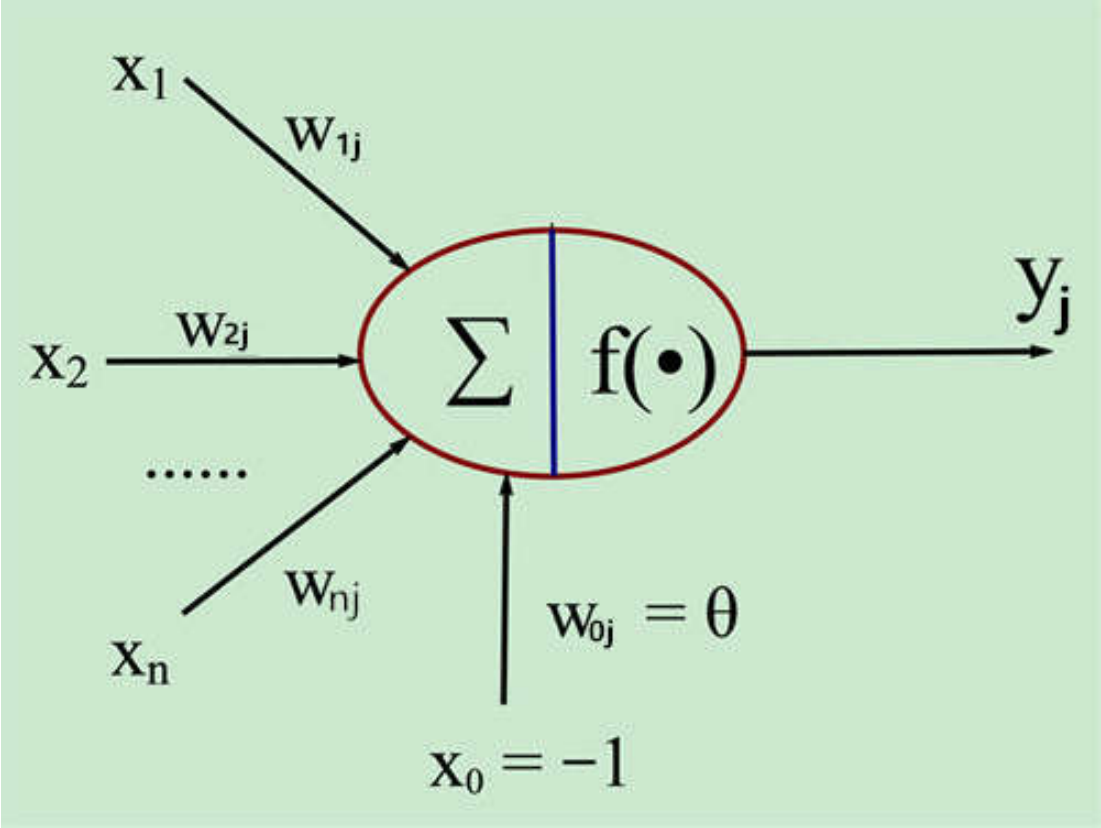


图7–1　单神经元模型

7.1 单神经元网络



7.1 单神经元网络

生物神经元与 MP 模型

生物神经元	神经元	输入信号	权值	输出	总和	膜电位	阈值
MP 模型	j	χ_i	ω_{ij}	y_j	$\sum$	$\sum_{i=1}^{n}\omega_{ij}\chi_i(t)$	θ_j

结合M-P模型示意图来看，对于某一个神经元j（注意别混淆成变量了，在这里j只是起到标识某个神经元的作用），它可能接受同时接受了许多个输入信号，用 χi 表示。

由于生物神经元具有不同的突触性质和突触强度，所以对神经元的影响不同，我们用权值 ω_{ij}来表示，

- 其正负模拟了生物神经元中突出的兴奋和抑制，
- 其大小则代表了突出的不同连接强度。

θ_j表示为一个阈值（threshold），或称为偏置（bias）。

7.1 单神经元网络

由于累加性，我们对全部输入信号进行累加整合，相当于生物神经元中的膜电位（水的变化总量），其值就为：

$$net_j(t) = \sum_{i=1}^{n} \omega_{ij} x_i(t) - \theta_j$$

神经元激活与否（外接专用水管流出与否）取决于某一阈值电平（水位高度），即只有当其输入总和超过阈值θ_j时，神经元才被激活而发放脉冲，否则神经元不会发生输出信号。整个过程可以用下面这个函数来表示：

$$y_j = f(net_j)$$

y_j表示神经元j的输出，函数f称为激活函数 (Activation Function)或转移函数 (Transfer Function)，$net_j(t)$称为净激活(net activation)。 若将阈值看成是神经元j的一个输入x0的权重w_{0j}，则上面的式子可以简化为：

$$net_j(t) = \sum_{i=0}^{n} \omega_{ij} x_i(t) \qquad y_j = f(net_j)$$

7.1 单神经元网络

若用X表示输入向量，用W表示权重向量，即：

$$X = [\chi_0, \chi_1, \cdots, \chi_n] \qquad W = \begin{bmatrix} \omega_{0j} \\ \omega_{1j} \\ \vdots \\ \omega_{nj} \end{bmatrix}$$

则神经元的输出可以表示为向量相乘的形式：

$$net_j = XW$$

$$y_j = f(net_j) = f(XW)$$

若神经元的净激活*net*为正，称该神经元处于激活状态或兴奋状态(fire)，若净激活*net*为负，则称神经元处于抑制状态。

7.1 单神经元网络

由此我们可以得到总结出M-P模型的6个特点：

1. 每个神经元都是一个多输入单输出的信息处理单元；
2. 神经元输入分兴奋性输入和抑制性输入两种类型；
3. 神经元具有空间整合特性和阈值特性；
4. 神经元输入与输出间有固定的时滞，主要取决于突触延搁；
5. 忽略时间整合作用和不应期；
6. 神经元本身是非时变的，即其突触时延和突触强度均为常数。

7.1 单神经元网络

前面4点和生物神经元保持一致。 结合公式来看，

- 输入χi的下标 i=1,2,..., ni=1,2,...,n，输出 yj 的下标j 体现了第1个特点“多输入单输出”；
- 权重值 ωij 的正负体现了第2个特点中“突触的兴奋与抑制”；
- θj代表第3个特点 中的阈值，当 netj(t)−Tj>0 时，神经元才能被激活；
- 为了简单起见，对膜电位的计算netj(t) 并没有考虑时间整合，只考虑了空间整合，即只对每条神经末梢传来的信号根据权重进行累加整合，而没有考虑输入输出间的突触时延,体现了第5个特点。

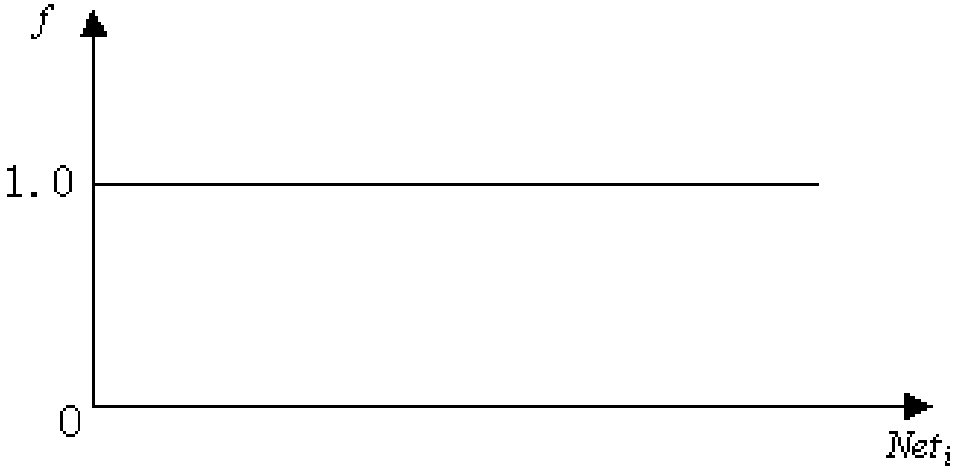
这种“阈值加权和”的神经元模型称为M-P模型
(McCulloch-Pitts Model)，也称为神经网络的一个处理单元
(PE, Processing Element)。

7.1 单神经元网络

常用的神经元非线性特性有以下3种：

（1）阈值型　　hardlim@Matlab

$$f(Net_i)=\begin{cases}1 & Net_i \geq 0 \quad \text{对应于神经元兴奋} \\ 0 & Net_i < 0 \quad \text{对应于神经元抑制}\end{cases}$$

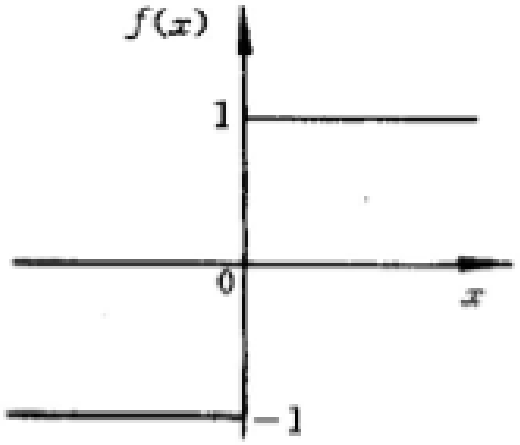


- 生物学背景：
  神经细胞的兴奋与抑制（单极性）

7.1 单神经元网络

符号函数　hardlims@Matlab

$$\mathrm{sgn}(x) = f(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

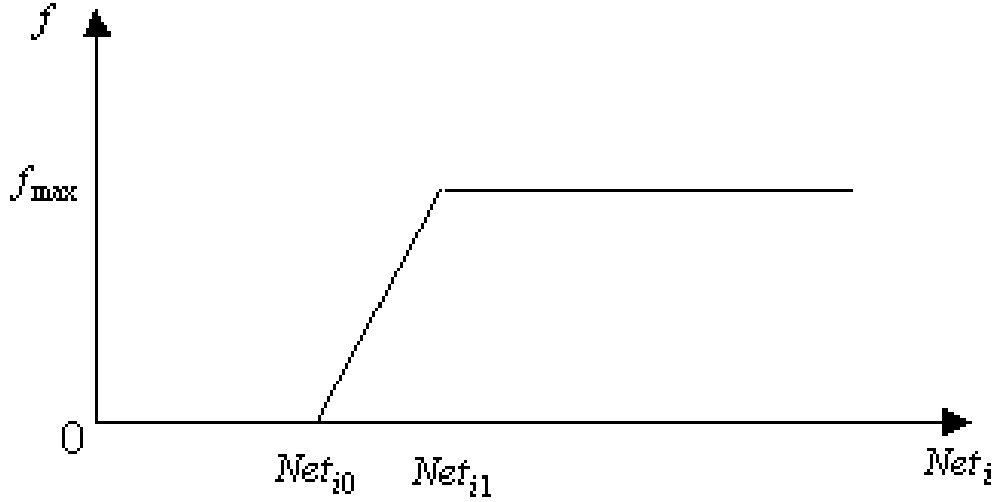


- 生物学背景：
  神经细胞的比例作用（全方位）

7.1 单神经元网络

（2）分段线性型

$$f(Net_i)=\begin{cases}0 & Net_i > Net_{i0}\\ kNet_i & Net_{i0} < Net_i < Net_{il}\\ f_{\max} & Net_i \geq Net_{il}\end{cases}$$

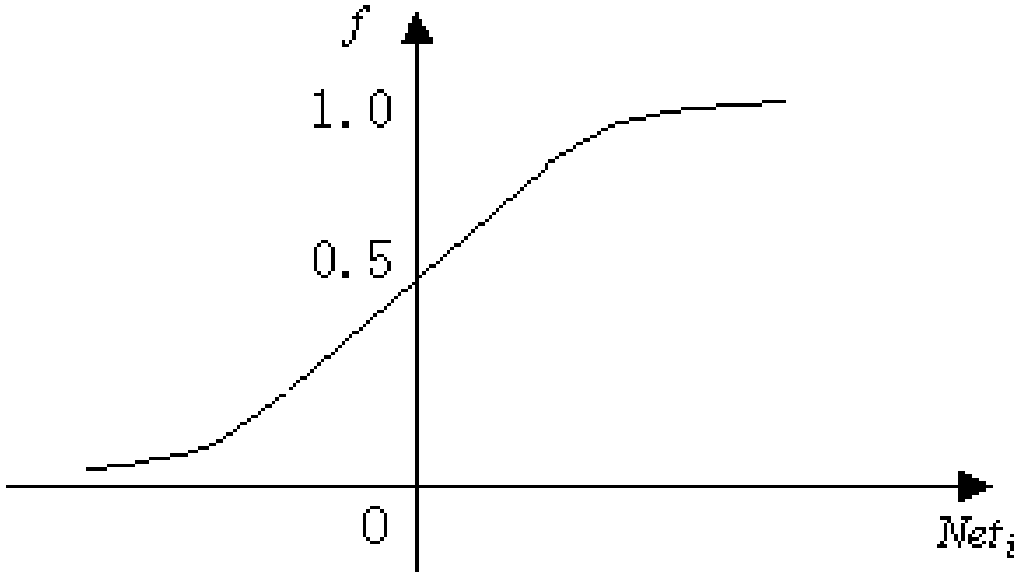


分段线性函数

7.1 单神经元网络

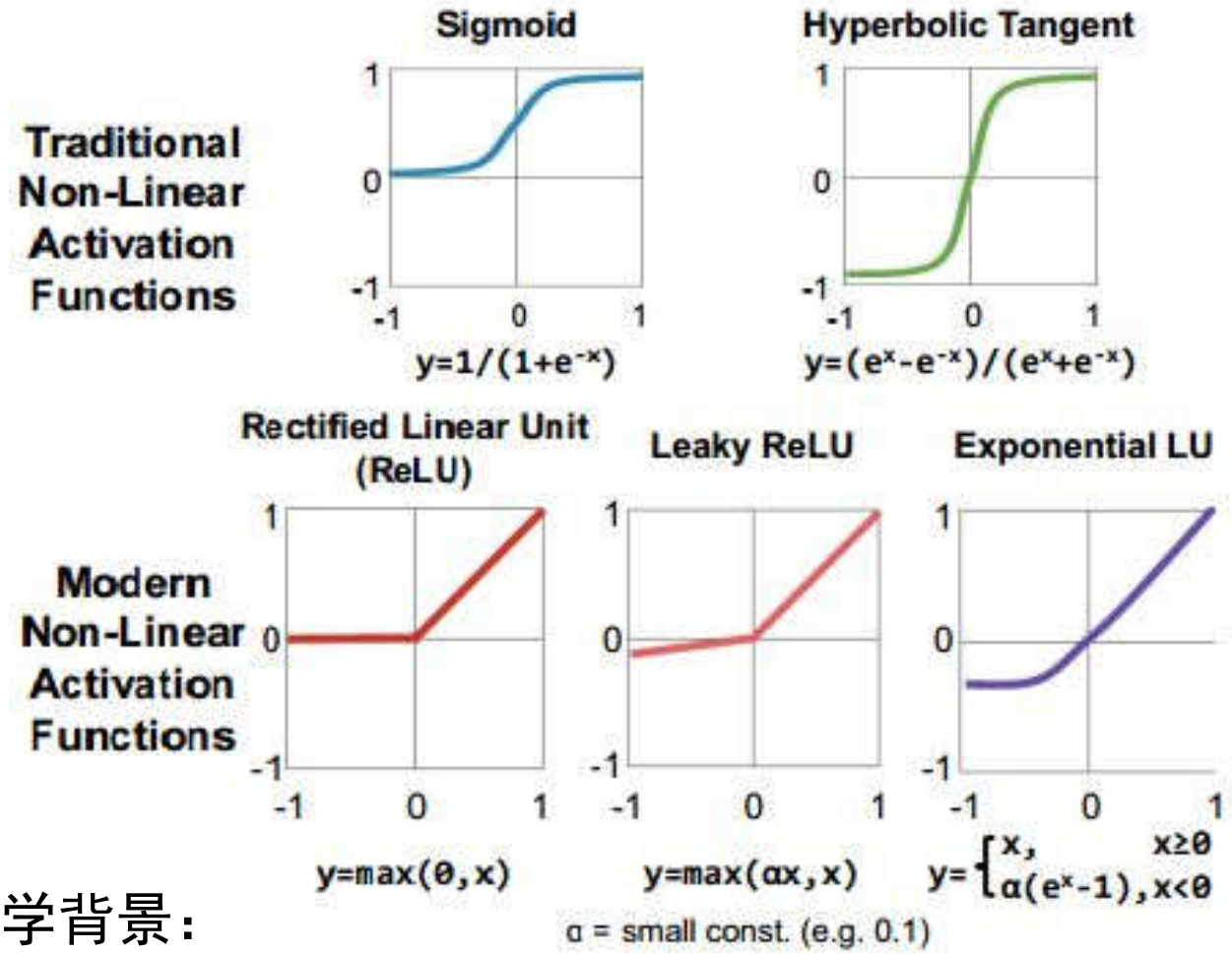
（3）Sigmoid函数型　logsig@Matlab

$$f(Net_i) = \frac{1}{1 + e^{-\frac{Net_i}{T}}}$$



- 生物学背景：神经细胞的非线性比例作用（单向）
- 阈值型函数具有不连续、不光滑等不太好的性质，因此实际常用Sigmoid函数。
- 把可能在较大范围内变化的输入值挤压到(0,1)输出范围内，因此也称为“挤压函数”(squashing function)

7.1 单神经元网络



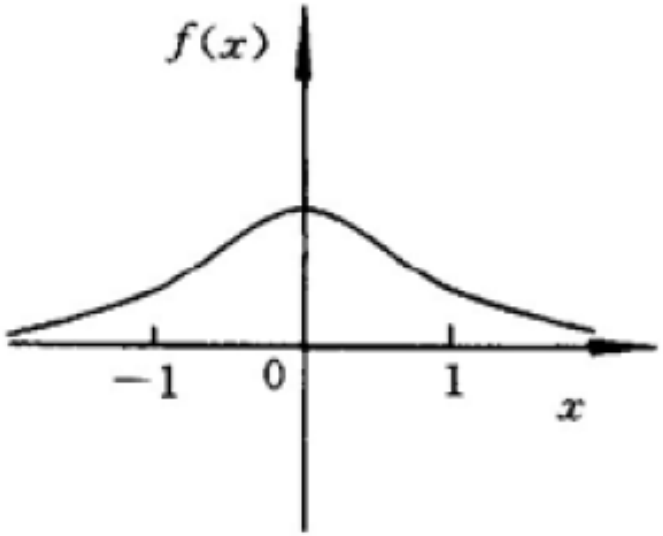
- 生物学背景：
  神经细胞的线性比例作用

7.1 单神经元网络

高斯函数（Radial Basis Function，RBF）

radbas@Matlab

$$f(x) = e^{-\frac{x^2}{\delta^2}}$$



- 生物学背景：
  视觉、听觉等神经细胞的区域性非线性作用

7.1 单神经元网络

Why use activation functions?

激活函数通常有如下一些性质：

- 非线性： 如果不用激励函数，每一层输出都是上层输入的线性函数，无论神经网络有多少层，输出都是输入的线性组合。如果使用的话，激活函数给神经元引入了非线性因素，使得神经网络可以任意逼近任何非线性函数，这样神经网络就可以应用到众多的非线性模型中。当激活函数是非线性的时候，一个两层的神经网络就可以逼近基本上所有的函数了。但是，如果激活函数是恒等激活函数的时候（即 $f(x) = x$），就不满足这个性质了，而且如果MLP使用的是恒等激活函数，那么其实整个网络跟单层神经网络是等价的。

7.1 单神经元网络

- 可微性： 当优化方法是基于梯度的时候，这个性质是必须的。
- 单调性： 当激活函数是单调的时候，单层网络能够保证是凸函数。
- $(x) \approx x$ ： 当激活函数满足这个性质的时候，如果参数的初始化是random的很小的值，那么神经网络的训练将会很高效；如果不满足这个性质，那么就需要很用心的去设置初始值。
- 输出值的范围： 当激活函数输出值是 有限 的时候，基于梯度的优化方法会更加稳定 ，因为特征的表示受有限权值的影响更显著；当激活函数的输出是 无限 的时候，模型的训练会更加高效，不过在这种情况小，一般需要更小的learning rate.

这些性质，正是我们使用激活函数的原因！

7.1 单神经元网络

感知器

- 一个由线性阈值神经元组成的最简单的前向神经网络。
- 一般包括输入层、中间层和输出层。
- 激活函数：

$$f(x)=\begin{cases}1 & x\geq 0\\ -1 & x<0\end{cases}$$

- 主要用于模式分类。

7.1 单神经元网络

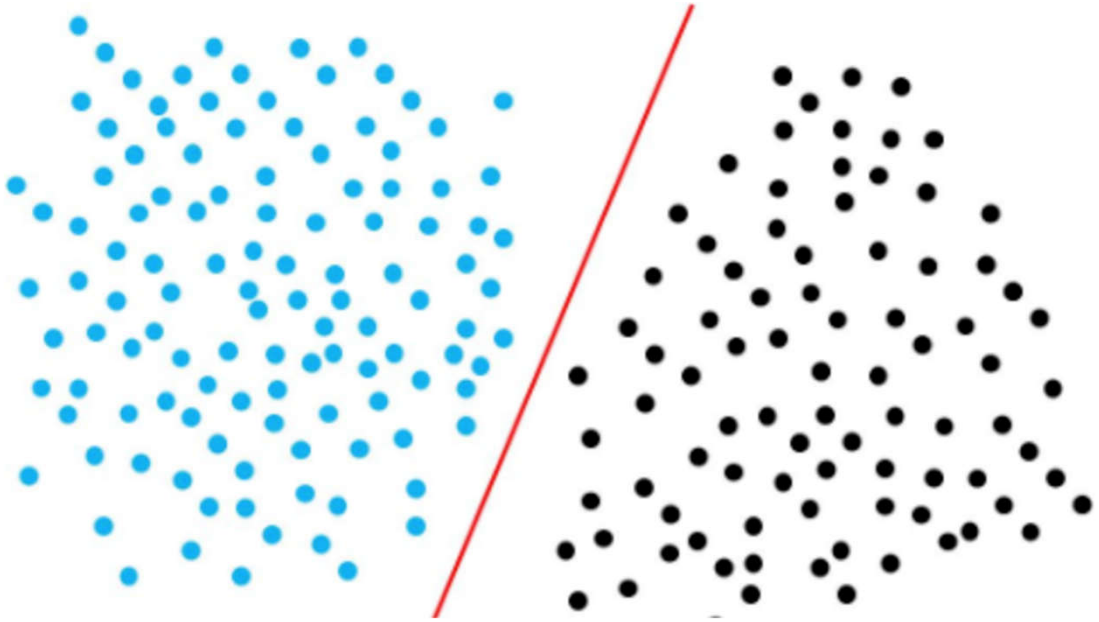
感知器模型与之前提到的神经元模型几乎是相同的，但是二者之间存在着一些关键的区别：

- 其输入可以选择使用实数向量，而不是神经元模型的二进制向量。
- 与神经元模型不同，感知器模型是一个可以学习的模型。

7.1 单神经元网络

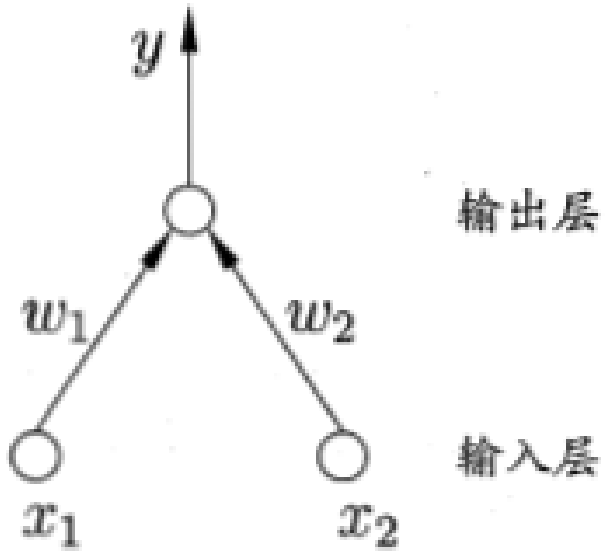
感知器是二分类的线性模型，其输入是实例的特征向量，输出的是实例的类别，分别是+1和−1，属于判别模型。

假设训练数据集是线性可分的，感知器学习的目标是求得一个能够将训练数据集**正实例点和负实例点完全正确分开的分离超平面**。



7.1 单神经元网络

感知机由两层神经元组成，如下图所示，输入层接收外界输入信号后传递给输出层，输出层是M-P神经元，亦称“阈值逻辑单元”（threshold logic unit）。

