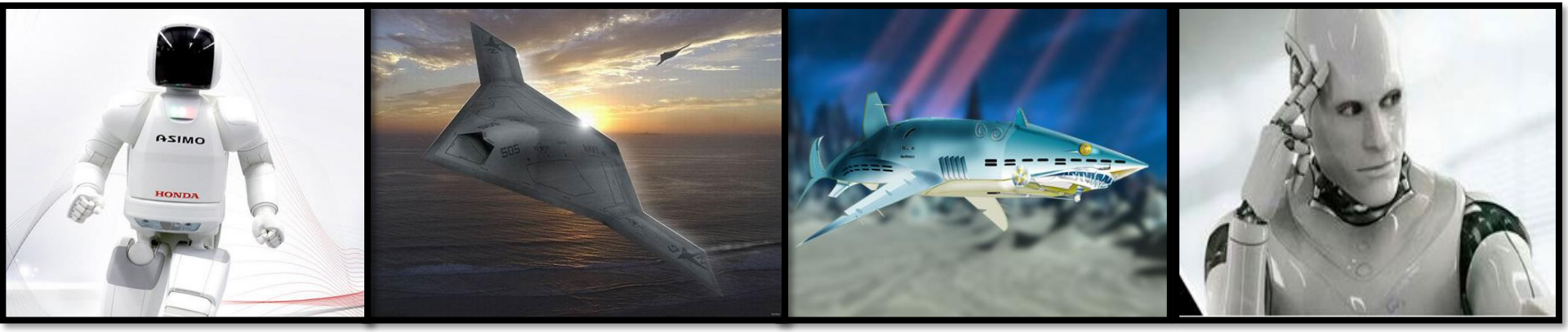
机器人智能控制

易建强 蒲志强 袁如意

中国科学院自动化研究所

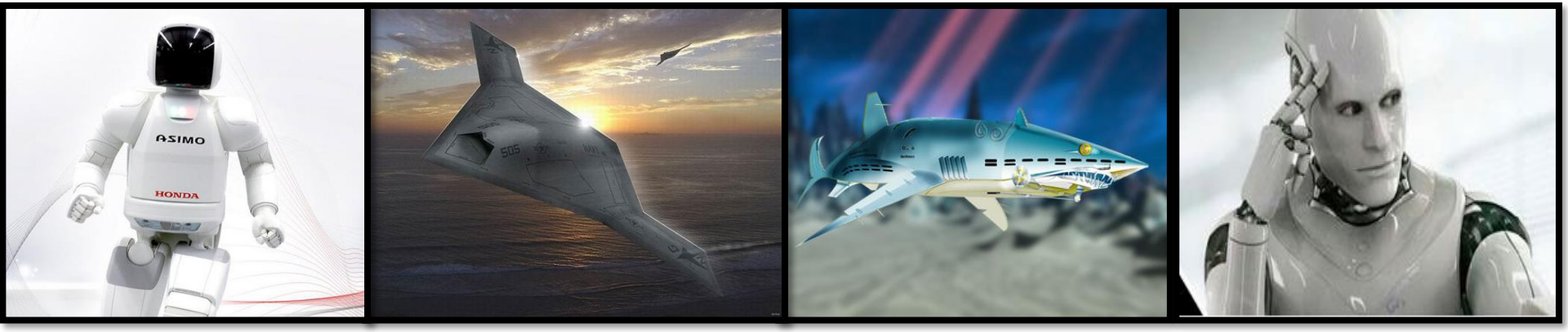
2020年秋季



神经网络控制

袁如意 高级工程师

ruyi.yuan@ia.ac.cn



本讲的主要内容

一、人工神经网络简介

二、神经网络控制概述

三、神经网络建模

四、神经网络控制

本讲的主要内容

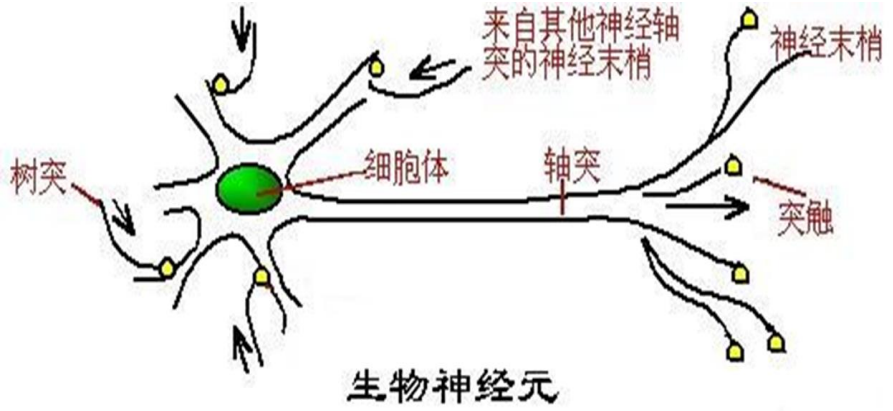
一、人工神经网络简介

二、神经网络控制概述

三、神经网络建模

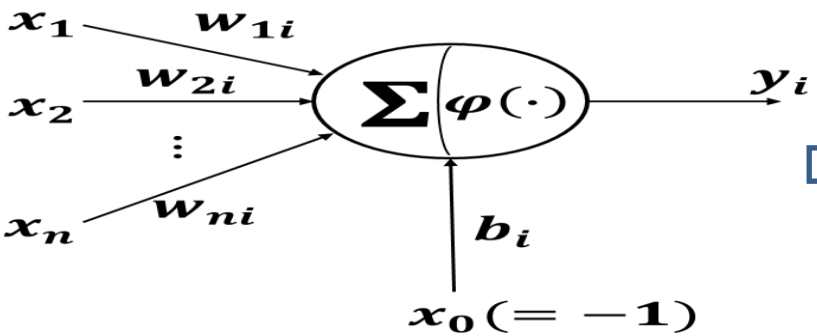
四、神经网络控制

人工神经网络

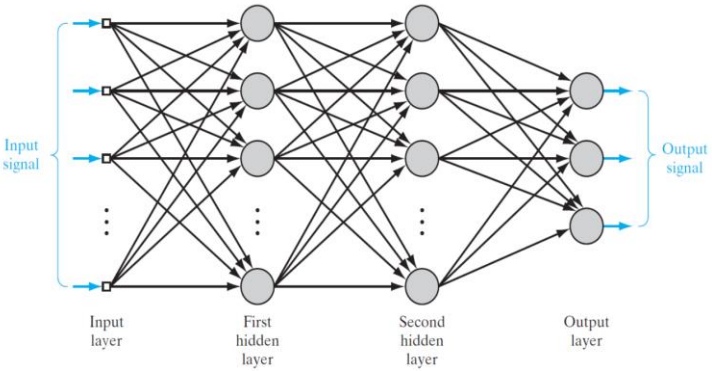


生物神经元

生物神经网络



人工神经元



人工神经网络

模拟：物理结构、计算模拟、存储操作、学习训练

人工神经元模型

人工神经元：对生物神经元的一种模拟与简化

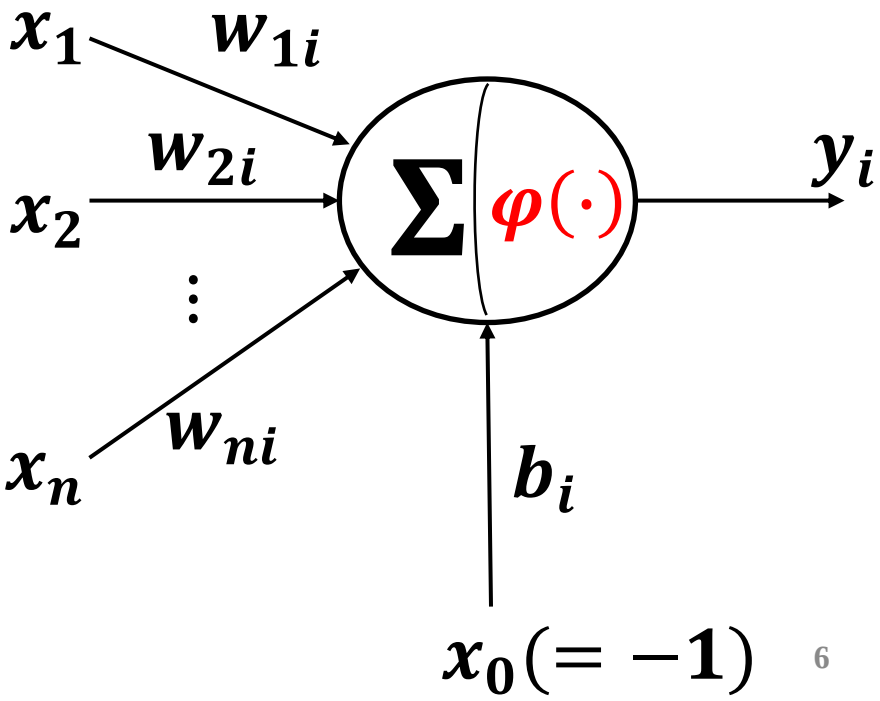
多输入单输出的非线性单元

$$I_i = \sum_{j=1}^{n} w_{ji} x_j - b_i$$

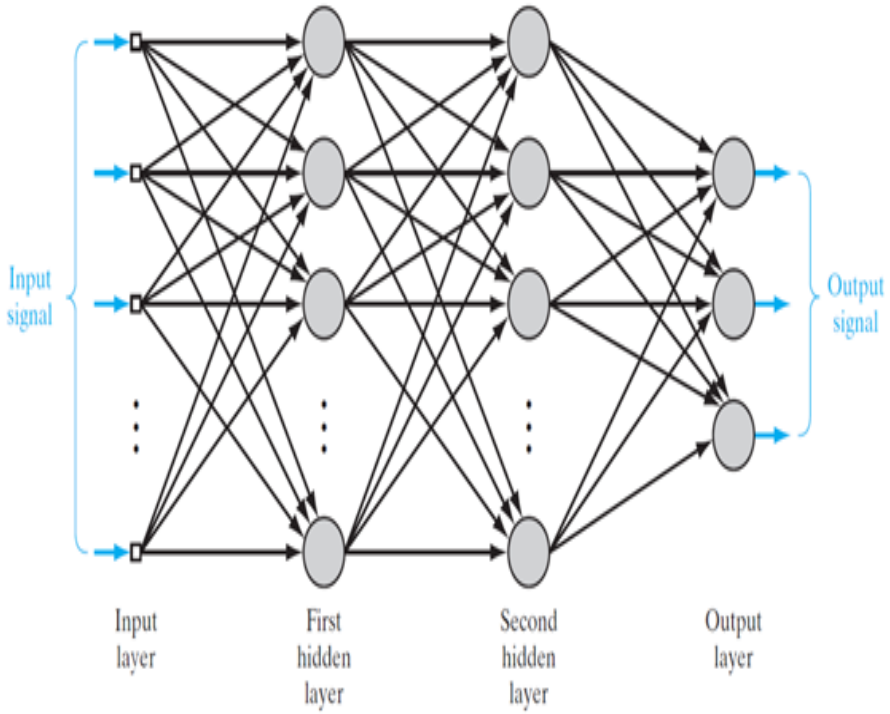
$$y_i = \varphi(I_i)$$

$$I_i = \sum_{j=0}^{n} w_{ji} x_j$$

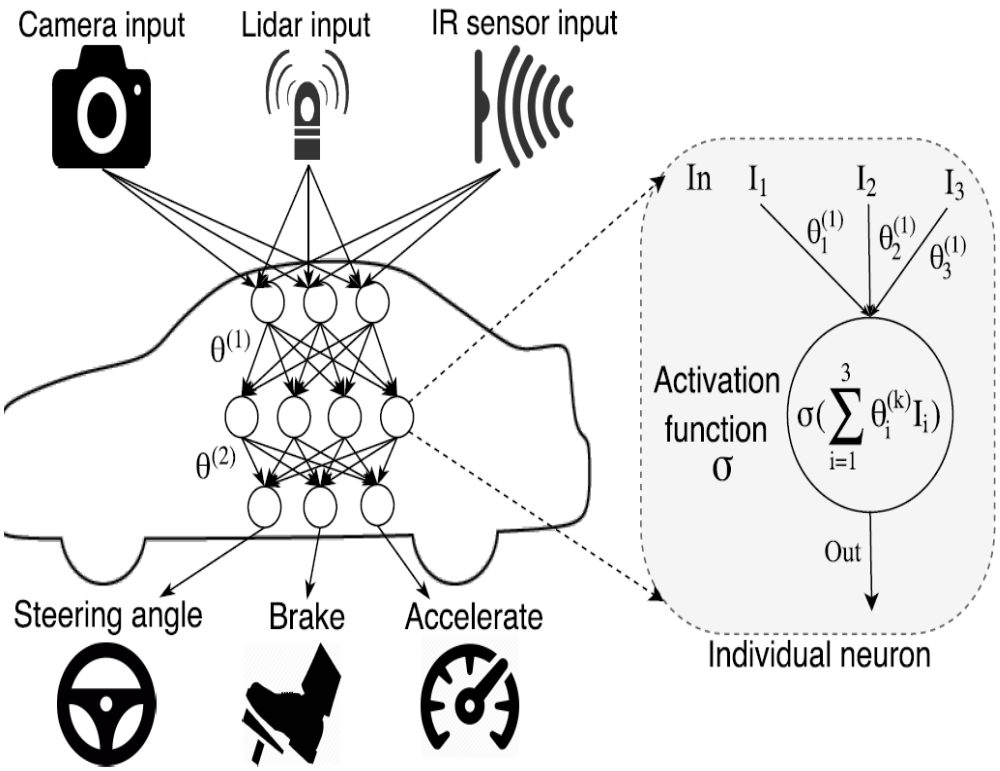
- ✓ 输入
- ✓ 连接权值
- ✓ 阈值（偏置）
- ✓ 激活函数



人工神经网络



人工神经网络



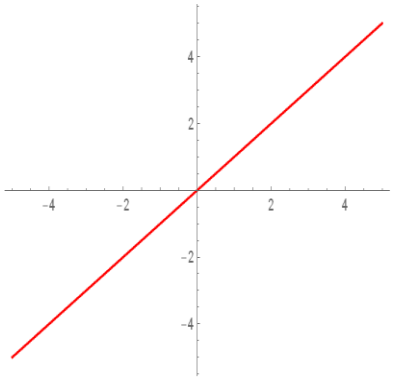
A simple autonomous car DNN that takes inputs from camera, light detection and ranging sensor (LiDAR), and IR (infrared) sensor, and outputs steering angle, braking decision, and acceleration decision

人工神经元模型

■ 激活函数模型

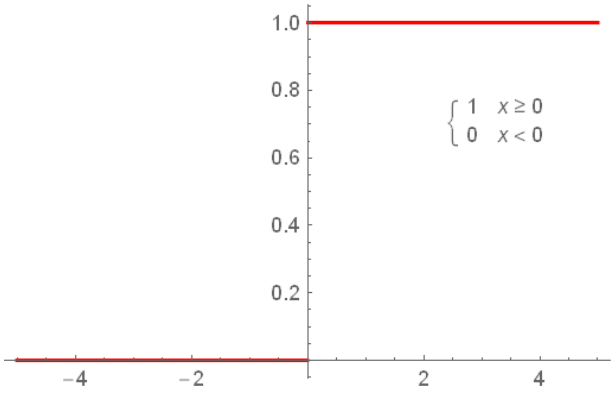
✓ 线性函数

$$\varphi(x) = x$$



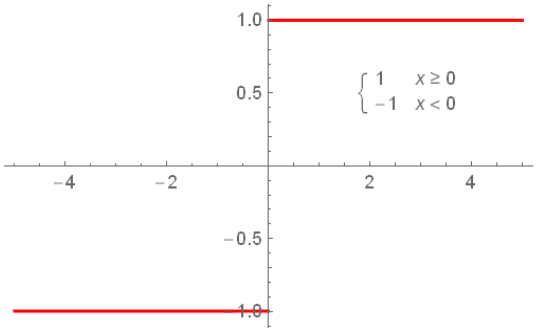
✓ 阶跃函数

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



✓ sgn符号函数

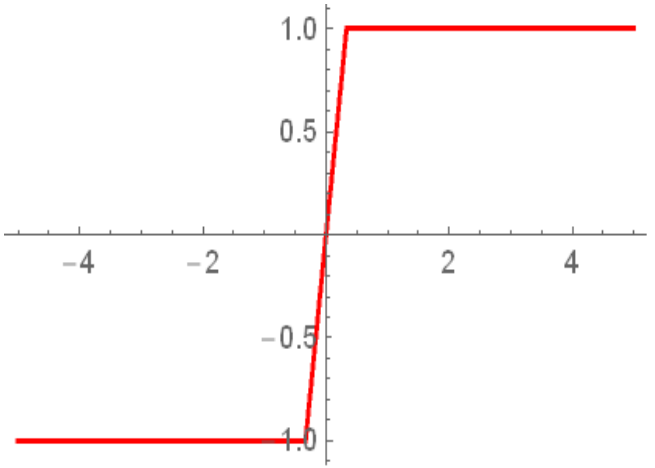
$$\mathrm{sgn}(x) = \varphi(x) = \begin{cases} -1, & x \geq 0 \\ 1, & x < 0 \end{cases}$$



人工神经元模型

✓ 饱和型函数（带限幅的放大器）

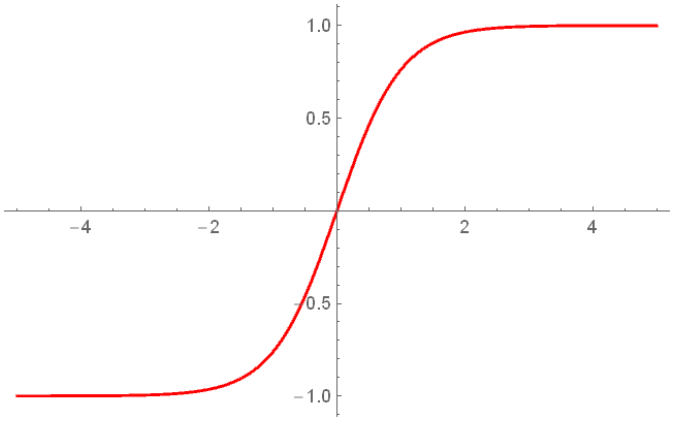
$$\varphi(x)=\begin{cases}1, & x\geq\frac{1}{k}\\ kx, & -\frac{1}{k}\leq x\leq\frac{1}{k}\\ -1, & x<-\frac{1}{k}\end{cases}$$



$$\begin{cases}1 & x>\frac{1}{3}\\ 3x & x\leq\frac{1}{3}\wedge x\geq-\frac{1}{3}\\ -1 & x<-\frac{1}{3}\end{cases}$$

✓ 双曲函数

$$\varphi(x)=\tanh(x)=\frac{1+e^{2x}}{1-e^{2x}}$$

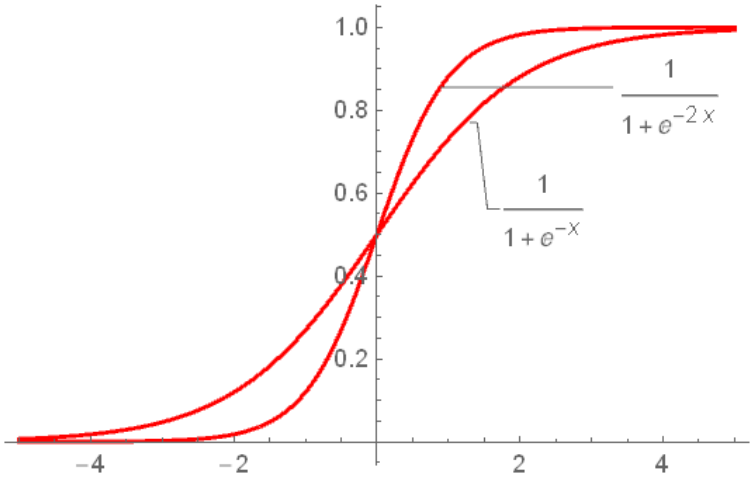


人工神经元模型

✓ S型函数

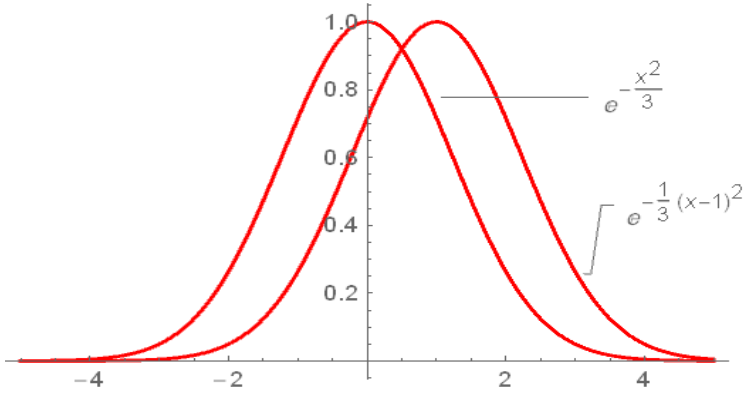
输入与状态之间的关系是在(0,1)内连续取值的单调可微函数，称为sigmod函数