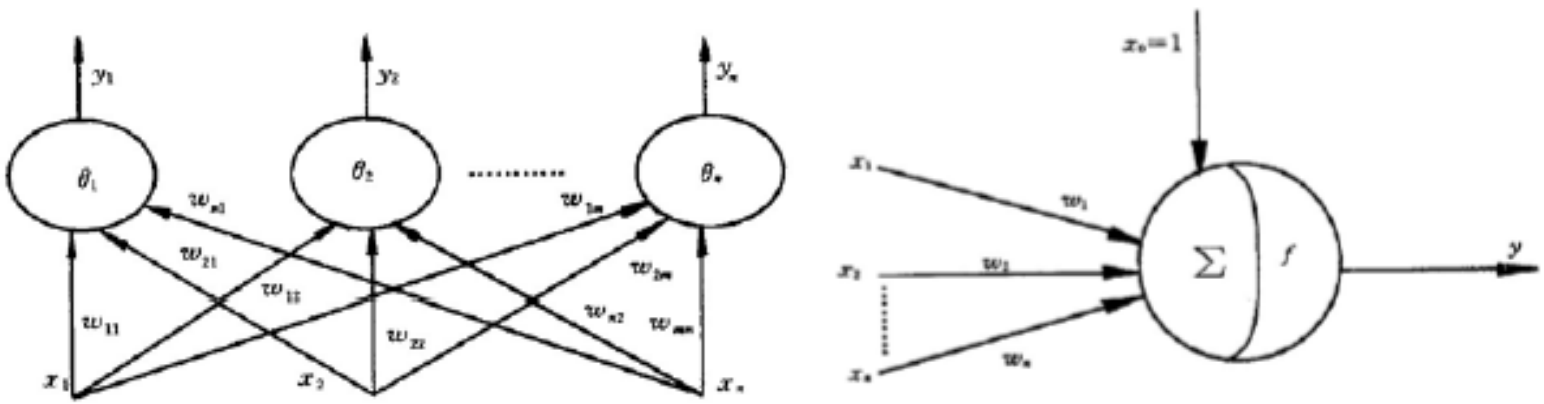


7.1 单神经元网络

单层感知器

- 单层感知器的一个神经元的输入输出关系：当其输入的加权和大于或等于阈值时，输出为1，否则为−1（或为0）。

$$y = f(\sum_{i=0}^{n} w_i x_i)$$



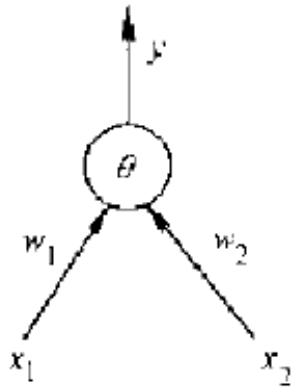
7.1 单神经元网络

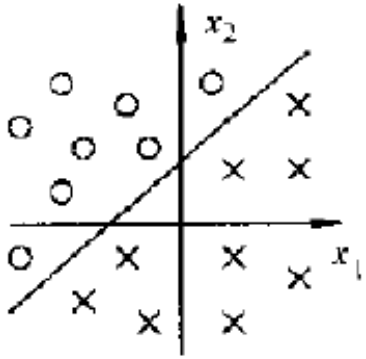
单层感知器为线性可分分类器

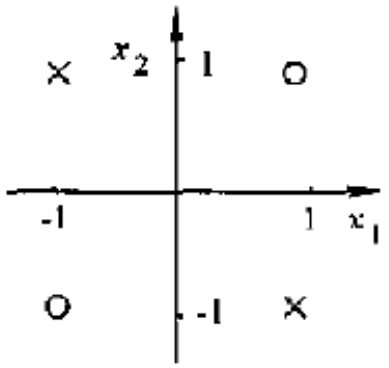
- 以二维空间为例，根据单层感知器的变换关系，分界线的方程为

$$w_1 x_1 + w_2 x_2 - \theta = 0$$

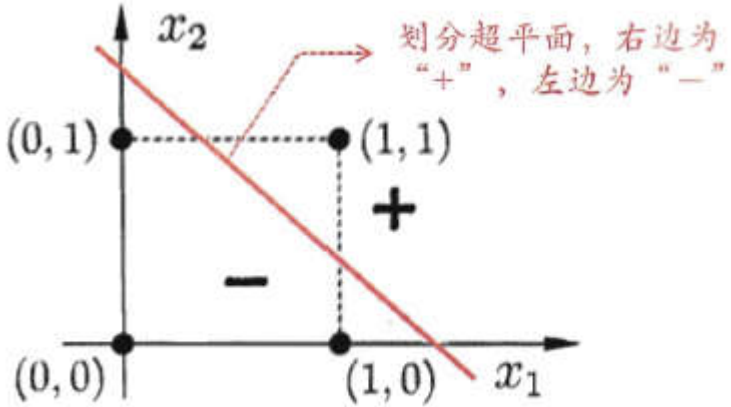
- 直线方程，只能区分线性可分模式类而无法区分如“异或”等非线性可分模式类。



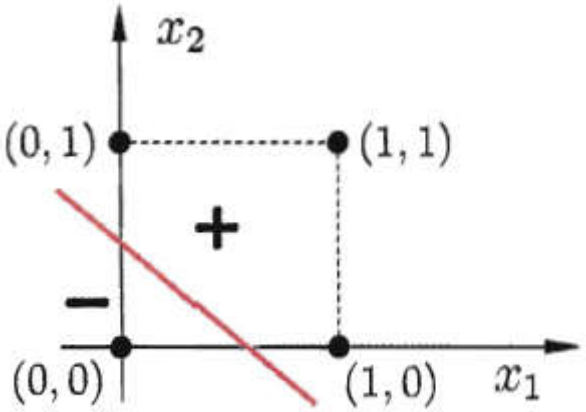




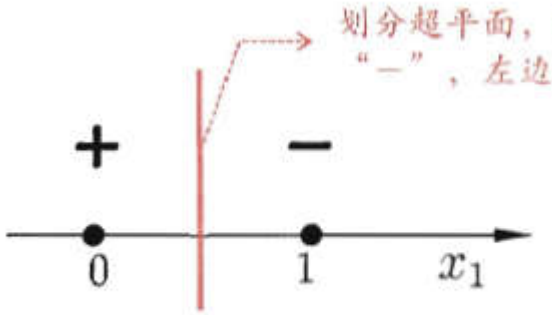
7.1 单神经元网络



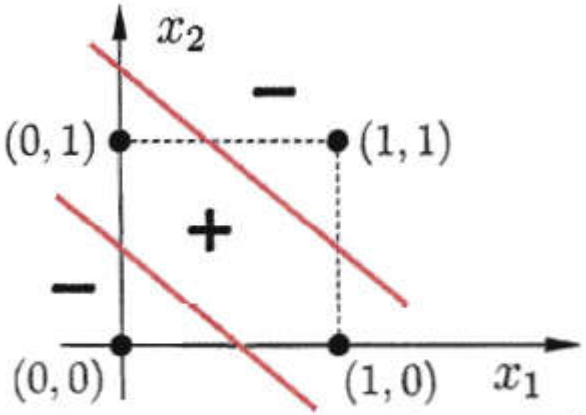
(a) “与”问题 $(x_1 \wedge x_2)$



(b) “或”问题 $(x_1 \vee x_2)$



(c) “非”问题 $(\neg x_1)$

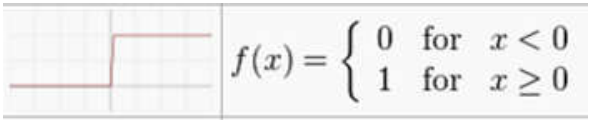


(d) “异或”问题 $(x_1 \oplus x_2)$

7.1 单神经元网络

感知机能容易地实现逻辑与、或、非运算，注意到

$$y = f(\sum_i w_i x_i - \theta_i)$$

假定f是阶跃函数  ，则有

$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

“与”（$x_1 \wedge x_2$）：令$w_1 = w_2 = 1$，$\theta = 2$，则$y = f(1 \cdot x_1 + 1 \cdot x_1 - 2)$，仅在$x_1 = x_2 = 1$时，y=1；

“或”（$x_1 \vee x_2$）：令$w_1 = w_2 = 1$，$\theta = 0.5$，则$y = f(1 \cdot x_1 + 1 \cdot x_1 - 0.5)$，仅在$x_1 = 1$或$x_2 = 1$时，y=1；

“非”（$\overline{x_1}$）：令$w_1 = -0.6$，$w_2 = 0$，$\theta = -0.5$，则$y = f(-0.6 \cdot x_1 + 0 \cdot x_2 + 0.5)$，当$x_1 = 1$时，y=0；当$x_1 = 0$时，y=1。

7.1 单神经元网络

单层感知器的一种学习算法

- 随机地给定一组连接权$w_i(0)$（较小的非零值）。
- 输入一组样本和期望的输出（亦称之为教师信号）。
- 计算感知器实际输出

$$y(k)=f\left(\sum_{i=0}^{n}w_i(k)x_i\right)=\begin{cases}1, & \sum_{i=0}^{n}w_i(k)x_i\geq 0\\ -1, & \sum_{i=0}^{n}w_i(k)x_i<0\end{cases}\qquad (x_0=1,\quad w_0(0)=-\theta)$$

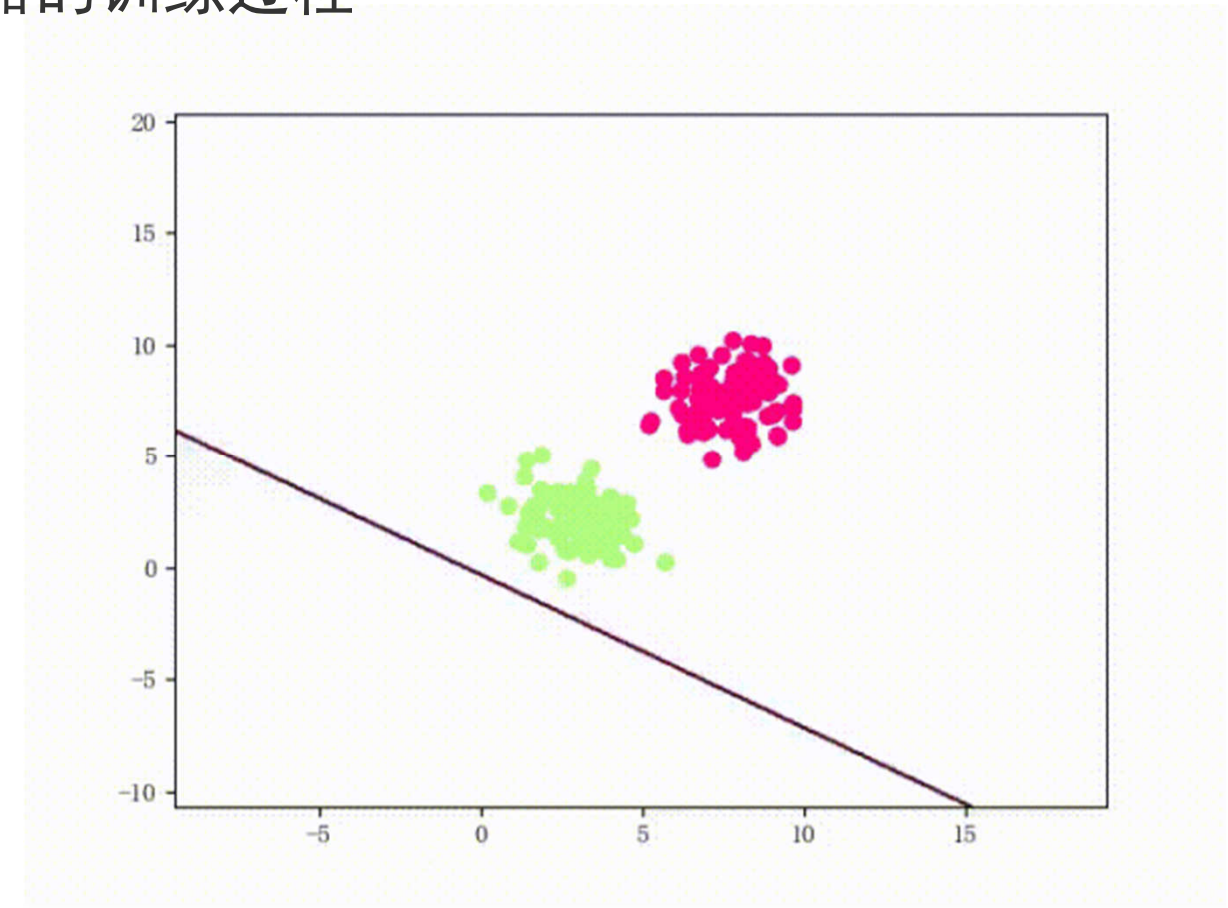
- 修正权值

$$w_i(k+1)=w_i(k)+\eta[d(k)-y(k)]x_i\qquad i=0,1,2,\cdots,n$$

- 选取另外一组样本，重复上述的过程，直到权值对一切样本均稳定不变为止，学习过程结束。
- 算法收敛的充要条件为输入样本是线性可分的。

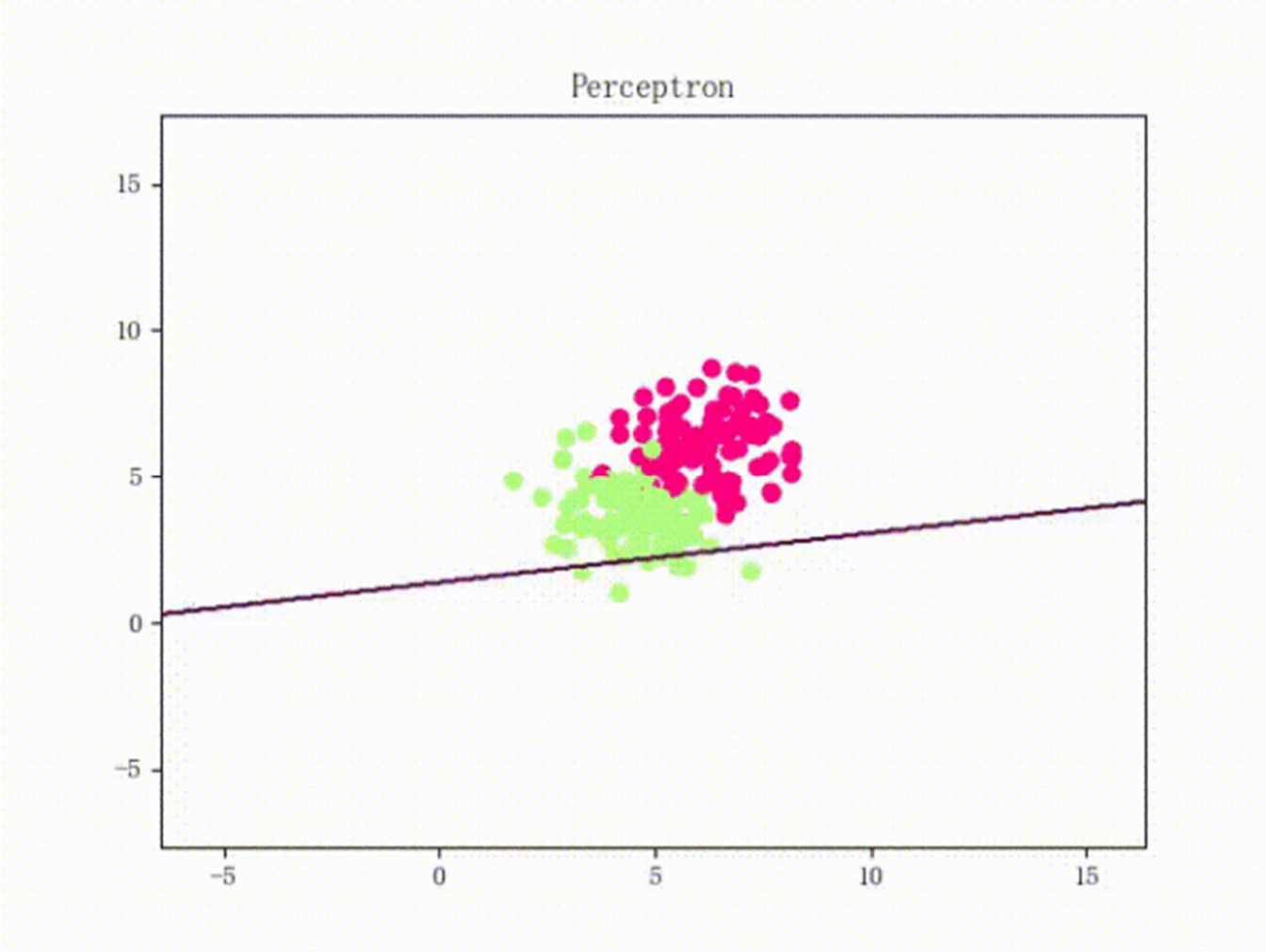
7.1 单神经元网络

感知器的训练过程



线性可分的过程

7.1 单神经元网络



线性不可分的过程

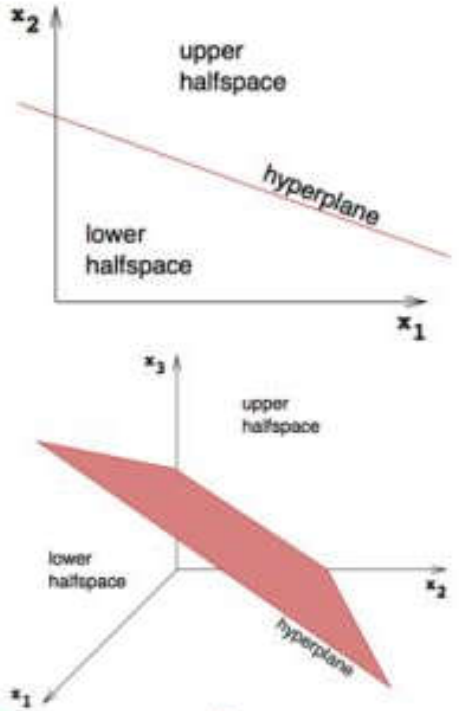
7.1 单神经元网络

我们可以把输入值$(x_1, \ldots, x_n)$看作是N维空间中的一个点的坐标，$w^T x - w_0 = 0$ 可以认为是N维空间中的一个超平面，显然：

- 当$w^T x - w_0 < 0$时，此时的点落在超平面的下方，
- 而当$w^T x - w_0 > 0$时，此时的点落在超平面的上方。

感知器模型对应的就是一个分类器的超平面，它可以将不同类别的点在N维空间中分离开。从下图中可以发现，感知器模型是一个线性的分类器。

7.1 单神经元网络

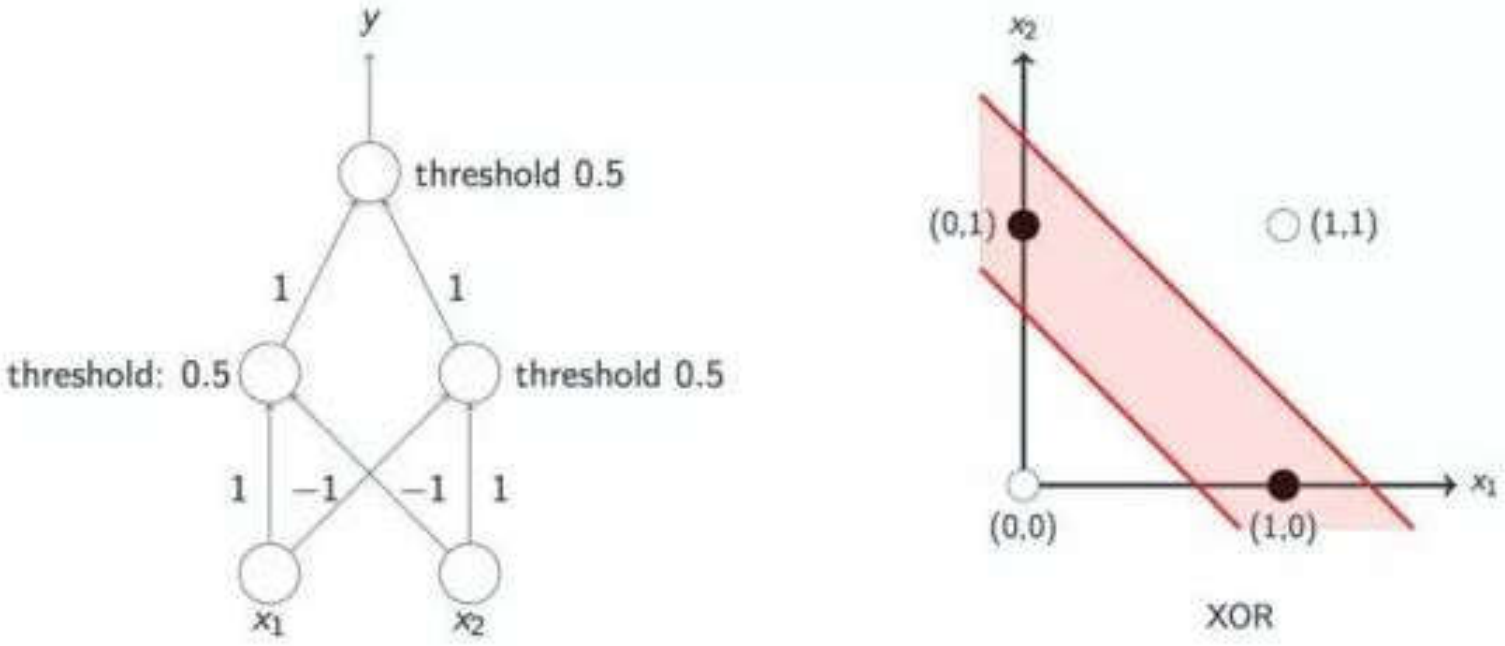


Represents a linear separator in input space.

- input patterns $(x_1, \ldots, x_n)$ are points in n-dimensional space.
- points with $\boldsymbol{w}^\top \boldsymbol{x} - w_0 = 0$ are on a hyperplane defined by $\boldsymbol{w}$ and w_0.
- points with $\boldsymbol{w}^\top \boldsymbol{x} - w_0 > 0$ are above the hyperplane.
- points with $\boldsymbol{w}^\top \boldsymbol{x} - w_0 < 0$ are below the hyperplane.
- perceptrons partition the input space into two halfspaces along a hyperplane.

7.1 单神经元网络

关于多层感知器模型实现异或操作的直观几何体现如下图所示：



We can model XOR using a hidden layer.

7.2 BP神经网络

多层网络的学习能力比单层感知器强得多。欲训练多层网络，简单的感知器学习规则显然不够，需要更强大的学习算法。

误差反向传播（error BackPropagation，简称BP）算法就是其中最杰出的代表，它是至今最成功的神经网络学习算法，其基本思想是梯度下降法。

它采用梯度搜索技术，以期使网络的实际输出值与期望输出值的误差均方值为最小。

7.2 BP神经网络

（一）BP网络特点

（1）是一种多层网络，包括输入层、隐含层和输出层；

（2）层与层之间采用全互连方式，同一层神经元之间不连接；

（3）权值通过δ学习算法进行调节；

（4）神经元激发函数为S函数；

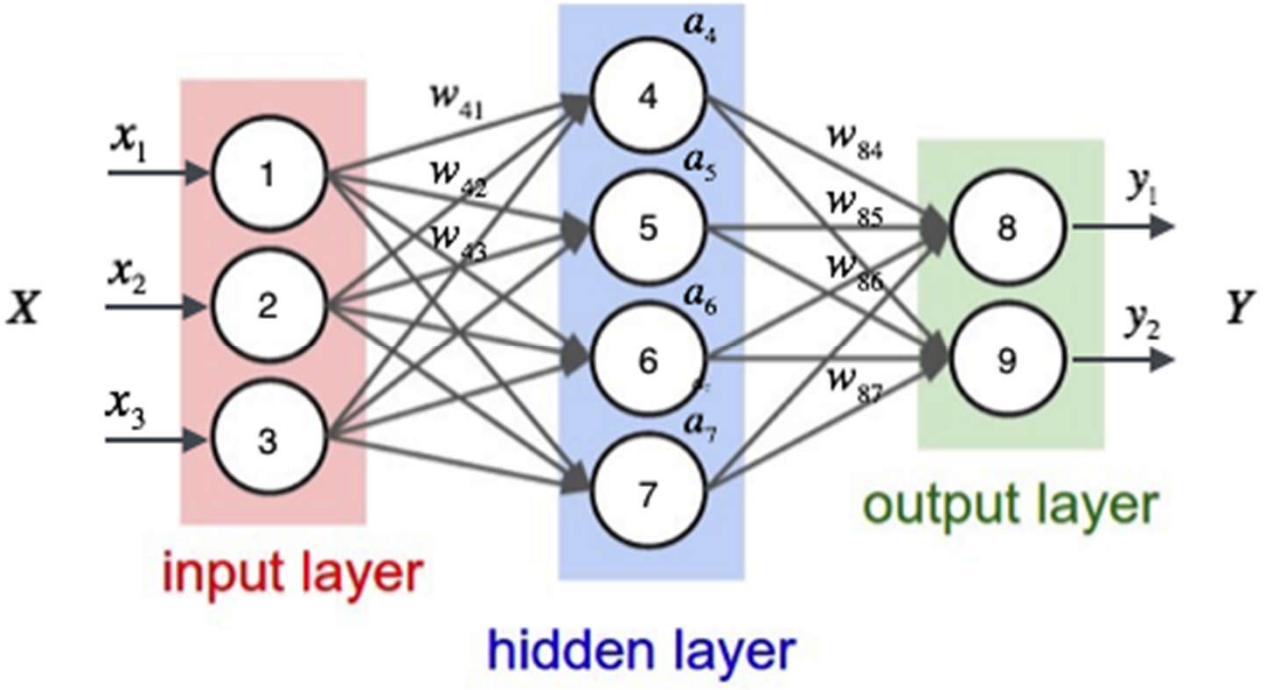
（5）学习算法由正向传播和反向传播组成；

（6）层与层的连接是单向的，信息的传播是双向的。

7.2 BP神经网络

（二）BP网络结构

含一个隐含层的BP网络结构如下图所示



7.2 BP神经网络

（额外）系统辨识

- 辨识、状态估计和控制理论是现代控制论三个相互渗透的领域。辨识和状态估计离不开控制理论的支持；控制理论的应用又几乎不能没有辨识和状态估计。

- 系统辨识是对难以通过机理和试验方法建模的复杂对象进行建模的一种方法。

- Zadeh把辨识定义为：“辨识就是在系统输入和输出观测数据的基础上，从一组给定的模型中，确定一个与被识别的系统等价的模型。”

7.2 BP神经网络

这个定义指出了辨识应具有三个要素：

（1）输入输出数据：能够观测到的按时间顺序排列的系统输入输出数据，简称为时序数据。
（2）模型类：明确给定所要辨识的系统属于哪一类系统，因为属性完全不同的系统可能具有相同的模型结构。
（3）等价准则：从一类模型中按所给定的性能等价准则，选择一个与实际系统最为接近的模型。等价准则就是用以衡量模型与实际系统接近程度的标准，一般表示为误差函数的泛函。

基于神经网络的系统辨识就是选择一个适当拓扑结构的神经网络模型，使该网络从待辨识的实际动态系统中不断地获得输入输出数据，神经网络通过学习算法自适应地调节神经元间的权重，从而获得利用神经网络逐渐逼近动态系统的模型（正模型）或逆模型（如果系统是可逆的）。这样的动态系统模型实际上是隐含在神经网络的权矩阵中。

7.2 BP神经网络

（三）BP网络的逼近

神经网络在机器学习中应用比较广泛，比如

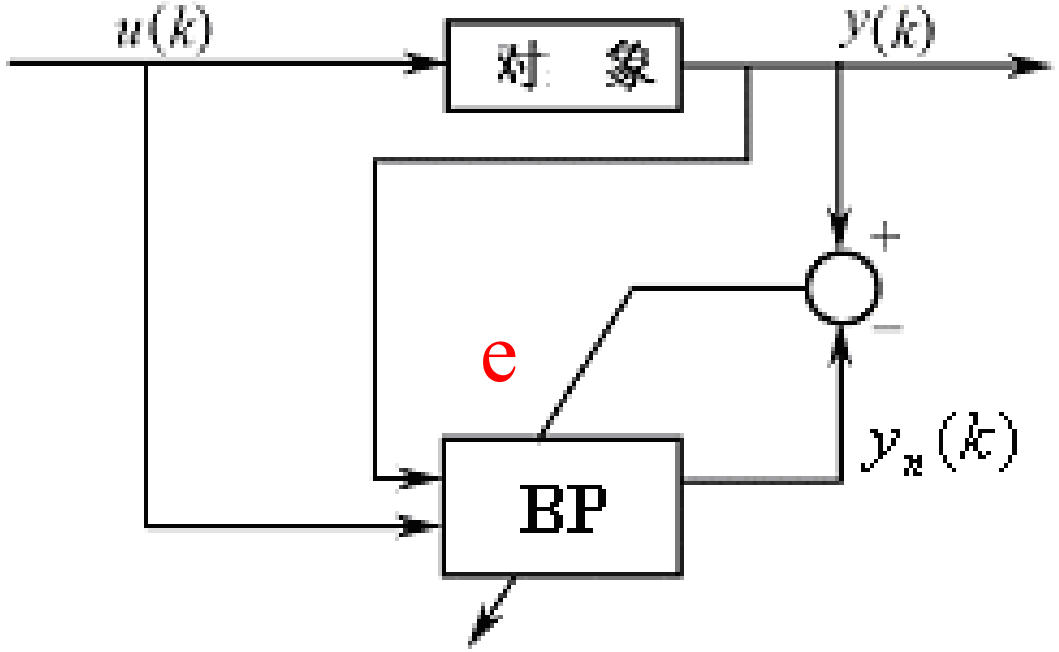
- 函数逼近
- 模式识别
- 分类
- 数据压缩
- 数据挖掘

等领域。

1989年，Robert Hecht-Nielson证明了对于任何在闭区间内的一个连续函数都可以用一个隐层的BP网络来逼近，因而一个三层的BP网络可以完成任意的n维到m维的映射。