$$\varphi(x)=\frac{1}{1+\exp(-\beta x)},\beta>0$$



✓ 高斯函数

$$\varphi(x)=e^{-(x-c)^2/\delta^2}$$

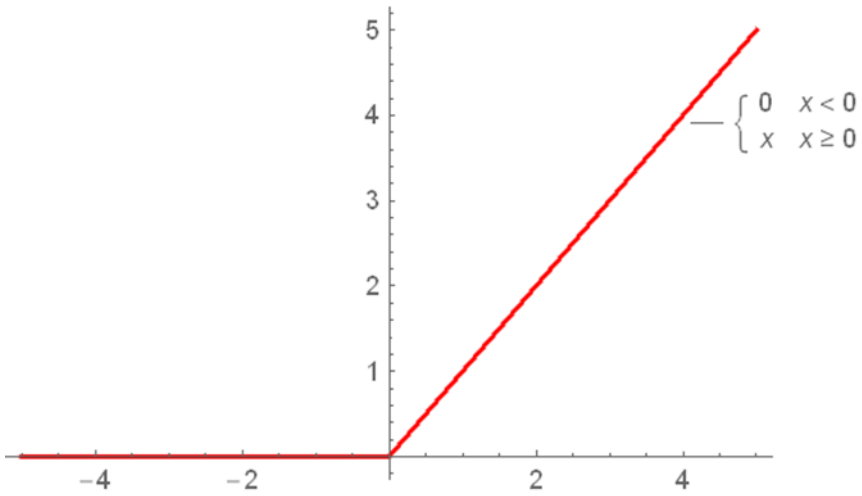


人工神经元模型

✓ ReLU函数（Rectified Linear Unit, ReLU）

线性整流函数，又称修正线性单元

- 更加有效率的梯度下降以及反向传播：避免了梯度爆炸和梯度消失问题，加快训练速度
- 计算简单



主要人工神经网络

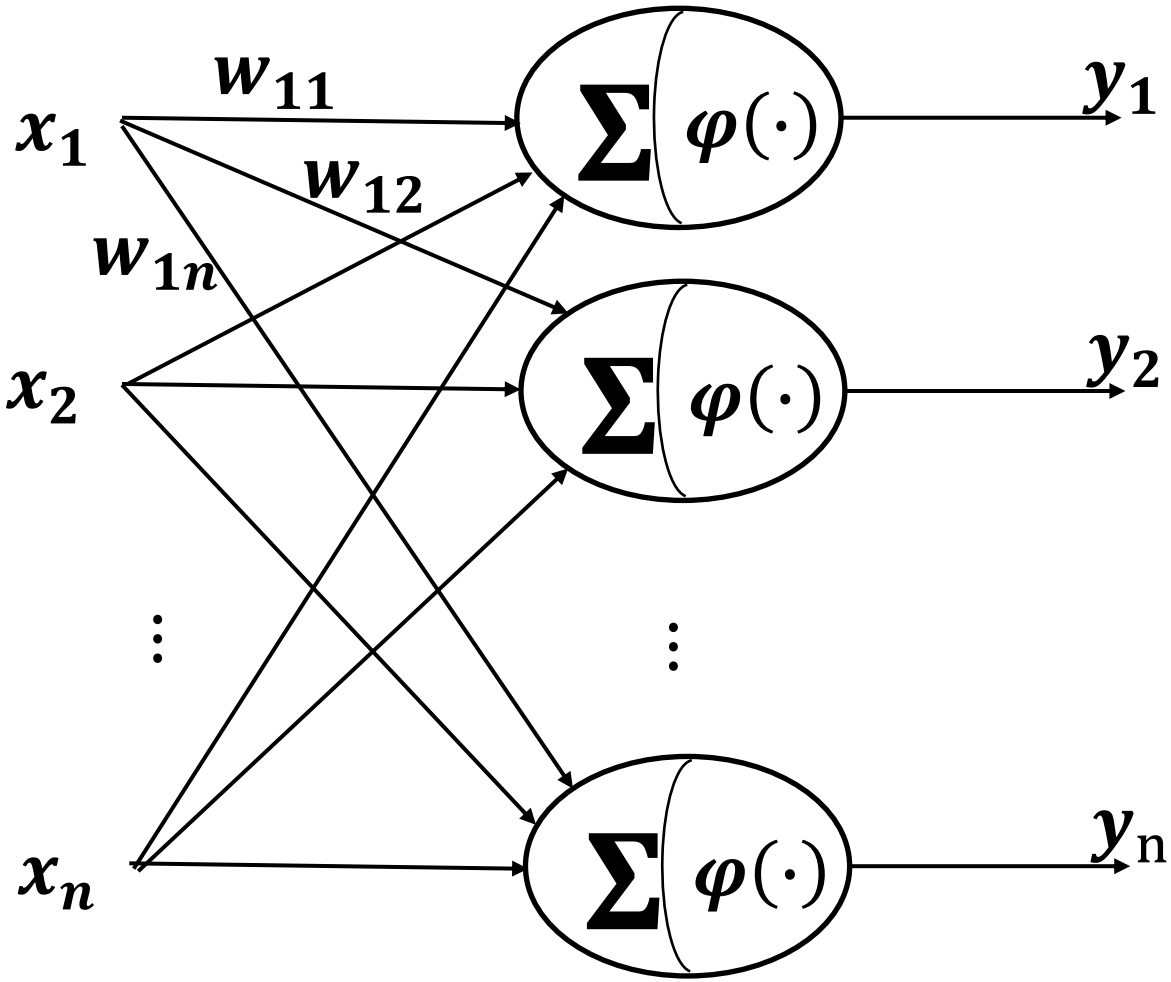
1)前向神经网络

- a) 感知器网络
- b) 多层感知器(反向传播网络)
- c) 径向基函数网络

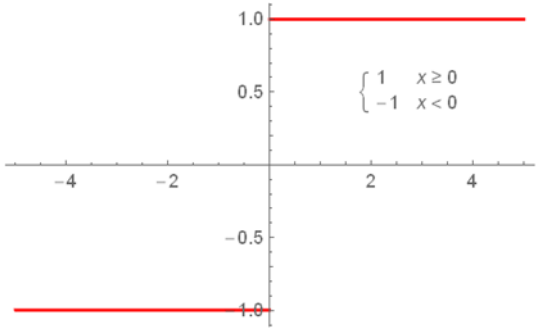
2)反馈神经网络

- a) 霍普菲尔德网络
- b) 波尔茨曼机
- c) 自组织映射网络
- d) 递归神经网络

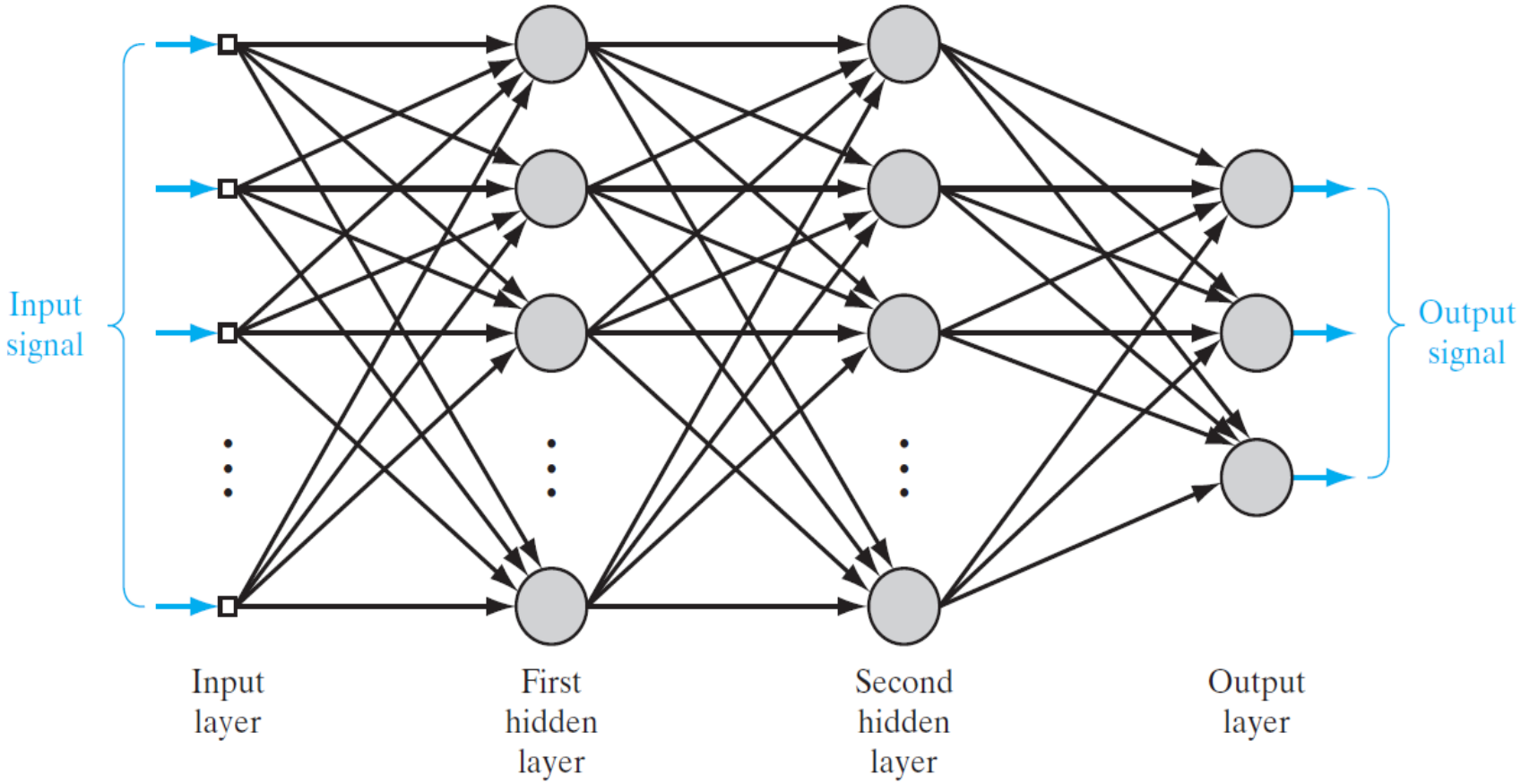
单层感知网络



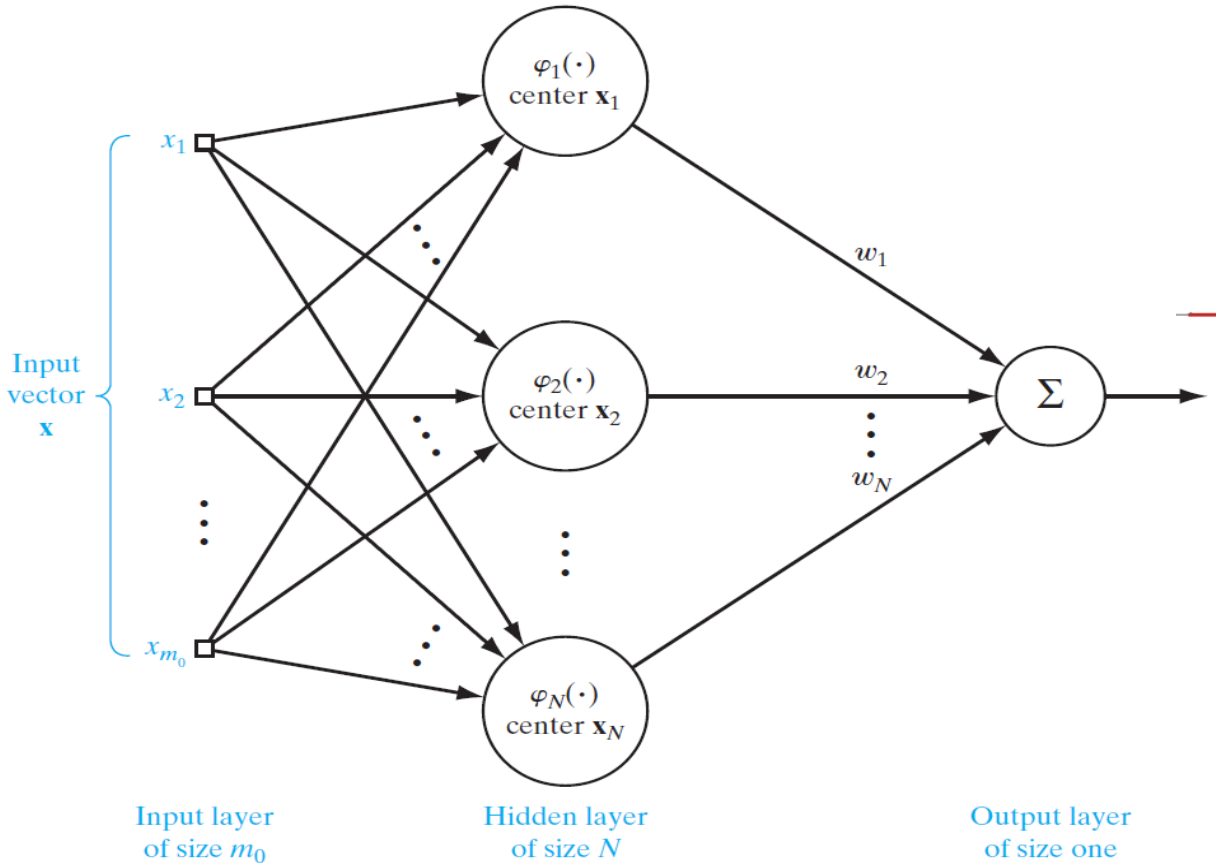
$$y_j = \varphi\left(\sum_{i=0}^{n} w_{ji}x_i\right) = \begin{cases} 1, \sum_{i=0}^{n} w_{ji}x_i \geq 0 \\ -1, \sum_{i=0}^{n} w_{ji}x_i < 0 \end{cases}$$

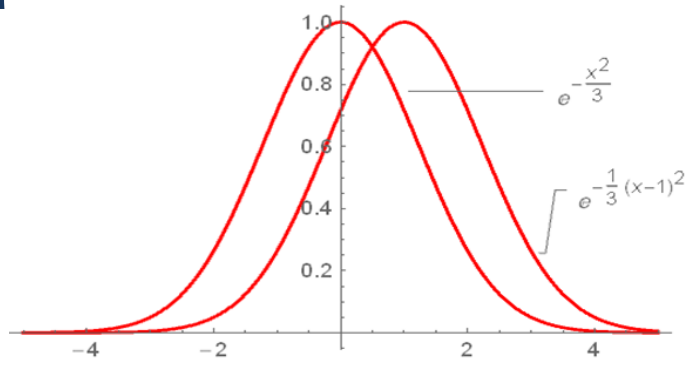


多层感知网络


Input signal
Output signal
Input layer
First hidden layer
Second hidden layer
Output layer

径向基网络

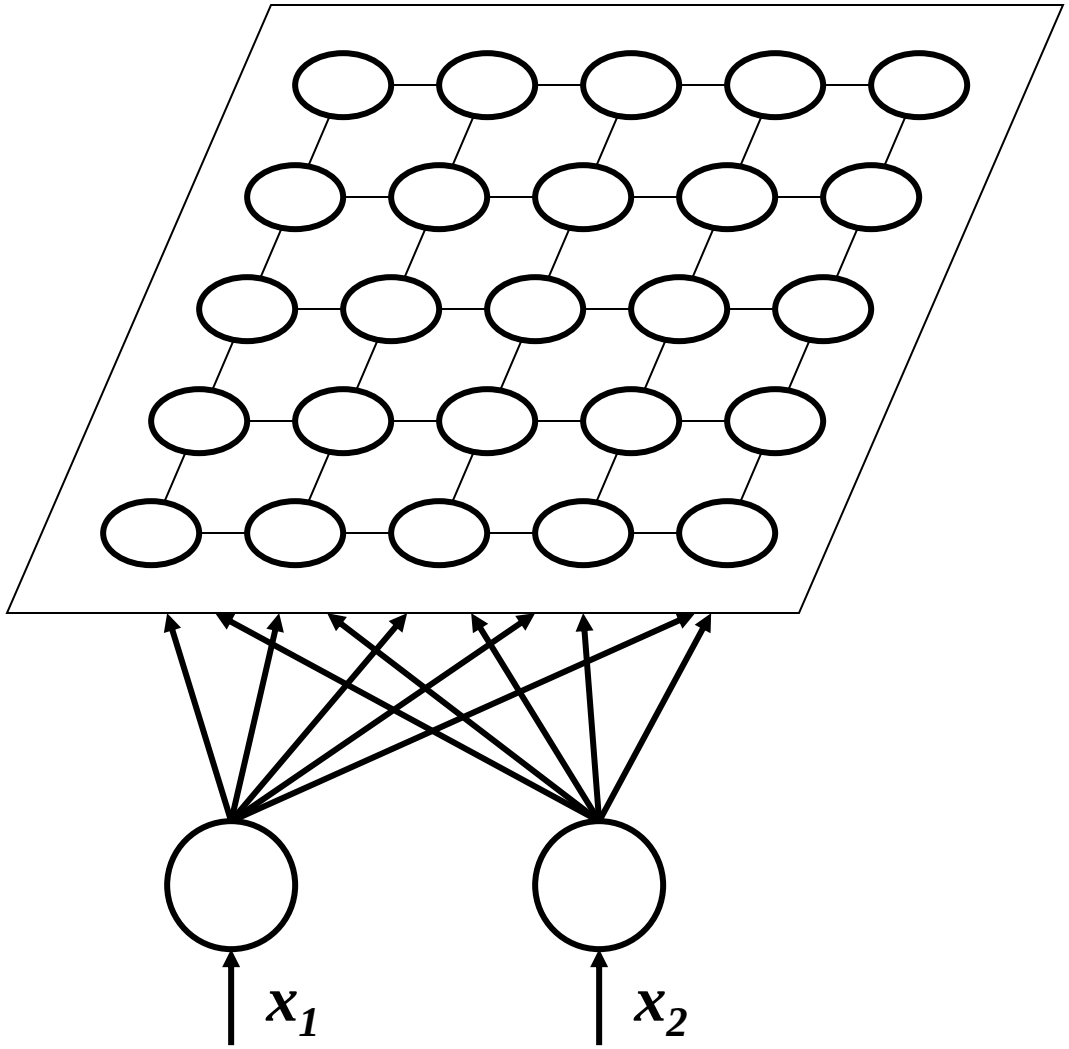




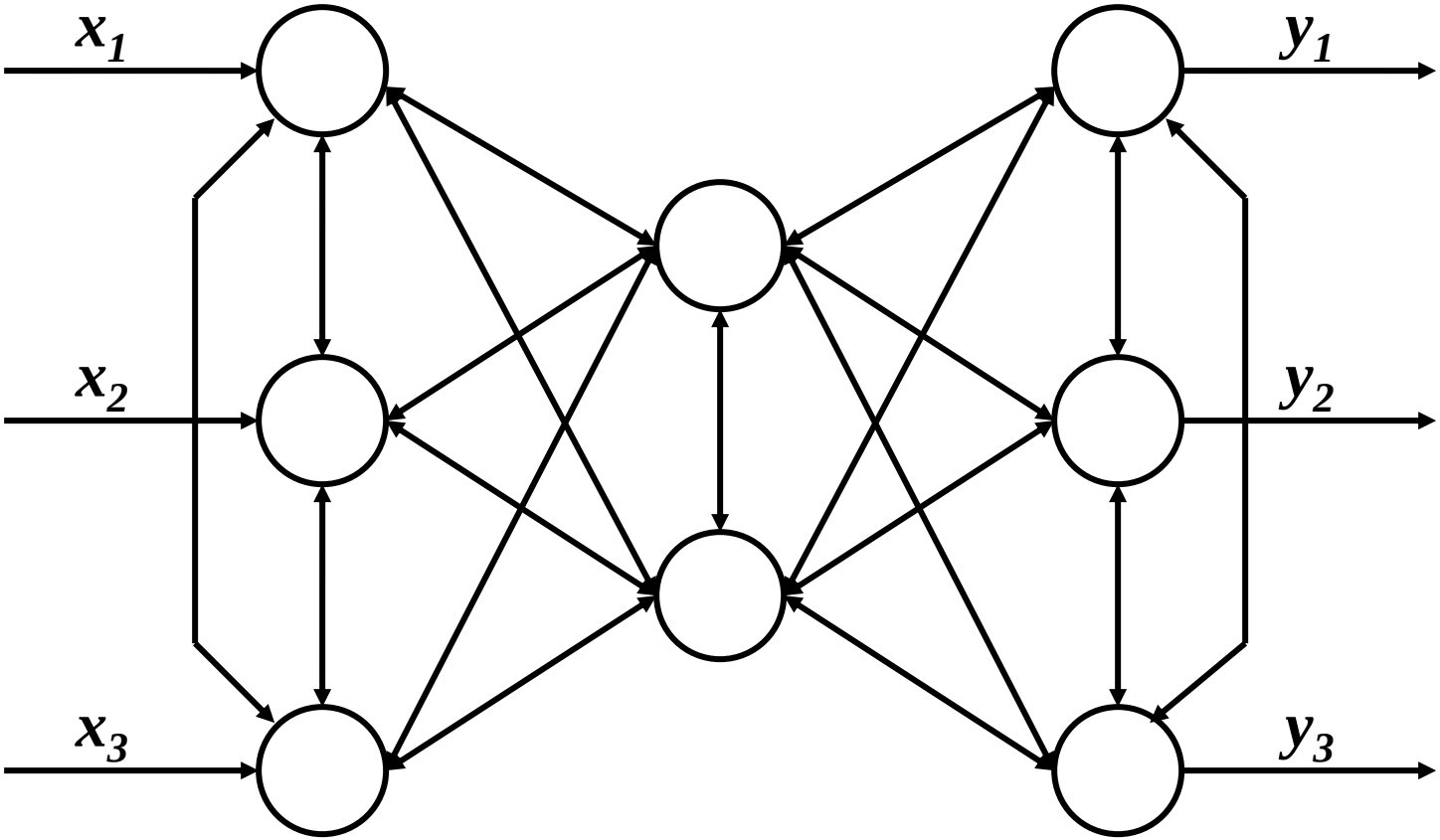
$$\varphi_j(\mathbf{x}) = \varphi(\mathbf{x} - \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}-\mathbf{x}_j\|^2}{2\sigma_j^2}}$$

$$y = \sum_{j=0}^{N} w_j \varphi_j$$

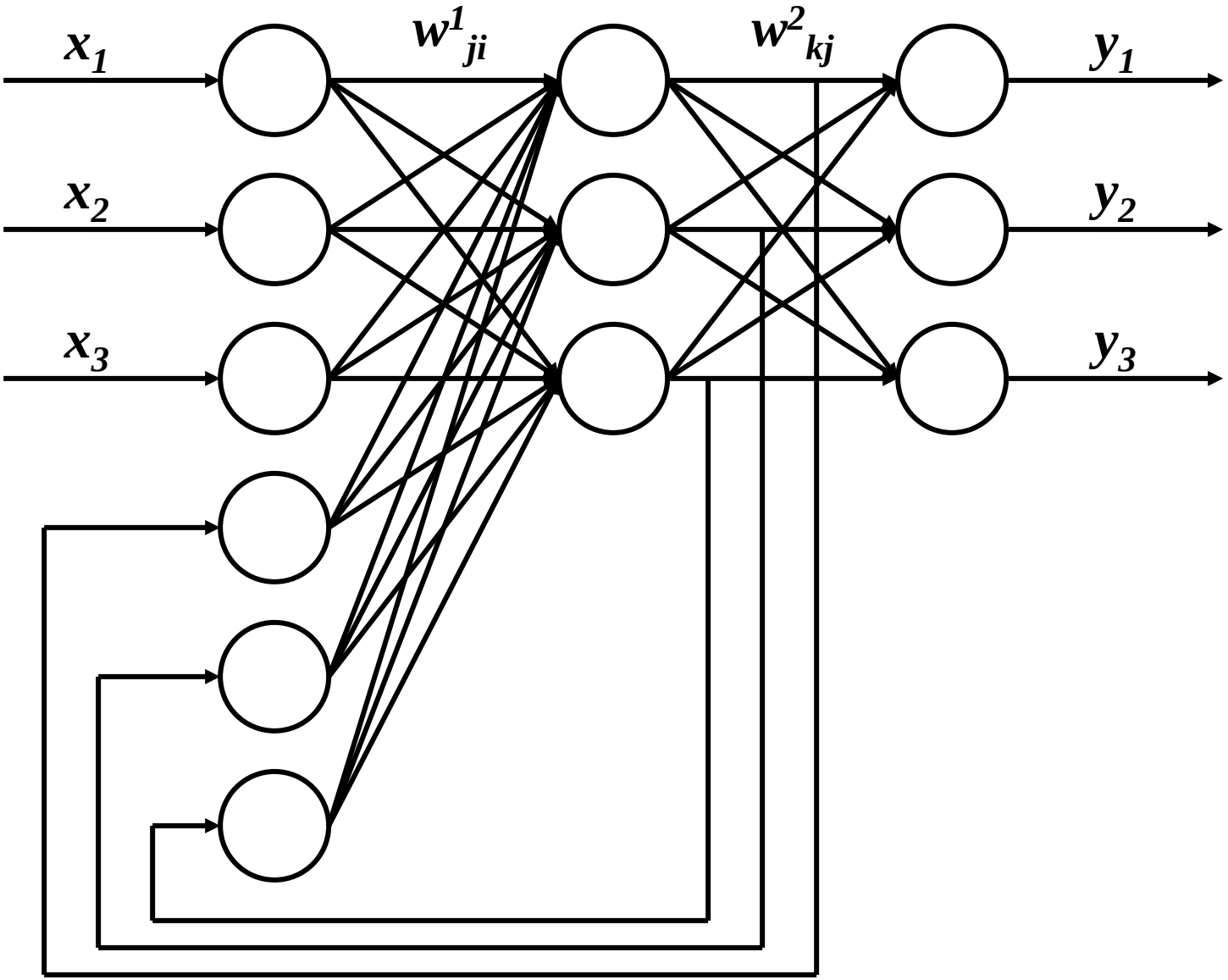
自组织映射网络



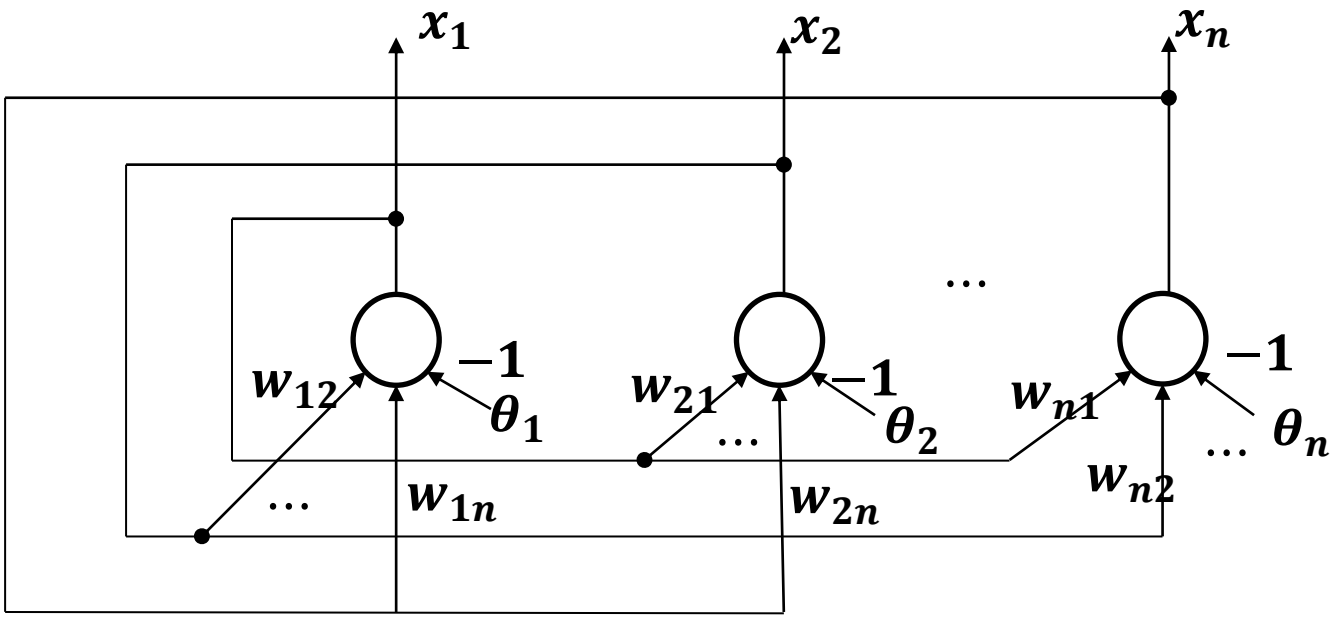
波尔茨曼机



递归神经网络



霍普菲尔德网络



$$\begin{cases} s_i = \sum_{j=1}^{n} w_{ij}\, x_j - \theta_i \\ x_i = \varphi(s_i) \end{cases} \qquad w_{ii} = 0$$