7.2 BP神经网络

BP网络逼近的结构图



图中k为网络的迭代步骤，$u(k)$和$y(k)$为逼近器的输入。BP为网络逼近器，$y(k)$为被控对象实际输出，$y_n(k)$为BP的输出。将系统输出$y(k)$及输入$u(k)$的值作为逼近器BP的输入，将系统输出与网络输出的误差作为逼近器的调整信号。

7.2 BP神经网络

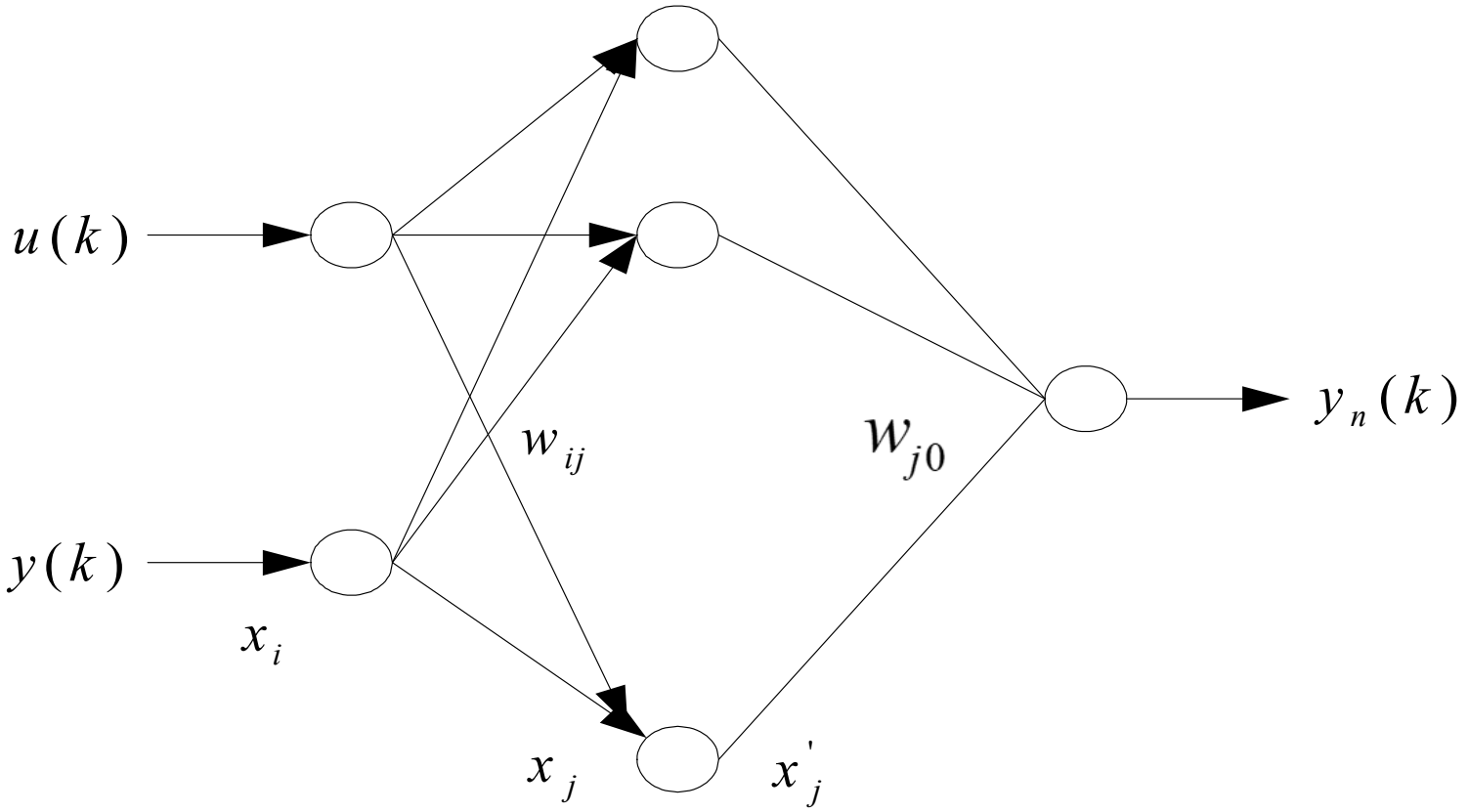


图7-7 用于逼近的BP网络

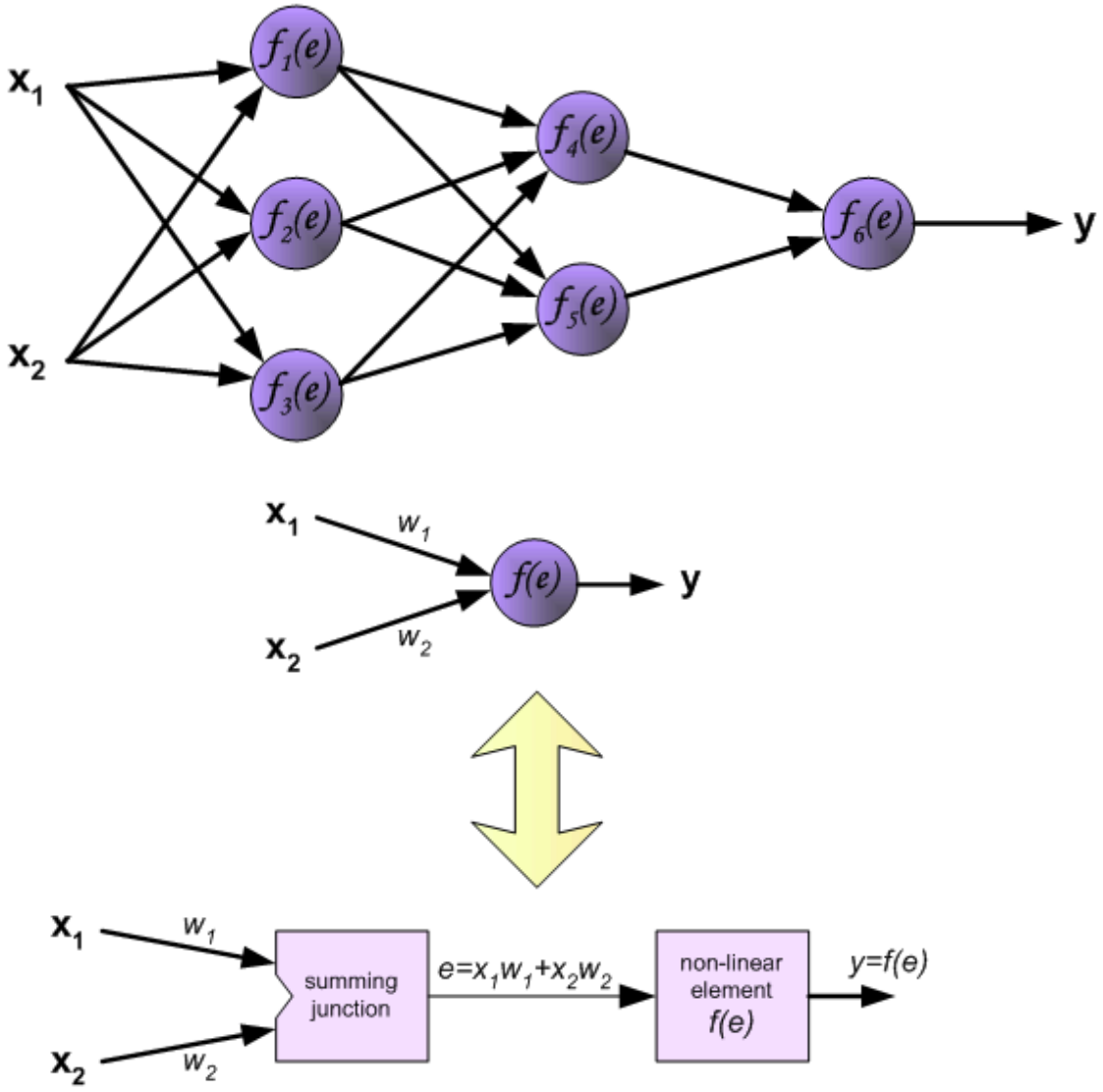
7.2 BP神经网络

BP算法的学习过程由正向传播和反向传播组成。

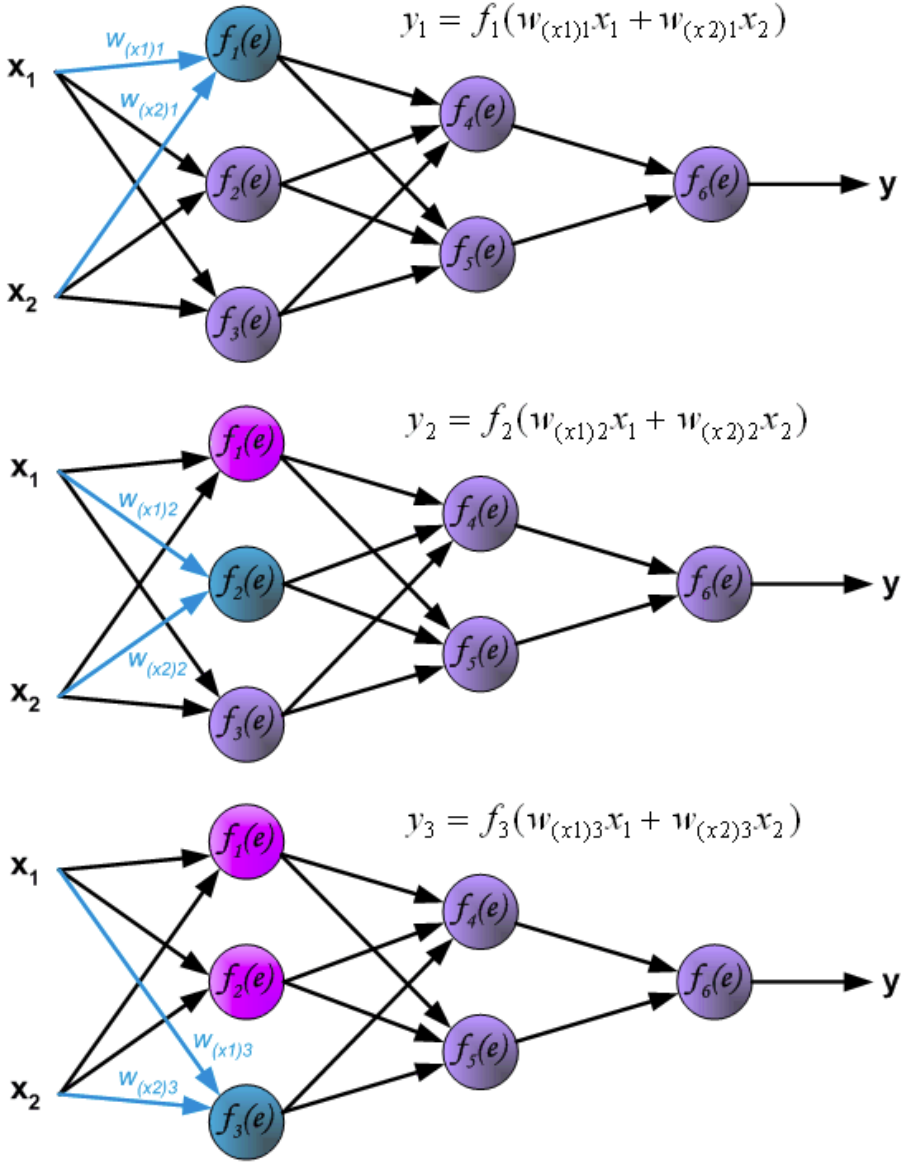
- 在正向传播过程中，输入信息从输入层经隐层逐层处理，并传向输出层，每层神经元（节点）的状态只影响下一层神经元的状态。

- 如果在输出层不能得到期望的输出，则转至反向传播，将误差信号（理想输出与实际输出之差）按联接通路反向计算，由梯度下降法调整各层神经元的权值，使误差信号减小。

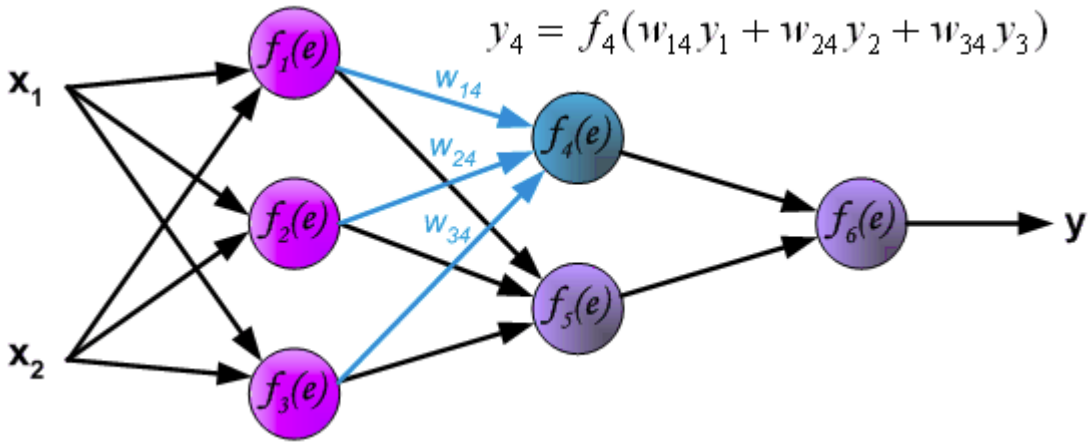
7.2 BP神经网络

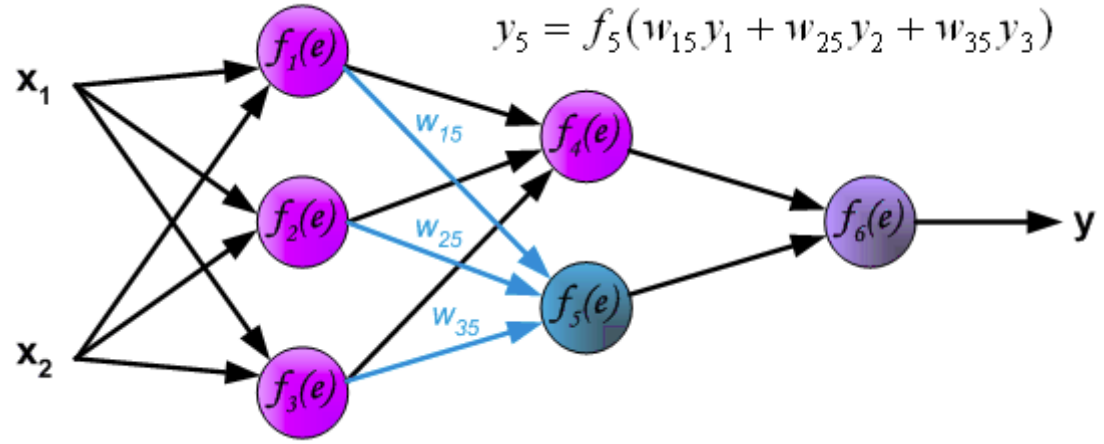

x1
x2
f1(e)
f2(e)
f3(e)
f4(e)
f5(e)
f6(e)
y
x1
w1
f(e)
y
x2
w2
x1
w1
summing junction
e=x1w1+x2w2
non-linear element f(e)
y=f(e)
x2
w2

7.2 BP神经网络

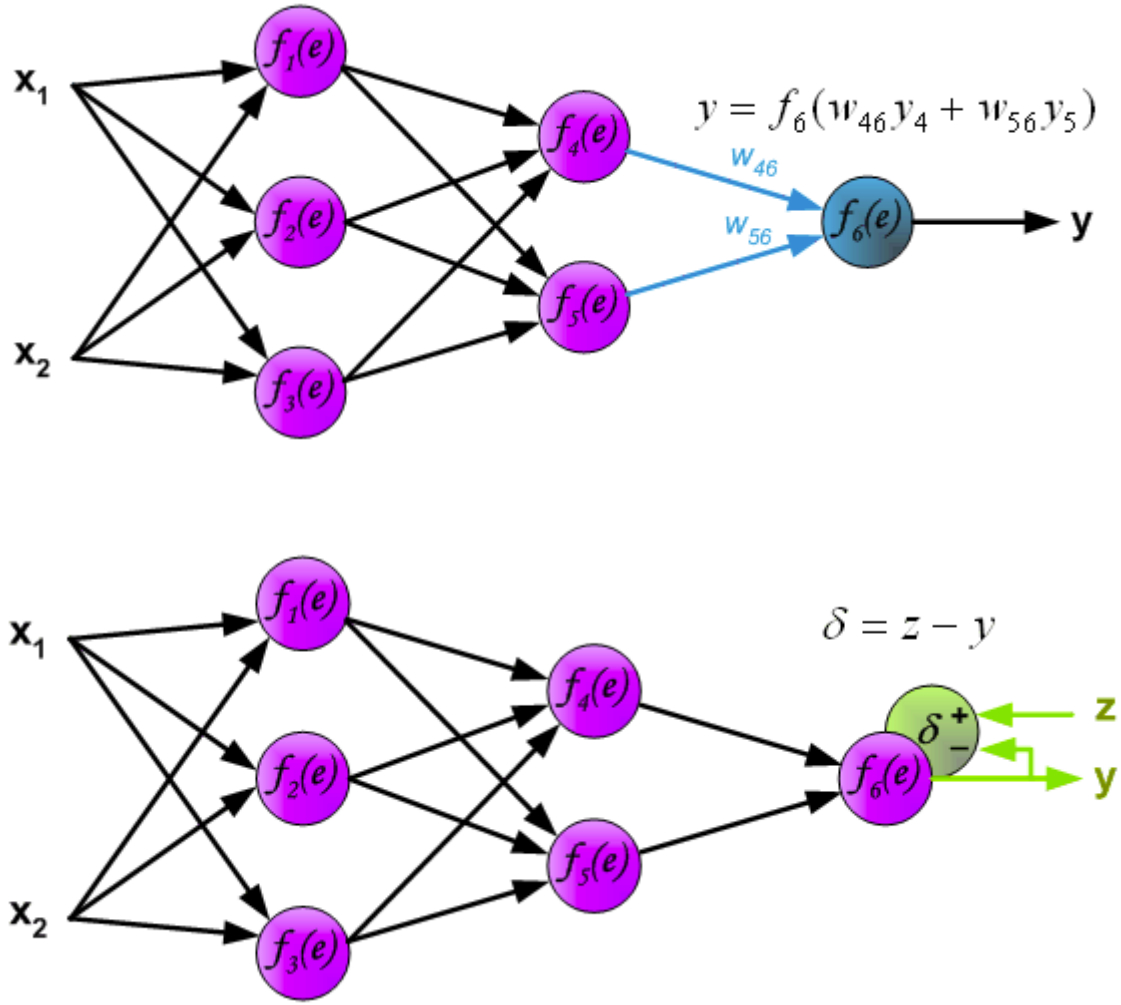

$y_1 = f_1(w_{(x1)1}x_1 + w_{(x2)1}x_2)$
$y_2 = f_2(w_{(x1)2}x_1 + w_{(x2)2}x_2)$
$y_3 = f_3(w_{(x1)3}x_1 + w_{(x2)3}x_2)$
x1
x2
y
$w_{(x1)1}$
$w_{(x2)1}$
$w_{(x1)2}$
$w_{(x2)2}$
$w_{(x1)3}$
$w_{(x2)3}$
$f_1(e)$
$f_2(e)$
$f_3(e)$
$f_4(e)$
$f_5(e)$
$f_6(e)$

7.2 BP神经网络

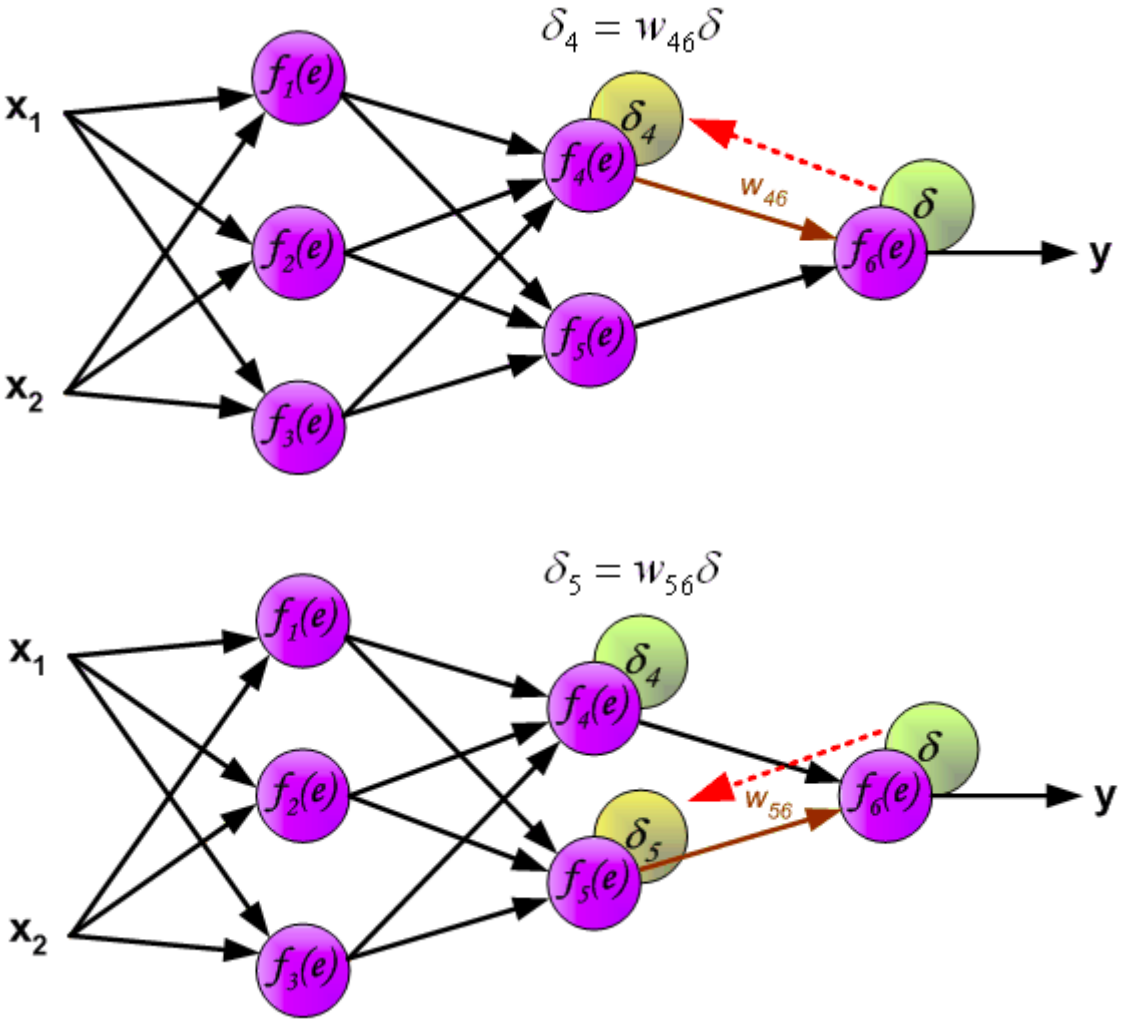

$y_4 = f_4(w_{14}y_1 + w_{24}y_2 + w_{34}y_3)$
x1
x2
f1(e)
f2(e)
f3(e)
w14
w24
w34
f4(e)
f5(e)
f6(e)
y


$y_5 = f_5(w_{15}y_1 + w_{25}y_2 + w_{35}y_3)$
x1
x2
f1(e)
f2(e)
f3(e)
w15
w25
w35
f4(e)
f5(e)
f6(e)
y

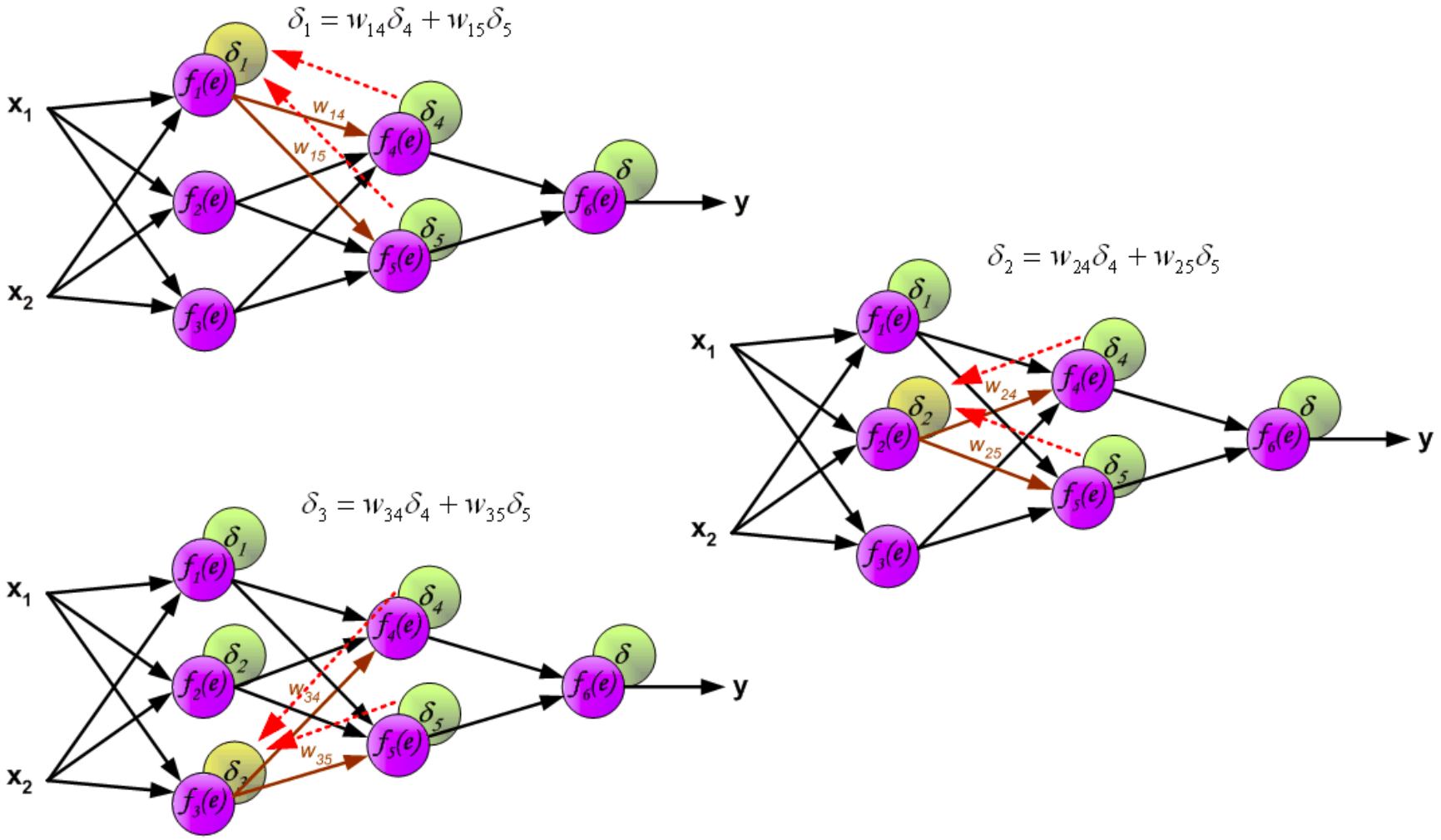
7.2 BP神经网络



7.2 BP神经网络


$\delta_4 = w_{46}\delta$
x1
x2
f1(e)
f2(e)
f3(e)
f4(e)
f5(e)
f6(e)
δ4
δ
w46
y
$\delta_5 = w_{56}\delta$
x1
x2
f1(e)
f2(e)
f3(e)
f4(e)
f5(e)
f6(e)
δ4
δ5
δ
w56
y