输入为网络初始状态 $x(0) = [x_1(0), x_2(0), \cdots, x_n(0)]^T$

输出为网络的稳定状态 $\lim_{k\to\infty} x(k)$

人工神经网络学习法则

学习的过程：

- 神经网络在外界输入样本的刺激下不断改变网络的连接权值，以使网络的输出不断地接近期望的输出

学习的本质：

- 对各连接权值的动态调整

学习方式

1）无监督学习算法

只有输入信息，没有期望的输出信息

2）有监督学习算法

同时提供输入信息和对应的期望输出信息

3）强化学习算法

没有明显的期望输出信息。通过与外界的交互获得反馈的评价信息（奖励或惩罚），学习如何执行恰当的动作。

人工神经网络学习法则

✓ Hebb学习法则

神经元连接强度的变化与两个相互连接的神经元的激活水平成正比。

两个神经元同时处于激活或抑制状态时，它们之间的连接强度将得到加强，反之应减弱

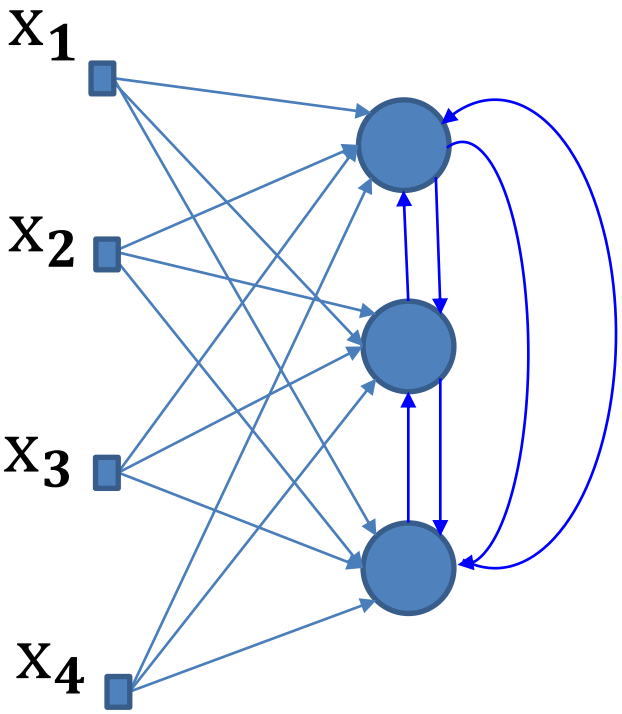
$$w_{ij}(k+1) = w_{ij}(k) + I_i I_j$$

人工神经网络学习法则

✓ 竞争学习法则

网络各输出单元相互竞争，最后达到只有一个最强者激活，最常见的一种情况是神经元之间有侧向抑制性连接。如输出单元中有一较强，则它将获胜并抑制其他单元，最优只有强者处于激活状态

$$\Delta w_{ji} = \begin{cases} \eta(x_i - w_{ji}), & \text{若}j\text{竞争获胜} \\ 0, & \text{若}j\text{竞争失败} \end{cases}$$



人工神经网络学习法则

✓ 误差学习（Delta学习规则）

$$e_k(n) = d_k(n) - y_k(n)$$

极小化目标函数 $J = E\left[\frac{1}{2}\sum_k e_k^2(n)\right]$

瞬时值 $\xi(n) = \frac{1}{2}\sum_k e_k^2(n)$

$$y_k = \boldsymbol{\varphi}(WX_k)$$

$$X_k = (x_{k0}, x_{k1}, \cdots, x_{kn})^T, k = 1,2, \cdots, m$$

$$W = (w_0, w_1, \cdots, w_n)$$

基本思想：沿着J的负梯度方向不断修正W的值，直到J达到最小 （梯度下降）

人工神经网络学习法则

求取梯度修正

$$\Delta W = \eta\left(-\frac{\partial J}{\partial W}\right)$$

$$\frac{\partial J}{\partial W} = \sum_{i=1}^{m} \frac{\partial J_k}{\partial w_i}$$

$$J_k = \frac{1}{2}(d_k - y_k)^2$$

$$\theta_k = WX_k$$

$$\frac{\partial J_k}{\partial w_i} = \frac{\partial J_k}{\partial \theta_k}\frac{\partial \theta_k}{\partial w_i} = \frac{\partial J_p}{\partial y_k}\frac{\partial y_k}{\partial \theta_k}x_{ki} = -(d_k - y_k)\boldsymbol{\varphi}'(\theta_k)x_{ki}$$

$$\Delta W = \eta\sum_{k=1}^{m}(d_k - y_k)\boldsymbol{\varphi}'(\theta_k)x_{ki}$$

人工神经网络学习法则

BP学习算法（多层网络）

第n次迭代学习输出端第j个神经元输出为$y_j(n)$

定义误差信号$e_j(n) = d_j(n) - y_j(n)$

输出端总平方误差

$$\xi(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n)$$

样本总数为N

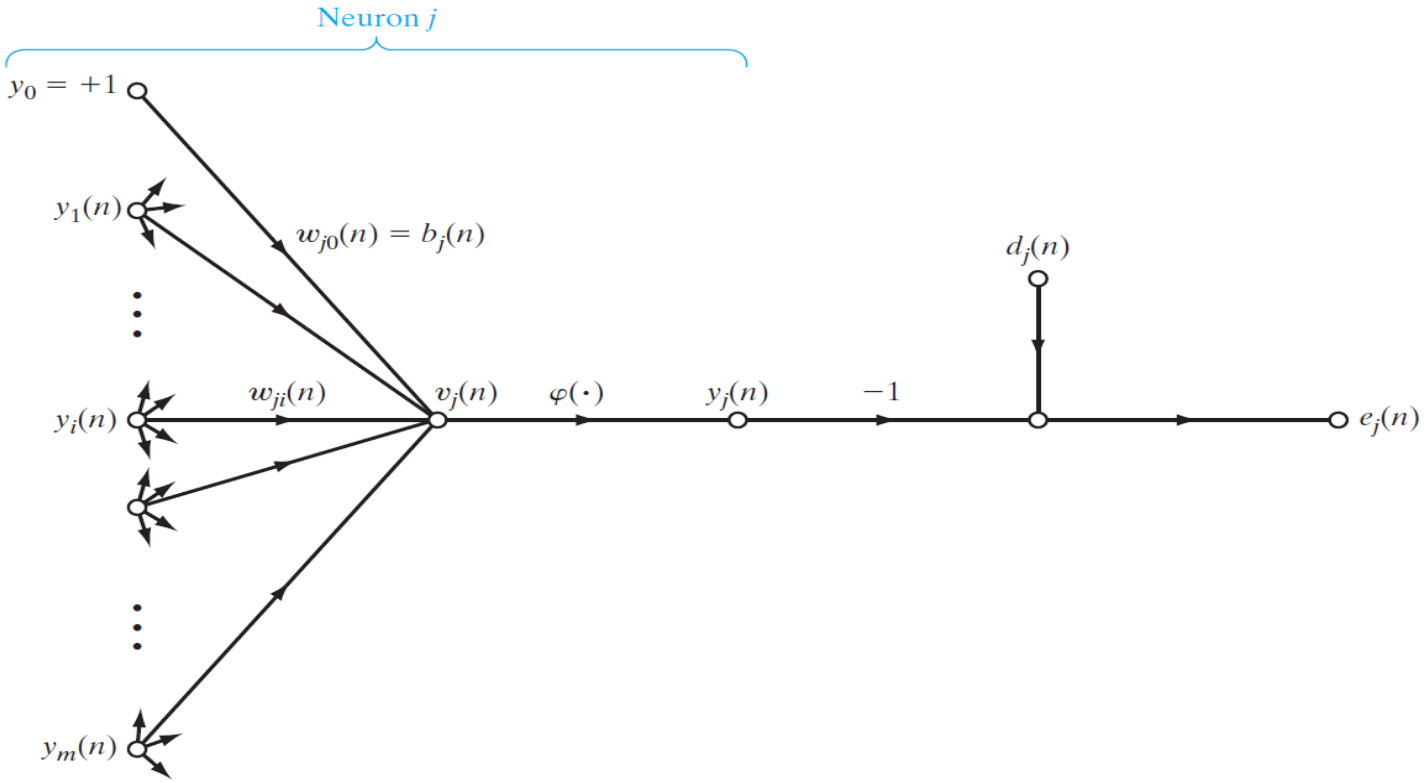
平方误差均值为

$$\xi_{AV} = \frac{1}{N}\sum_{n=1}^{N}\xi(n) = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C} e_j^2(n)$$

人工神经网络学习法则

BP学习算法（多层网络）

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n) \qquad y_j(n) = \varphi_j\big(v_j(n)\big)$$



人工神经网络学习法则

j是输出节点

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \xi(n)}{\partial e_j(n)} = e_j(n) \qquad \frac{\partial e_j(n)}{\partial y_j(n)} = -1 \qquad \frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'\left(v_j(n)\right) \qquad \frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'\left(v_j(n)\right) y_i(n)$$

$$\Delta w_{ij}(n) = -\eta \frac{\partial \xi(n)}{\partial w_{ji}(n)} = -\eta \delta_j(n) y_i(n)$$

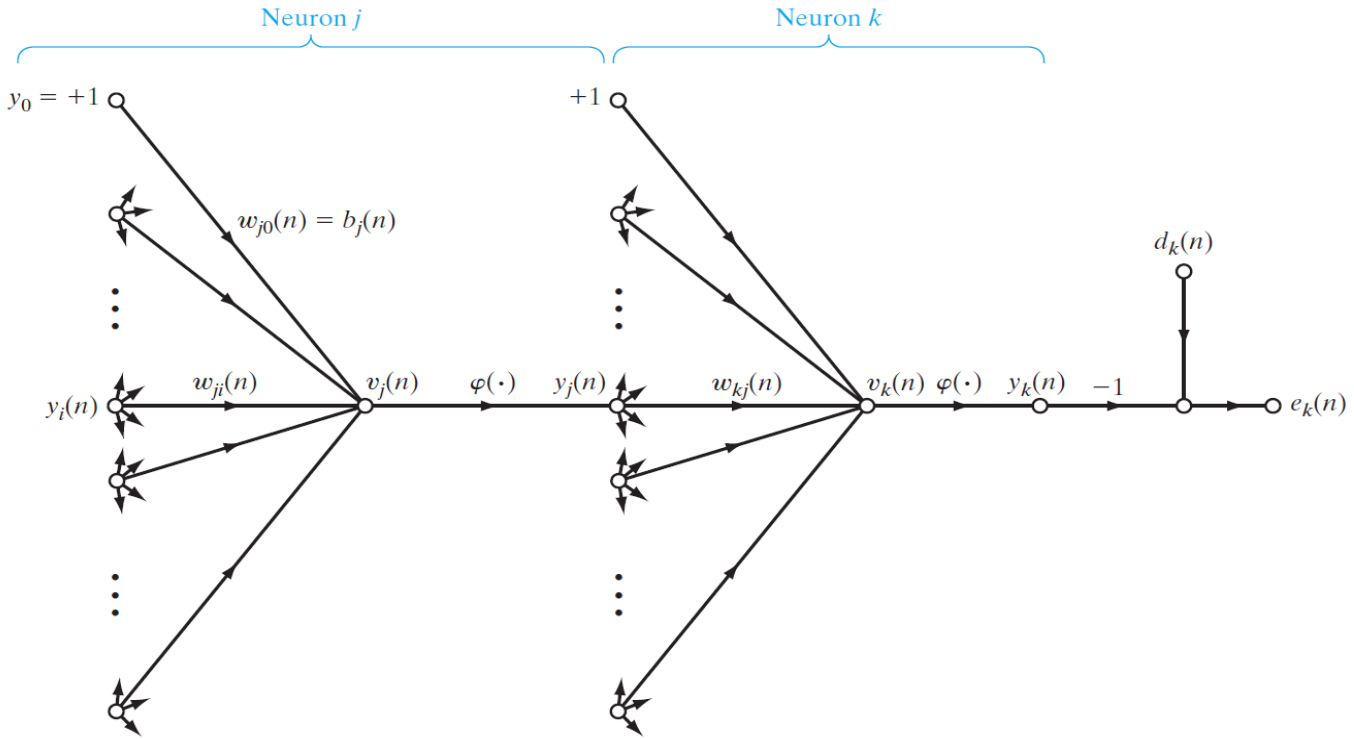
$$\delta_j(n) = -\frac{\partial \xi(n)}{\partial v_j(n)} = -\frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'\left(v_j(n)\right) e_j(n)$$

$\delta_j(n)$局域梯度：权值所需的变化

人工神经网络学习法则

j是隐藏层节点， k是输出节点

$$\delta_j(n) = -\frac{\partial \xi(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial \xi(n)}{\partial y_j(n)} \varphi_j'\left(v_j(n)\right)$$



人工神经网络学习法则

j是隐藏层节点, k是输出节点

$$\xi(n) = \frac{1}{2}\sum_{k\in C} e_k^2(n)$$

$$\frac{\partial \xi(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)}$$

$e_k(n) = d_k(n) - y_k(n) = d_k(n) - \varphi_k\big(v_k(n)\big)$ $\quad \frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi_k'\big(v_k(n)\big)$

$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n)$ $\quad \frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n)$

$$\frac{\partial \xi(n)}{\partial y_j(n)} = -\sum_k e_k(n)\varphi_k'\big(v_k(n)\big) w_{kj}(n) = -\sum_k \delta_k(n)\, w_{kj}(n)$$

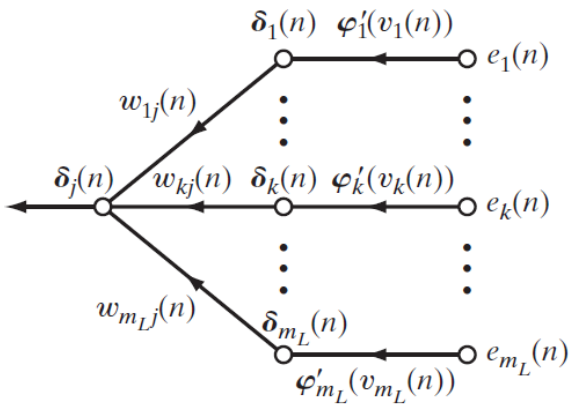
$$\delta_j(n) = \varphi_j'\left(v_j(n)\right)\sum_k \delta_k(n)\, w_{kj}(n)$$

人工神经网络学习法则

$$\begin{pmatrix} \text{校正} \\ \text{权值} \\ \Delta w_{ji}(n) \end{pmatrix} = \begin{pmatrix} \text{学习率} \\ \text{参数} \\ \eta \end{pmatrix} \cdot \begin{pmatrix} \text{局部} \\ \text{梯度} \\ \delta_j(n) \end{pmatrix} \cdot \begin{pmatrix} \text{神经元} j \\ \text{输入信号} \\ y_i(n) \end{pmatrix}$$

$$\delta_j(n) = \varphi_j'\left(v_j(n)\right) e_j(n)$$

$$\delta_j(n) = \varphi_j'\left(v_j(n)\right) \sum_k \delta_k(n)\, w_{kj}(n)$$



人工神经网络学习法则

反向传播算法步骤归纳

1、初始化，选择合适的网络结构，置所有可调参数（权值和阈值）为均匀分布的较小的值

2、对每个样本做如下计算：

①前向计算：对第l层第j单元

$$v_j^{(l)}(n)=\sum_{i=0}^{p} w_{ji}^{(l)}(n)\, y_i^{(l-1)}(n)$$

如单元j的激活函数为Sigmoid函数，则

$$y_j^{(l)}(n)=\frac{1}{1+e^{-v_j^{(l)}(n)}} \qquad \varphi_j'\left(v_j(n)\right)=\frac{\partial y_j^{(l)}(n)}{\partial v_j(n)}=y_j^{(l)}(n)\left(1-y_j^{(l)}(n)\right)$$

人工神经网络学习法则

若神经元j属于第一隐层$(l=1)$，则有

$$y_j^{(0)}(n)=x(n)$$

若神经元j属于输出层$(l=L)$

$y_j^{(L)}(n)=o_j(n)$ 且 $e_j(n)=d_j(n)-o_j(n)$

② 反向计算δ:

$$\delta_j^{(l)}(n)=\begin{cases} e_j^{(L)}\varphi_j'\left(v_j^{(L)}(n)\right), & \text{对输出层L的神经元}j \\ \varphi_j'\left(v_j^{(l)}(n)\right)\sum_k \delta_k^{(l+1)}(n)\,w_{kj}^{(l+1)}(n), & \text{对隐藏层L的神经元}j \end{cases}$$

③按照下式修正权值

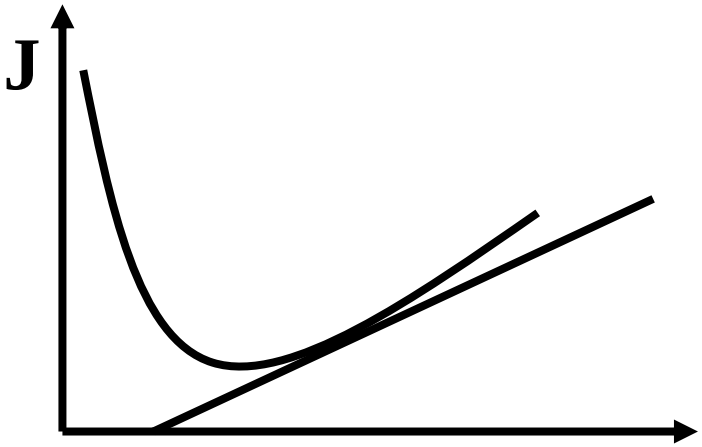
$$w_{ij}^{(l)}(n+1)=w_{ij}^{(l)}(n)+\eta\delta_j^{(l)}(n)y_j^{(l-1)}(n)$$

3、$n=n+1$，输入新样本进行训练

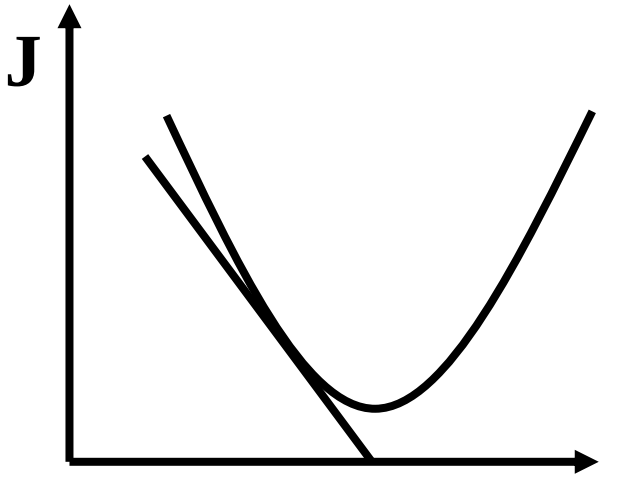
直到ξ_{AV}达到预定要求

人工神经网络学习法则

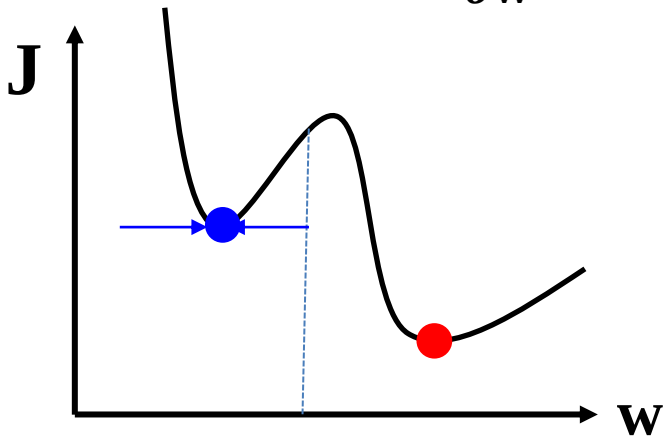
梯度下降直观解释



$\frac{\partial J}{\partial w}>0$，此时$\Delta w<0$



$\frac{\partial J}{\partial w}<0$, 此时$\Delta w>0$



人工神经网络学习法则

BP网络缺点：

- ✓ 收敛速度慢
- ✓ 目标函数存在局部极小值
- ✓ 难以确定隐层和隐层节点的数量

改进BP网络收敛速度：加入动量项

$$\Delta w_{ji}(n) = \alpha w_{ji}(n-1) + \eta \delta_j(n) y_j(n) \quad 0 < \alpha < 1$$

$$\Delta w_{ji}(n) = \eta \sum_{t=0}^{n} \alpha^{n-1} \delta_j(t) y_i(t)$$

$$= -\eta \sum_{t=0}^{n} \alpha^{n-1} \frac{\partial \xi(t)}{\partial w_{ji}(t)}$$

通用逼近定理(*universal approximation theorem*)

Let $\varphi(\cdot)$ be a nonconstant, bounded, and monotone-increasing continuous function. Let I_{m_0} denote the m_0-dimensional unit hypercube $[0, 1]^{m_0}$. The space of continuous functions on I_{m_0} is denoted by $C(I_{m_0})$. Then, given any function $f \ni C(I_{m_0})$ and $\varepsilon > 0$, there exist an integer m_1 and sets of real constants α_i, b_i, and w_{ij}, where $i = 1, \ldots, m_1$ and $j = 1, \ldots, m_0$ such that we may define

$$F(x_1, \ldots, x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \varphi\left(\sum_{j=1}^{m_0} w_{ij} x_j + b_i \right) \tag{4.88}$$

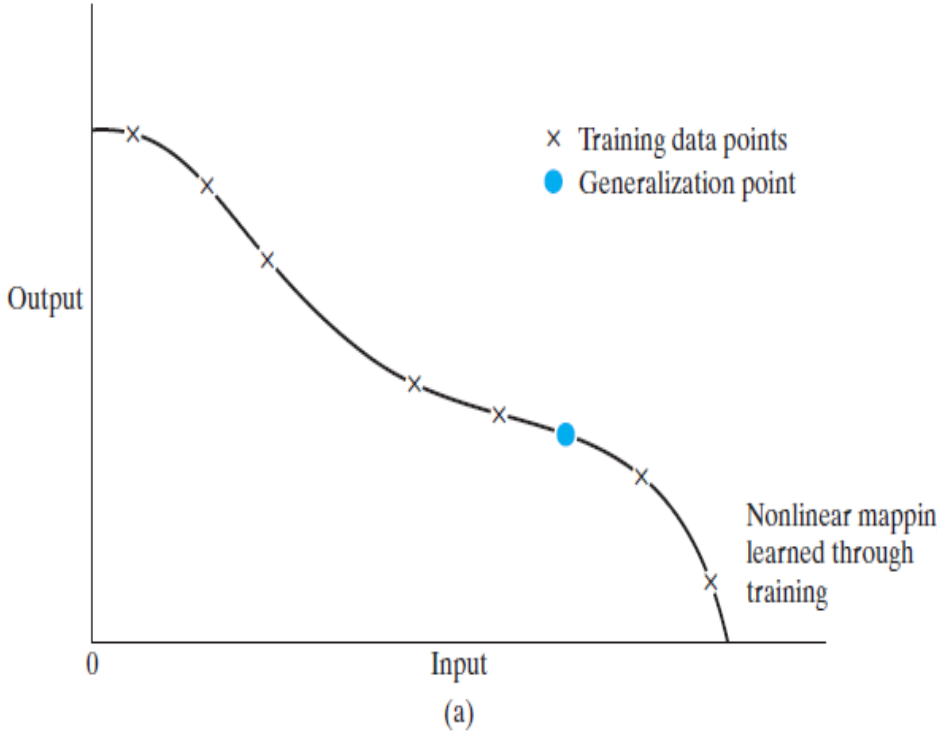
as an approximate realization of the function $f(\cdot)$; that is,

$$|F(x_1, \ldots, x_{m_0}) - f(x_1, \ldots, x_{m_0})| < \varepsilon$$

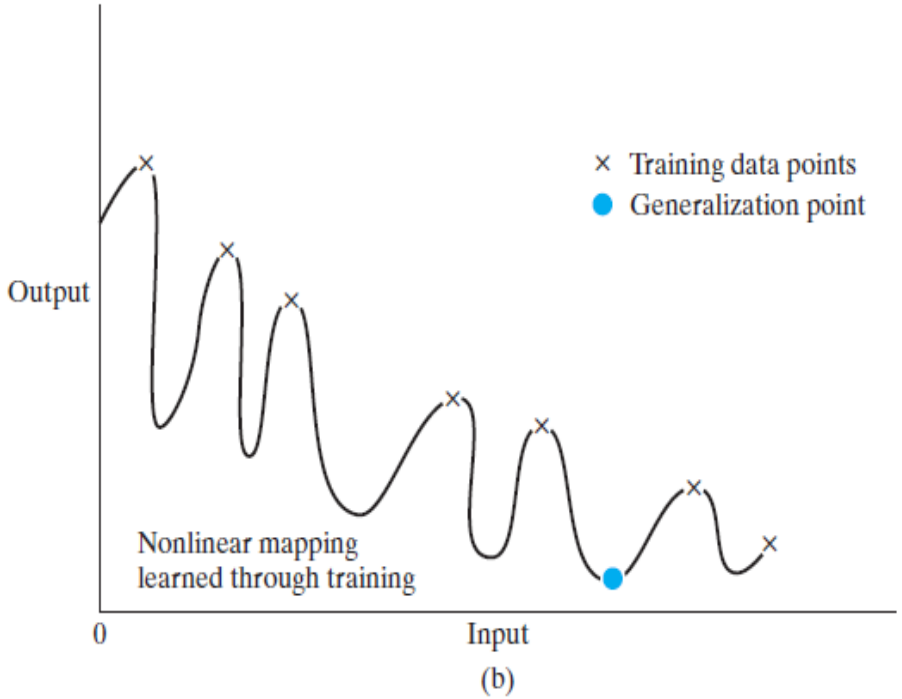
for all $x_1, x_2, \ldots, x_{m_0}$ that lie in the input space.