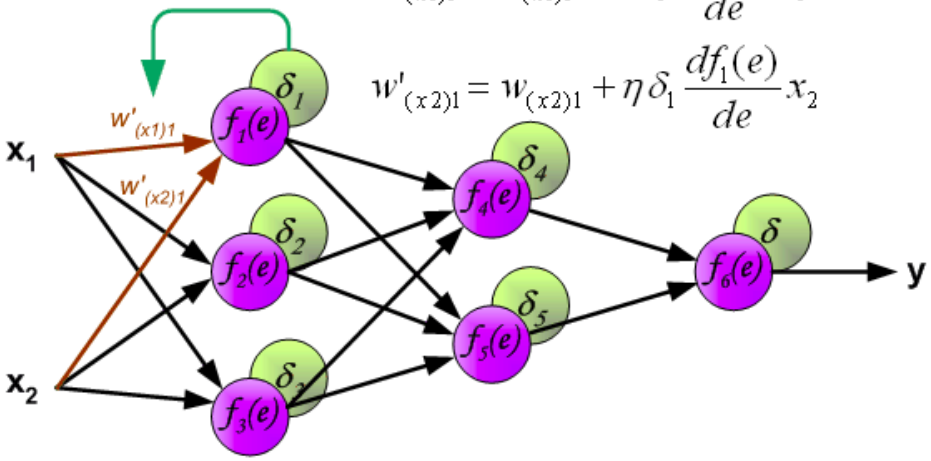
7.2 BP神经网络


$\delta_1 = w_{14}\delta_4 + w_{15}\delta_5$
$\delta_2 = w_{24}\delta_4 + w_{25}\delta_5$
$\delta_3 = w_{34}\delta_4 + w_{35}\delta_5$
x1
x2
y
w14
w15
w24
w25
w34
w35

7.2 BP神经网络

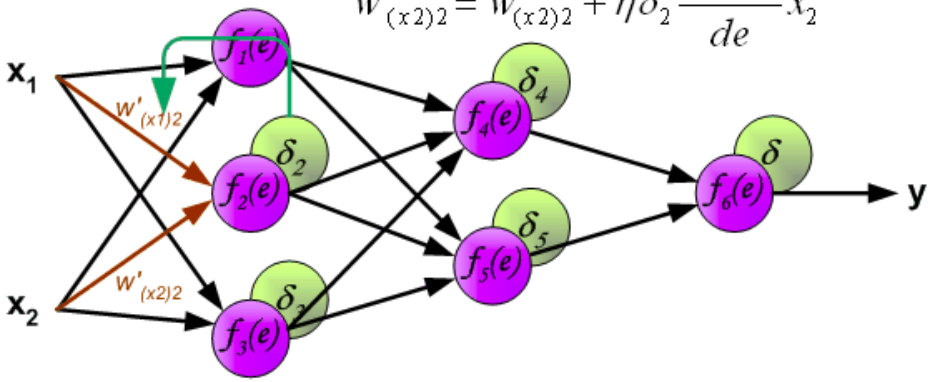
$$w'_{(x1)1} = w_{(x1)1} + \eta\delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta\delta_1 \frac{df_1(e)}{de} x_2$$



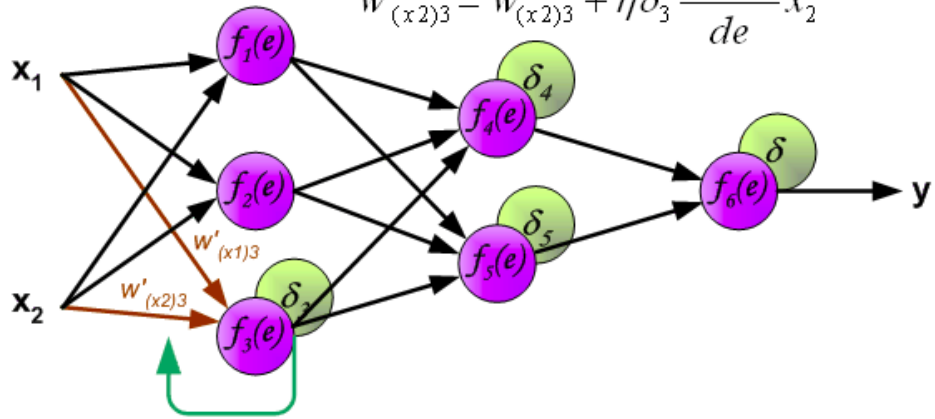
$$w'_{(x1)2} = w_{(x1)2} + \eta\delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta\delta_2 \frac{df_2(e)}{de} x_2$$



$$w'_{(x1)3} = w_{(x1)3} + \eta\delta_3 \frac{df_3(e)}{de} x_1$$

$$w'_{(x2)3} = w_{(x2)3} + \eta\delta_3 \frac{df_3(e)}{de} x_2$$

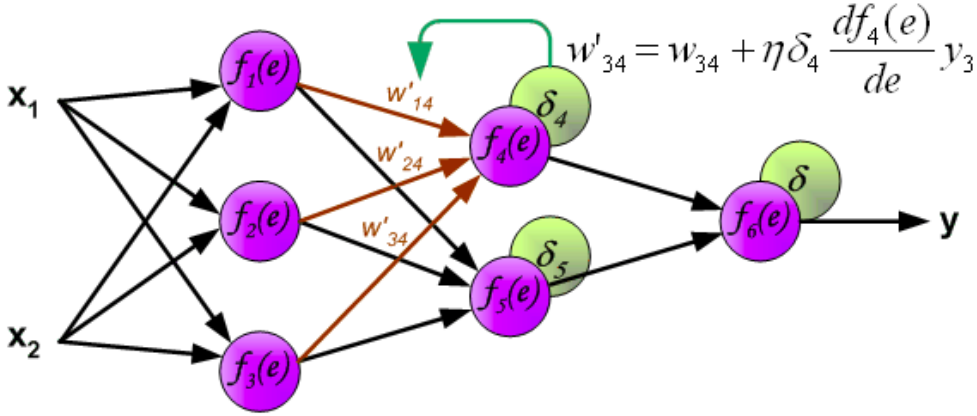


7.2 BP神经网络

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

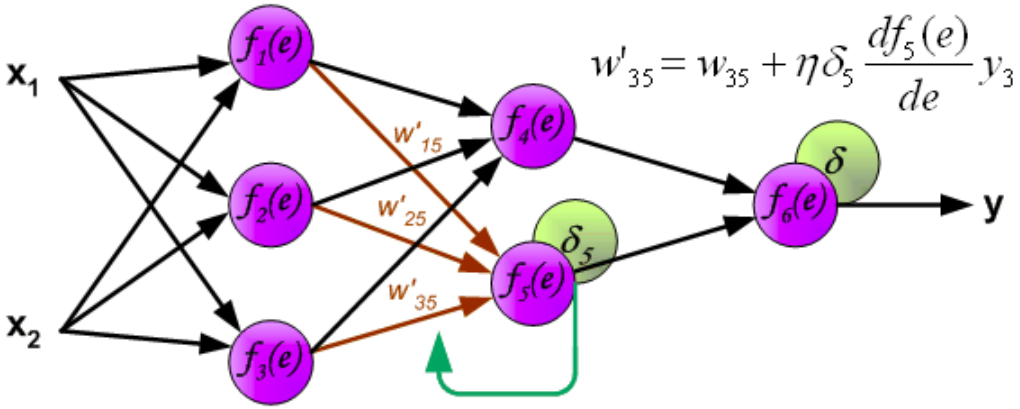
$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



$$w'_{15} = w_{15} + \eta\delta_5 \frac{df_5(e)}{de} y_1$$

$$w'_{25} = w_{25} + \eta\delta_5 \frac{df_5(e)}{de} y_2$$

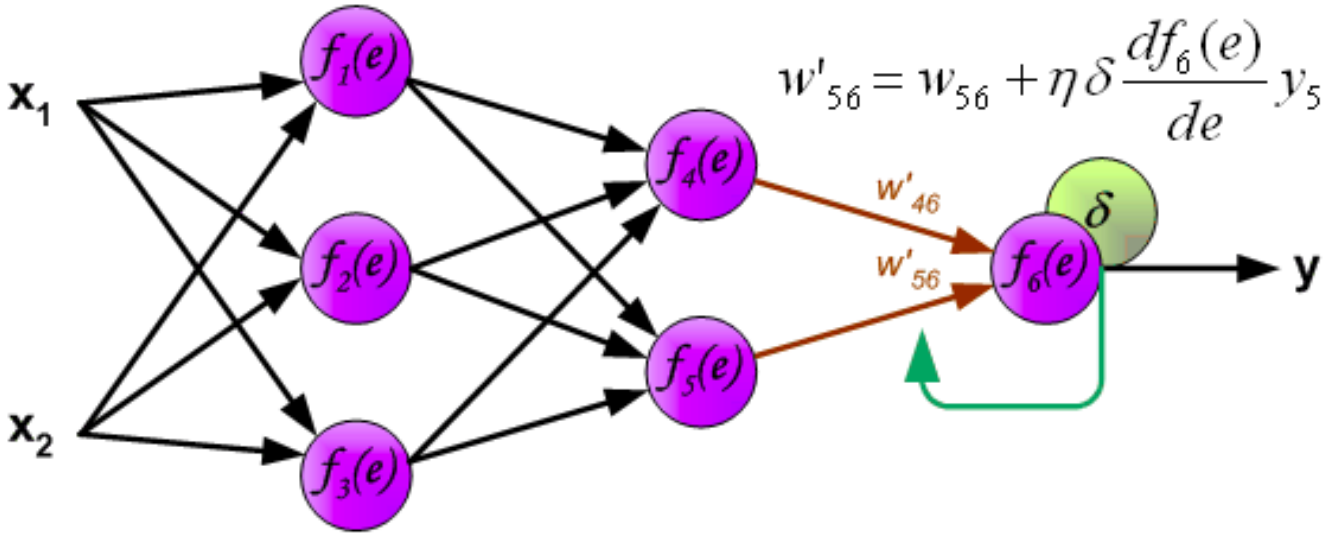
$$w'_{35} = w_{35} + \eta\delta_5 \frac{df_5(e)}{de} y_3$$



7.2 BP神经网络

$$w'_{46} = w_{46} + \eta\delta\frac{df_6(e)}{de}y_4$$

$$w'_{56} = w_{56} + \eta\delta\frac{df_6(e)}{de}y_5$$



7.2 BP神经网络

（1）正向传播：计算网络的输出。

隐层神经元的输入为所有输入的加权之和：

$$x_j = \sum_i w_{ij} x_i$$

隐层神经元的输出采用S函数激发：

$$x_j' = f(x_j) = \frac{1}{1+e^{-x_j}}$$

则

$$\frac{\partial x_j'}{\partial x_j} = x_j'(1 - x_j')$$

7.2 BP神经网络

输出层神经元的输出：

$$y_n(\mathrm{k}) = \sum_j w_{j0} x_j^{'}$$

网络输出与理想输出误差为：

$$e(k) = y(k) - y_n(k)$$

误差性能指标函数为：

$$E = \frac{1}{2} e(k)^2$$

7.2 BP神经网络

（2）反向传播：

采用δ学习算法，调整各层间的权值。

根据梯度下降法，权值的学习算法如下：

输出层及隐层的连接权值学习算法为

$$\Delta w_{j0} = -\eta \frac{\partial E}{\partial w_{j0}} = \eta \cdot e(k) \cdot \frac{\partial y_n}{\partial w_{j0}} = \eta \cdot e(k) \cdot x_j'$$

k+1时刻网络的权值为

$$w_{j0}(\mathrm{k}+1) = w_{j0}(\mathrm{k}) + \Delta w_{jo}$$

7.2 BP神经网络

隐层及输入层连接权值学习算法为：

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \eta \cdot e(k) \cdot \frac{\partial y_n}{\partial w_{ij}}$$

其中

$$\frac{\partial y_n}{\partial w_{ij}} = \frac{\partial y_n}{\partial x_j^{'}} \cdot \frac{\partial x_j^{'}}{\partial x_j} \cdot \frac{\partial x_j}{\partial w_{ij}} = w_{j0} \cdot \frac{\partial x_j^{'}}{\partial x_j} \cdot x_i = w_{j0} \cdot x_j^{'}(1 - x_j^{'}) \cdot x_i$$

k+1时刻网络的权值为：

$$w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}$$

7.2 BP神经网络

如果考虑上次权值对本次权值变化的影响，需要加入动量因子α，此时的权值为：

$$w_{j0}(k+1)=w_{j0}(k)+\Delta w_{j0}+\alpha(w_{j0}(k)-w_{j0}(k-1))$$

$$w_{ij}(k+1)=w_{ij}(k)+\Delta w_{ij}+\alpha(w_{ij}(k)-w_{ij}(k-1))$$

其中，α为动量因子 $\alpha\in[0,1]$

Jacobian阵(即为对象的输出对控制输入的灵敏度信息)可由神经网络辨识得到，其算法为：

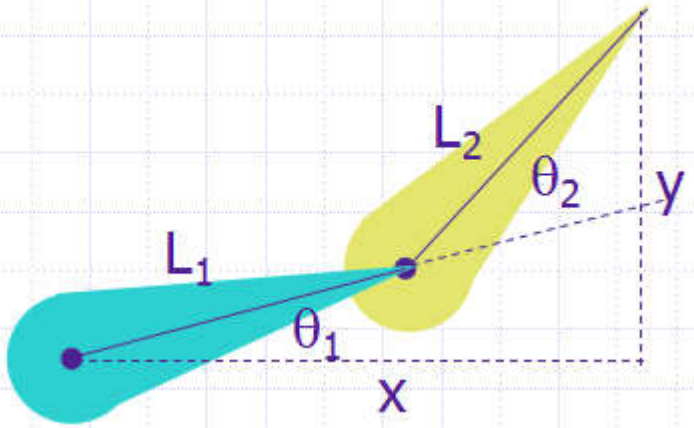
$$\frac{\partial y(\mathrm{k})}{\partial u(k)}\approx\frac{\partial y_n(\mathrm{k})}{\partial u(k)}=\frac{\partial y_n(\mathrm{k})}{\partial x_j^{'}}\times\frac{\partial x_j^{'}}{\partial x_j}\times\frac{\partial x_j}{\partial x_1}=\sum_j w_{j0}x_j^{'}\left(1-x_j^{'}\right)w_{1j}$$

7.2 BP神经网络

Forward Kinematics

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$



$$\Delta x = J \Delta q$$

$$J = \begin{bmatrix} \partial x/\partial\theta_1 & \partial x/\partial\theta_2 \\ \partial y/\partial\theta_1 & \partial y/\partial\theta_2 \end{bmatrix} = \begin{bmatrix} -L_1 \sin\theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

What is Jacobian? A linear approximation to f(x). 雅克比矩阵相当于函数f(x)的一阶导数，即线性近似。

7.2 BP神经网络

（四）BP网络的优缺点

BP网络的优点：

（1）只要有足够多的隐层和隐层节点，BP网络可以逼近任意的非线性映射关系；

（2）BP网络的学习算法属于全局逼近算法，具有较强的泛化能力。

（3）BP网络输入输出之间的关联信息分布地存储在网络的连接权中，个别神经元的损坏只对输入输出关系有较小的影响，因而BP网络具有较好的容错性。

7.2 BP神经网络

BP网络的缺点：

（1）待寻优的参数多，收敛速度慢；
（2）目标函数存在多个极值点，按梯度下降法进行学习，很容易陷入局部极小值；
（3）难以确定隐层及隐层节点的数目。目前，如何根据特定的问题来确定具体的网络结构尚无很好的方法，仍需根据经验来试凑。
（4）难以解决应用问题的实例规模和网络规模间的矛盾。
（5）新加入的样本要影响已学习成功的网络，而且刻画每个输入样本的特征的数目也必须相同。
（6）网络的预测能力（也称泛化能力、推广能力）与训练能力（也称逼近能力、学习能力）的矛盾。

7.2 BP神经网络

由于BP网络具有很好的逼近非线性映射的能力，该网络在模式识别、图像处理、系统辨识、函数拟合、优化计算、最优预测和自适应控制等领域有着较为广泛的应用。

由于BP网络具有很好的逼近特性和泛化能力，可用于神经网络控制器的设计。但由于BP网络收敛速度慢，难以适应实时控制的要求。

7.2 BP神经网络

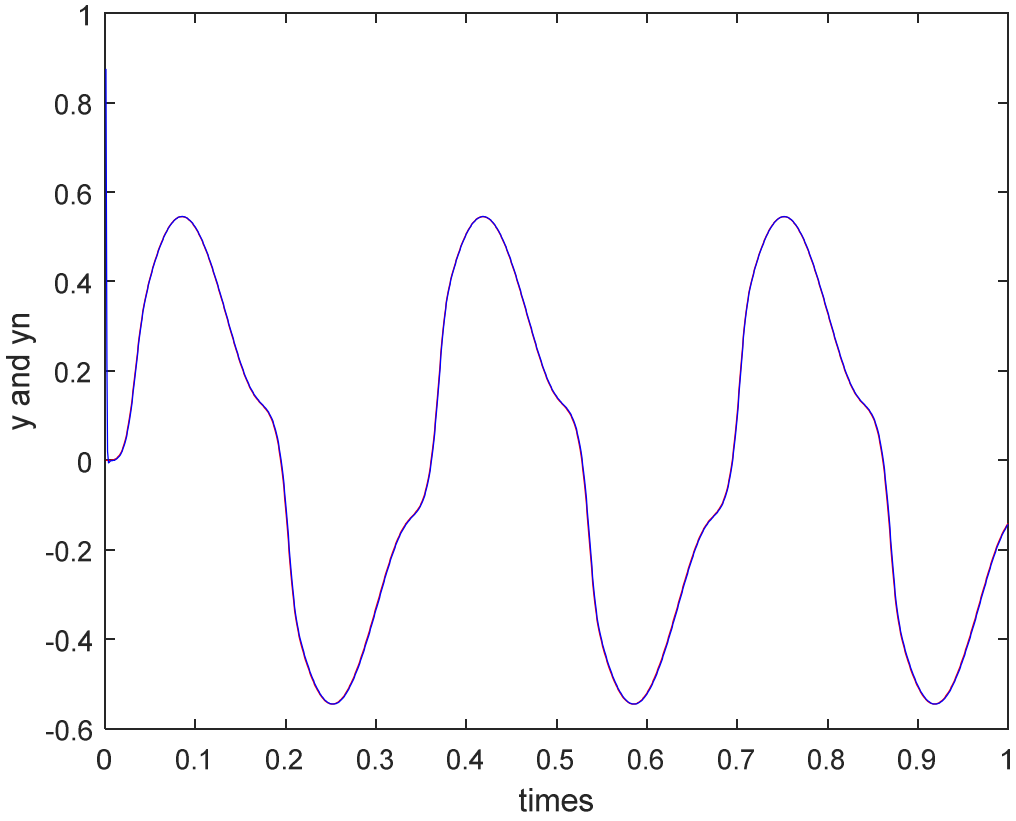
（五）BP网络逼近仿真实例

使用BP网络逼近对象：

$$y(k)=u(k)^3+\frac{y(k-1)}{1+y(k-1)^2}$$

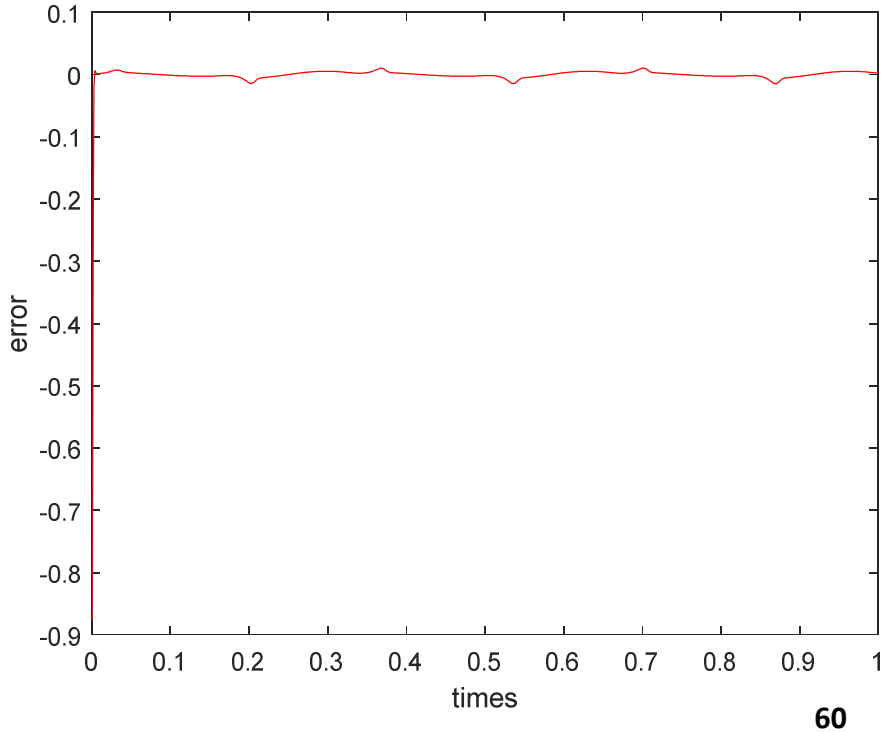
BP网络逼近程序见chap7_1.m

7.2 BP神经网络

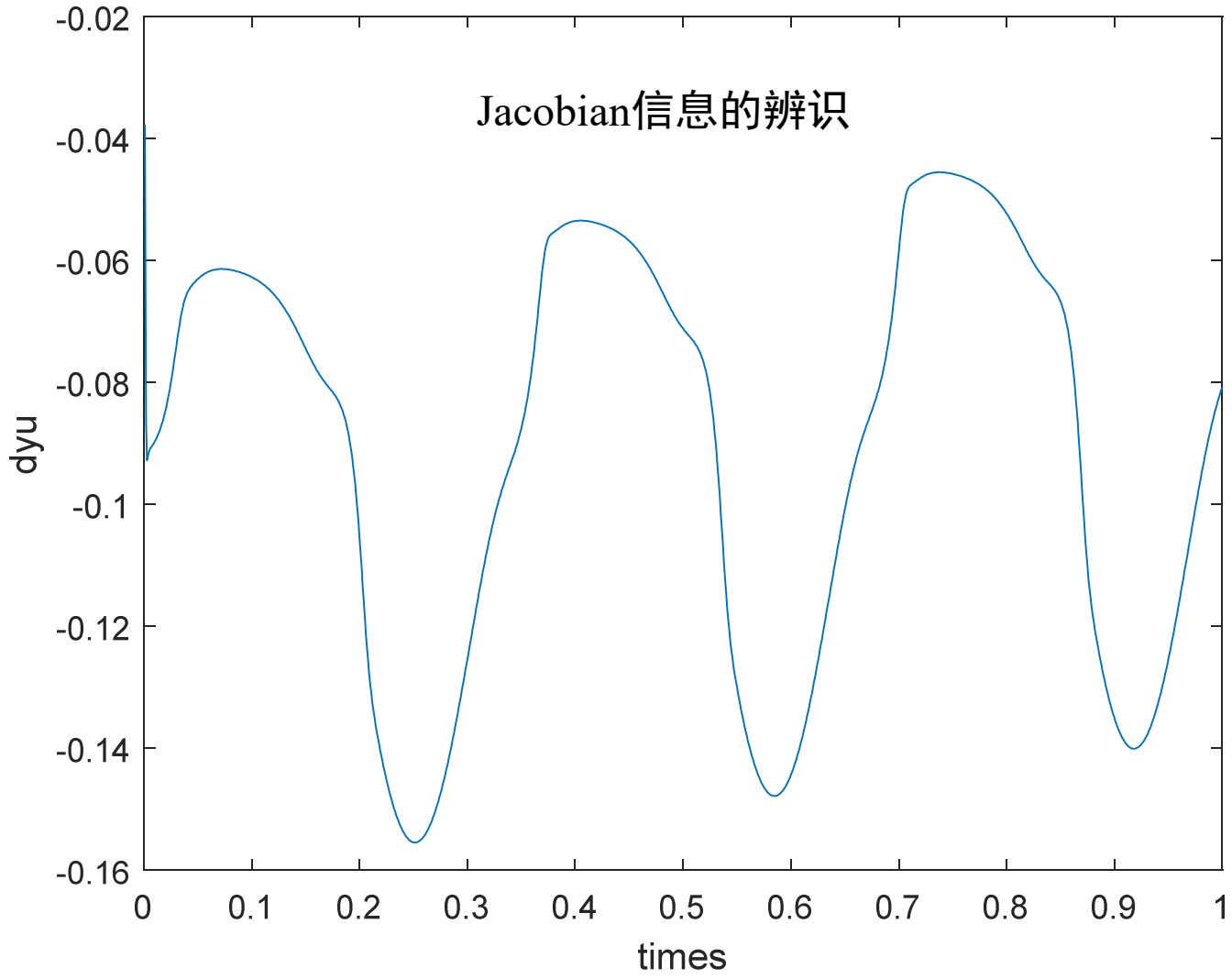


BP网络逼近效果

BP网络逼近误差



7.2 BP神经网络


Jacobian信息的辨识
dyu
times
-0.02
-0.04
-0.06
-0.08
-0.1
-0.12
-0.14
-0.16
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