Simon Haykin, Neural Networks and Learning Machines, Prentice Hall (2008)

人工神经网络训练和泛化



恰当地拟合数据
（好的泛化）



过拟合数据
（差的泛化）

人工神经网络

要素

神经元结构

网络拓扑

学习算法

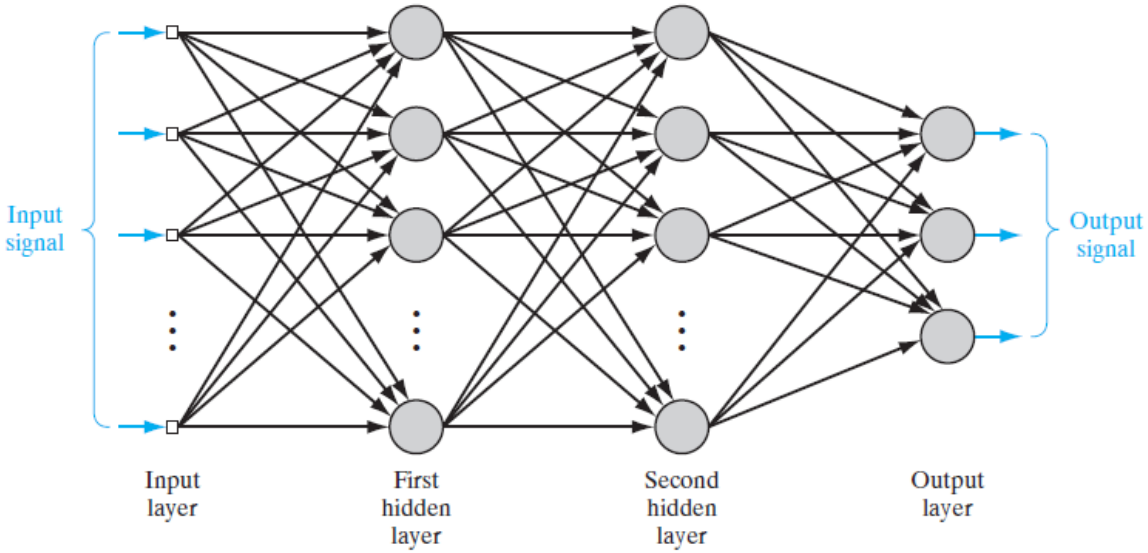
特点

大规模并行性

冗余性、容错性

本质非线性

强大的自组织、自学习、自适应能力



本讲的主要内容

一、人工神经网络简介

二、神经网络控制概述

三、神经网络建模

四、神经网络控制

经典和现代控制理论

- 建立被控对象数学模型
- 根据数学模型进行分析(Analysis)
- 根据数学模型和分析结果设计合适的控制器(Synthesis)

存在问题

- 对非线性系统尚没有通用的分析设计方法
- 对相当多的被控对象尚不能用机理分析或系统辨识的方法获得足够精度的数学模型
- 自适应控制理论与方法对线性定常或慢时变系统比较成熟，对非线性系统尚缺乏通用的自适应控制理论与方法

智能控制

自动控制发展的新方向：将传统控制和人工智能理论结合，模仿人的智能来研究解决复杂控制问题

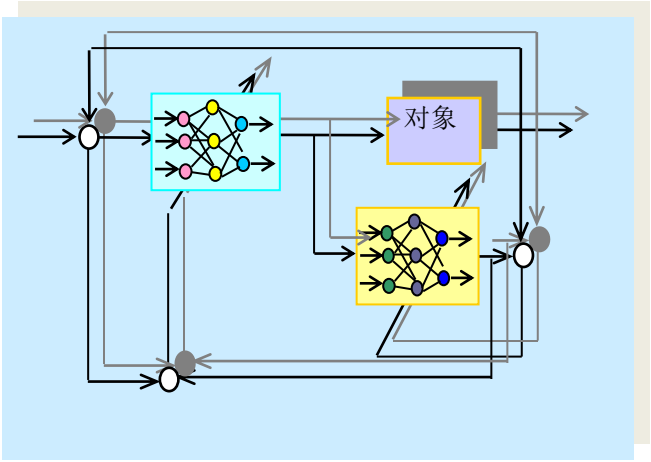
- 神经网络控制
- 模糊控制
- 专家系统
- 强化学习
- 遗传算法….

神经网络控制概述

神经网络

控制

- 处理那些难以用模型或规则描述的对象
- 采用并行分布式信息处理方式，具有很强的容错性
- 可以实现任意非线性映射，用于非线性系统控制
- 具有很强的信息综合能力，能够同时处理大量不同类型的输入，能够很好地解决输入信息之间的互补性和冗余性问题
- 硬件实现愈趋方便，大规模集成电路技术的发展为神经网络的硬件实现提供了技术手段



神经网络控制

(1) 神经网络用于辨识系统：可在已知常规模型结构的情况下，估计模型的参数；或利用神经网络的线性、非线性特性，建立线性、非线性系统的静态、动态、逆动态及预测模型；

(2) 神经网络用作控制器：神经网络作为控制器，可实现对不确定系统或未知系统进行有效的控制，使控制系统达到所要求的动态、静态特性；

(3) 神经网络与其他算法相结合：神经网络与专家系统、模糊逻辑、遗传算法等相结合可构成新型控制器；

神经网络控制

(4) 优化计算：在常规控制系统的设计中，常遇到求解约束优化问题，神经网络为这类问题提供了有效的途径；

(5) 控制系统的故障诊断：利用神经网络的逼近特性，可对控制系统的各种故障进行模式识别，从而实现控制系统的故障诊断。

神经网络控制

1985年Rumelhart突破性研究为界，覆盖控制理论中的绝大多数控制问题

- **系统建模与辨识、PID参数整定、极点配置**
- **预测控制、最优控制、自适应控制、滤波与预测、容错控制、内模控制**
- **模糊控制、专家控制和学习控制等**
- **与控制相关的问题如A/D，D/A转换、矩阵求逆、Jacobi矩阵计算，QR分解、Lyapunov方程和Riccati方程求解等**

神经网络控制

- 神经网络的稳定性与收敛性问题；
- 神经网络学习算法的实时性；
- 神经网络控制器和辨识器的模型和结构；
- 神经网络控制系统的稳定性与收敛性问题；

本讲的主要内容

一、人工神经网络简介

二、神经网络控制概述

三、神经网络建模

四、神经网络控制

神经网络建模（辨识）

- **辨识建模**：是在输入和输出数据的基础上，从一组定的模型中，确定一个与所测系统等价的模型

- **神经网络建模**：用神经网络作为被辨识对象的正模型、逆模型、预测模型等，辨识只需利用对象本身的输入输出数据即可

- 关键：网络结构确定及连接权值等的参数确定和调整对同样的初始条件和任何特定输入，模型和过程产生同样的输出

建模的两种基本情况

前向建模：建立系统本身的模型，也称正向建模；

逆向建模：建立系统的逆模型。

神经网络建模（辨识）

神经网络辨识建模的考虑因素

✓ **模型的选择**

精确性和复杂性的权衡：网络隐含层数的选择和隐含层内节点的选择。

权衡的有效途径：进行多次仿真实验。

✓ **输入信号的选择**

时域上，要求输入信号持续加在系统对象上，以便在辨识时间内充分激励系统的所有模态、反映系统对象的完整动态过程。（这里的输入信号是加在系统上的信号，也将构成神经网络的输入或输出信号）

频域上，要求输入信号的频谱覆盖系统的频谱。

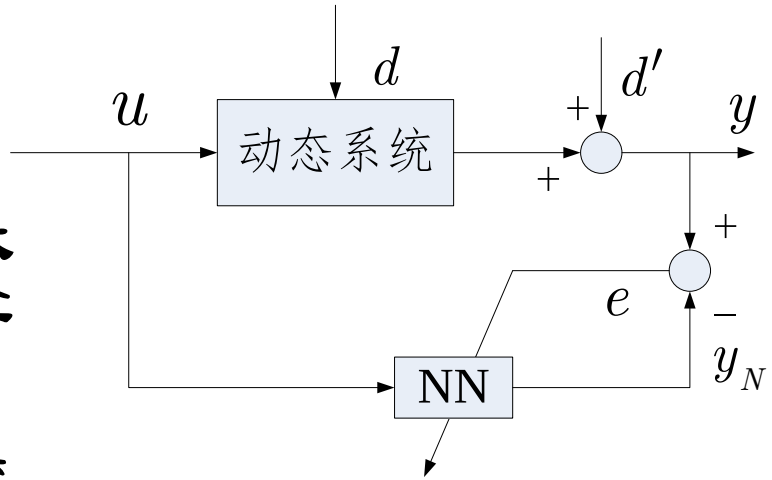
✓ **等价准则的选择**

等价意味着按照某种误差评价准则，使确定的神经网络模型最好地拟合所关心的被辨识系统的静态或动态特性。

正模型辨识

- 正向建模：利用多层前馈神经网络，通过训练或学习，使其能够表达系统正向动力学特性的模型

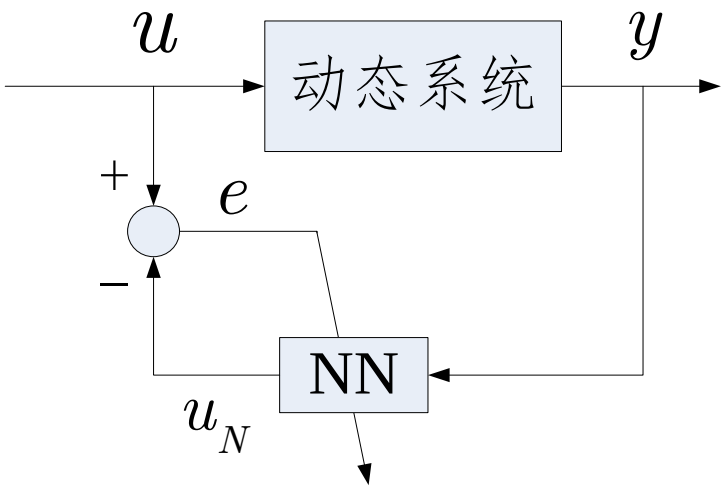
- NN与待辨识系统并联
- 输出误差作为训练信号
- 监督学习（实际系统作为教师，为神经网络提供学习算法所需要的期望输出）
- 系统可以是被控对象或传统控制器
- 可认为是被辨识系统的一个物理实现，用于在线控制



逆模型辨识

直接逆建模

- 建立逆模型对控制意义重大
- 需假定系统逆性
- 学习样本需选择，一般应比未知系统的运行范围更大
- 缺点：不是目标导向的，系统输入也不可能预先定义



神经网络直接逆辨识

正逆模型

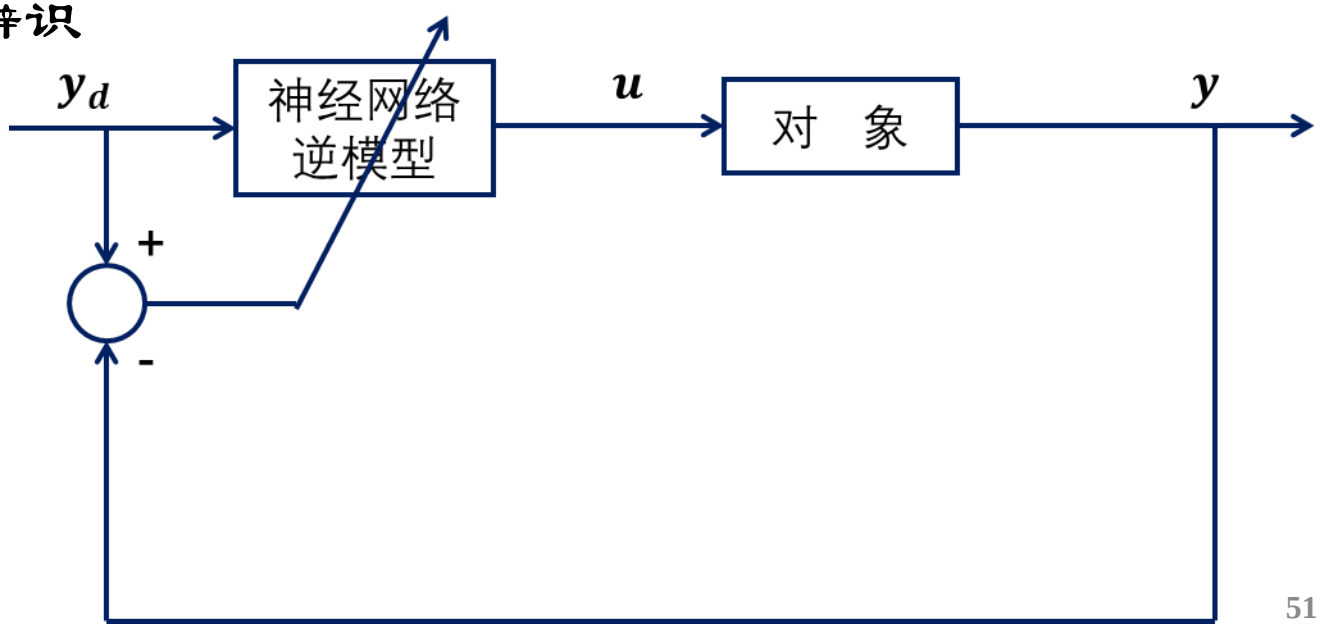
- 正—逆建模(狭义逆学习Specialized Inverse Learning)

辨识的主要目的：建立对象的逆模型；

训练误差性能：$e(k) = y_d(k) - y(k)$或$e(k) = y_d(k) - y_N(k)$。

优点：是目标导向的，即训练信号是期望输出与实际输出之差。

一般用于在线辨识



正逆模型

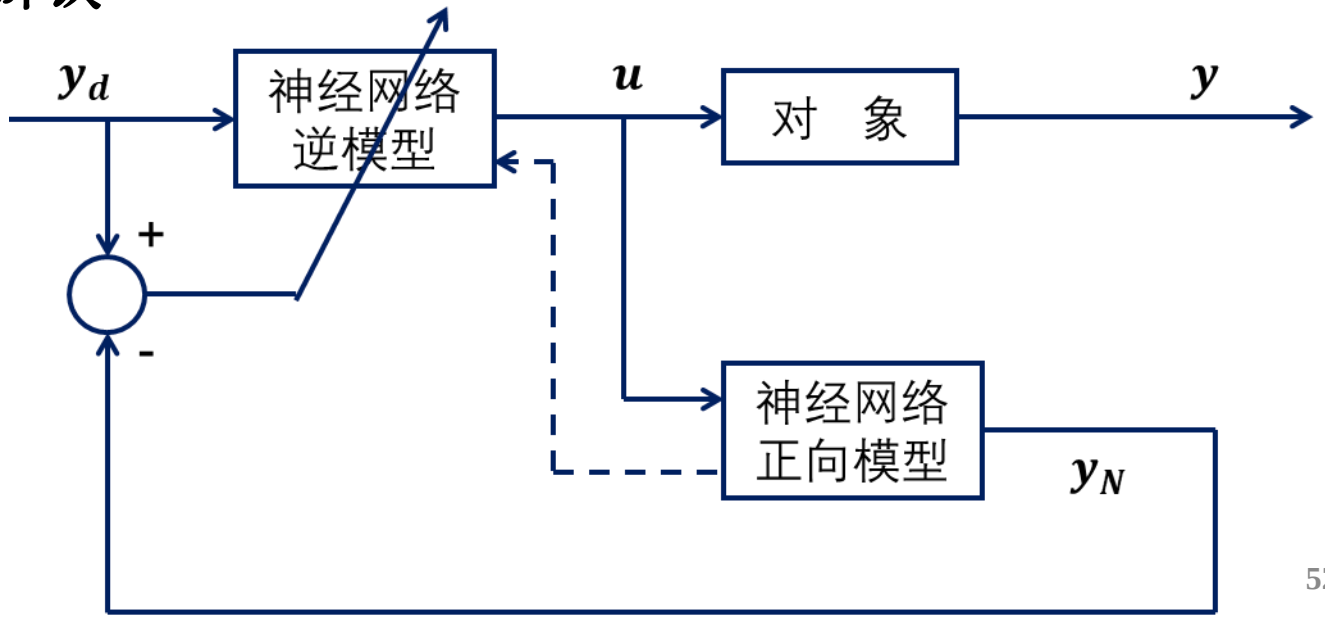
- 正—逆建模(狭义逆学习Specialized Inverse Learning)

辨识的主要目的：建立对象的逆模型；

训练误差性能：$e(k) = y_d(k) - y(k)$或$e(k) = y_d(k) - y_N(k)$。

优点：是目标导向的，即训练信号是期望输出与实际输出之差。

一般用于在线辨识



正逆模型

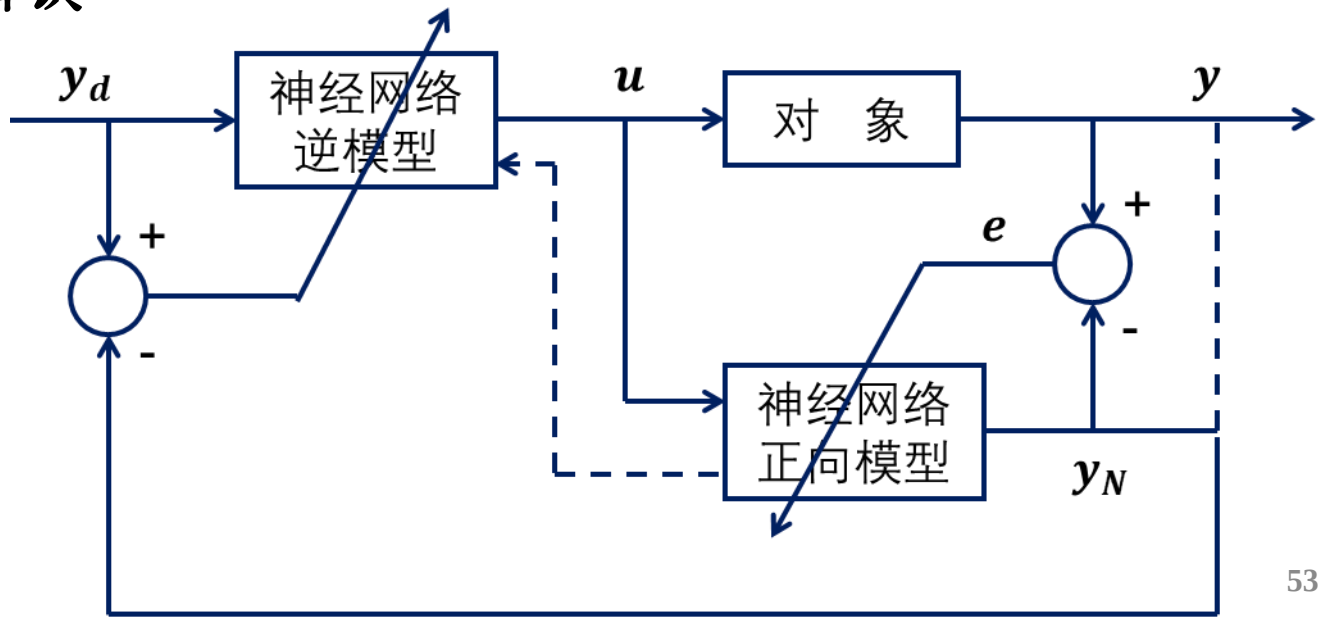
- 正—逆建模(狭义逆学习Specialized Inverse Learning)

辨识的主要目的：建立对象的逆模型；

训练误差性能：$e(k) = y_d(k) - y(k)$或$e(k) = y_d(k) - y_N(k)$。

优点：是目标导向的，即训练信号是期望输出与实际输出之差。

一般用于在线辨识



辨识工作模式

离线辨识与在线辨识

①在线辨识：在对象系统实际运行的过程中进行的，辨识过程要求实时性，即必须在一个采样周期的时间间隔内至少进行一次网络权值的调整；

②离线辨识：在取得对象系统的一批输入输出数据后再进行辨识，故辨识过程与实际系统是分离的，无实时性要求。

最好的辨识方式：先进行离线训练、再进行在线学习，将离线训练得到的权值作为在线学习的初始权，以加快在线学习的速度。

神经网络辨识特点

- **神经网络辨识的特点**(与传统辨识方法相比)

✓神经网络本身作为一种辨识模型，其可调参数反映在网络内部的权值上，无需建立实际系统的辨识格式。

✓借助网络外部的输入/输出数据拟合系统的输入/输出关系，可对本质非线性系统进行辨识。(网络内部隐含着系统的特性)

✓辨识的收敛速度不依赖于被辨识系统的维数，只与神经网络本身所采用的学习算法有关。

✓神经网络具有大量的连接，连接权值构成神经网络模型的参数，通过调节这些权值使网络输出逼近系统输出。

✓神经网络作为实际系统的辨识模型，构成系统的一个物理实现，可用于在线控制。

辨识网络选择

- **用什么类型的神经网络建模？**

静态系统：

- ✓ 稳态系统，系统的各状态变量不随时间变化
- ✓ 实际上就是一个非线性映射关系
- ✓ 前向神经网络（静态网络）进行建模

具有n维输入矢量和m维输出矢量的非线性静态系统的输入输出关系：

$$\begin{cases} y_1 = f_1(x_1, x_2, \cdots, x_n) \\ \vdots \\ y_m = f_m(x_1, x_2, \cdots, x_n) \end{cases}$$

辨识网络选择

- **用什么类型的神经网络建模？**

动态系统：

- ✓ 直接使用动态神经网络，分析相对复杂：网络本身引入动态环节（动态递归网络）或在神经元中引入动态特性
- ✓ 将动态系统的神经网络辨识转化为静态系统的神经网络辨识，简单、常用

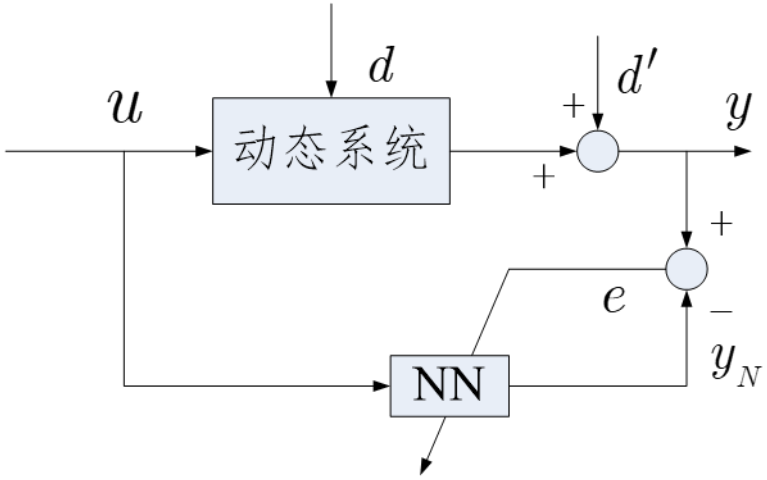
n维输入矢量和m维输出矢量的非线性动态系统：

$$\begin{cases} \dot{x}_1 = f_1(x_1, x_2, \cdots, x_n) \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \cdots, x_n) \end{cases} \quad \begin{cases} y_1 = g_1(x_1, x_2, \cdots, x_n) \\ \vdots \\ y_m = g_m(x_1, x_2, \cdots, x_n) \end{cases}$$

静态网络辨识

辨识非线性系统需解决如下两个问题

- 确定辨识模型NN的结构
- 确定参数学习算法，使学习过程尽快收敛。



静态网络辨识

多入多出系统两种常见的神经网络辨识结构：

第一，每个非线性方程分别构造一个神经网络。假设对应于第i个方程的神经网络输出为$\hat{y}_i$，则调整网络权系数的误差评价函数为

$$J(W_i)=\frac{1}{2}e_i^2=\frac{1}{2}(y_i-\hat{y}_i)^2$$

其中，W_i为第i个神经网络的权系数。

第二，仅用一个神经网络构成静态系统的神经网络模型，用神经网络的输出矢量为$\hat{Y}$逼近系统的实际输出矢量$Y=[y_1,y_2,\cdots,y_m]^T$，调整权系数的误差评价函数为

$$J(W)=\frac{1}{2}\sum_{i=1}^{m}e_i^2=\frac{1}{2}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2$$

静态网络辨识—学习算法

定义辨识系统误差评价函数为

$$J = \frac{1}{2T}\sum_{k=0}^{T-1} e^2(k)$$

$$e(k) = y(k) - \hat{y}(k)$$

BP算法

$$w_k = w_{k-1} - \eta \frac{\partial J}{\partial w}$$

$$\frac{\partial J}{\partial w} = \frac{-1}{T}\sum_{k=0}^{T-1} e(k)\frac{\partial \hat{y}(k)}{\partial w}$$

$\eta > 0$是学习率（步长）

静态网络辨识—时间序列模型

线性 AR (auto-regressive) 模型 （n阶自回归模型）

$$y(k+1)=\sum_{i=0}^{n-1}\alpha_i y(k-i)$$

线性 MA(moving average) 模型 （m阶滑动平均模型）

$$y(k+1)=\sum_{j=0}^{m-1}\beta_i u(k-j)$$

线性 ARMA(auto-regressive moving average) 模型

$$y(k+1)=\sum_{i=0}^{n-1}\alpha_i y(k-i)+\sum_{j=0}^{m-1}\beta_i u(k-j)$$

自回归滑动平均模型

静态网络辨识

非线性系统的四种类型（NARMA模型）的辨识

① 关于过去n时刻的输出是线性的

$$y(k+1)=\sum_{i=0}^{n-1}\alpha_i y(k-i)+g[u(k),u(k-1),\cdots,u(k-m+1)]$$

② 关于过去m时刻的输入是线性的

$$y(k+1)=f[y(k),y(k-1),\cdots,y(k-n+1)+\sum_{i=0}^{m-1}\beta_i u(k-i)$$

静态网络辨识

非线性系统的四种类型（NARMA模型）的辨识

③ 过去n时刻的输出和过去m时刻的输入可分离

$$y(k+1)=f[y(k),y(k-1),\cdots,y(k-n+1)]+g[u(k),u(k-1),\cdots,u(k-m+1)]$$

④ 过去n时刻的输出和过去m时刻的输入不可分离

$$y(k+1)=f[y(k),y(k-1),\cdots,y(k-n+1);u(k),u(k-1),\cdots,u(k-m+1)]$$

f,g为非线性函数；系统本身的阶次n,m已知；

输入输出数据$\{u(k-i),i=0,1,\cdots\}$与$\{y(k-j),j=0,1,\cdots\}$可测量

辨识方法

对于模型①②，

线性部分的参数已知：仅需用神经网络辨识非线性部分g, f；

线性部分的参数未知：对线性部分可借助传统的最小二乘辨识法，对非线性部分则用神经网络辨识出g, f。

对于模型④，

输入输出无特殊性；利用一个多层前馈网络即可；

网络输入：模型④方程右边的$n+m$个时刻的输入输出数据；

网络输出：$y(k+1)$。

对于模型③，

过去时刻的输入输出可分离；

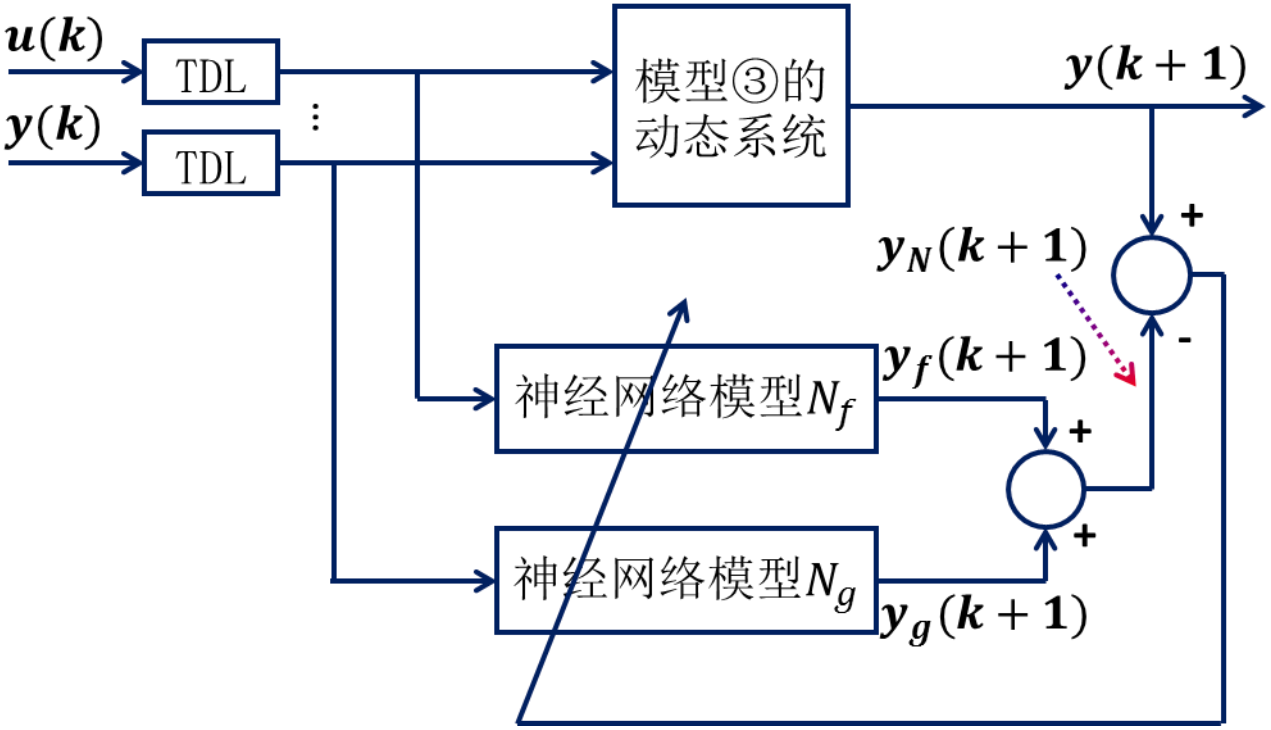
可利用一个多层前馈网络，等同于模型④的辨识；

也可对分离的两部分同时使用两个前馈网络，假定网络的串联辨识模型如下：

$$\hat{y}(k+1) = N_f[y(k), y(k-1), \cdots, y(k-n+1)] + N_g[u(k), u(k-1), \cdots, u(k-m+1)]$$

其中，网络N_f和N_g分别用以逼近系统的非线性函数f和g。

神经网络建模



网络N_f的输入：$[u(k), u(k-1), \cdots, u(k-m+1)]$；

网络N_g的输入：$[y(k), y(k-1), \cdots, y(k-n+1)]$；

网络N_f与N_g的期望输出均未知，但期望输出之和$N_f + N_g$可测量，即$y(k+1)$。这两个网络不能同时独立使用，需共用同一期望输出。

模型③的算法： 假定两个前向传播网络的层数均为L；

对于给定的P组样本矢量$[X_p, T_p], p = 1,2,\cdots P$，误差性能指标为：

$$E_p = \frac{1}{2}\sum_{j=1}^{n_o}\left[t_{pj} - \left(o_{pj}^1 + o_{pj}^2\right)\right]^2$$

$$E = \sum_{p=1}^{P} E_p$$

其中，o_{pj}^1为第一个神经网络输出层的第j个神经元，o_{pj}^2为第二个神经网络输出层的第j个神经元；n_o为两个网络输出层的神经元个数，也即系统输出矢量y的维数。

基于梯度下降法，并考虑误差反向传播算法，可以同时得到两个网络的权值修正公式：

对输出层：

$$\delta_{pj}^{1(L)} = \left(t_{pj} - \left(o_{pj}^{1(L)} + o_{pj}^{2(L)}\right)\right) f^{1'}{}_{Net_{pj}^{1(L)}}$$

$$\delta_{pj}^{2(L)} = \left(t_{pj} - \left(o_{pj}^{1(L)} + o_{pj}^{2(L)}\right)\right) f^{2'}{}_{Net_{pj}^{2(L)}}$$

对于第r个隐含层：

$$\delta_{pj}^{1(r)} = f^{1'}{}_{Net_{pj}^{1(r)}} \sum_{l} \delta_{pl}^{1(r+1)} w_{lj}^{1(r+1)}$$

$$\delta_{pj}^{2(r)} = f^{2'}{}_{Net_{pj}^{2(r)}} \sum_{l} \delta_{pl}^{2(r+1)} w_{lj}^{2(r+1)}$$

其中，l对各网络第$r+1$层的所有神经元求和。

权值修正公式：

$$\Delta_p w_{ji}^{1(r)} = \eta_1 \delta_{pj}^{1(r)} o_i^{1(r)}$$

$$\Delta_p w_{ji}^{2(r)} = \eta_2 \delta_{pj}^{2(r)} o_i^{2(r)}$$

上标中的1、2表示第1、2个神经网络，(r)表示网络的第r层。

静态网络辨识

将动态系统转化为静态系统

首先假定拟辨识对象为线性或非线性离散时间系统，或者人为地离散化动态系统，将动态系统由下列差分方程表示：(NARMA模型)

$$\dot{y} = F(y, u) \Rightarrow$$

$$y(k+1) = f[y(k), y(k-1), \cdots, y(k-n+1); u(k), u(k-1), \cdots, u(k-m+1)]$$

- 将非线性函数f中不同时刻的所有变量$y(k), y(k-1), \cdots, y(k-n+1), u(k), u(k-1), \cdots, u(k-m+1)$全部看做自变量，将$y(k+1)$看做因变量
- 表示系统动力学变化规律就变成了一个**非线性的静态映射f**

- 建立系统本身的模型：逼近未知非线性映射f
- 若f可逆，则得动态系统的逆模型：

$$u(k)=f^{-1}[y(k),y(k-1),\cdots,y(k-n+1),y(k+1);u(k-1),\cdots,u(k-m+1)]$$

$$u(k)=f^{-1}[y(k),y(k-1),\cdots,y(k-n+1),y_d(k+1);u(k-1),\cdots,u(k-m+1)]$$

建立逆模型：逼近未知非线性f^{-1}

神经网络输入：$y(k),y(k-1),\cdots,y(k-n+1),y_d(k+1),u(k-1),\cdots,u(k-m+1)$；

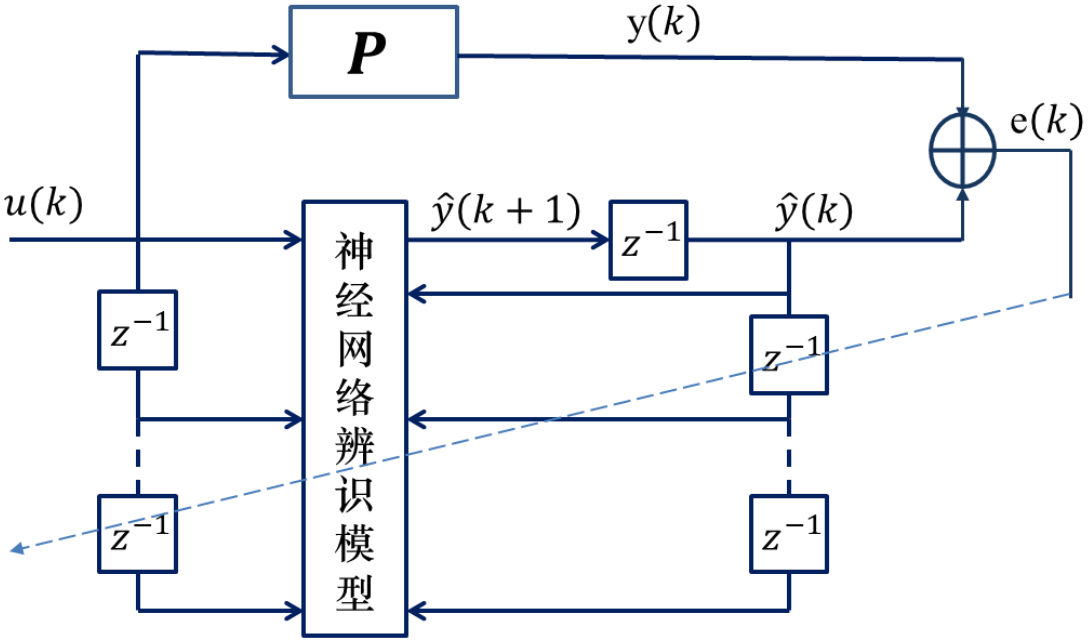
神经网络输出：

$$u(k)$$

两种辨识模型

(1) 并列模型

$$\hat{y}(k+1) = N[\hat{y}(k), \hat{y}(k-1), \cdots, \hat{y}(k-n+1), u(k), u(k-1), \cdots, u(k-m+1)]$$

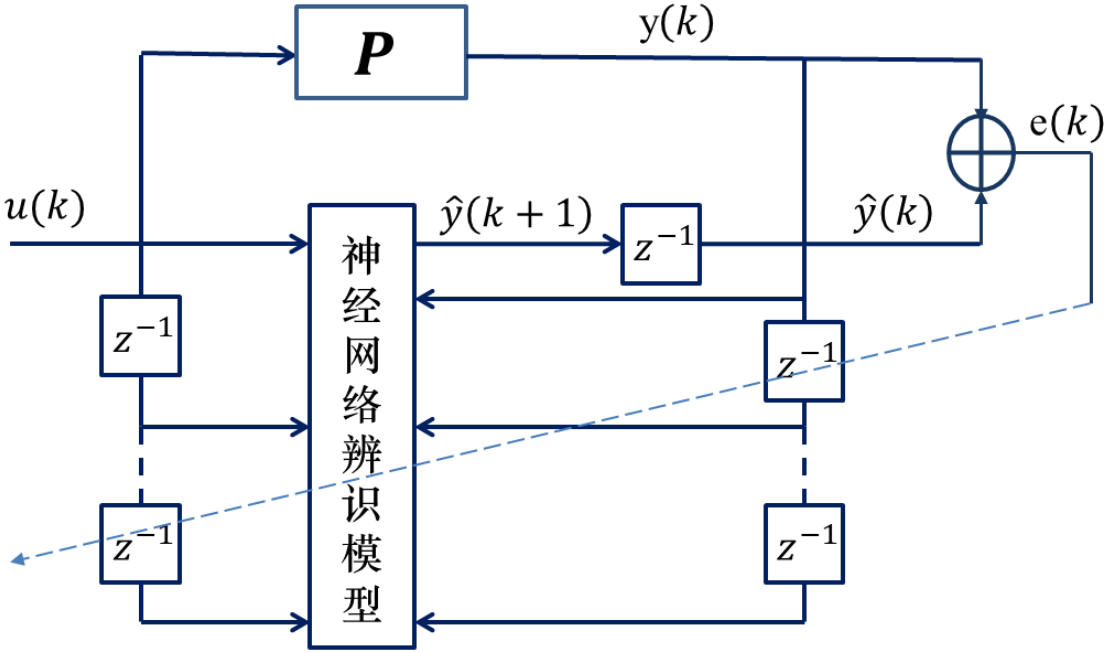


输入到神经网络的量包含了网络本身输出的过去值，即网络含到自身的反馈，故所用网络为动态神经网络

两种辨识模型

(2) 串联并列模型

$$\hat{y}(k+1)=N[y(k),y\ (k-1),\cdots,y\ (k-n+1),\quad u(k),u(k-1),\cdots,u(k-m+1)]$$



网络自身不含反馈，可采用BP算法调整权系数。可使辨识过程简化，收敛性较好，一般场合均采用

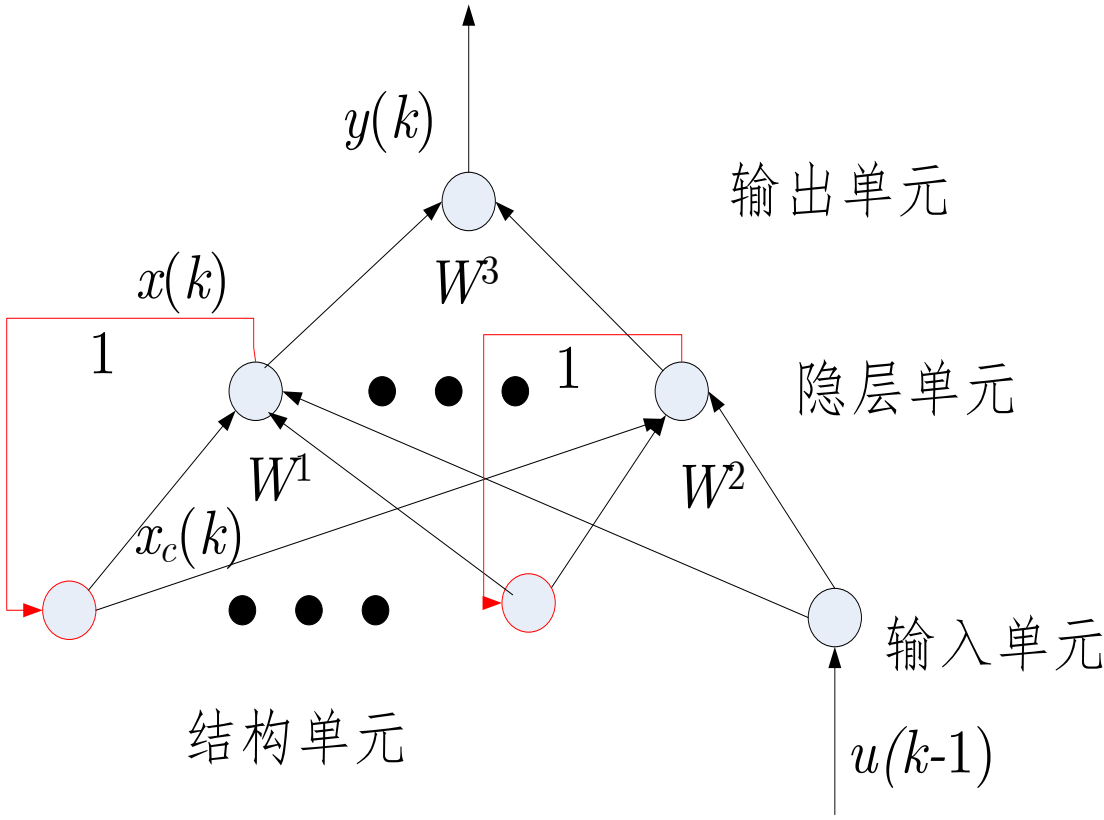
动态网络辨识

转换成静态系统进行静态辨识：

- 需先验假定系统的NARMA模型类
- 需要对结构模型进行定阶
- 较多的输入节点将使相应的辨识系统对外部噪声特别敏感
- 阶次增加或未知时，网络结构膨胀，学习收敛速度变缓慢

基本Elman动态递归网络

- 完全递归与部分递归
- **完全递归**具有任意的前馈与反馈连接，且所有连接权都可以进行修正
- **部分递归**网络的主要网络结构是前馈，连接权可以修正，反馈连接由“结构”单元构成，权不可修正



基本Elmann网络结构

部分递归

基本Elman动态递归网络

数学模型

输入$u(k-1) \in R^r$，输出$y(k) \in R^m$，隐层输出$x(k) \in R^n$

$$x(k) = f\left(W^1 x_c(k) + W^2 u(k-1)\right)$$

$$x_c(k) = x(k-1)$$

$$y(k) = g\left(W^3 x(k)\right)$$

隐层与输出单元采用线性函数且隐层及输出层阈值为0，则

$$x(k) = W^1 x(k-1) + W^2 u(k-1)$$

$$y(k) = W^3 x(k)$$

基本Elman动态递归网络

$$x_c(k) = x(k-1) = f(W^1_{k-1}x_c(k-1) + W^2_{k-1}u(k-2))$$

$$x_c(k-1) = x(k-2)$$

$$E = \sum_{p=1}^{N} E_p \qquad E_p = \frac{1}{2}(y_d(k) - y(k))^T(y_d(k) - y(k))$$

$$\Delta w^3_{ij} = \eta \delta^0_i x_j(k) \quad i = 1,2,\cdots,m; j = 1,2,\cdots,n$$

$$\Delta w^2_{jq} = \eta \delta^h_j u_q(k-1) \quad j = 1,2,\cdots,n; q$$

$$\Delta w^1_{jl} = \eta \sum_{i=1}^{m}(\delta^0_i w^3_{ij}) \frac{\partial x_j(k)}{\partial w^1_{jl}} \quad j = 1,2,\cdots,$$