7.2 BP神经网络

（六）BP网络模式识别

由于神经网络具有自学习、自组织和并行处理等特征，并具有很强的容错能力和联想能力，因此，神经网络具有模式识别的能力。

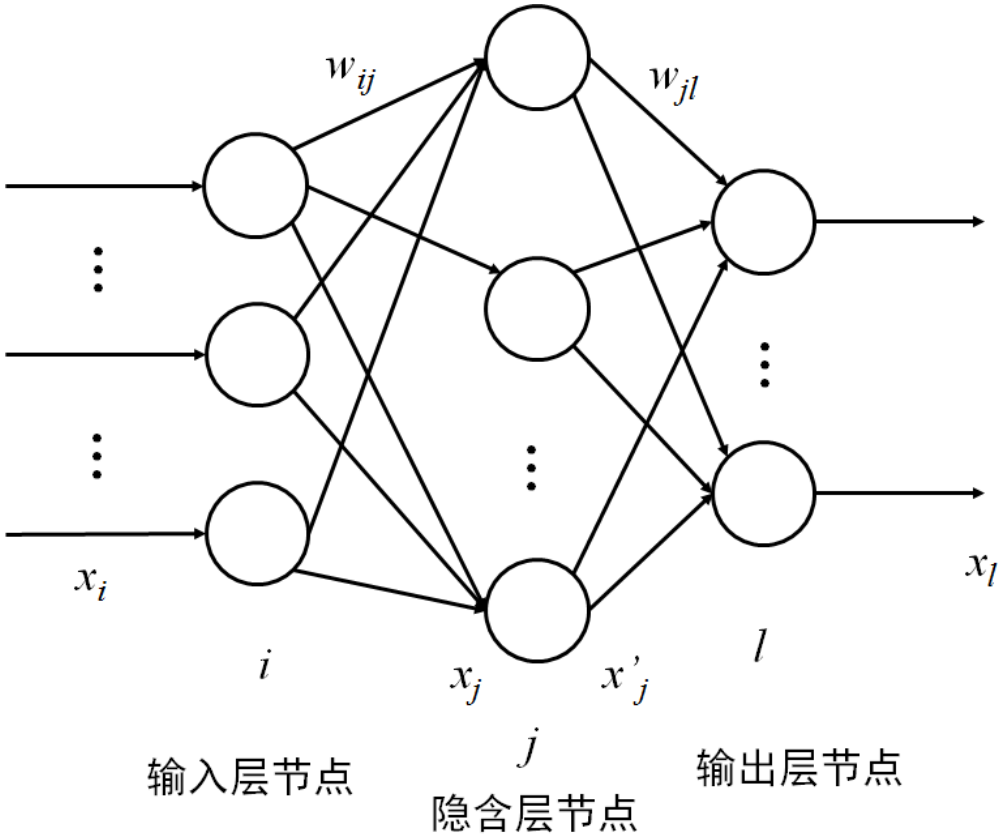
在神经网络模式识别中，根据标准的输入输出模式对，采用神经网络学习算法，以标准的模式作为学习样本进行训练，通过学习调整神经网络的连接权值。当训练满足要求后，得到的神经网络权值构成了模式识别的知识库，利用神经网络并行推理算法对所需要的输入模式进行识别。

7.2 BP神经网络

- 当待识别的输入模式与训练样本中的某个输入模式相同时，神经网络识别的结果就是与训练样本中相对应的输出模式。
- 当待识别的输入模式与训练样本中所有输入模式不完全相同时，则可得到与其相近样本相对应的输出模式。
- 当待识别的输入模式与训练样本中所有输入模式相差较远时，就不能得到正确的识别结果，此时可将这一模式作为新的样本进行训练，使神经网络获取新的知识，并存储到网络的权值矩阵中，从而增强网络的识别能力。

7.2 BP神经网络

BP网络的训练过程如下：正向传播是输入信号从输入层经隐层传向输出层，若输出层得到了期望的输出，则学习算法结束；否则，转至反向传播。以第p个样本为例，用于训练的BP网络结构如下图所示。



7.2 BP神经网络

网络的学习算法如下：

（1）前向传播：计算网络的输出。

隐层神经元的输入为所有输入的加权之和：

$$x_j = \sum_i w_{ij} x_i$$

隐层神经元的输出 $x_j^{'}$ 采用S函数激发 x_j：

$$x_j^{'} = f(x_j) = \frac{1}{1+e^{-x_j}}$$

则

$$\frac{\partial x_j^{'}}{\partial x_j} = x_j^{'}(1-x_j^{'})$$

7.2 BP神经网络

输出层神经元的输出：

$$x_l = \sum_j w_{jl} x_j^{'}$$

网络第l个输出与相应理想输出x_l^0的误差为：

$$e_l = x_l^0 - x_l$$

第p个样本的误差性能指标函数为：

$$E_p = \frac{1}{2}\sum_{l=1}^{N} e_l^{\,2}$$

其中N为网络输出层的个数。

7.2 BP神经网络

（2）反向传播：采用梯度下降法，调整各层间的权值。权值的学习算法如下：

输出层及隐层的连接权值 w_{jl} 学习算法为：

$$\Delta w_{jl} = -\eta \frac{\partial Ep}{\partial w_{jl}} = \eta el \frac{\partial x_l}{\partial w_{jl}} = \eta e_l x_j'$$

$$w_{jl}(k+1) = w_{jl}(k) + \Delta w_{jl}$$

7.2 BP神经网络

隐层及输入层连接权值 w_{ij} 学习算法为：

$$\Delta w_{ij} = -\eta \frac{\partial E_p}{\partial w_{ij}} = \eta \sum_{l=1}^{N} e_l \frac{\partial x_l}{\partial w_{ij}}$$

$$w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}$$

其中

$$\frac{\partial x_l}{\partial w_{ij}} = \frac{\partial x_l}{\partial x_j^{'}} \cdot \frac{\partial x_j^{'}}{\partial x_j} \cdot \frac{\partial x_j}{\partial w_{ij}} = w_{il} \cdot \frac{\partial x_j^{'}}{\partial x_j} \cdot x_i = w_{jl} \cdot x_j^{'}(1-x_j^{'}) \cdot x_i$$

7.2 BP神经网络

如果考虑上次权值对本次权值变化的影响，需要加入动量因子α，此时的权值为：

$$w_{jl}(k+1)=w_{jl}(k)+\Delta w_{jl}+\alpha\left(w_{jl}(k)-w_{jl}(k-1)\right)$$

$$w_{ij}(\mathrm{k}+1)=w_{ij}(\mathrm{k})+\Delta w_{ij}+\alpha(w_{ij}(\mathrm{k})-w_{ij}(\mathrm{k}-1))$$

其中α为动量因子

$$\alpha \in [0,1]$$

7.2 BP神经网络

（七）仿真实例：

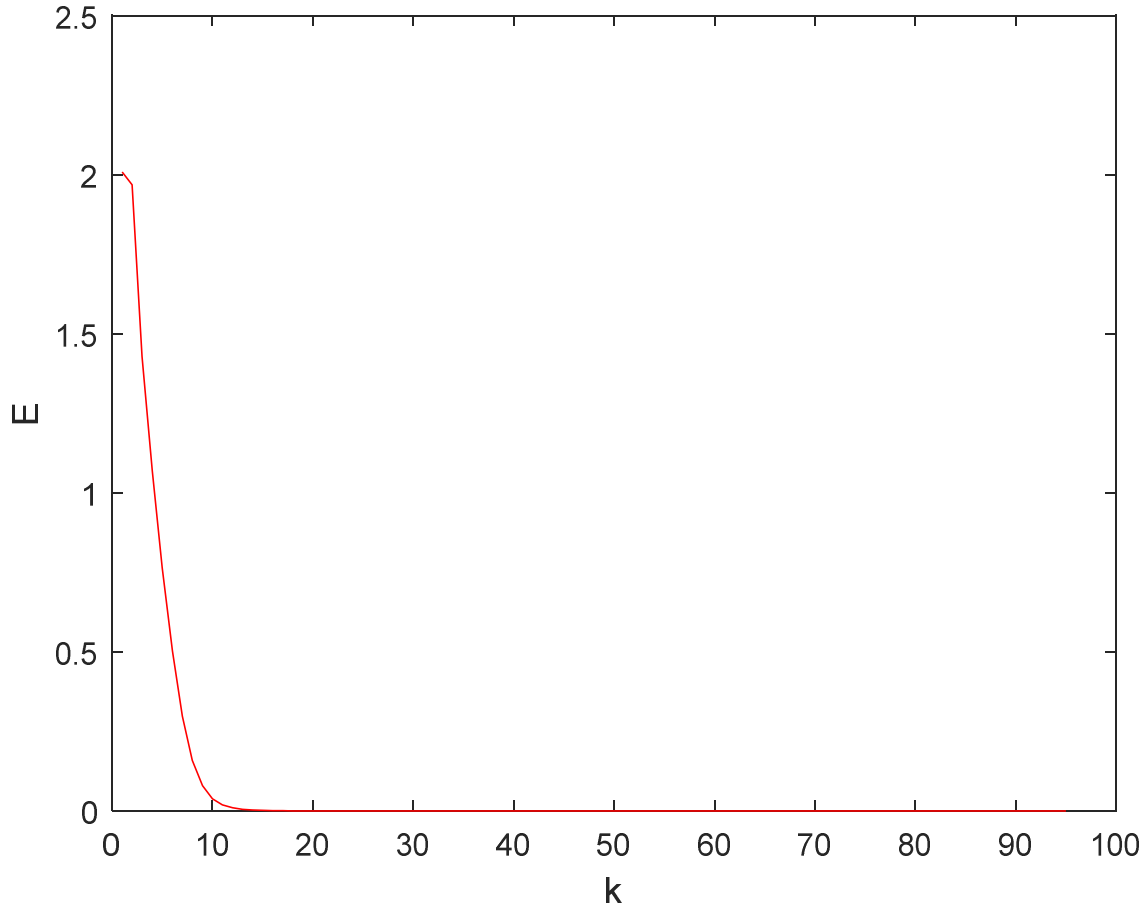
取标准样本为三输入两输出样本，如表7-1所示。

表7-1 训练样本

输 入			输 出	
1	0	0	1	0
0	1	0	0	0.5
0	0	1	0	1

BP网络模式识别程序包括网络训练程序chap7_2a.m和网络测试程序chap7_2b.m。

7.2 BP神经网络



7.2 BP神经网络

输入

```
x=[0.970,0.001,0.001;
   0.000,0.980,0.000;
   0.002,0.000,1.040;
   0.500,0.500,0.500;
   1.000,0.000,0.000;
   0.000,1.000,0.000;
   0.000,0.000,1.000];
```

输出

```
y =

    0.9853    0.0102
    0.0088    0.4975
   -0.0156    1.0205
    0.2497    0.5769
    1.0000    0.0000
   -0.0000    0.5000
    0.0000    1.0000
```

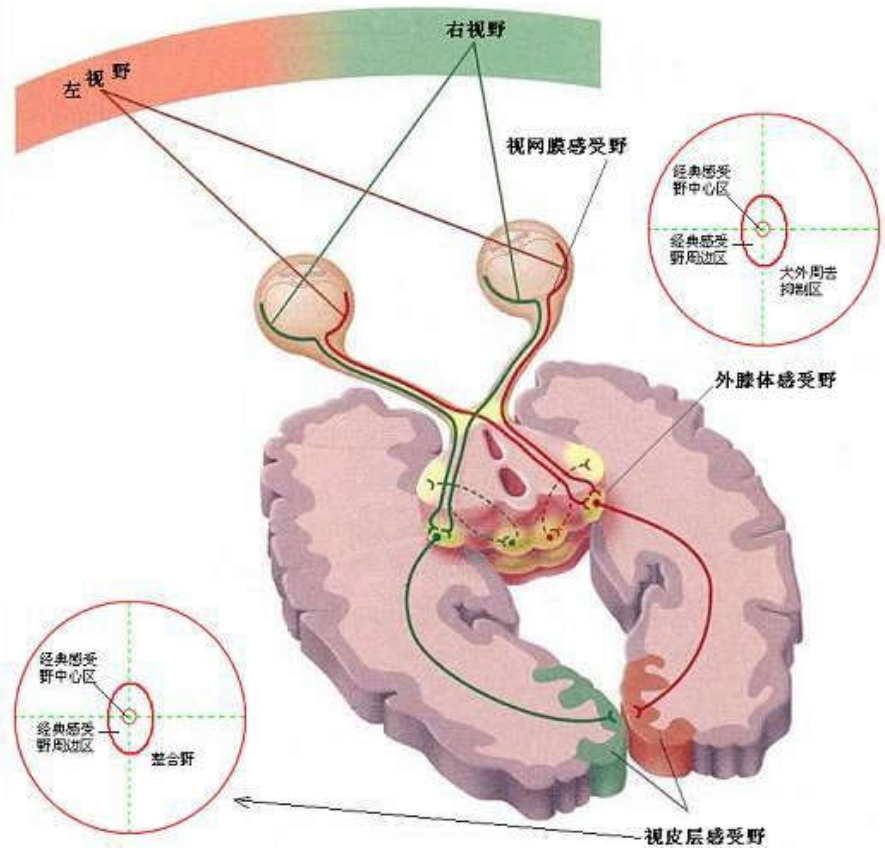
7.3 RBF神经网络

径向基函数(RBF-Radial Basis Function)神经网络是由J.Moody和C.Darken在80年代末提出的一种神经网络，它是具有单隐层的三层前馈网络。由于它模拟了人脑中局部调整、相互覆盖接收域（或称感受野-Receptive Field）的神经网络结构，因此，RBF网络是一种局部逼近网络，已证明它能任意精度逼近任意连续函数。

7.3 RBF神经网络

感受野，感受器受刺激兴奋时，通过感受器官中的向心神经元将神经冲动（各种感觉信息）传到上位中枢，一个神经元所反应（支配）的刺激区域就叫做神经元的感受野（receptive field）。

【例】在视觉通路上，视网膜上的光感受器（杆体细胞和锥体细胞）通过接受光并将它转换为输出神经信号而来影响许多神经节细胞，外膝状体细胞以及视觉皮层中的神经细胞。反过来，任何一种神经细胞（除起支持和营养作用的神经胶质细胞外）的输出都依赖于视网膜上的许多光感受器。我们称直接或间接影响某一特定神经细胞的光感受器细胞的全体为该特定神经细胞的感受野。



7.3 RBF神经网络

与全局作用对应的是局部作用。在局部作用中，每个局部神经元只对特定区域的输入产生响应。如果输入在空间上是相近的，对这些输入的反应应该是相似的，那么被这些输入激活的神经元也应该是同一批神经元。

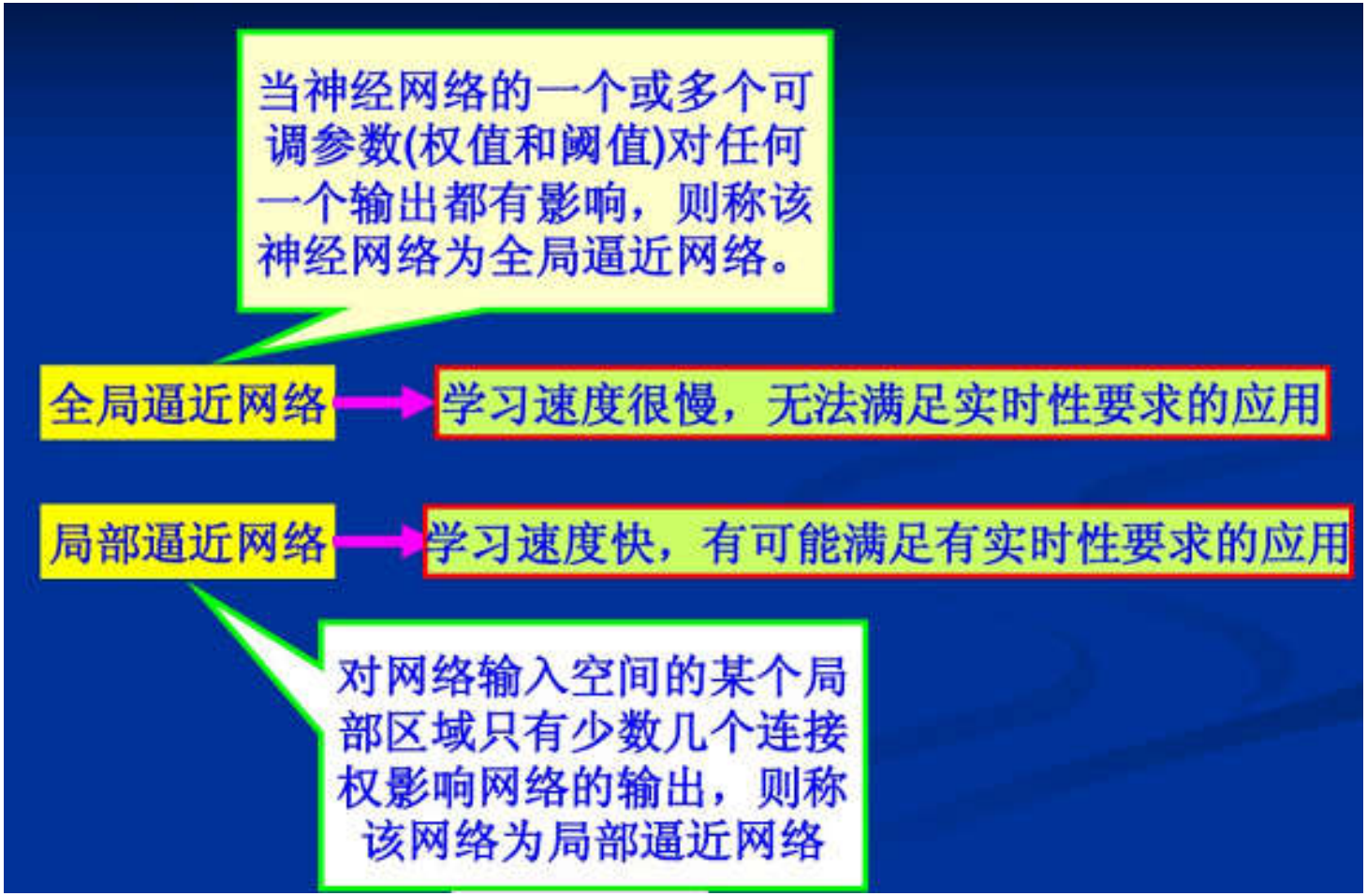
神经元的局部作用原理有它的生理学依据。

【例】当你仰望夜空中的点点繁星时，茫茫暗夜中的星光激活的是视觉神经的特定部分。随着地球的自转，星光也会移动，虽然亮度没有变化，但不同位置的星光激活的就是视觉神经中的不同部分，因而产生响应的神经元也会发生变化。有些原本被激活的神经元会因为目标对象的移出而被抑制，有些原本被抑制的神经元则因为目标对象的移入而被激活。

在神经科学中，这个概念被称为“感受野（receptive field）”。一个感觉神经元的感受野指的是位于这一区域内的适当刺激能够引起该神经元反应的区域。人类神经的感受野的变化方式可以在人工神经网络中以权重系数的形式体现出来，而按照感受野的变化规律设置权重系数，得到的就是“径向基函数神经网络”。

7.3 RBF神经网络

RBF网络的学习过程与BP网络类似，区别在于：



7.3 RBF神经网络

Poggio和Girosi已经证明，RBF网络是连续函数的最佳逼近，而BP网络不是。

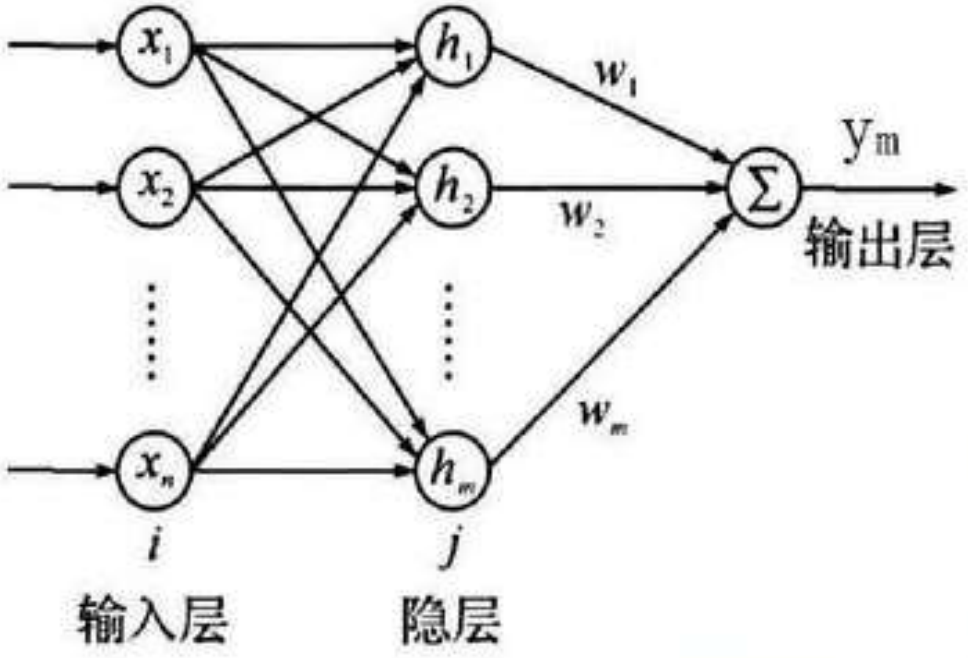
In comparison, both the RBF and BP networks can approximate any non-linear linear function in reasonable precision. However, because the RBF uses a different transfer function in the hidden layer than that of the BP network, the accuracy in approximating the input function is also different. Poggio and Girosi [13] have proven that the RBF network is superior in approximating continuous functions. Although sigmoid function commonly used in the BP network has better ability in generalization, its results will affect much more neurons in adjusting their weights. Moreover, the transfer function overlaps in a large range of the input values, so the outputs inter-reference each other. The RBF network increases the speed of training by using a local transfer function so that only a few neurons have a non-zero response and become active to each input value. Therefore when new data are adopted, only the weights of a few active neurons are required to be modified. In summary, the training of the BP network is slower and more difficult to be convergent than the RBF network. Moreover, the RBF network can avoid falling into local minimum when the training is in progress [14].

Lai Wuxing et al., Classification of gear faults using cumulants and the radial basis function network, Mechanical Systems and Signal Processing, Volume 18, Issue 2, March 2004, Pages 381-389

7.3 RBF神经网络

（一）RBF网络结构与算法

多输入单输出的RBF神经网络结构如下图所示。



7.3 RBF神经网络

在RBF神经网络中，$\boldsymbol{x}=\begin{bmatrix}x_1 & x_2 & \cdots & x_n\end{bmatrix}^T$为网络输入，$h_j$为隐含层第$j$个神经元的输出，即

$$h_j=\exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_j\|^2}{2b_j^2}\right),j=1,2,\cdots m \tag{7.20}$$

式中，$\boldsymbol{c}_j=\begin{bmatrix}c_{j1}, & \cdots, & c_{jn}\end{bmatrix}$

为第j个隐层神经元的中心点矢量值。

高斯基函数的宽度矢量为

$$\boldsymbol{b}=\begin{bmatrix}b_1, & \cdots, & b_m\end{bmatrix}^T$$

其中$b_j>0$为隐含层神经元j的高斯基函数的宽度。

7.3 RBF神经网络

网络的权值为

$$\boldsymbol{w} = [w_1, \quad \cdots, \quad w_m]^{\mathrm{T}} \tag{7.21}$$

RBF网络的输出为

$$y_m(t) = w_1 h_1 + w_2 h_2 + \cdots\cdots + w_m h_m \tag{7.22}$$

$$= \sum_{j=1}^{m} w_j \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{2b_j^2} \right)$$

由于RBF网络只调节一层（隐层与输出层之间）权值，因此，RBF网络较BP网络有算法简单、运行时间快的优点。但由于RBF网络中，输入空间到输出空间是非线性的，而隐含空间到输出空间是线性的，因而其非线性能力不如BP网络。

7.3 RBF神经网络

RBF网络特点

(1) 已证明RBF网络具有唯一最佳逼近的特性，且无局部极小。

(2) 如何确定RBF网络隐层节点的中心c_j及基宽度参数b_j是一个困难的问题；

(3) 径向基函数，即径向对称函数有多种。对于一组样本，如何选择合适的径向基函数、隐节点数，使网络学习达到要求的精度，是一个还没有解决的问题。

(4) RBF网络适用于非线性系统辨识与控制，上述问题是该网络难以广泛应用的原因。

7.3 RBF神经网络

为什么高斯核函数就是映射到高维空间？

首先给出高斯核函数的定义公式：

$$K(\vec{X_i},\vec{X_j}) = e^{-\frac{\|\vec{X_i}-\vec{X_j}\|^2}{2\sigma^2}}$$

实际上，可以化简为：

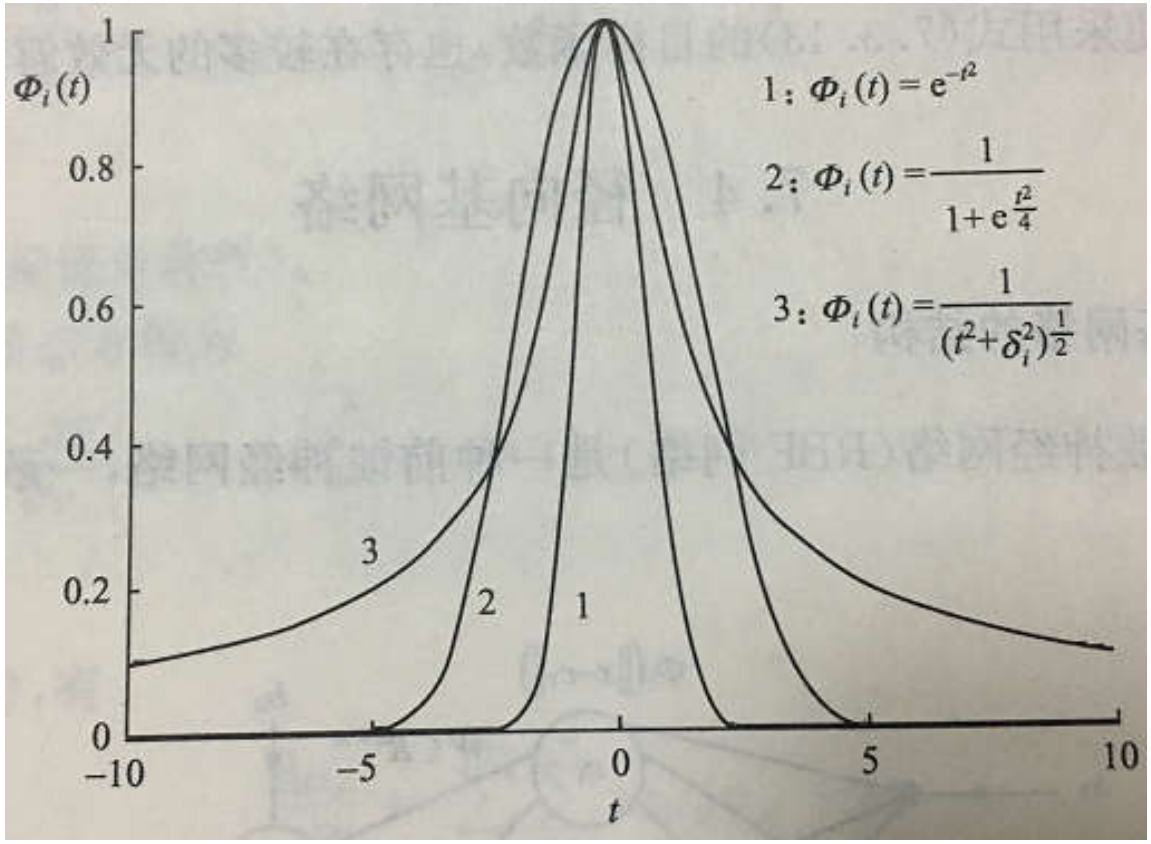
$$K'(\vec{X_i},\vec{X_j}) = e^{-\frac{\vec{X_i}\cdot\vec{X_j}}{\sigma^2}}$$