梯度动态递推

$$\frac{\partial x_j(k)}{\partial w^1_{jl}} = f'_j(\cdot)\{x_l(k-1) + \sum_{i=1}^{n} w^1_{ji} \frac{\partial x_j(k-1)}{\partial w^1_{jl}}\}$$

$$\delta^0_i = (y_{d,i}(k) - y_i(k))g'_j(\cdot)$$

$$\delta^h_j = \sum_{i=1}^{m}(\delta^0_i w^3_{ij}) f'_j(\cdot)$$

采用标准BP算法，辨识一阶动态系统效果好、高阶系统效果差

修改的Elman网络

$$x(k) = f\left(W^1 x_c(k) + W^2 u(k-1)\right)$$

$$x_c(k) = \alpha x_c(k-1) + x_l(k-1)$$

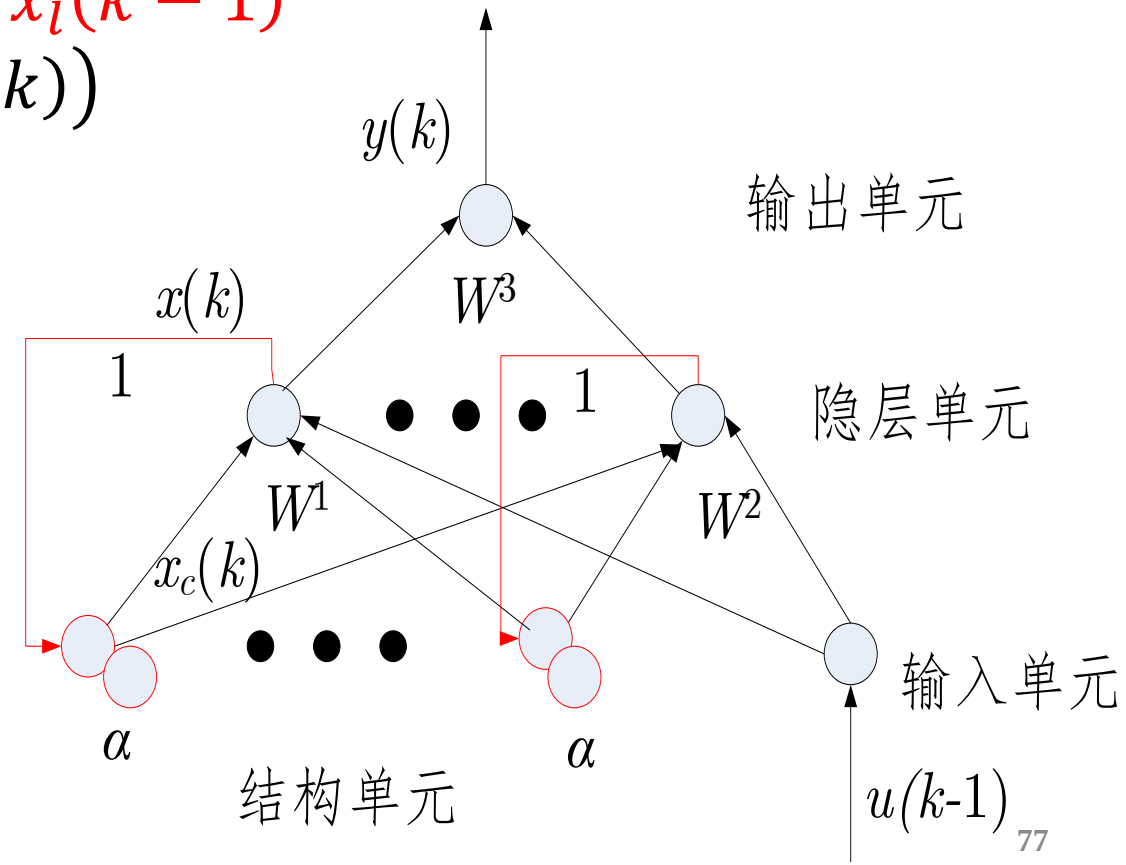
$$y(k) = g\left(W^3 x(k)\right)$$

结构单元增加自反馈

模拟高阶系统，

解决高阶系统辨识

采用BP学习算法



修改的Elman网络

$$\Delta w_{ij}^3 = \eta \delta_i^0 x_j(k) \quad i = 1,2,\cdots,m; j = 1,2,\cdots,n$$

$$\Delta w_{jq}^2 = \eta \delta_j^h u_q(k-1) \quad j = 1,2,\cdots,n; q = 1,2,\cdots,r;$$

$$\Delta w_{jl}^1 = \eta \sum_{i=1}^{m} (\delta_i^0 w_{ij}^3) \frac{\partial x_j(k)}{\partial w_{jl}^1} \quad j = 1,2,\cdots,n; l = 1,2,\cdots,n;$$

$$\frac{\partial x_j(k)}{\partial w_{jl}^1} = f'_j(\cdot) x_l(k-1) + \alpha \frac{\partial x_j(k-1)}{\partial w_{jl}^1}$$

$$\delta_i^0 = (y_{d,i}(k) - y_i(k)) g'_j(\cdot)$$

$$\delta_j^h = \sum_{i=1}^{m} (\delta_i^0 w_{ij}^3) f'_j(\cdot)$$

基本Elman动态递归网络

- 相对于静态网络

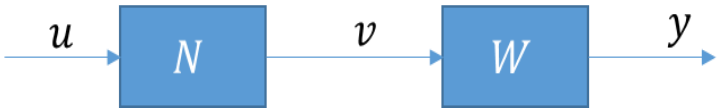
  用于单输入单输出系统，只需要一个输入和一个输出单元，即使有n个结构单元，隐层输入也只有$n+1$个

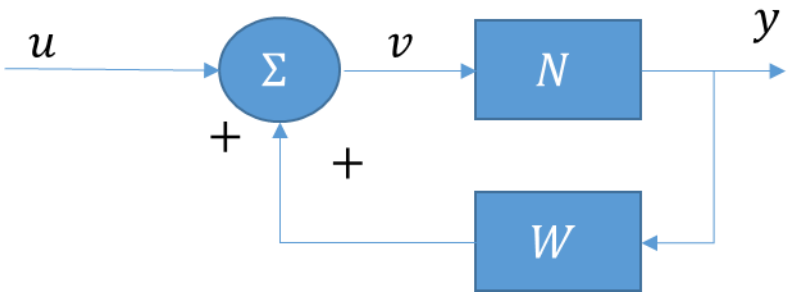
  Elmann网络的动态特性仅由内部连接提供，无需使用状态作为输入或训练信号

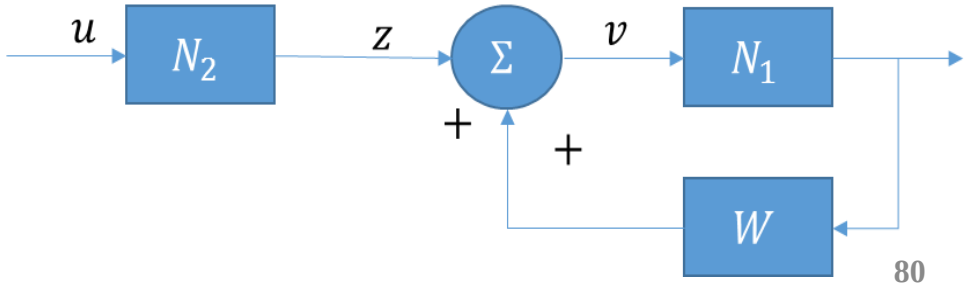
通用模型辨识

■ 几类辨识模型基本组成单元

- 一个或多个多层前馈网络$N^l(i_1, i_2, \cdots, i_L)$
- 线性动态单元$W(z^{-1})$（已知）
- 纯滞后单元z^{-1}（可看做$W(z^{-1})$的特例）







通用模型辨识

- 系统辨识的指标函数

$$J = \frac{1}{2}[y_d(k) - y(k)]^2$$

被辨识系统输出

辨识系统输出

BP算法

$$w_{new} = w_{old} - \eta \frac{\partial J}{\partial w}$$

$$\frac{\partial J}{\partial w} = [y_d(k) - y(k)] \frac{\partial y(k)}{\partial w}$$

通用模型辨识

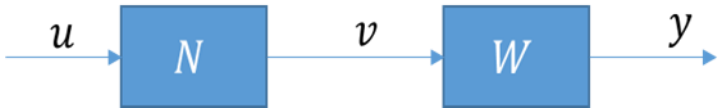
- 模型Ⅰ

$$\frac{\partial y(k)}{\partial w} = W(z^{-1})\frac{\partial v(k)}{\partial w}$$

$\frac{\partial v(k)}{\partial w}$可由一般BP算法得到

$\frac{\partial y(k)}{\partial w}$可看做是由$\frac{\partial v(k)}{\partial w}$经过一个动态系统产生，

故称之为动态BP算法



通用模型辨识

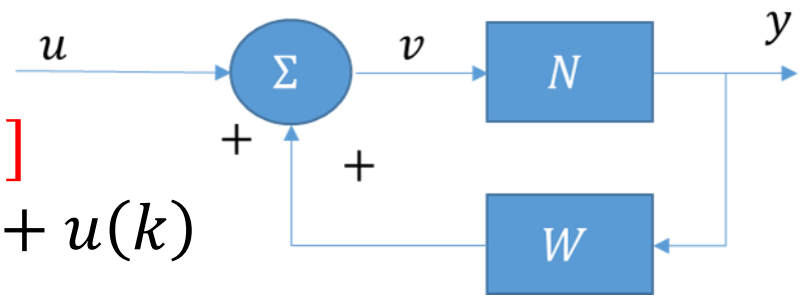
模型II：

N1在输出端，参数可由BP算法辨识，而且可以得到$\frac{\partial y(k)}{\partial v(k)}$，同时

$$\frac{\partial y(k)}{\partial w_2} = \frac{\partial y(k)}{\partial v(k)} W(z^{-1}) \frac{\partial z(k)}{\partial w_2}$$

通用模型辨识

模型III可用下面的两个方程描述



$$y(k) = N[v(k)]$$

$$v(k) = W(z^{-1})y(k) + u(k)$$

对w求导有

$$\frac{\partial y(k)}{\partial w} = \frac{\partial N[v]}{\partial v}\frac{\partial v(k)}{\partial w} + \frac{\partial N[v]}{\partial w}$$

$$\frac{\partial v(k)}{\partial w} = W(z^{-1})\frac{\partial y(k)}{\partial w}$$

$$\frac{\partial y(k)}{\partial w} = \frac{\partial N[v]}{\partial v}W(z^{-1})\frac{\partial y(k)}{\partial w} + \frac{\partial N[v]}{\partial w}$$

通用模型辨识

$\frac{\partial N[v]}{\partial v}$及$\frac{\partial N[v]}{\partial w}$可以用一般BP算法求出，

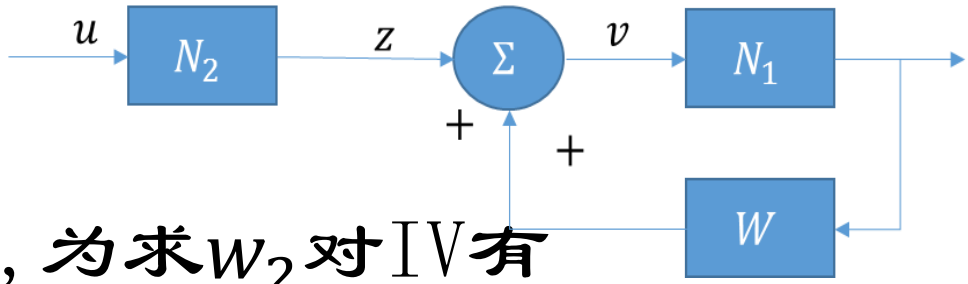
上式是一个线性化的差分方程，写成传递函数形式有

$$\frac{\partial y(k)}{\partial w} = \frac{1}{1 - \frac{\partial N[v]}{\partial v} W(z^{-1})} \frac{\partial N[v]}{\partial w}$$

$$= W_1(z^{-1}) \frac{\partial N[v]}{\partial w}$$

$W_1(z^{-1})$稳定，则可用DBP算法来辨识神经网络参数w

通用模型辨识



模型Ⅳ

- 对w_1的辨识同模型Ⅲ，为求w_2对Ⅳ有

$$y(k) = N_1[v(k)]$$

$$v(k) = W(z^{-1})y(k) + N_1[u(k)]$$

$$\frac{\partial y(k)}{\partial w_2} = \frac{\partial N_1[v]}{\partial v}\frac{\partial v(k)}{\partial w_2}$$

$$\frac{\partial v(k)}{\partial w_2} = W(z^{-1})\frac{\partial y(k)}{\partial w_2} + \frac{\partial N_2[u]}{\partial w_2}$$

通用模型辨识

$$\frac{\partial y(k)}{\partial w_2}=\frac{\partial N_1[v]}{\partial v}\left[W(z^{-1})\frac{\partial y(k)}{\partial w_2}+\frac{\partial N_2[u]}{\partial w_2}\right]$$

写成传递函数形式

$$\frac{\partial y(k)}{\partial w_2}=\frac{\frac{\partial N_1[v]}{\partial v}}{1-\frac{\partial N_1[v]}{\partial v}W(z^{-1})}\frac{\partial N_2[u]}{\partial w_2}=W_2(z^{-1})\frac{\partial N_2[u]}{\partial w_2}$$

若$W_2(z^{-1})$稳定，则可求出$\frac{\partial y(k)}{\partial w_2}$，在$k$时刻，一旦求出$\frac{\partial y(k)}{\partial w_2}$，便可求出

$$\frac{\partial N_2[u]}{\partial w_2}=\left[\frac{1}{\frac{\partial N_1[v]}{\partial v}}-W(z^{-1})\right]\frac{\partial y(k)}{\partial w_2}$$

考虑第III类系统 $$y(k+1)=\frac{y(k)}{1+y^2(k)}+u^3(k)$$

$u(k)\in[-2,2], y(k)\in[-10,10]$,BIBO稳定

辨识模型M为$\hat{y}(k+1)=N_f[y(k)]+N_g[u(k)]$

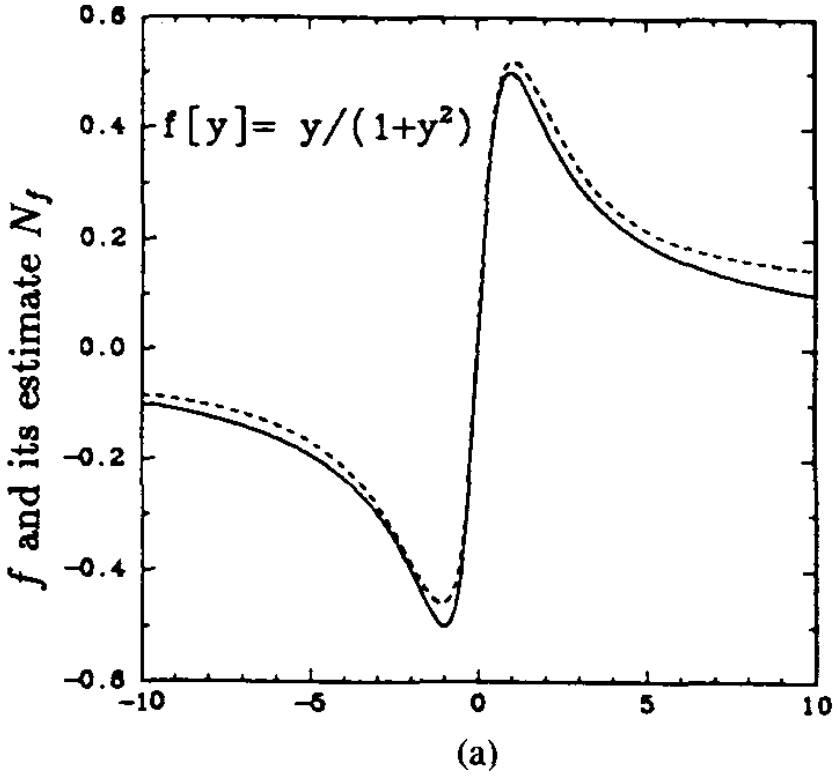
N_f, N_g采用四层前馈神经网络，结构为$N^4(1,20,10,1)$

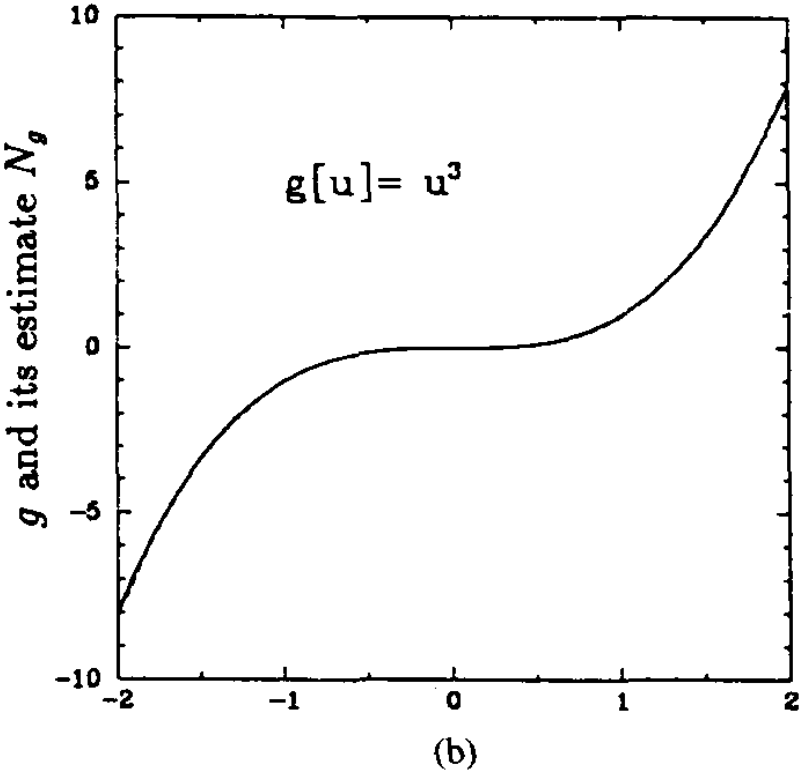
$u(k)$在$[-2,2]$随机变化，得到相应的y(k+1),作为训练数据

$\eta=0.1$

串-并联模式 → 并联模式验证

Kumpati S. Narendra. Identification and Control of Dynamical Systems Using Neural Networks, IEEE transactions on neural networks. Vol. I . No. I . 1990

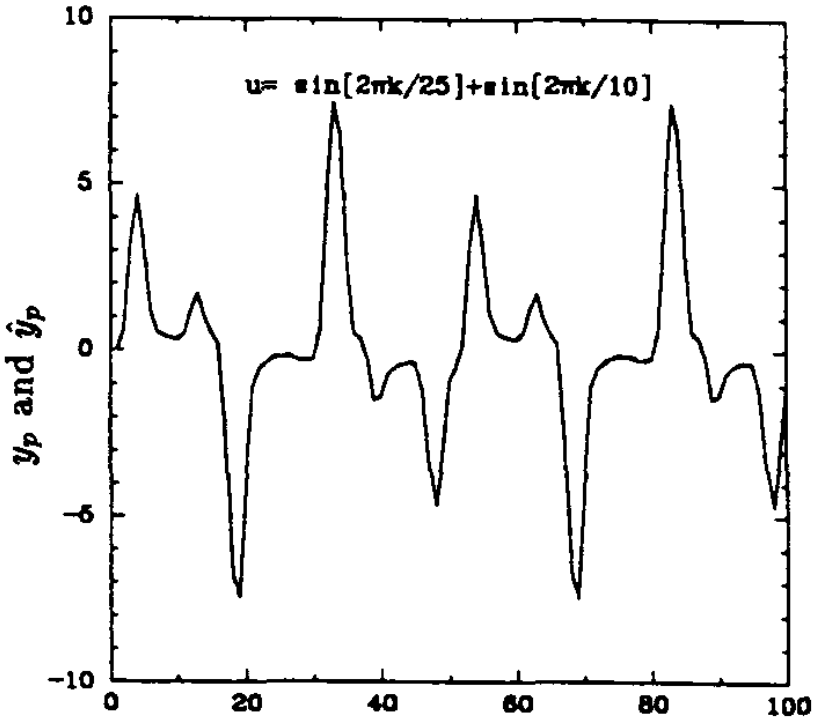




$$N_f[\hat{y}(k)] \text{与} \frac{y(k)}{1+y^2(k)}$$

$$N_g[u(k)] \text{与} u^3(k)$$

Kumpati **S.** Narendra. Identification and Control of Dynamical Systems Using Neural Networks, IEEE transactions on neural networks. Vol. I . No. I . 1990



$$\text{输入} u(k) = \sin\left(\frac{2\pi k}{25}\right) + \sin\left(\frac{2\pi k}{10}\right), k = 0, 1, 2, \cdots, 100$$

Kumpati **S.** Narendra. Identification and Control of Dynamical Systems Using Neural Networks, IEEE transactions on neural networks. Vol. I . No. I . 1990

本讲的主要内容

一、人工神经网络简介

二、神经网络控制概述

三、神经网络建模

四、神经网络控制

典型的神经网络控制结构

- 神经网络监督控制（神经网络学习控制）
- 神经网络自适应控制（自校正、模型参考控制）
- 神经网络内模控制
- 神经网络预测控制
- 神经网络自适应评价控制（神经网络再励控制）
- 神经网络混合控制

- 充当系统模型，构成各种控制结构，如在内模控制、模型参考自适应控制、预测控制中充当模型
- 直接用作控制器
- 在控制系统中起优化计算作用

神经网络控制的学习机制

神经网络控制器的特殊性：

控制器的样本信息通常无法预先知道(如：控制器的期望输出通常是系统的最佳控制量，一般无法通过测量获得)。

解决控制器的学习问题是关键。

神经网络控制器的学习：

寻找一种有效的途径进行网络连接权值或网络结构的修改，从而使得网络控制器输出的控制信号能够保证系统输出跟随系统的期望输出。

神经网络控制器的学习类别

(1) 监督式学习(有导师指导下的控制网络学习)

外界提供适当形式的导师信号，学习系统根据导师信号与相对应的实际输出量之差调节网络参数。

- ✓ 离线学习法
- ✓ 在线学习法
- ✓ 反馈误差学习法
- ✓ 多网络学习法

(2) 无监督学习(通过某一评价函数指定下的学习)

无导师信号，按照环境所提供数据的某些规则或适当的评价函数调节网络参数。

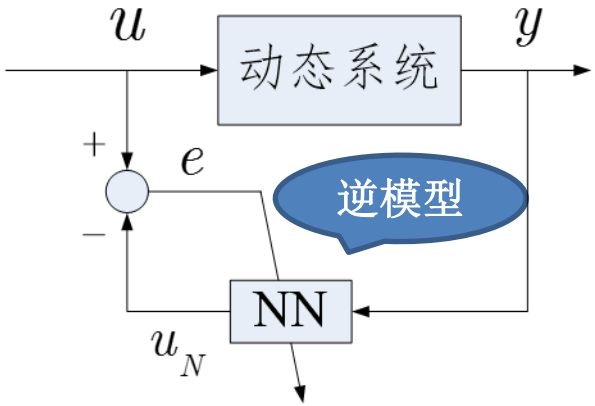
(3) 再励学习(强化学习)

介于上述两种情况之间，外部环境只对输出结果给出评价，而不给出具体答案，学习系统通过强化那些受奖励的动作来改善自身的性能。

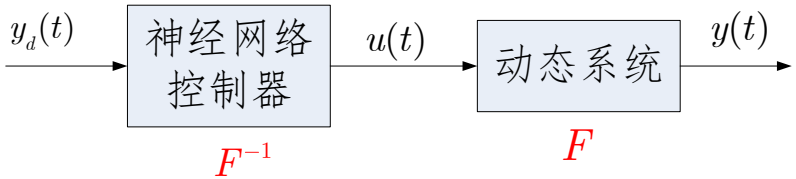
离线学习

对一批实现给定的系统输入输出样本数据进行离线学习，建立系统的一个逆模型，然后用此逆模型进行在线控制。

适合静态环境



离线学习(建模)