当然通过幂级数展开：

$$K'(\vec{X_i},\vec{X_j}) = \sum_{n=0}^{+\infty}\frac{(\vec{X_i}\cdot\vec{X_j})^n}{\sigma^n n!}$$

可以看到，其中X向量会生成类似多项式核展开的形式，譬如原来的参数有x1,x2。映射后，参数包含了x1*x1, x1*x2, x2*x2将原来2维映射到3维上了。

7.3 RBF神经网络



Gaussian函数

$$\Phi(t) = e^{-\frac{t^2}{\delta^2}}$$

Reflected Sigmoidal函数

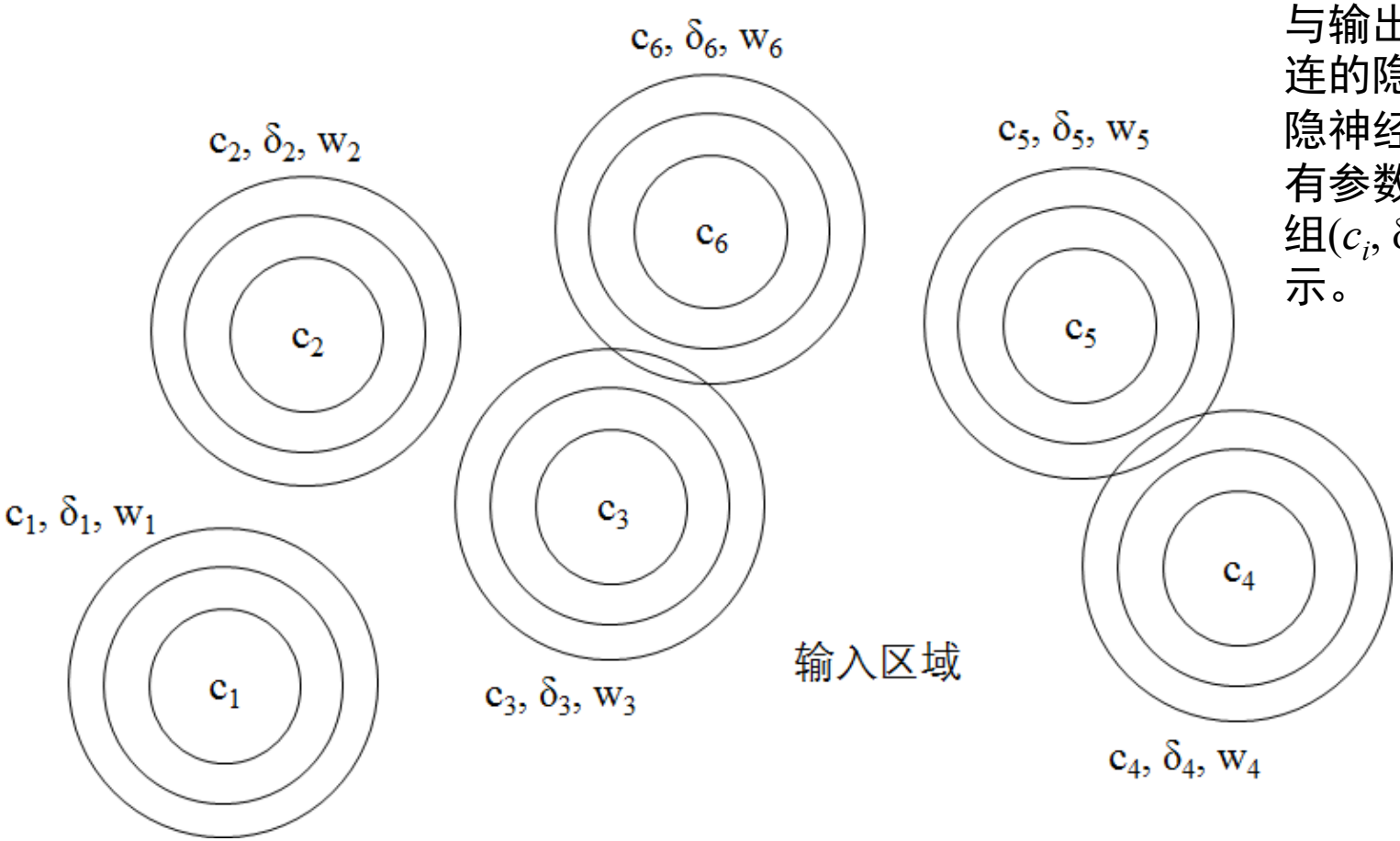
$$\Phi(t) = \frac{1}{1+e^{\frac{t^2}{\delta^2}}}$$

逆Multiquadric函数

$$\Phi(t) = \frac{1}{\left(t^2+\delta^2\right)^{\alpha}}, \quad \alpha > 0$$

δ越小，径向基函数的宽度就越小，基函数就越具有选择性。

7.3 RBF神经网络



与输出节点相连的隐层第i个隐神经元的所有参数用三元组(c_i, δ_i, w_i)表示。

每个隐层神经元都对输入产生响应，且响应特性呈径向对称（即使一个个同心圆）。

输入区域中有6个神经元，每个神经元都对输入x产生一个响应$\Phi(\|x-c_i\|)$，而神经网络的输出为这些响应的加权和。

7.3 RBF神经网络

RBF网络的学习算法

无监督学习

（1）随机选取训练样本数据作为聚类中心向量初始化，确定J个初始聚类中心向量。

（2）将输入训练样本数据xi按最邻近聚类原则选择最近的聚类j^*。

$$c_j^* = \arg\min_j \left\| x_i - c_j \right\|$$

（3）聚类中心向量更新。若全部聚类中心向量不再发生变化，则所得到的聚类中心即为RBF网络最终基函数中心，否则返回（2）。

有监督学习

（4）采用较小的随机数对隐层和输出层间的权值初始化。

（5）利用最小二乘法计算隐层和输出层间的权值w_{kj}。

（6）权值更新。首先求出各输出神经元的误差 $e_k = d_k - y_k(x)$

其中d_k是输出神经元k的期望输出。然后，再更新权值为

其中η是学习率。 $$w_{kj}^{new} = w_{kj}^{old} + \eta e_k G_j \left(x - c_j \right)$$

（7）若满足终止条件，结束。否则返回（5）。

7.3 RBF神经网络

（二）RBF网络设计实例

(1) 结构为1-5-1的RBF网络

考虑结构为1-5-1的RBF网络，取网络输入为 $x = x_1$

令 $\boldsymbol{b} = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 \end{bmatrix}^{\mathrm{T}}$ $\boldsymbol{c} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} \end{bmatrix}$

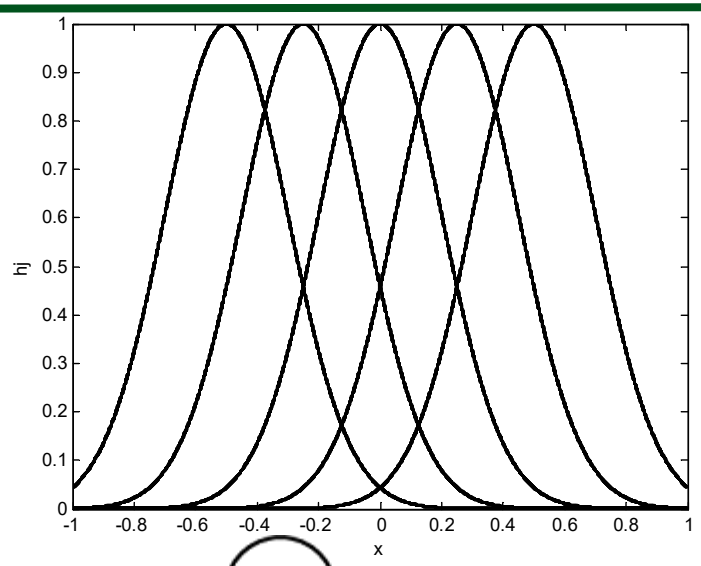
$\boldsymbol{h} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 \end{bmatrix}^{\mathrm{T}}$ $\boldsymbol{w} = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix}^{\mathrm{T}}$

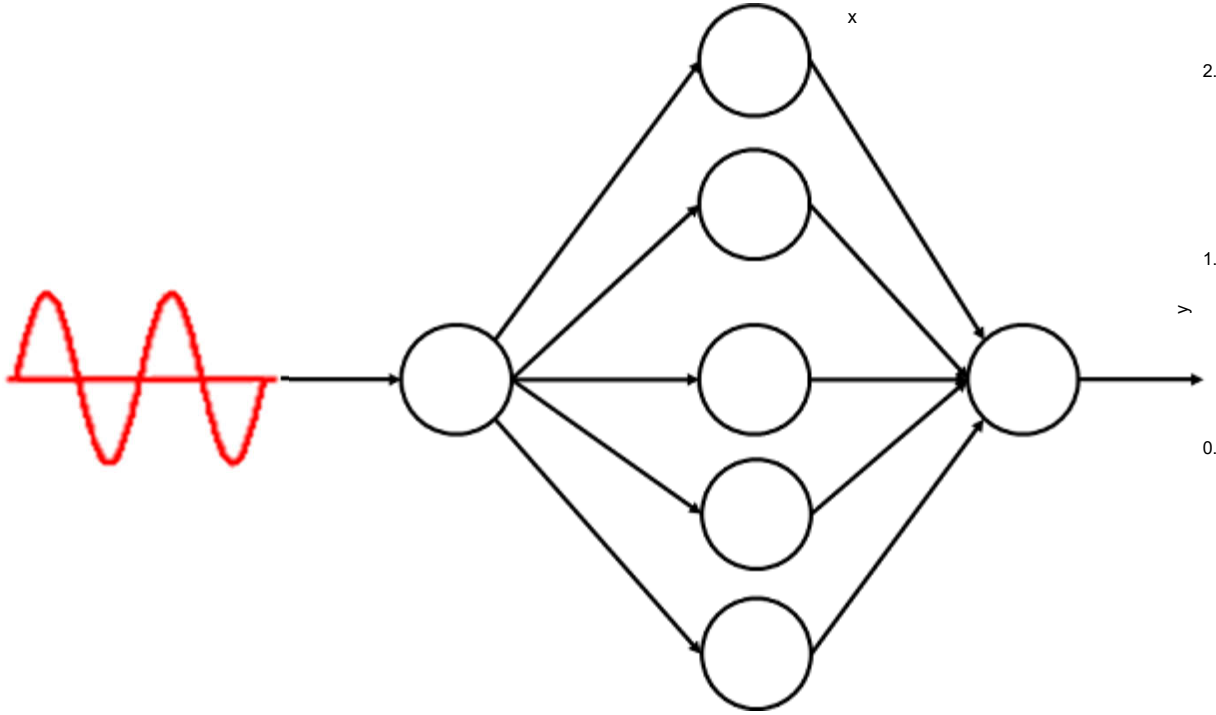
则网络输出为

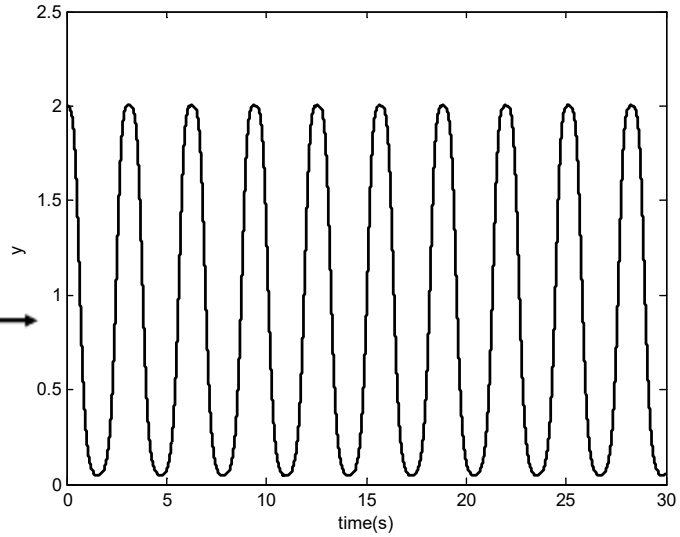
$$y_{\mathrm{m}}(t) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{h} = w_1 h_1 + w_2 h_2 + w_3 h_3 + w_4 h_4 + w_5 h_5$$

取网络的输入为sin$t$ 时，网络的输出和隐含层的输出如下图所示。仿真程序为chap7_3sim.mdl。

7.3 RBF神经网络


hj
x
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1




y
time(s)
2.5
2
1.5
1
0.5
0
0
5
10
15
20
25
30

7.3 RBF神经网络

(2) 结构为2-5-1的RBF网络

考虑结构为2-5-1的RBF网络，取网络输入为$x=[x_1, x_2]^{\mathrm{T}}$，令

$$\boldsymbol{b}=\begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 \end{bmatrix}^{\mathrm{T}} \quad \boldsymbol{c}=\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} \\ c_{21} & c_{22} & c_{23} & c_{24} & c_{25} \end{bmatrix}$$

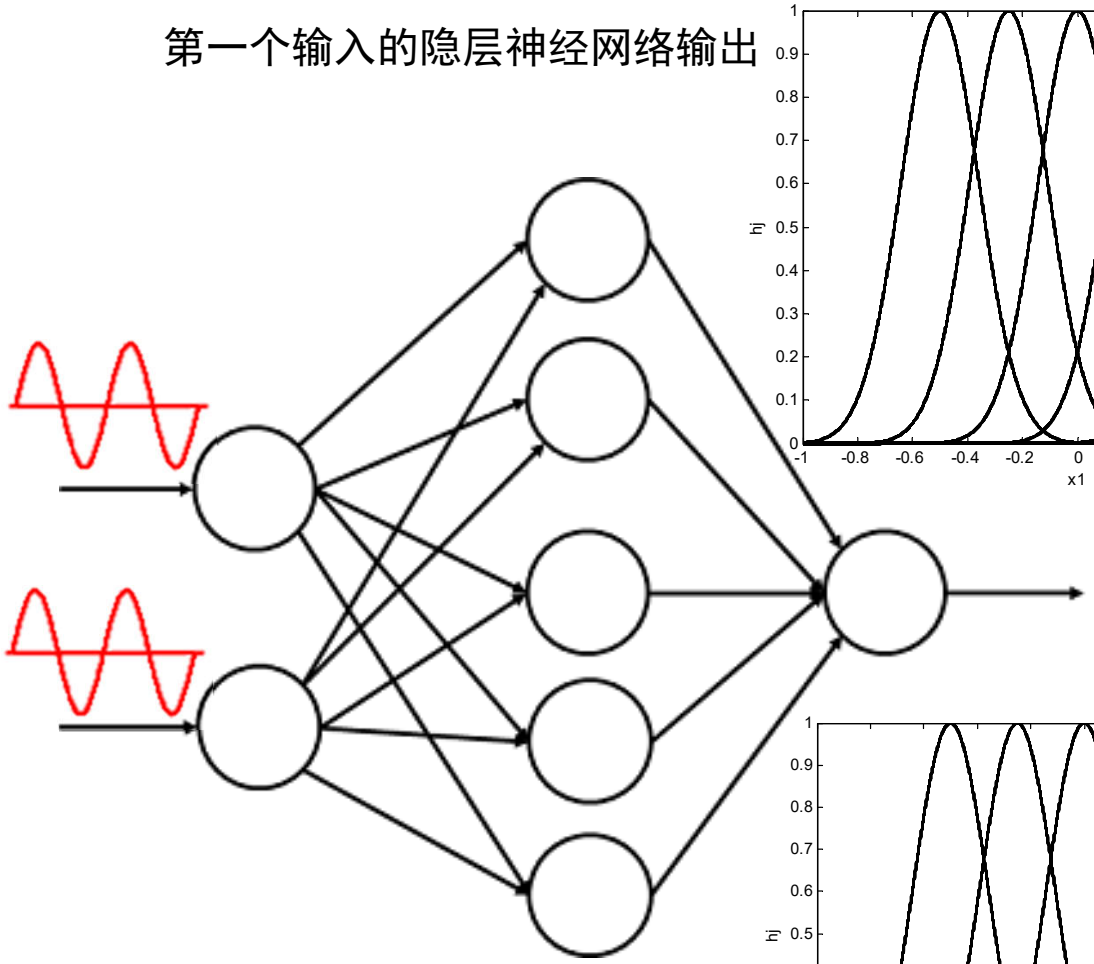
$$\boldsymbol{h}=\begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 \end{bmatrix}^{\mathrm{T}} \quad \boldsymbol{w}=\begin{bmatrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix}^{\mathrm{T}}$$

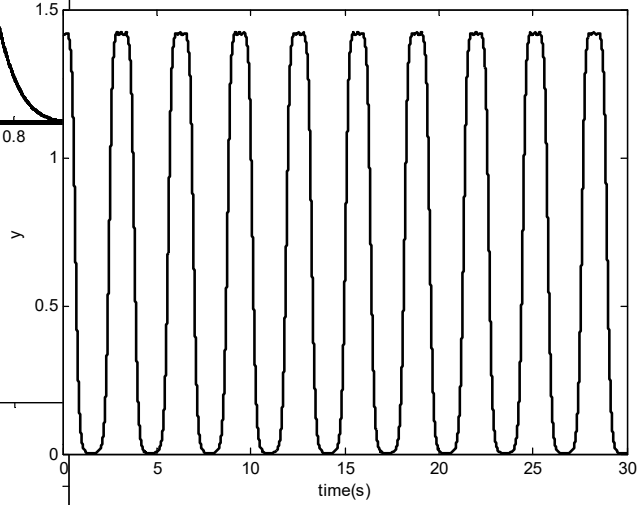
网络输出为 $y_{\mathrm{m}}(t)=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{h}=w_1h_1+w_2h_2+w_3h_3+w_4h_4+w_5h_5$

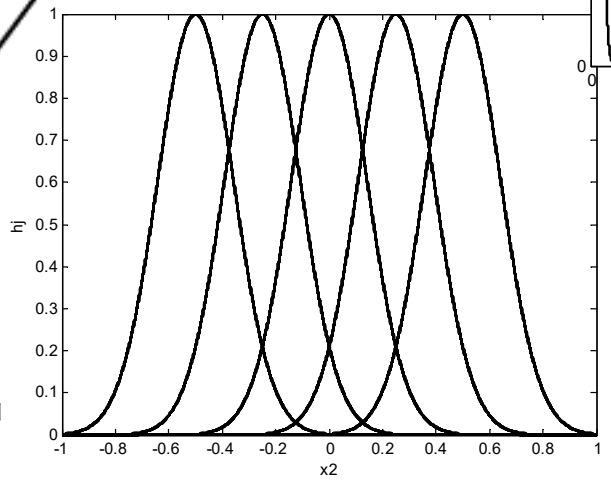
取网络的输入为sin$t$ 时，网络的输出如图7-16所示，网络隐含层的输出如图7-17和图7-18所示。仿真程序为chap7_4sim.mdl。

7.3 RBF神经网络

第一个输入的隐层神经网络输出







第二个输入的隐层神经网络输出

7.3 RBF神经网络

（三）RBF网络的逼近

1. 基本原理

采用RBF网络对模型进行逼近，结构如图7-19所示。

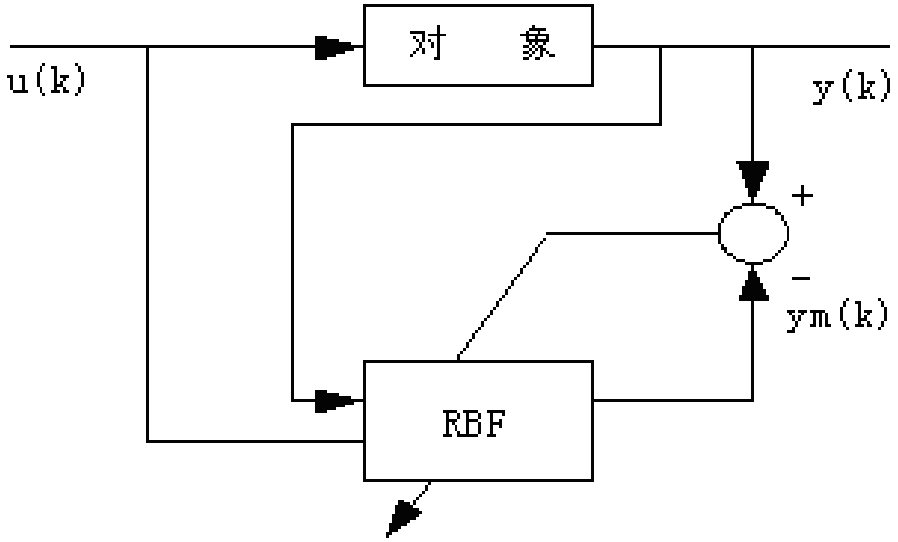


图7-19 RBF神经网络逼近

7.3 RBF神经网络

- 在RBF网络设计中，需要注意的是将cj和 b值设计在网络输入有效的映射范围内，否则高斯基函数将不能保证实现有效的映射，导致RBF网络失效。
- 如果将 cj和b的初始值设计在有效的映射范围内，则只调节网络的权值便可实现RBF网络的有效学习。

7.3 RBF神经网络

网络逼近的误差指标为

$$E(t)=\frac{1}{2}\left(y(t)-y_{\mathrm{m}}(t)\right)^2 \tag{7.23}$$

根据梯度下降法，权值按以下方式调节

$$\Delta w_j(t)=-\eta\frac{\partial E}{\partial w_j}=\eta\left(y(t)-y_m(t)\right)h_j$$

$$w_j(t)=w_j(t-1)+\Delta w_j(t)+\alpha\left(w_j(t-1)-w_j(t-2)\right) \tag{7.24}$$

其中 $\eta\in(0,1)$ 为学习速率，$\alpha\in(0,1)$为动量因子。

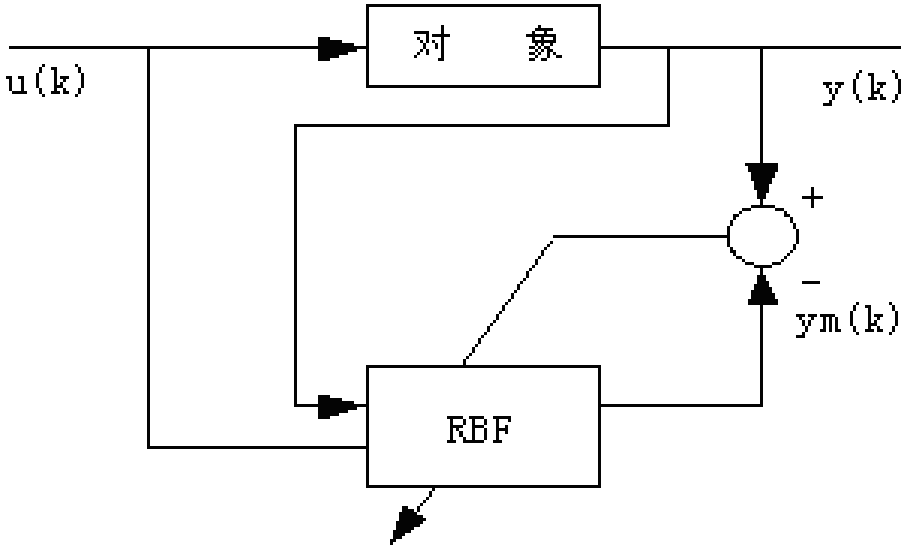
7.3 RBF神经网络

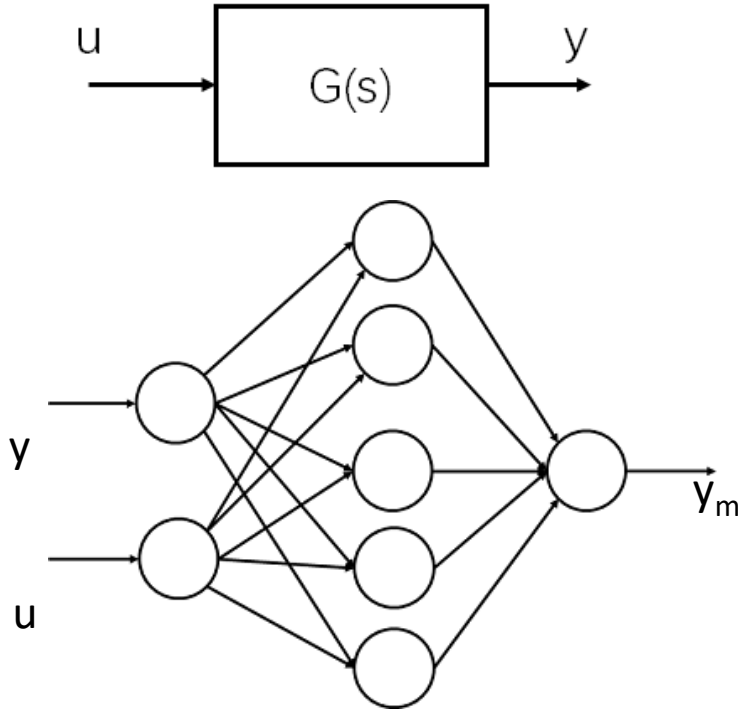
2. 仿真实例

实例1:连续系统

采用RBF网络对如下模型进行逼近 $G(s)=\dfrac{133}{s^2+25s}$

网络结构为2-5-1，取 x(1)=u(t)，x(2)=y(t)，$\alpha=0.05$ $\eta=0.5$





7.3 RBF神经网络

网络的初始权值取0至1之间的随机值。考虑到网络的第一个输入范围为[0,1]，离线测试可得第二个输入范围为[0,10]，取高斯基函数的参数取值为

$$\boldsymbol{c}_j = \begin{bmatrix} -1 & -0.5 & 0 & 0.5 & 1 \\ -10 & -5 & 0 & 5 & 10 \end{bmatrix}^{\mathrm{T}}$$

$$\boldsymbol{b}_j = 1.5 \qquad j = 1,2,3,4,5$$

网络的第一个输入为u(t)=sin$t$，仿真中，只调节权值w，取固定的cj和b，仿真结果如图7-20所示。仿真程序为chap7_5.m。

7.3 RBF神经网络

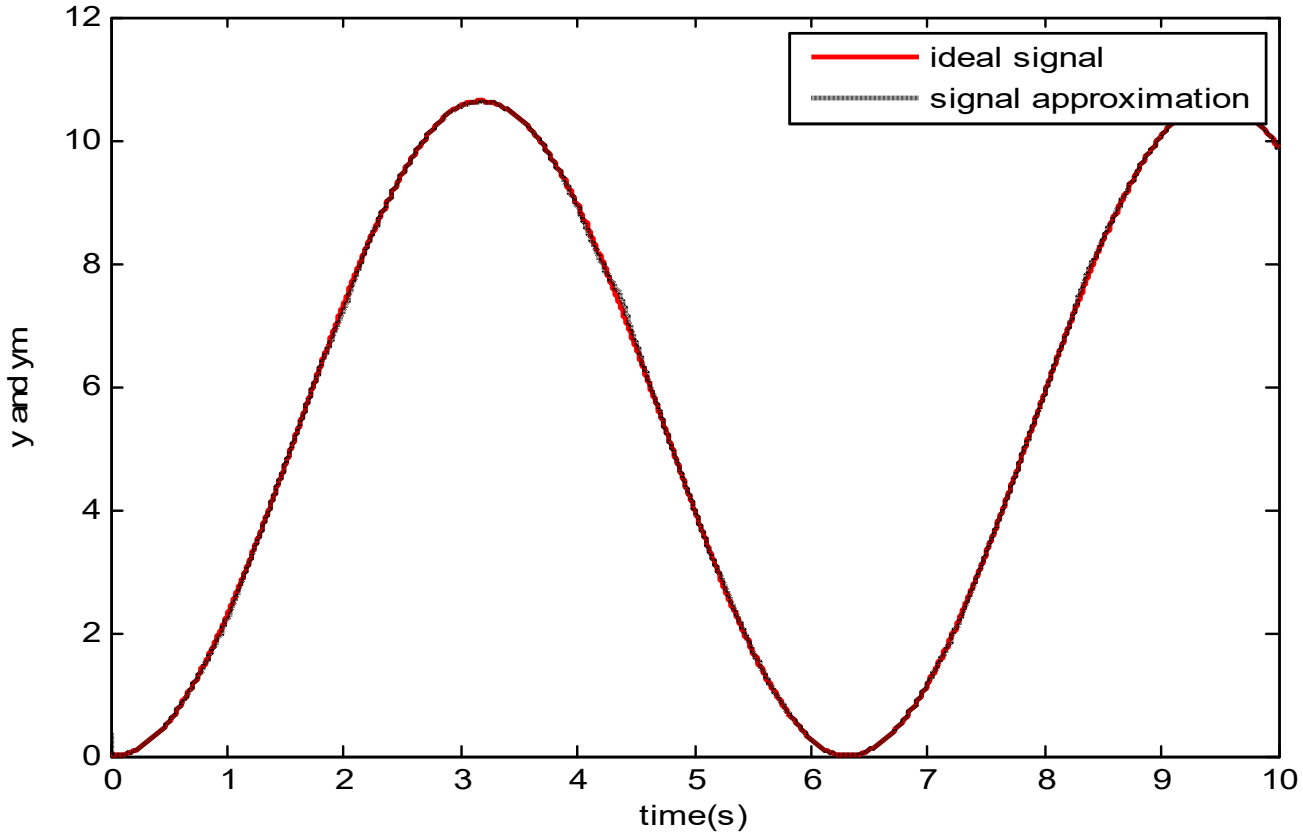


图7-20 基于权值调节的RBF网络逼近

7.3 RBF神经网络

实例2:离散系统

采用RBF网络对如下离散模型进行逼近

$$y(k)=u(k)^3+\frac{y(k-1)}{1+y(k-1)^2}$$

网络的第一个输入为$u(t)=sint$，$t=k\times T$，$T=0.001$。

网络结构为2-5-1，取x(1)=u(t)，x(2)=y(t)，$\alpha=0.05$，$\eta=0.15$

网络的初始权值取0至1之间的随机值。考虑到网络的第一个输入范围为[0,1]，离线测试可得第二个输入范围为[0,10]，取高斯基函数的参数取值为

$$\boldsymbol{c}_j=\begin{bmatrix}-1 & -0.5 & 0 & 0.5 & 1\\ -1 & -0.5 & 0 & 0.5 & 1\end{bmatrix}^{\mathrm{T}} \qquad \boldsymbol{b}_j=3.0 \qquad j=1,2,3,4,5$$