在线控制

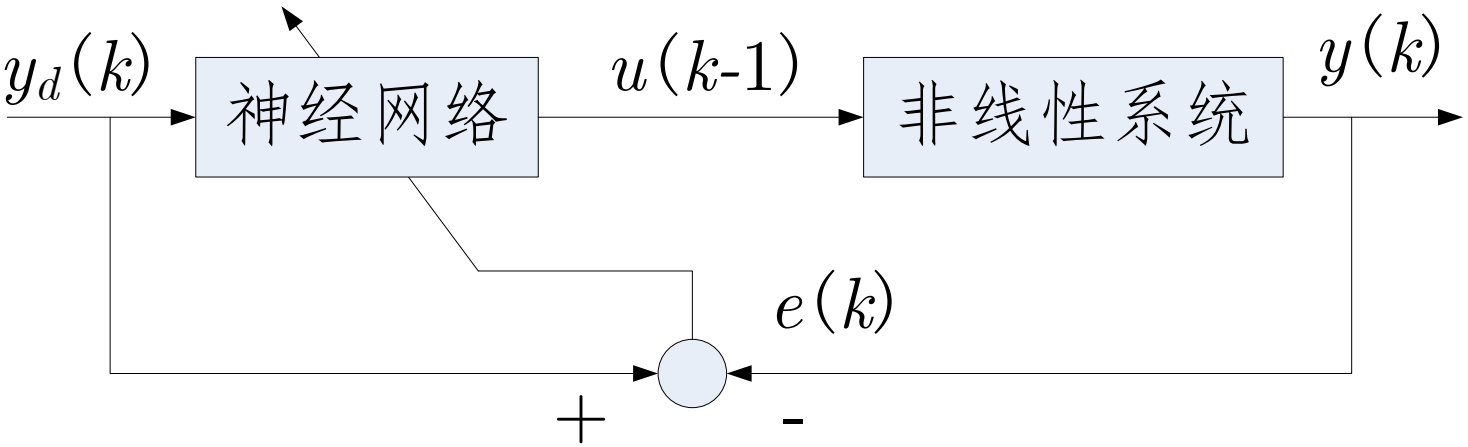
缺点：

① 导师信号u应遍及整个控制域，才能保证网络能最大范围地逼近系统的逆模型，而实际上很难构造这样的导师信号。

② 在环境或对象特性发生变化时无法使用，离线学习结束控制器的学习能力即停止。

③ 性能$(u-u_N)^2$的极小不能保证$(y_d-y)^2$的极小。

神经网络控制在线学习



学习目的：找到最优控制量u，使系统输出y趋于期望输出y_d。

非线性系统：

$$y(k)=f(y(k-1),\cdots,y(k-n),u(k-1),\cdots,u(k-m));$$

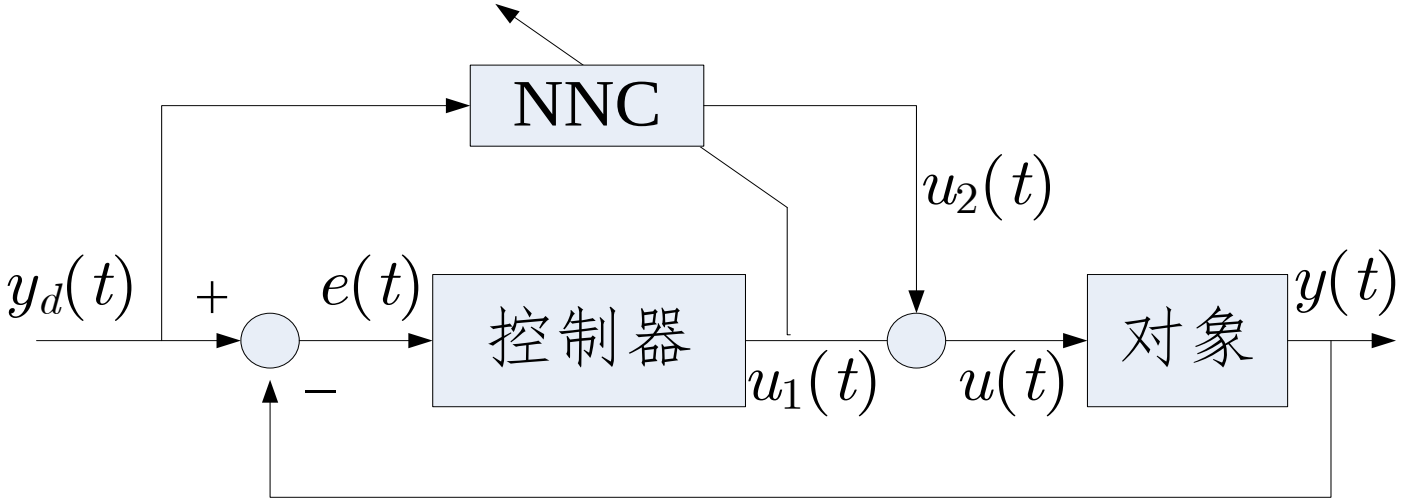
性能指标：$E(k)=\frac{1}{2}(y_d(k)-y(k))^2$

最速下降法调整权值：

$$\Delta w_{ji}^{(r)}(k)=\eta(y_d(k)-y(k))\frac{\partial y(k)}{\partial u(k-1)}\frac{\partial u(k-1)}{\partial w_{ji}^{(r)}}$$

适合模型已知的动态环境

神经网络控制反馈误差学习



动机：解决模型未知情况下的神经网络学习。

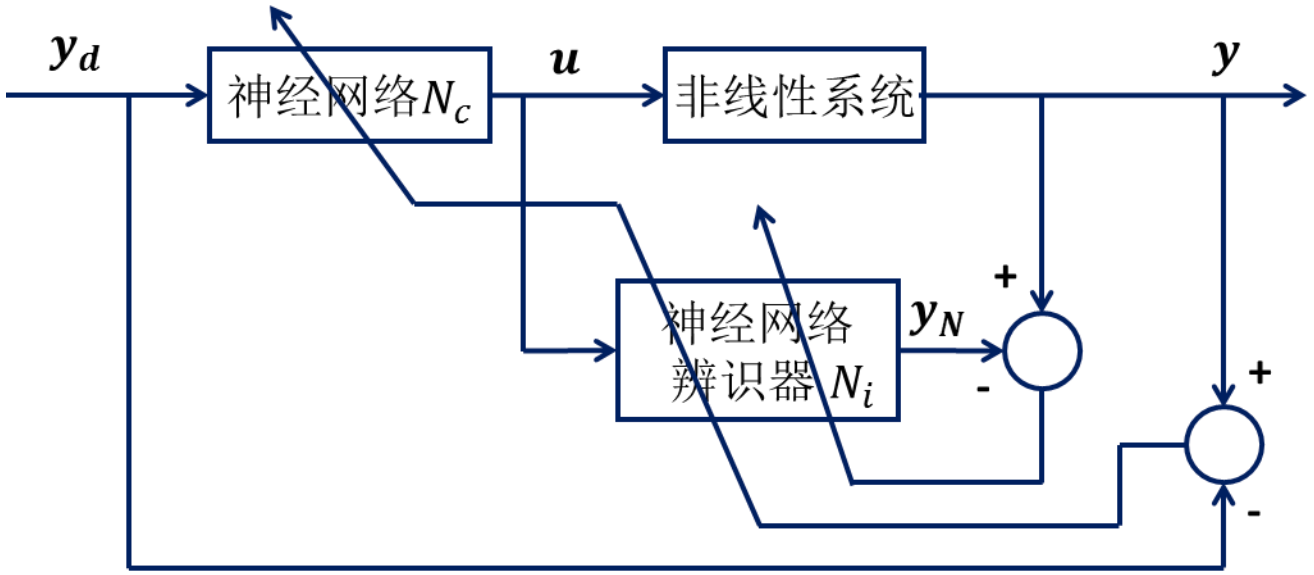
e：反馈误差；训练网络

训练中，神经网络控制器逐渐占主导地位，反馈仅用来克服扰动

网络训练中不涉及系统的动态性能，可能导致学习不收敛。

适用于非线性系统线性绝对占优条件下的网络学习。

前向多神经网络学习

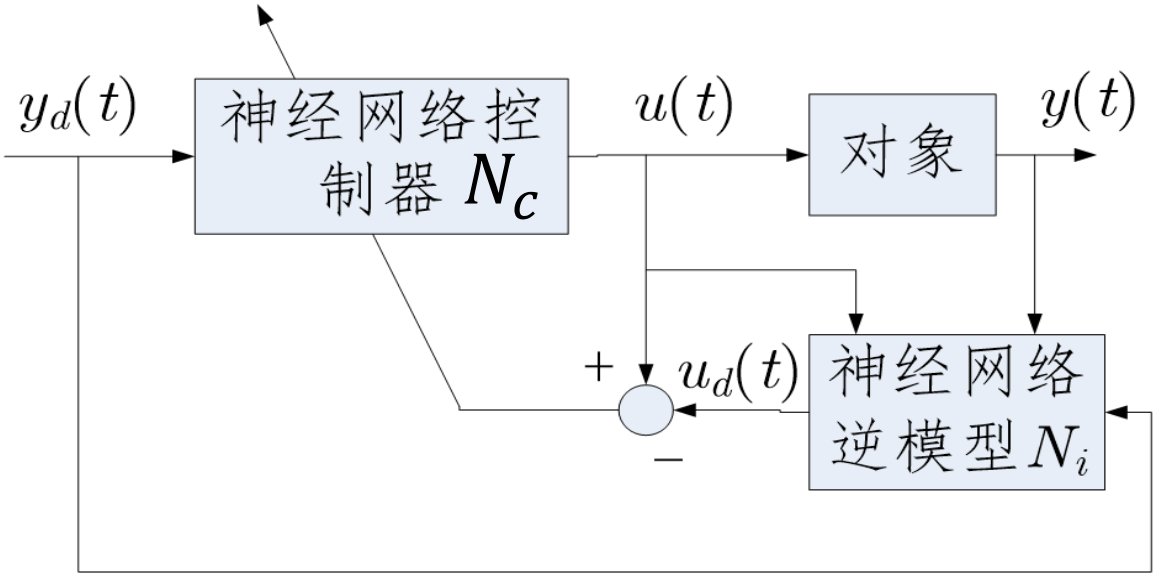


在神经网络控制器的学习过程中以及控制系统的实时控制过程中，辨识器N_i均可不断学习，以不断提高模型精度。

利用前向模型实现系统误差信息的反向传播，完成网络控制器的学习。

N_i的精度影响N_c的控制性能；N_c学习过程的收敛性和收敛速度有待深入研究。

逆向建模多神经网络学习



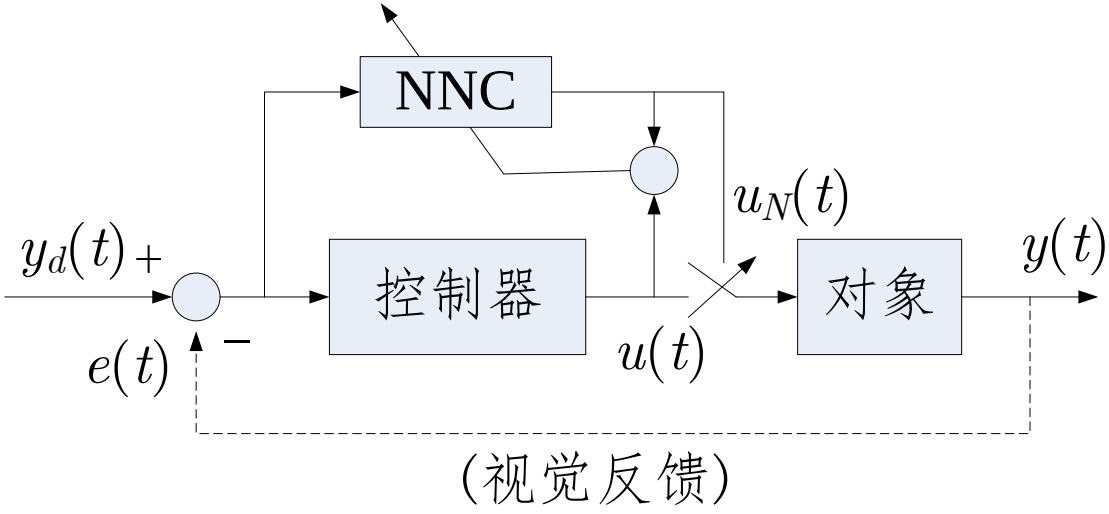
期望输出y_d作为逆模型的输入

- 产生期望的控制信号u_d
- 与实际的网络控制器N_c的输出信号u比较
- 产生的误差作为神经网络控制器N_c的学习信号。

神经网络监督控制

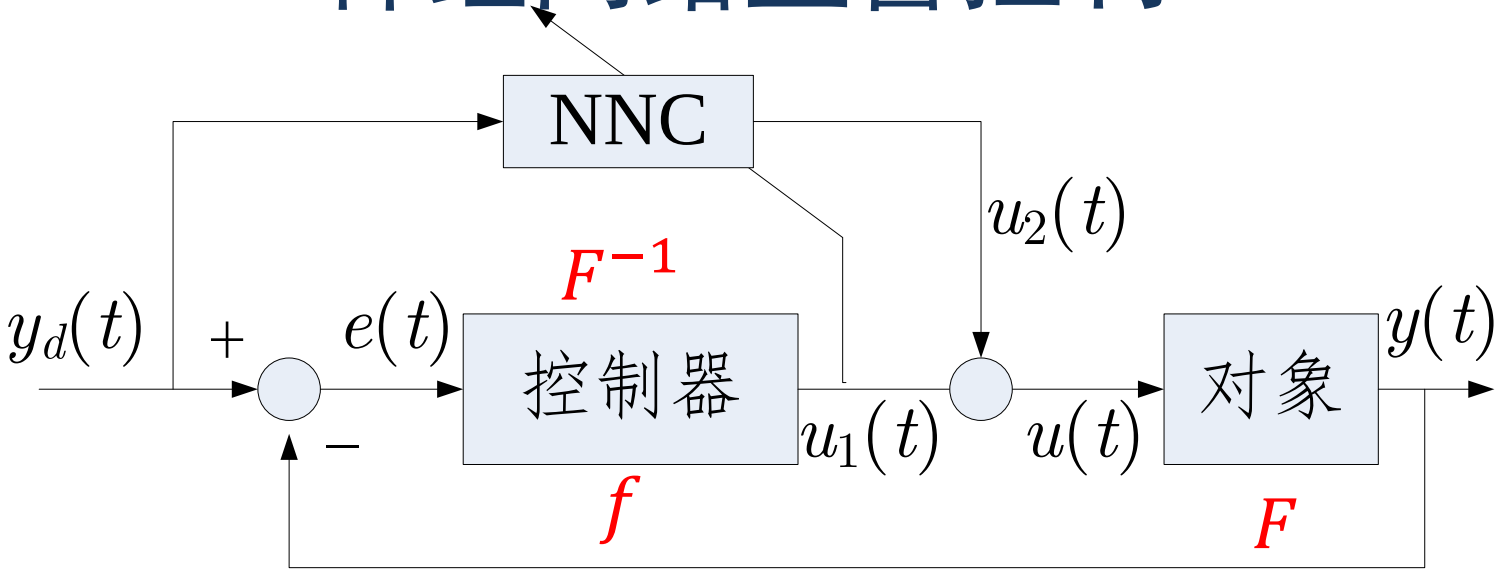
取代人工控制途径：

- **总结规则经验，构造专家或模糊系统**
- **知识难以表达时，用神经网络学习人的控制行为，对人工控制器建模，然后用神经网络取代**
- **实质：建立人工控制的正向模型**



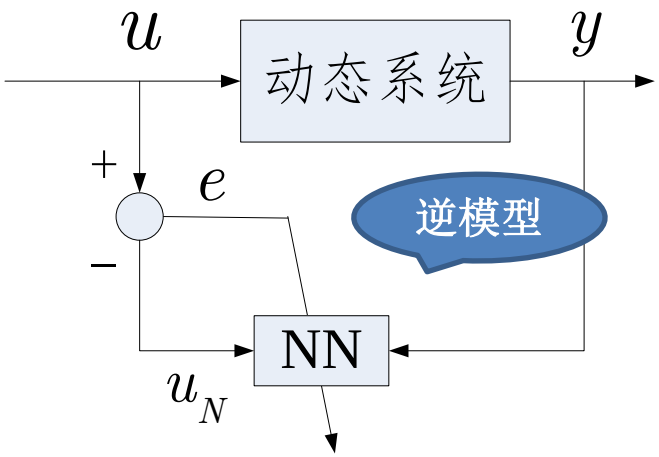
也叫神经网络COPY控制

神经网络监督控制

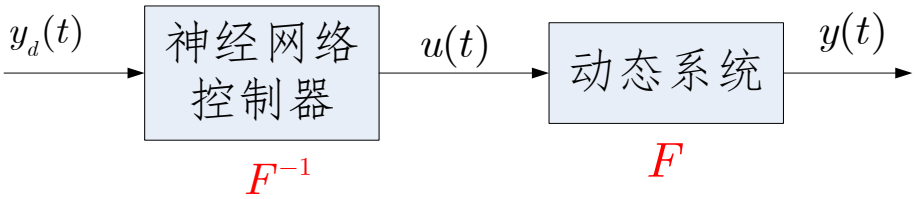


- 神经网络向传统控制器的输出学习，在线调整，目标使误差反馈或$e(t)$或$u_1(t)$趋于0
- 反馈控制器仍存在，一旦出现干扰，反馈控制器可重新起作
- 当$u_1 = 0$时，$y = F(u) = F(u_2) = F\left(F^{-1}(y_d)\right) = y_d$，此时再从反馈回路看，有：$e = y_d - y = 0$。

神经网络直接逆控制



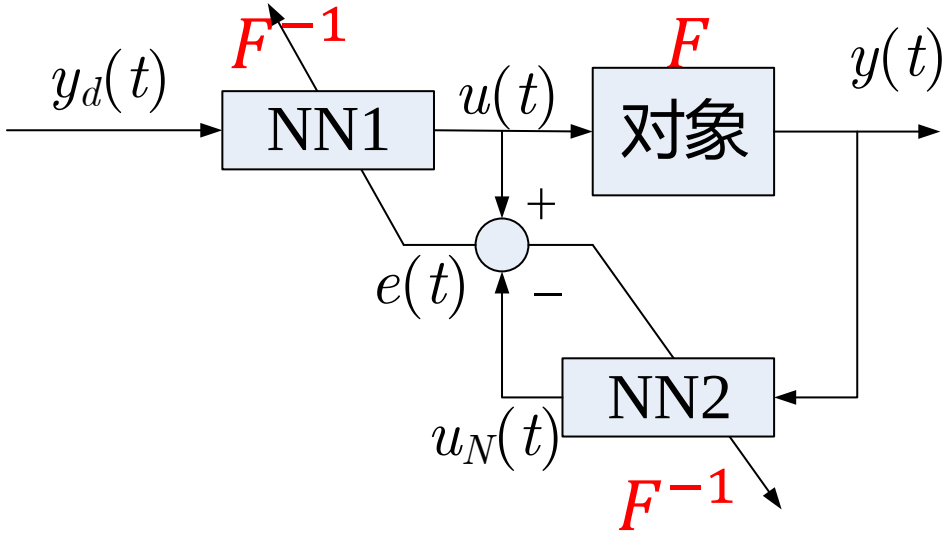
离线学习(建模)



在线控制

- 使期望输出与实际输出之间的传递函数等于1
- 前馈控制器
- 缺点：无反馈，用作控制器的神经网络逆模型不准确时，抗干扰能力差，缺乏鲁棒性

直接逆控制结构1



- 在开环结构的基础上增加神经网络2，以实现对于网络1权值的在线调整；
- 网络1和2映射特性相同（结构相同、权值相同）
- 系统通过偏差e调整两个网络的权值。当$e=0$时，网络具有对象的逆特性，因为此时意味着$u=u_N$，故

对网络2：　$u_N = F^{-1}(y)$；

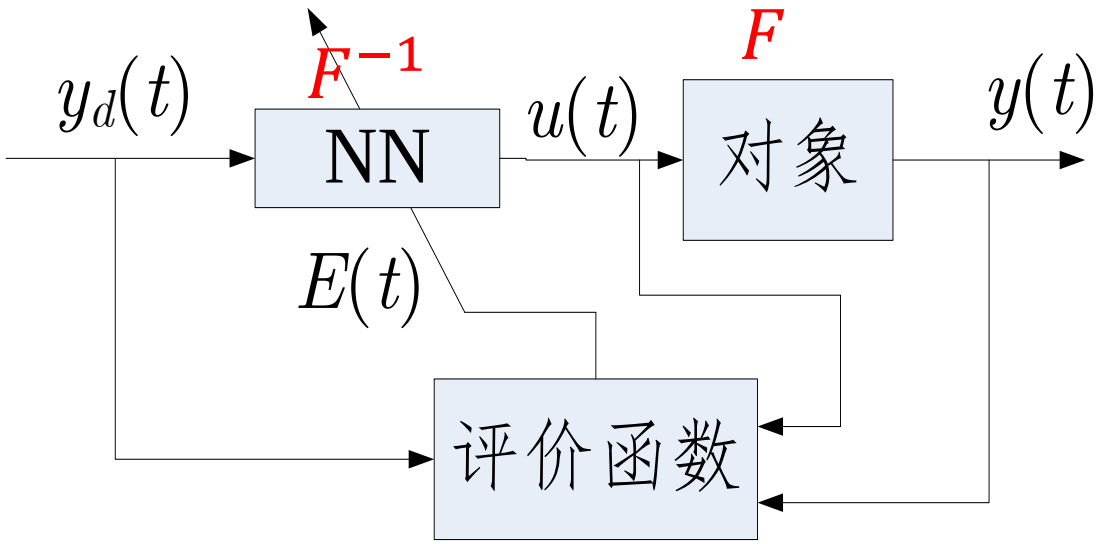
对网络1：　$u = F^{-1}(y_d)$；

$F^{-1}(y) = F^{-1}(y_d) \Rightarrow y = F(F^{-1}(y_d)) = y_d$。

神经网络直接逆控制

直接逆控制结构2

- 用评价函数$E(t)$作为性能指标，调整神经网络控制器的权值
- 当性能指标为0时，神经网络控制器即为对象的逆模型



神经网络自适应控制

自适应控制

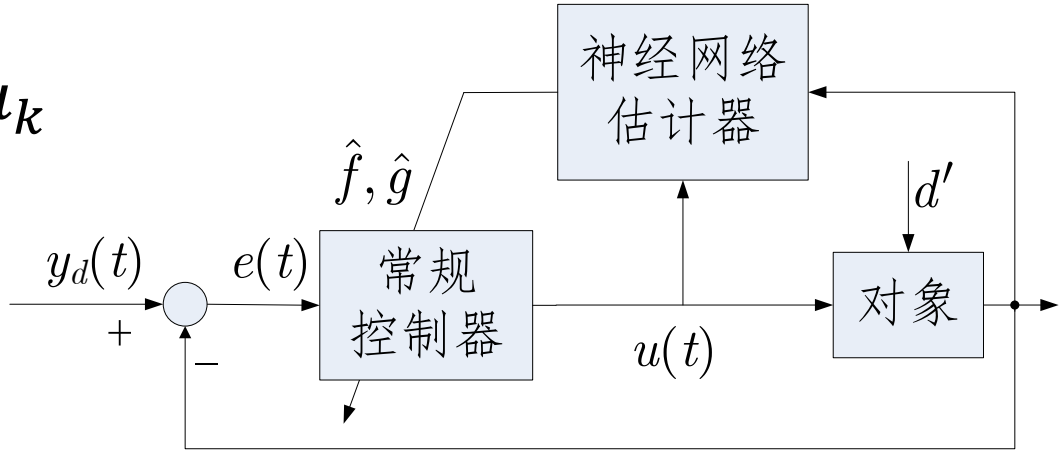
- 自校正控制
  - ✓ 直接自校正控制
  - ✓ 间接自校正控制
- 模型参考自适应控制
  - ✓ 直接模型参考自适应控制
  - ✓ 间接模型参考自适应控制

- **神经网络自校正控制**：根据系统正向或逆模型的输出结果调节神经或传统控制器的内部参数，使系统满足给定的指标。
  - **直接自校正**：调整的是神经网络控制器本身的参数，**本质等同于神经网络直接逆控制**。
  - **间接自校正**：同时使用常规控制器和神经网络估计器，神经网络估计器主要用来调整常规控制器的参数。

神经网络自适应控制

■ 间接自校正控制

- $y_{k+1} = f(y_k) + g(y_k)u_k$
- $u_k = \dfrac{y_{d,k+1} - \hat{f}(y_k)}{\hat{g}(y_k)}$