仿真中，调节权值w，取固定的cj和b，仿真结果如图7-21所示。

7.3 RBF神经网络

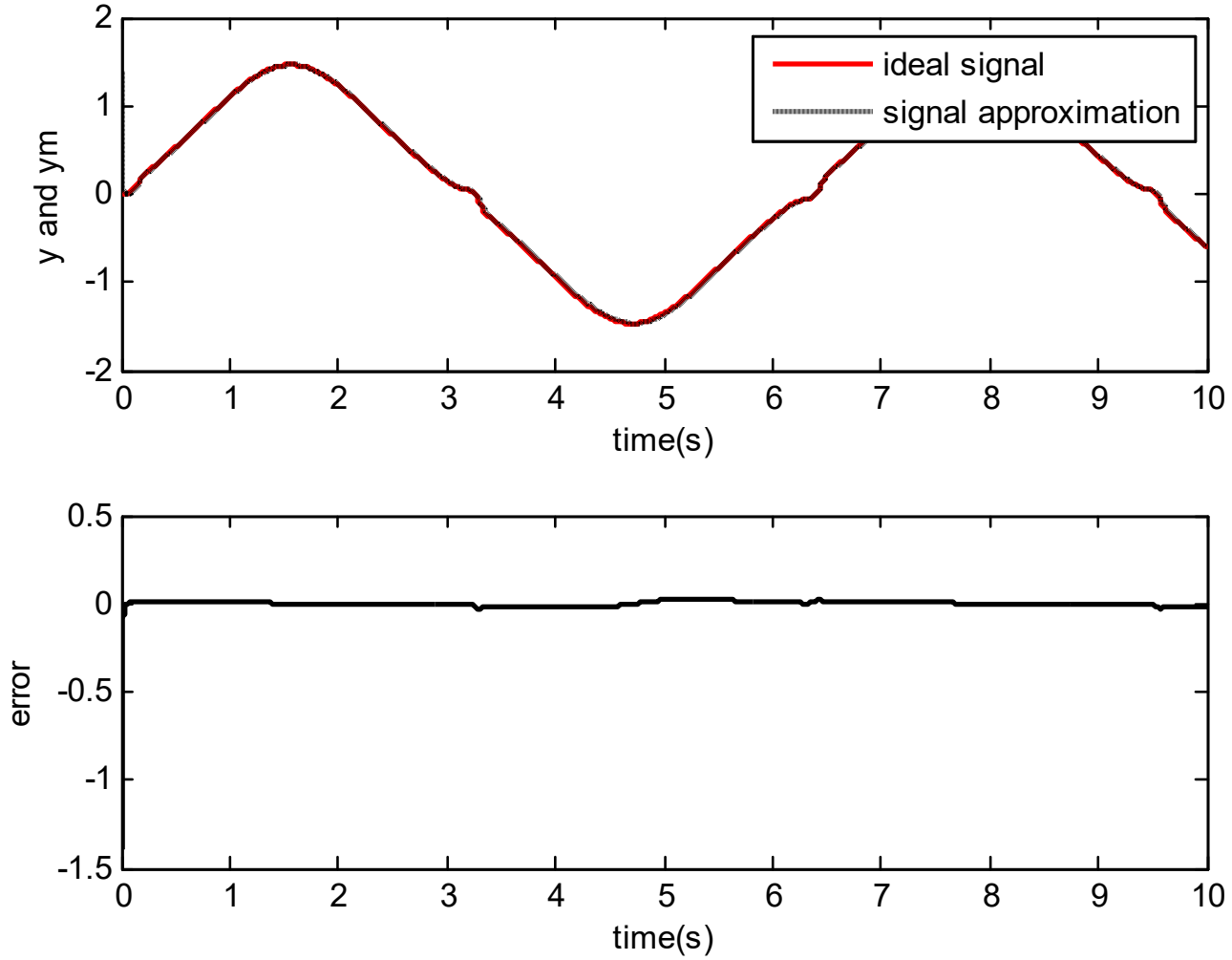


图7-21 基于权值调节的RBF网络逼近

由仿真结果可见，采用梯度下降法可实现很好的逼近效果，其中高斯基函数的参数值c_j和b的取值很重要。仿真程序为chap7_6.m。

7.3 RBF神经网络

（四）高斯基函数的参数对RBF网络逼近的影响

由高斯函数的表达式可知，高斯基函数受参数c_j和b_j的影响，c_j和b_j的设计原则如下：

（1）b_j为隐含层第j 个神经元高斯基函数的宽度。b_j值越大，表示高斯基函数越宽。高斯基函数宽度是影响网络映射范围的重要因素，高斯基函数越宽，网路对输入的映射能力越大，否则，网路对输入的映射能力越小。一般将b_j值设计为适中的值。

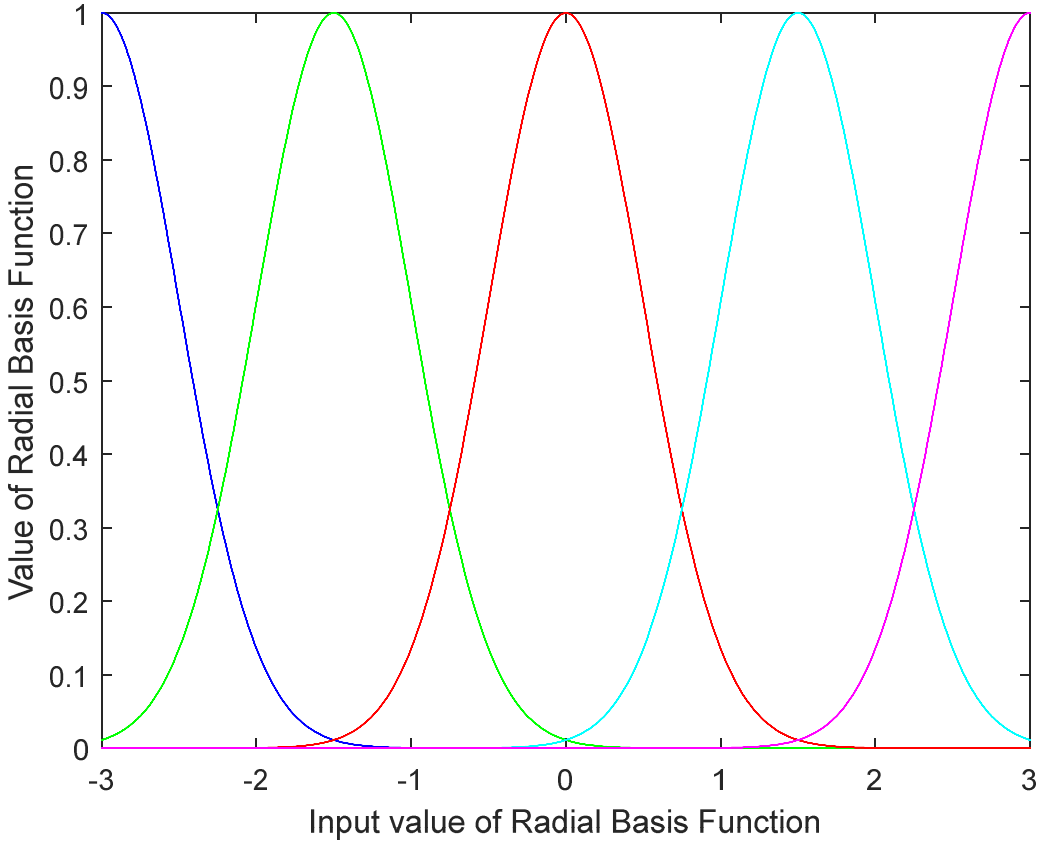
（2）c_j为隐含层第j个神经元高斯基函数中心点的坐标向量。c_j值离输入越近，高斯函数对输入越敏感，否则，高斯函数对输入越不敏感；

（3）中心点坐标向量c_j应使高斯基函数在有效的输入映射范围内。例如，RBF网络输入为[-3,+3]，则c_j为[-3,+3]。

7.3 RBF神经网络

仿真中，应根据网络输入值得范围来设计c_j和b_j,从而保证有效的高斯基函数映射，如图7-22为5个高斯基函数。仿真程序为chap7_7.m。

$$h_j = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_j\|^2}{2b_j^2}\right)$$



$$x = 3\sin(2\pi t)$$

7.3 RBF神经网络

采用RBF网络对如下离散模型进行逼近

$$y(k)=u(k)^3+\frac{y(k-1)}{1+y(k-1)^2}$$

仿真中，取RBF网络输入为$0.5\sin(2\pi t)$，网络结构为2-5-1，通过改变高斯基函数c_j和b_j值，可分析c_j和b_j对RBF网络逼近性能的影响，具体验证以下四种情况：

7.3 RBF神经网络

(1) 合适的b_j和c_j值对RBF网络逼近的影响(Mb=1, Mc=1)；

(2) 不合适的b_j和c_j合适的值对RBF网络逼近的影响(Mb=2, Mc=1)；

(3) 合适的b_j与不合适的c_j值对RBF网络逼近的影响(Mb=1, Mc=2)；

(4) 不合适的b_j和c_j值对RBF网络逼近的影响(Mb=2, Mc=2)。

7.3 RBF神经网络

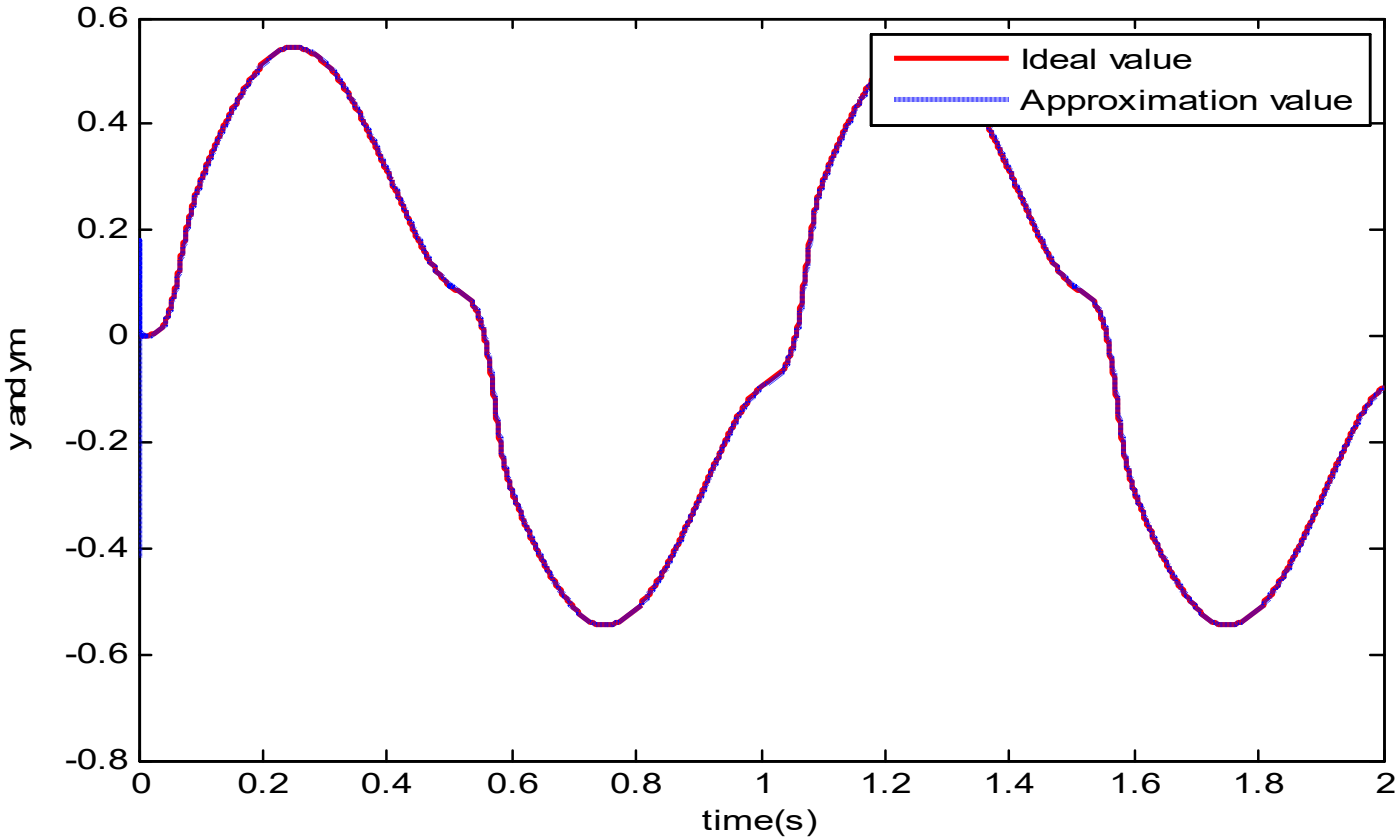


图7-23 合适b_j与c_j值的RBF网络逼近RBF (Mb=1, Mc=1)

7.3 RBF神经网络

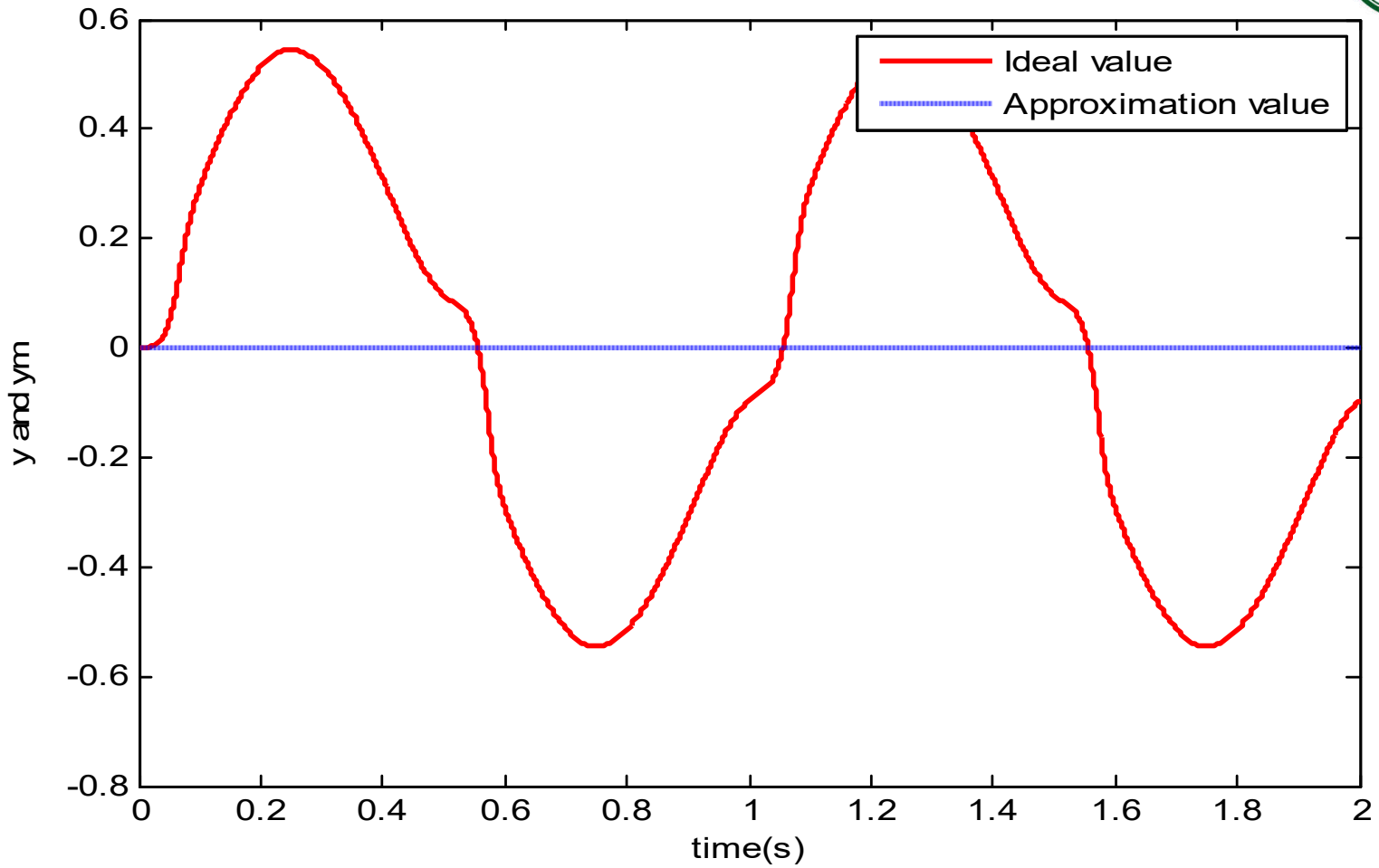


图7-24 不合适b_j与合适c_j值的RBF网络逼近 (Mb=2, Mc=1)

7.3 RBF神经网络

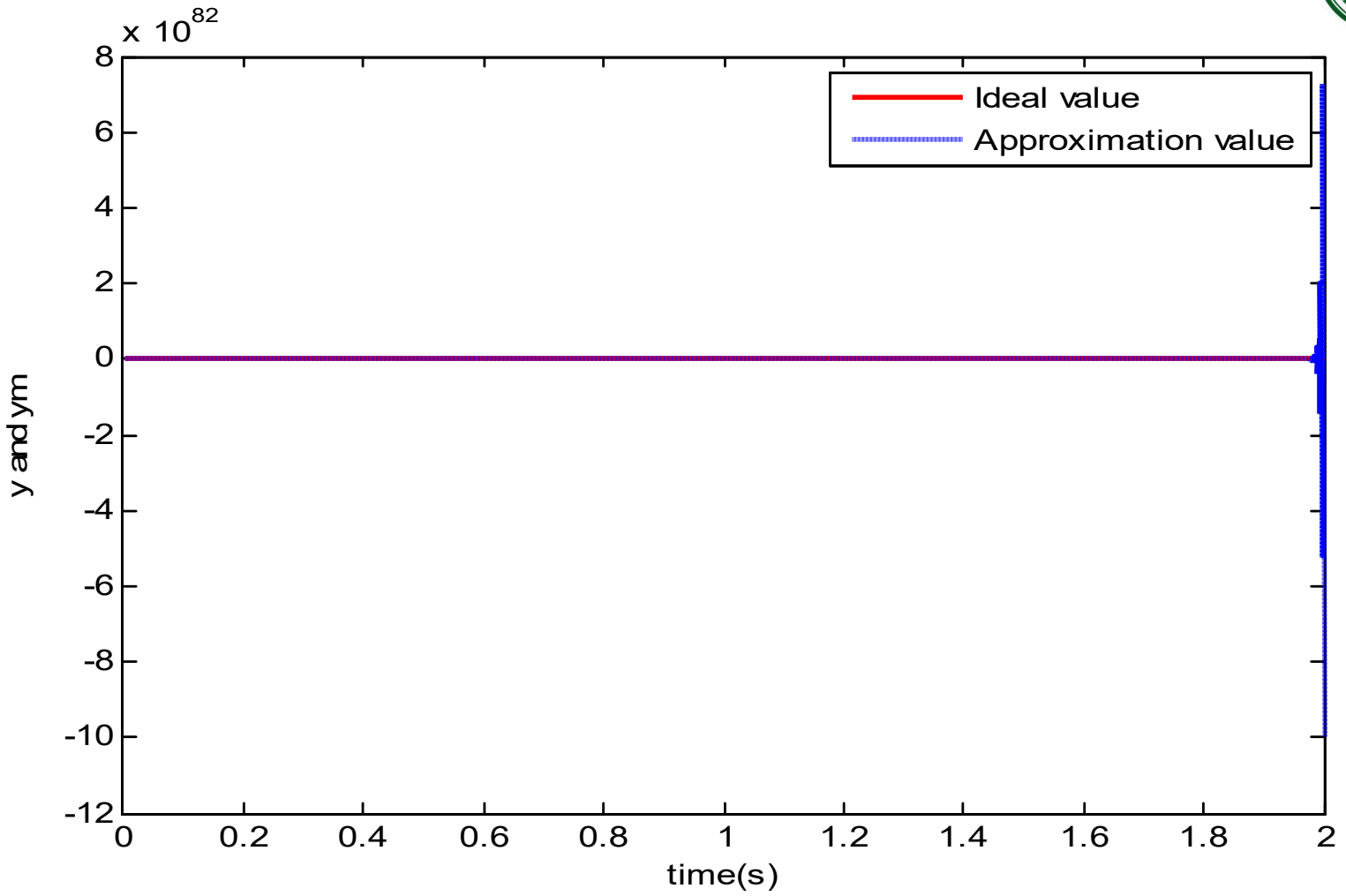


图7-25 合适的 b_j与不合适的c_j值的RBF网络逼近 (Mb=1, Mc=2)

7.3 RBF神经网络

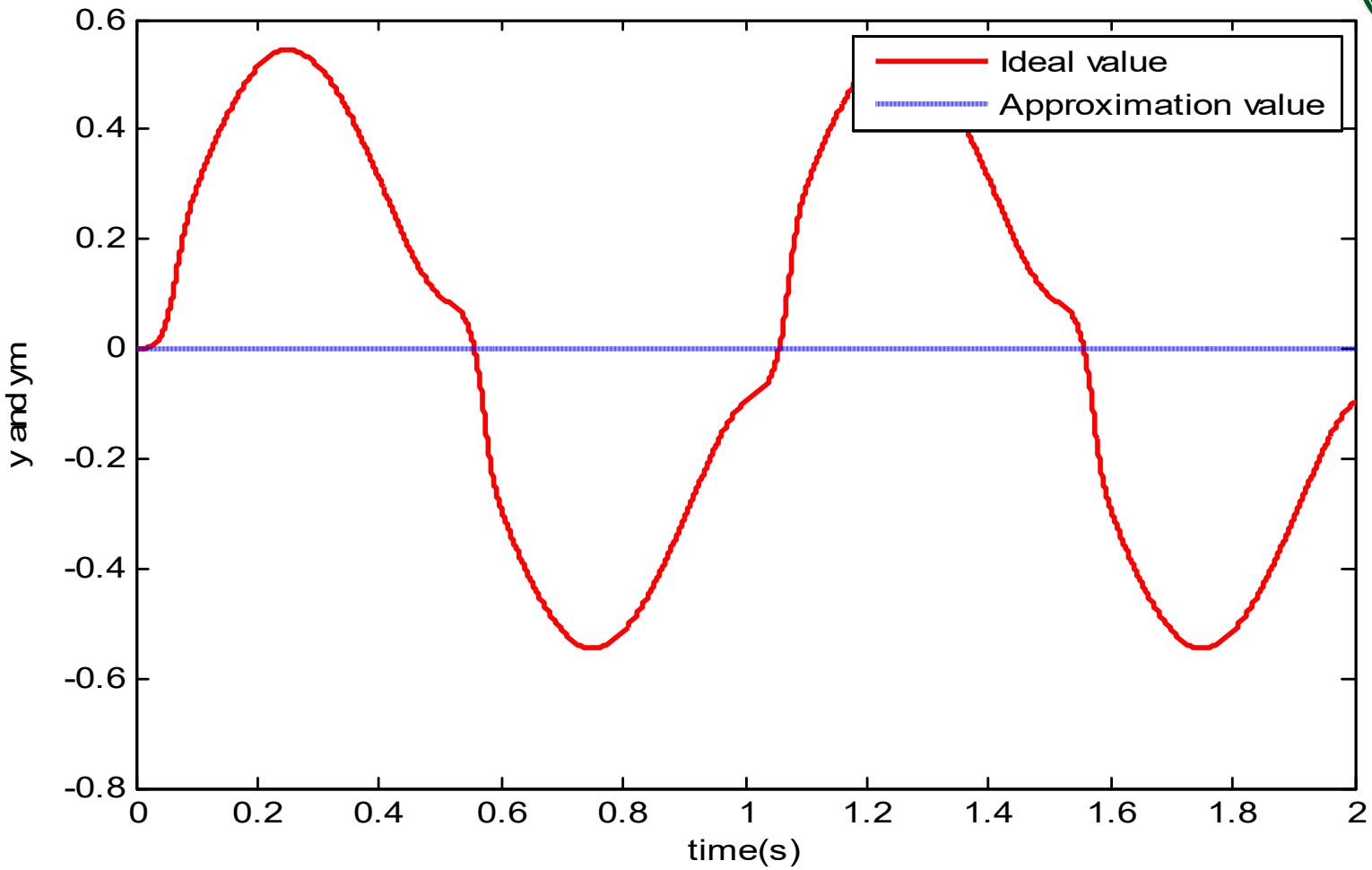


图7-26 不合适b_j和c_j值的RBF网络逼近(Mb=2, Mc=2)

7.3 RBF神经网络

```
if Mb==1          %The width b of Guassian function is
moderate
    b=1.5*ones(5,1);
elseif Mb==2      %The width of Guassian function is too
narrow, most overlap of the function is near to zero
    b=0.0005*ones(5,1);
end
```

```
if Mc==1  %The center position c of Guassian function is
moderate
c=[-1.5 -0.5 0 0.5 1.5;
   -1.5 -0.5 0 0.5 1.5];   %cij
elseif Mc==2  %The center position of Guassian function
is improper
c=0.1*[-1.5 -0.5 0 0.5 1.5;
       -1.5 -0.5 0 0.5 1.5];   %cij
end
```

7.3 RBF神经网络

（五）隐含层节点数对RBF网络逼近的影响

由高斯函数的表达式可见，逼近误差除了与高斯函数的中心点坐标 c_j 和宽度参数b_j有关，还与隐含层神经元节点数量有关。

采用RBF网络对如下离散模型进行逼近

$$y(k)=u(k)^3+\frac{y(k-1)}{1+y(k-1)^2}$$

仿真中，取$\alpha=0.05$ ，$\eta=0.3$ 。神经网络权值的初始值取零，取高斯基函数参数b_j=1.5 。取RBF网络的输入为u(k)=sin$t$ 和y(k)，网络结构取2-m-1, m为隐含层节点数。为了表明隐含层节点数对网络逼近的影响，分别取m=1，m=3，m=7，所对应的 c_j分别取c_j=0，$\boldsymbol{c}_j=\frac{1}{3}\begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}^{\mathrm{T}}$和

$$\boldsymbol{c}_j=\frac{1}{9}\begin{bmatrix}-3 & -2 & -1 & 0 & 1 & 2 & 3\\ -3 & -2 & -1 & 0 & 1 & 2 & 3\end{bmatrix}^{\mathrm{T}}$$

7.3 RBF神经网络

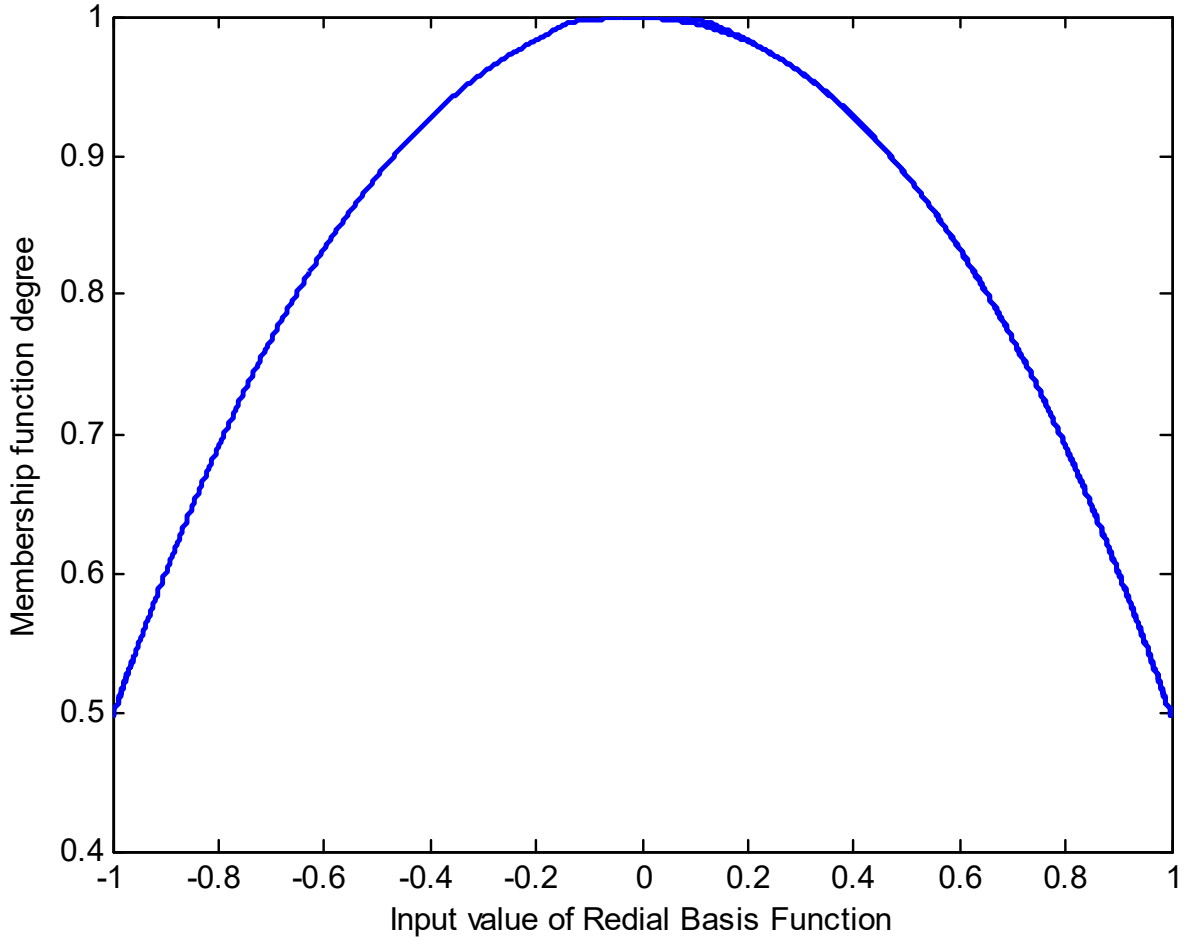


图7-27 单个隐含神经元的高斯基函数(m=1)

7.3 RBF神经网络

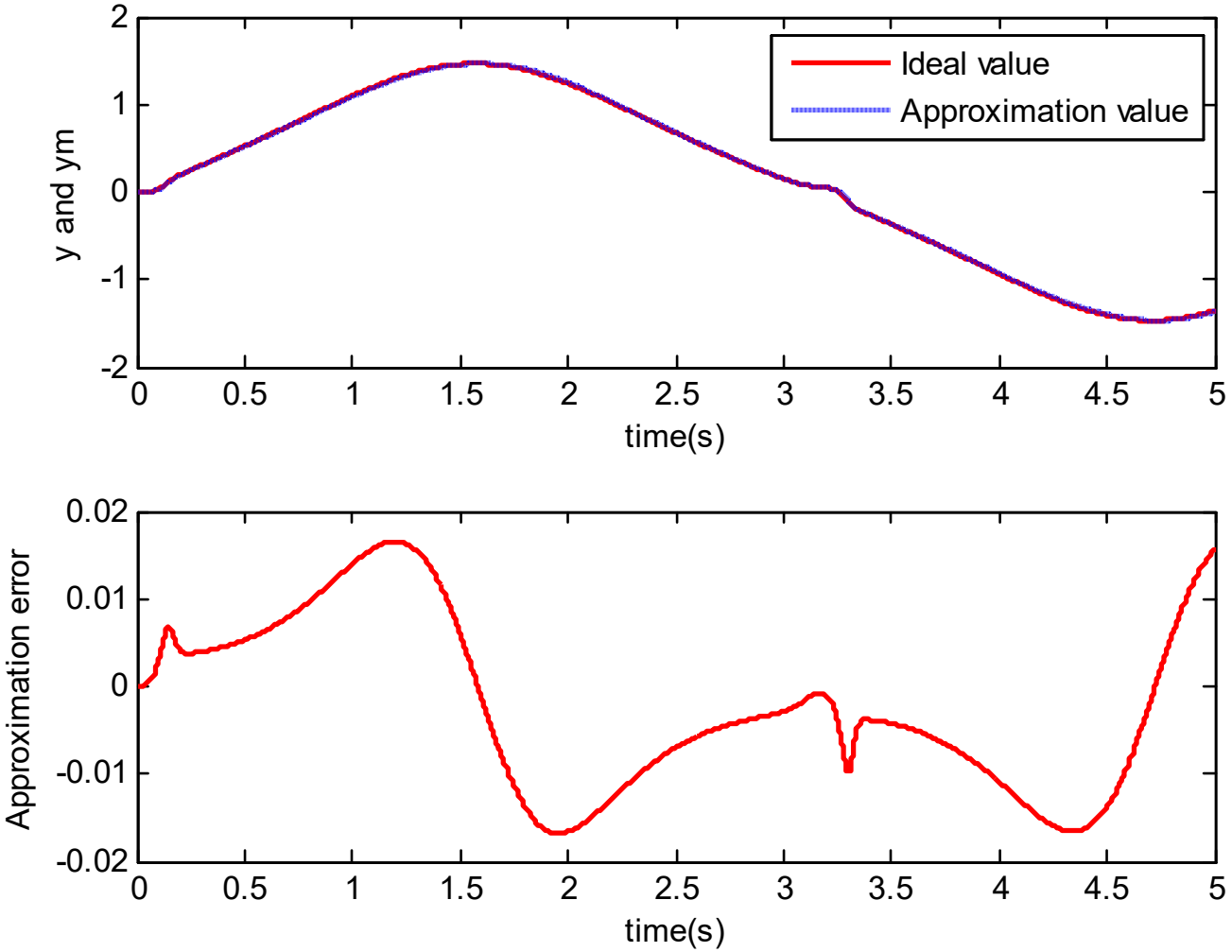


图7-28 含有单个隐含层神经元的逼近 (m=1)

7.3 RBF神经网络

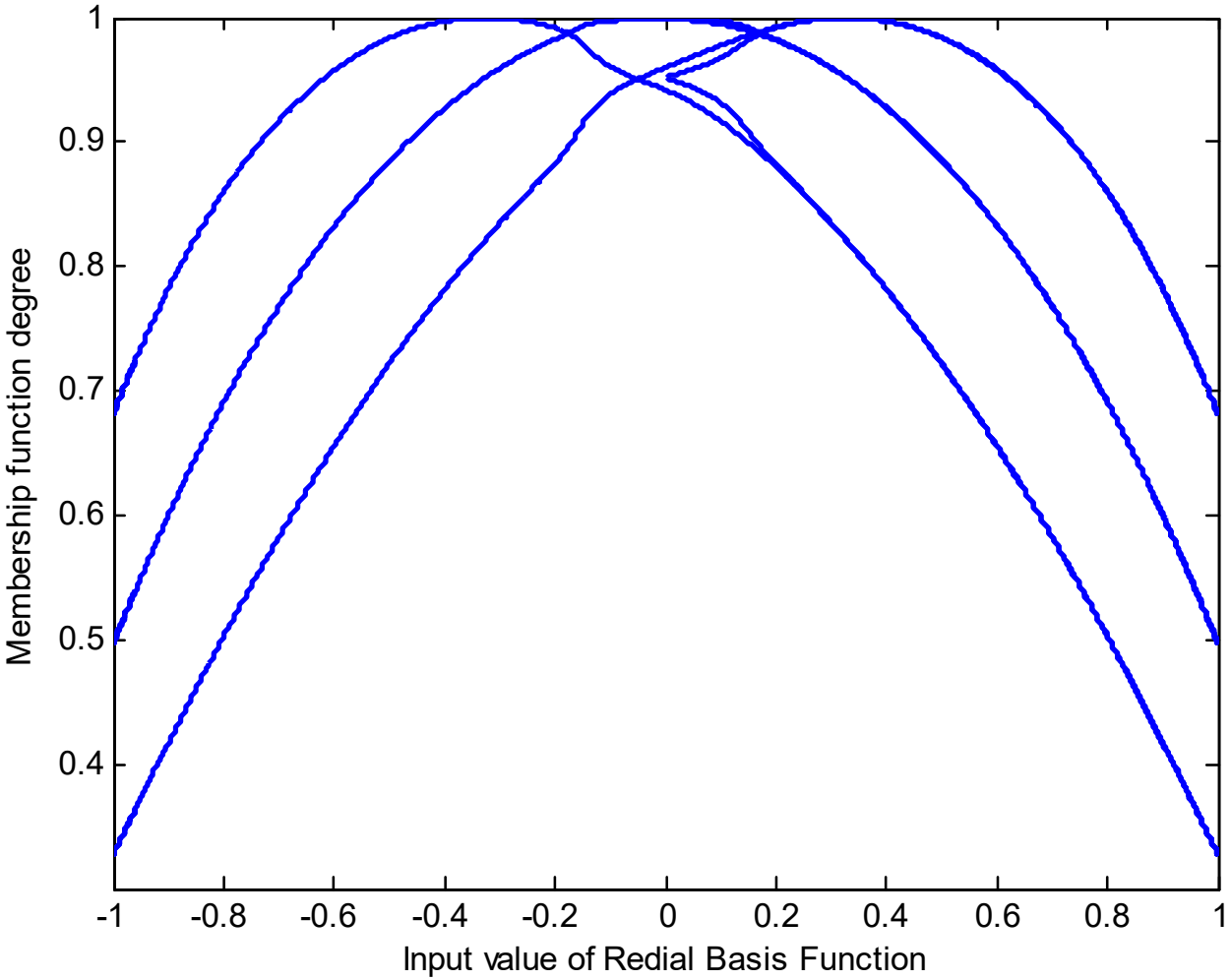


图7-29 3个隐含层神经元的高斯基函数(m=3)

7.3 RBF神经网络

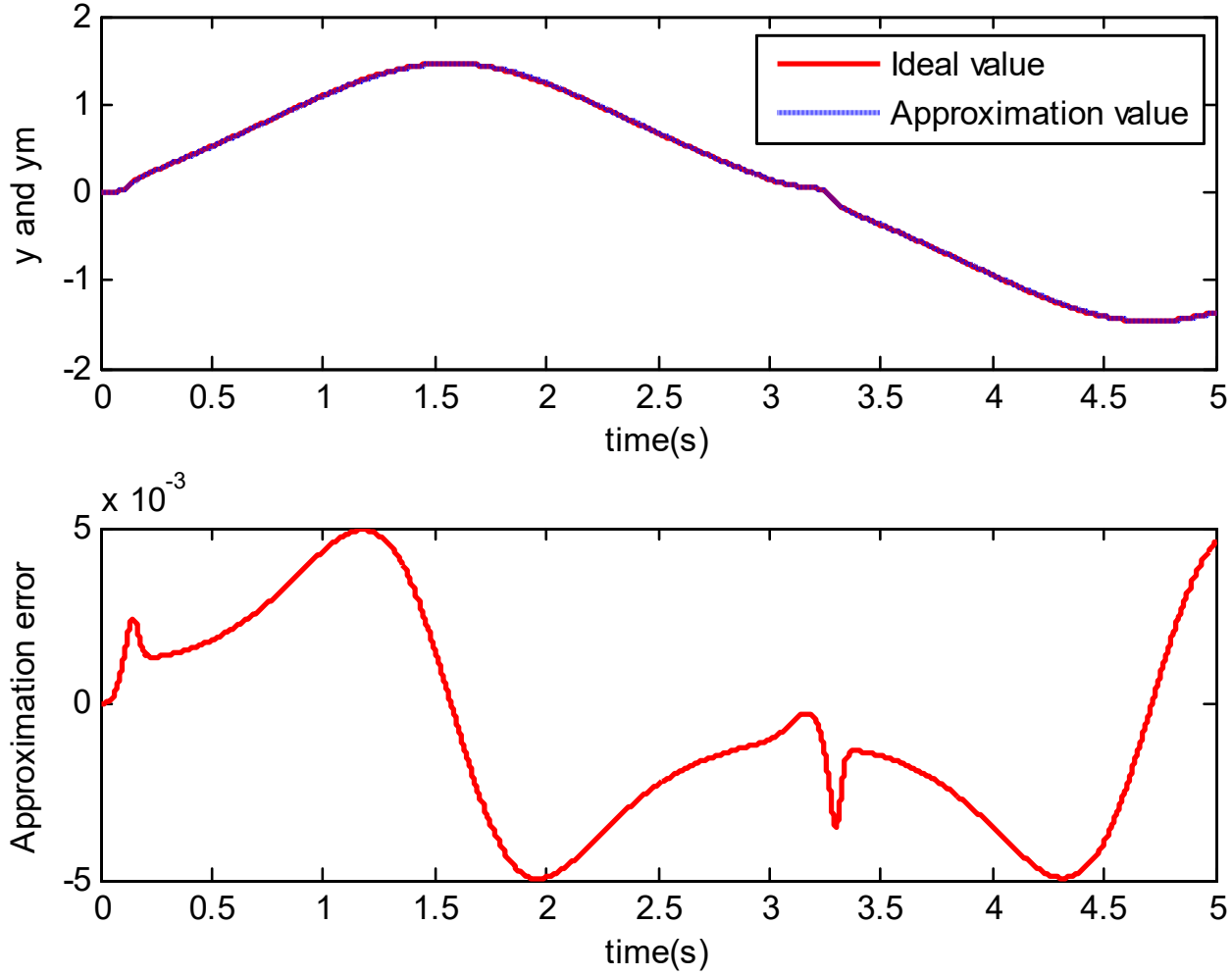


图7-30 含有3个隐含层神经元的逼近 (m=3)

7.3 RBF神经网络

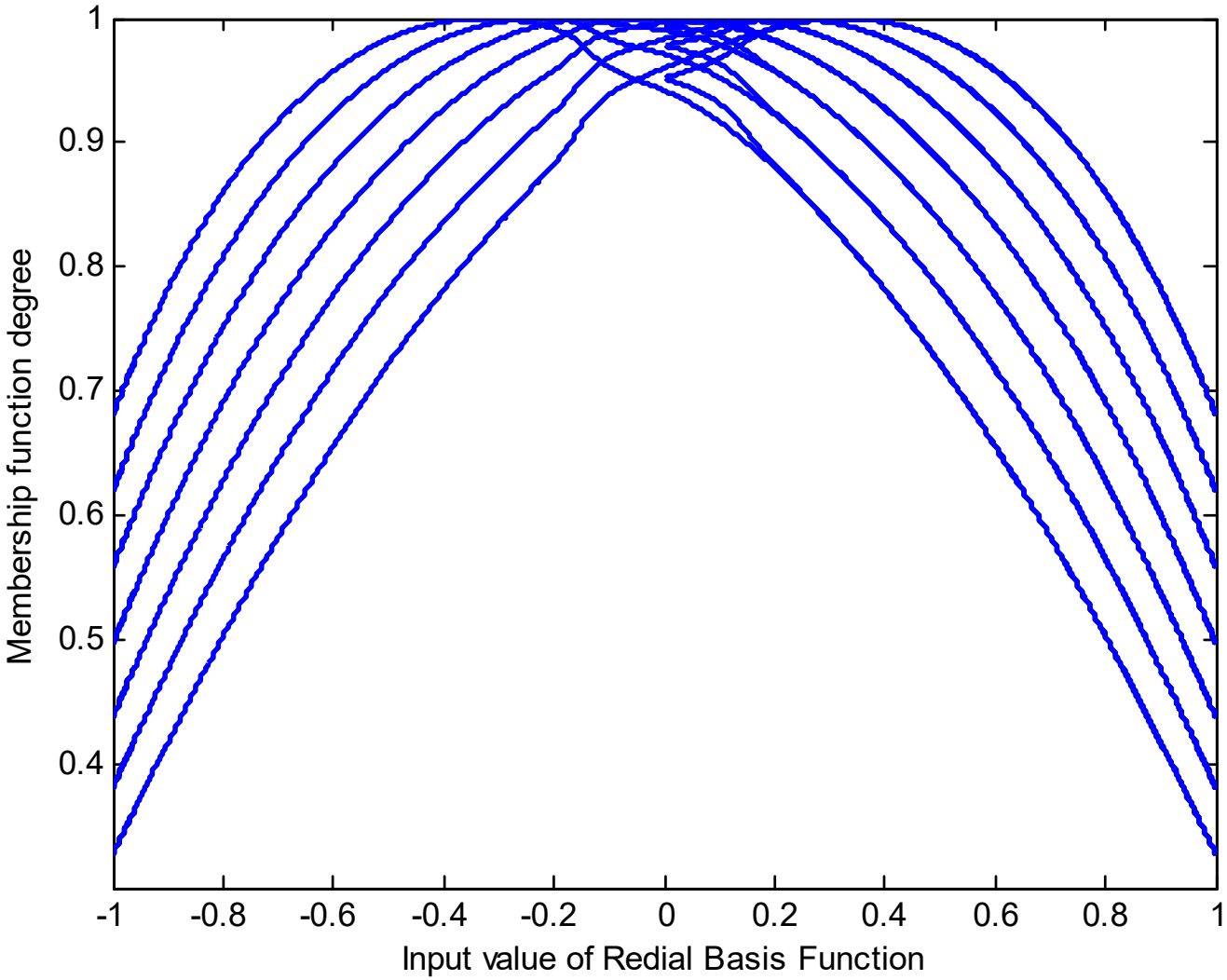


图7-31 7个隐含层神经元的高斯基函数(m=7)

7.3 RBF神经网络

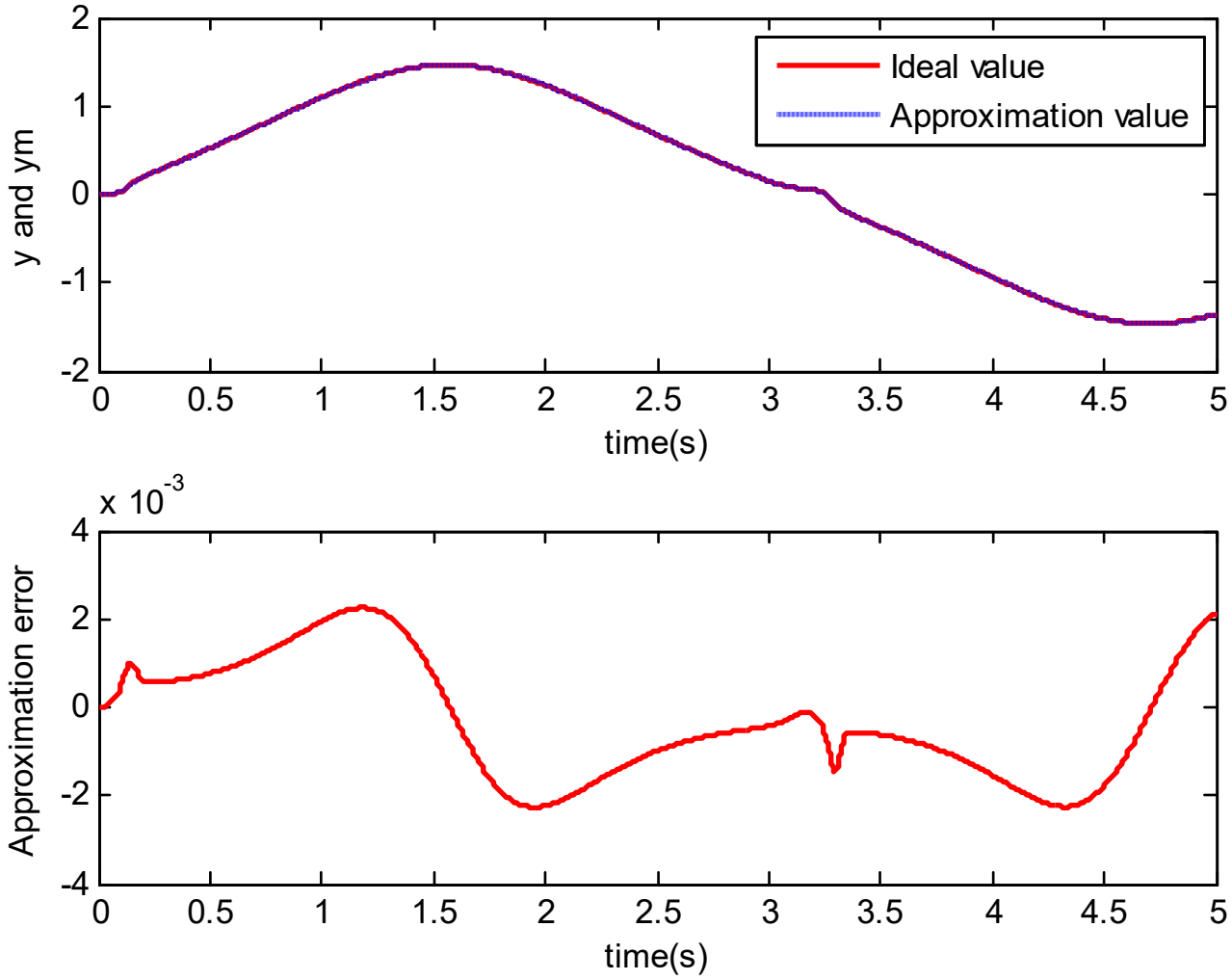


图7-32 含有7个隐含层神经元的逼近 (m=7)

7.3 RBF神经网络

仿真结果如图7-27至图7-32所示。由仿真结果可见，随着隐含层神经元节点数的增加，逼近误差下降。同时，随着隐含层神经元节点数的增加，为了防止梯度下降法的过度调整造成学习过程发散，应适当降低学习速率η。仿真程序为chap7_9.m。

7.3 RBF神经网络

（六）控制系统设计中RBF网络的逼近

RBF网络可对任意未知非线性函数进行任意精度的逼近。在控制系统设计中，采用RBF网络可实现对未知函数的逼近。

【例】为了估计未知函数$f(x)$，可采用如下RBF网络算法进行逼近

$$h_j = g\left(\left\|\boldsymbol{x}-\boldsymbol{c}_{ij}\right\|^2 / b_j^2\right) \tag{7.25}$$

$$f = \boldsymbol{W}^{*\mathrm{T}}\boldsymbol{h}(\boldsymbol{x}) + \varepsilon$$

其中x 为网络输入，i表示输入层节点，j为隐含层节点，$\boldsymbol{h}=[h_1,h_2,\cdots,h_n]^{\mathrm{T}}$ 为隐含层的输出，$\boldsymbol{W}^*$为理想权值，ε为网络的逼近误差，$\varepsilon \le \varepsilon_{\mathrm{N}}$

7.3 RBF神经网络

在控制系统设计中，可采样RBF网络对未知函数f进行逼近。一般可采用系统状态作为网络的输入，网络输出为

$$\hat{f}(\boldsymbol{x}) = \hat{\boldsymbol{W}}^{\mathrm{T}}\boldsymbol{h}(\boldsymbol{x}) \qquad (7.26)$$

其中$\hat{w}$为估计权值。

在实际的控制系统设计中，为了保证网络的输入值处于高斯基函数的有效范围，应根据网络的输入值实际范围确定高斯基函数中心点坐标向量c值，为了保证高斯基函数的有效映射，需要将高斯基函数的宽度bj 取适当的值。$\hat{w}$ 的调节是通过闭环的Lyapunov函数的稳定性分析中进行设计的。

思考与练习

第七章的全部课后习题

7月8日前交！
命名规则：姓名_学号_第七章作业
ic_sysu@163.com