- 或利用神经网络估计系统响应的特性参数，如上升时间t_s、超调量σ，或二阶系统的自然振荡频率ω_n及阻尼系数ξ等，再用常规的极点配置等方法调整控制器参数

神经网络自适应控制

■模型参考自适应控制

- 稳定参考模型描述闭环系统的期望性能，参考模型定义为$\{r(t), y^m(t)\}$输入—输出对

$$\ddot{y}^m = -2\dot{y}^m - y^m + r$$

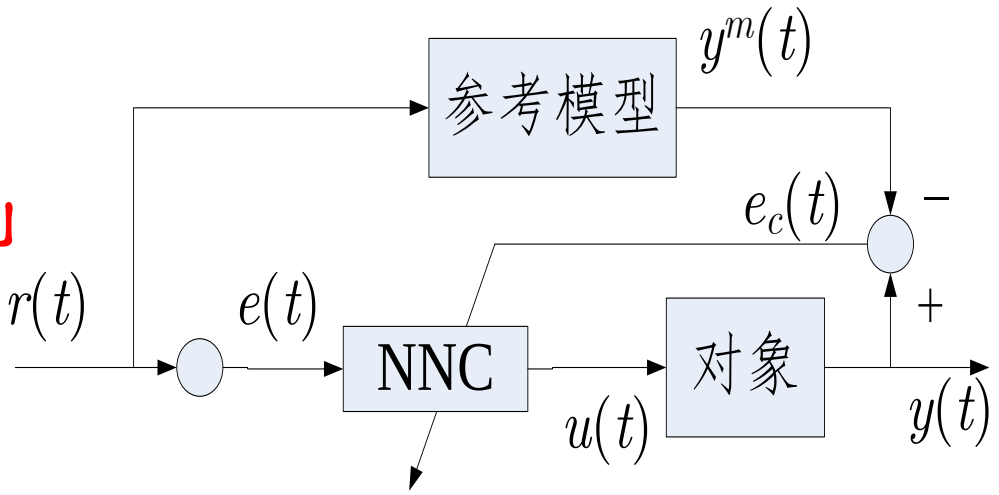
- 控制目标：使被控对象输出$y(t)$一致地趋于参考模型输出

$$\lim_{t\to\infty} \| y(t) - y^m(t) \| \le \varepsilon$$

神经网络自适应控制

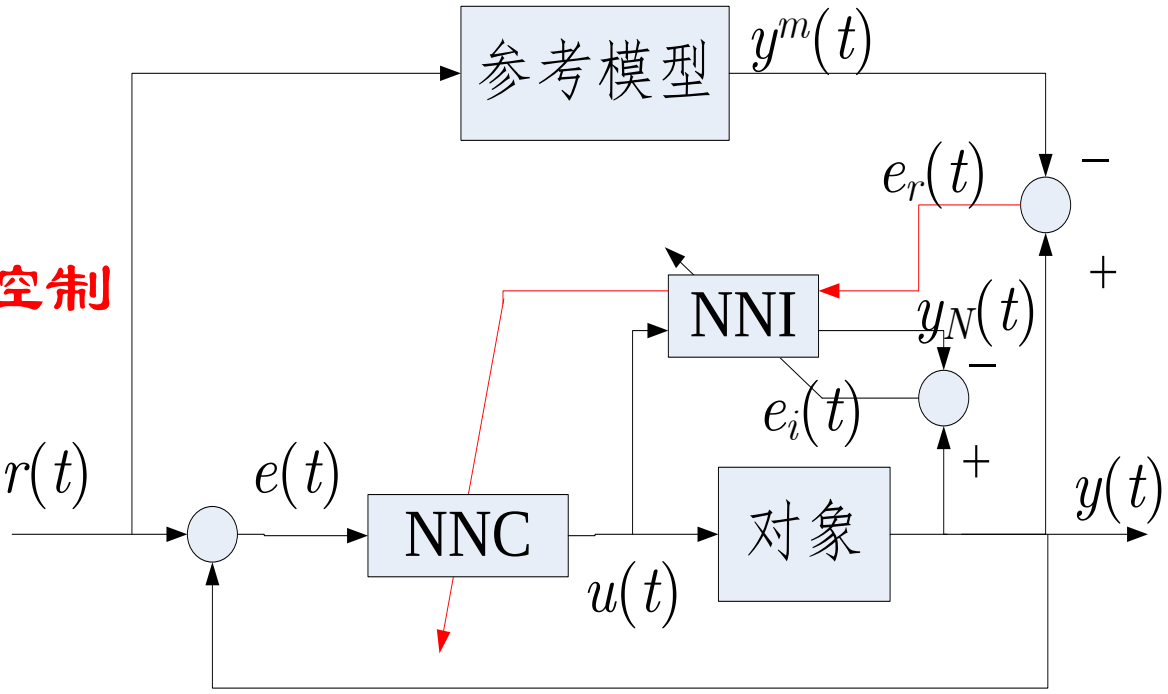
- 使被控对象与参考模型输出之差$e_c(t) = y(t) - y^m(t) \to 0$或$e_c(t)$的二次型最小
- 需要知道对象的数学模型(Jacobian信息$\frac{\partial y}{\partial u}$)才能通过误差反向传播算法修正网络权值，但对象通常含有未知参数

直接参考模型自适应控制



- NNI首先离线辨识被控对象的正向模型，并可由$e_i(t)$进行在线学习
- NNI可为NNC提供误差$e_r(t)$或其梯度的反向传播通道
- 神经网络辨识器NNI向神经网络控制器提供对象的Jacobian信息

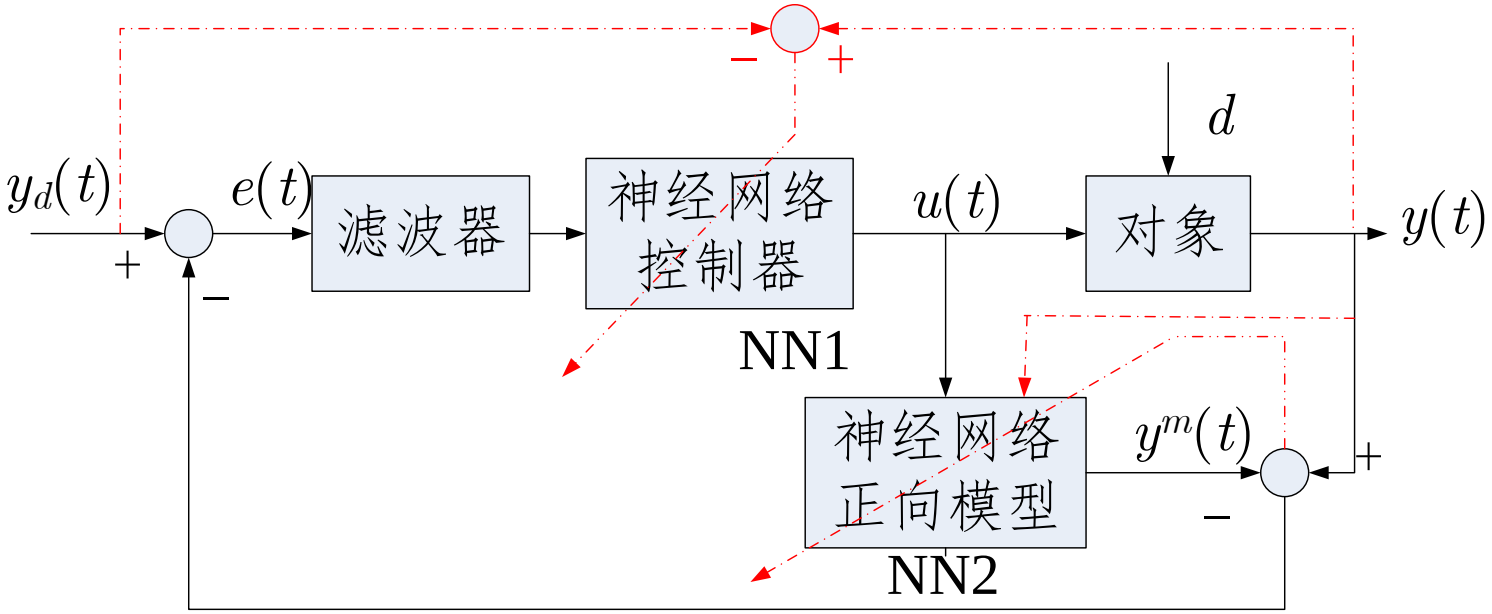
间接参考模型自适应控制



神经网络内模控制

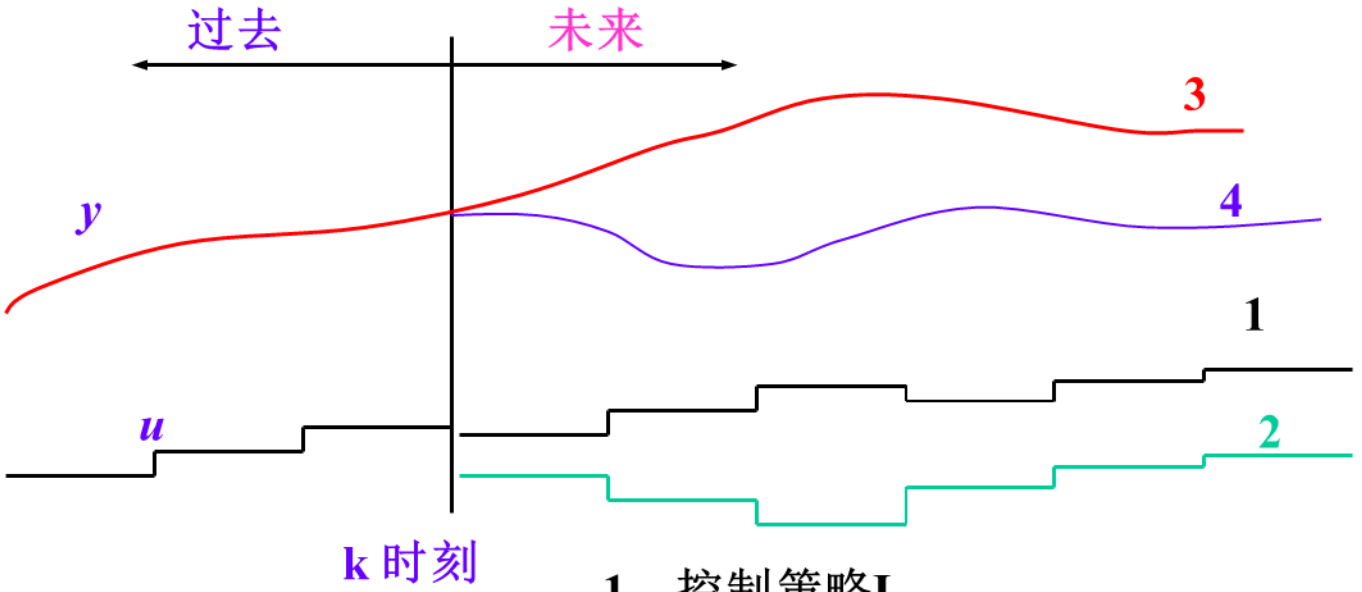
■经典内模控制

- 将被控系统的正向模型和逆模型直接加入反馈回路
- 正向模型作为被控对象的近似模型，与实际对象并联；
- 控制器与对象的逆有关，可以是对象的逆；
- 滤波器通常为线性的，可提高系统的鲁棒性。



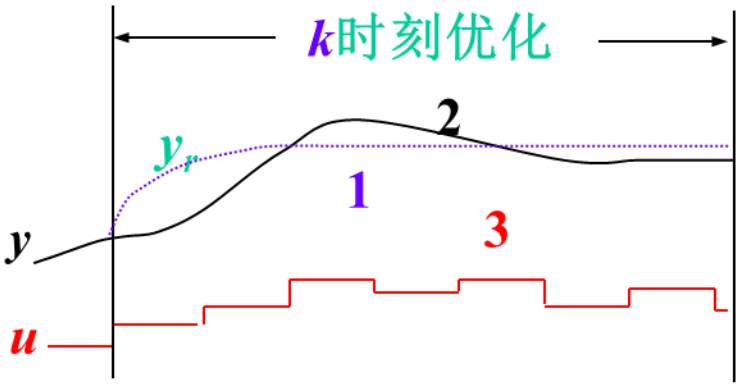
神经网络预测控制

预测模型



1—控制策略I
2—控制策略II
3—对应于控制 策略I的输出
4—对应于控制策略II的输出

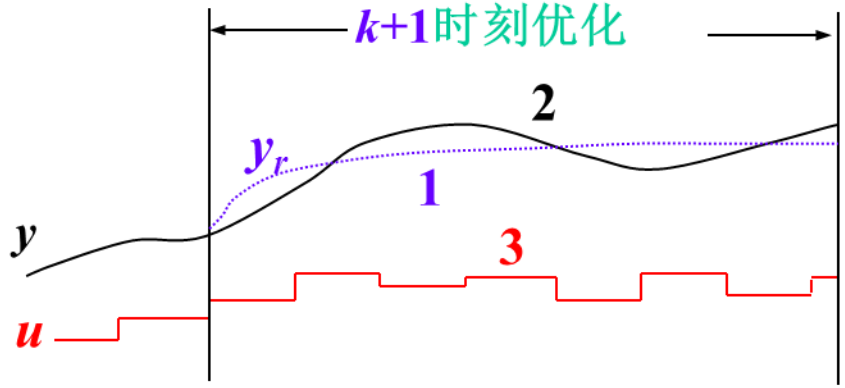
神经网络预测控制



1—参考轨迹y_r(虚线)

2—最优预测输出y(实线)

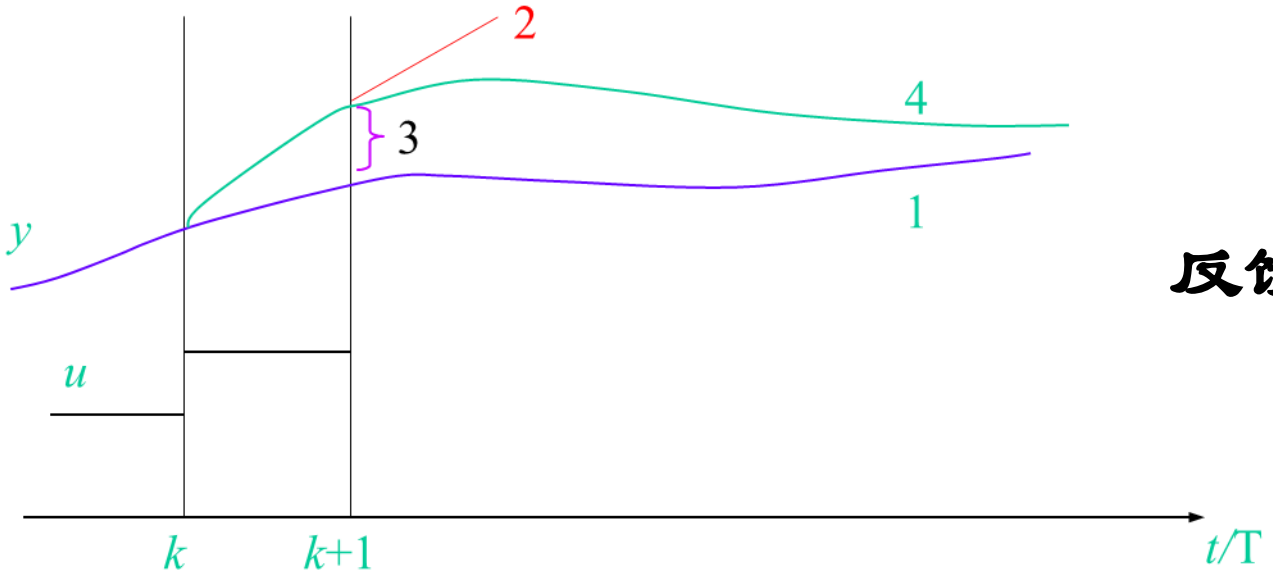
3—最优控制作用u



滚动优化

神经网络预测控制

误差校正示意图



反馈校正

1—*k*时刻的预测输出
2—*k*＋ 1 时刻实际输出
3—预测误差
4—*k*＋ 1 时刻校正后的预测输出

神经网络预测控制

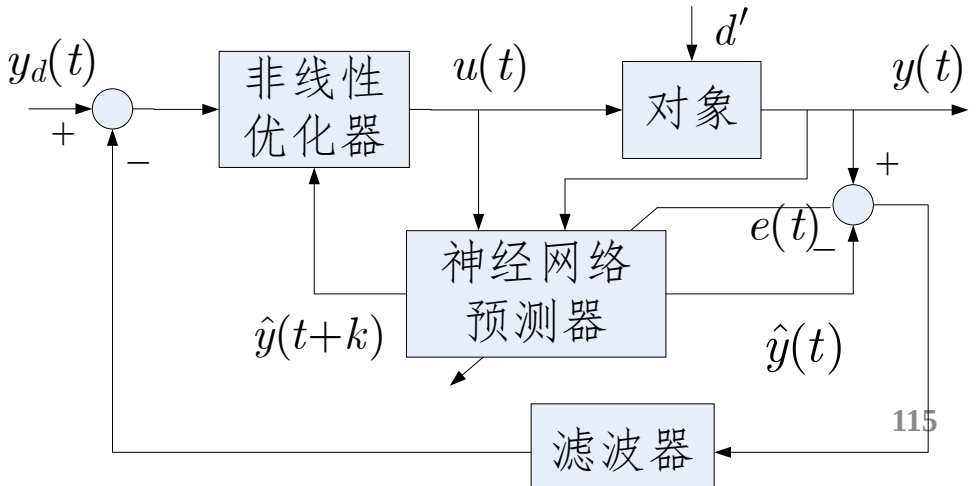
- 神经网络建立被控对象的预测模型，并在线学习修正

根据目前输入$u(t)$，预测未来输出$y(t+j|t)$ $j=N_1, N_1+1, \cdots, N_2$

$$e(t+j)=y_d(t+j)-y(t+j|t)$$

$$J=\frac{1}{2}\sum_{j=N_1}^{N_2}e^2(t+j)+\sum_{j=1}^{N_2}\lambda_j\Delta u^2(t+j-1)$$

$$\Delta u(t+j-1)=u(t+j-1)-u(t+j-2)$$



神经网络预测控制

1)计算未来期望输出序列$y_d(t+j)$, $j=N_1,N_1+1,\cdots,N_2$

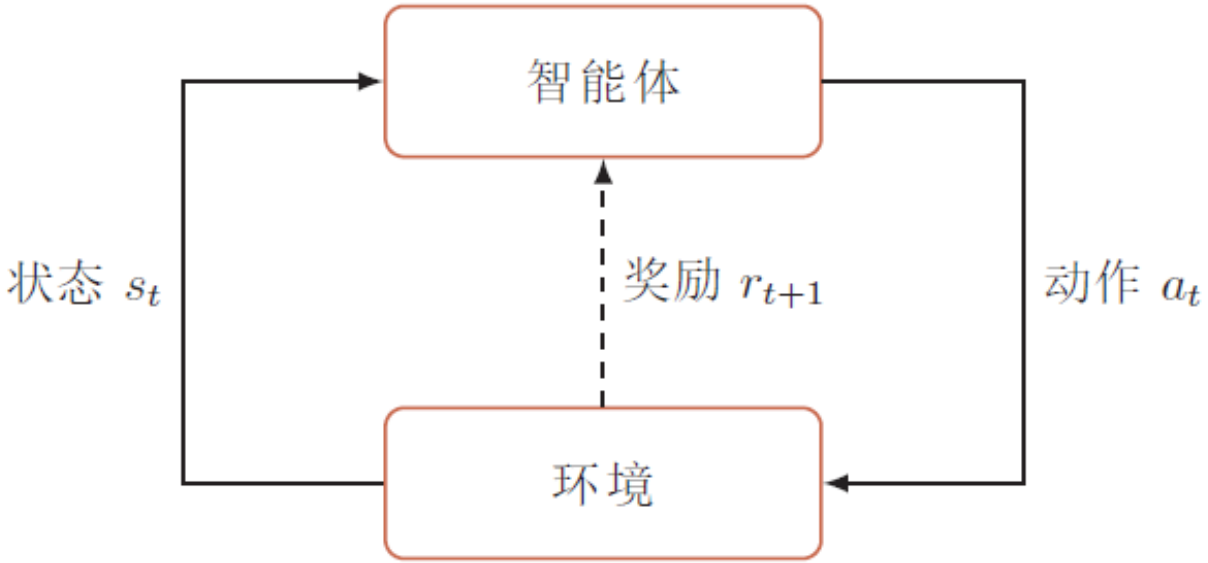
2)利用神经网络产生预报输出$y(t+j|t)$ $j=N_1,N_1+1,\cdots,N_2$

3)计算预报误差$e(t+j)=y_d(t+j)-y(t+j|t)$, $j=N_1,N_1+1,\cdots,N_2$

4)极小化性能指标J,获得控制序列$u(t+j)$ $j=0,1,\cdots,N_2$

5)采用第一控制量$u(t)$,然后返回 (1)

神经网络自适应评价控制



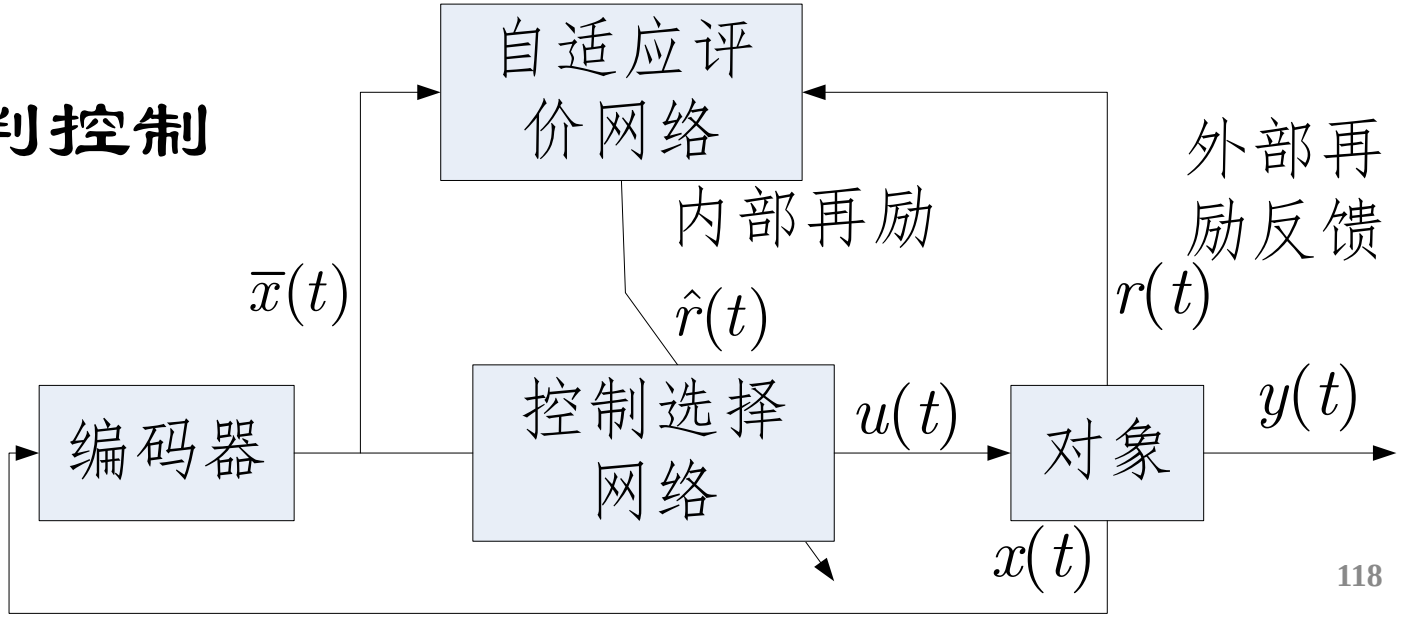
行动—评价—改进行动方案—再行动

（增强学习、强化学习、再励学习）

- 自适应评价网络
  - ✓ 通过不断奖励、惩罚等再励学习，使自己逐渐成为一个合格的教师；
  - ✓ 在学习完成后，根据被控系统目前状态和外部再励反馈信号$r(t)$，产生再励信号$p(t)$，给出内部激励信号$\hat{r}(t)$，以期对目前控制作用的效果作出评价
- 控制选择网络
  - ✓ 在内部再励学习信号指导下进行学习的多层前馈神经网络控制器
  - ✓ 根据编码后的系统状态，在允许控制集中选择下一步控制作用

神经网络

自适应评判控制



神经网络混合控制

集成人工智能各分支的优点，由神经网络技术与模糊控制、专家系统等相结合而形成的一种具有很强学习能力的智能控制系统。神经网络混合控制可使控制系统同时具有学习、推理和决策能力。

模糊神经网络控制

神经网络专家系统

直接逆模型神经网络控制器的设计实例

基本思想： 假设被控系统可逆，通过离线建模得到系统的逆模型网络，然后用这一逆模型网络作为控制器去直接控制被控对象

SISO系统模型：

$$y(k+1) = f[y(k), y(k-1), \cdots, y(k-n+1); u(k), u(k-1), \cdots, u(k-m+1)]$$

逆系统：

$$u(k) = f^{-1}[y(k+1), y(k), y(k-1), \cdots, y(k-n+1); u(k-1), \cdots, u(k-m+1)]$$

用期望值$y_d(k+1)$代替k时刻的未来值$y(k+1)$，得动态系统的逆模型：

$$u(k)=f^{-1}[y_d(k+1),y(k),y(k-1),\cdots,y(k-n+1);\ u(k-1),\cdots,u(k-m+1)]$$

将右边的$n+m$个时刻的输入输出值作为前向网络的$n+m$个输入、并记为X，将左边的输入值$u(k)$作为网络的期望输出，则网络具有$n+m$个输入神经元和1个输出神经元。

离线建模，使用批处理训练方式。

假定：已测得动力系统的以下数据序列

$$\{(y(k),u(k-1)),(y(k-1),u(k-2)),\cdots,(y(k-n-P+1),u(k-n-P))\}$$

则可构造出到k时刻为止网络的P组输入样本：

$$X_p(k)=[y(k-p+1),y(k-p),\cdots,y(k-n-p+1),u(k-p-1),\cdots,u(k-m-p+1)]^T$$

网络的期望输出为：

$$u_p(k)=u(k-p)$$

其中，$p=1,2,\cdots,P$。

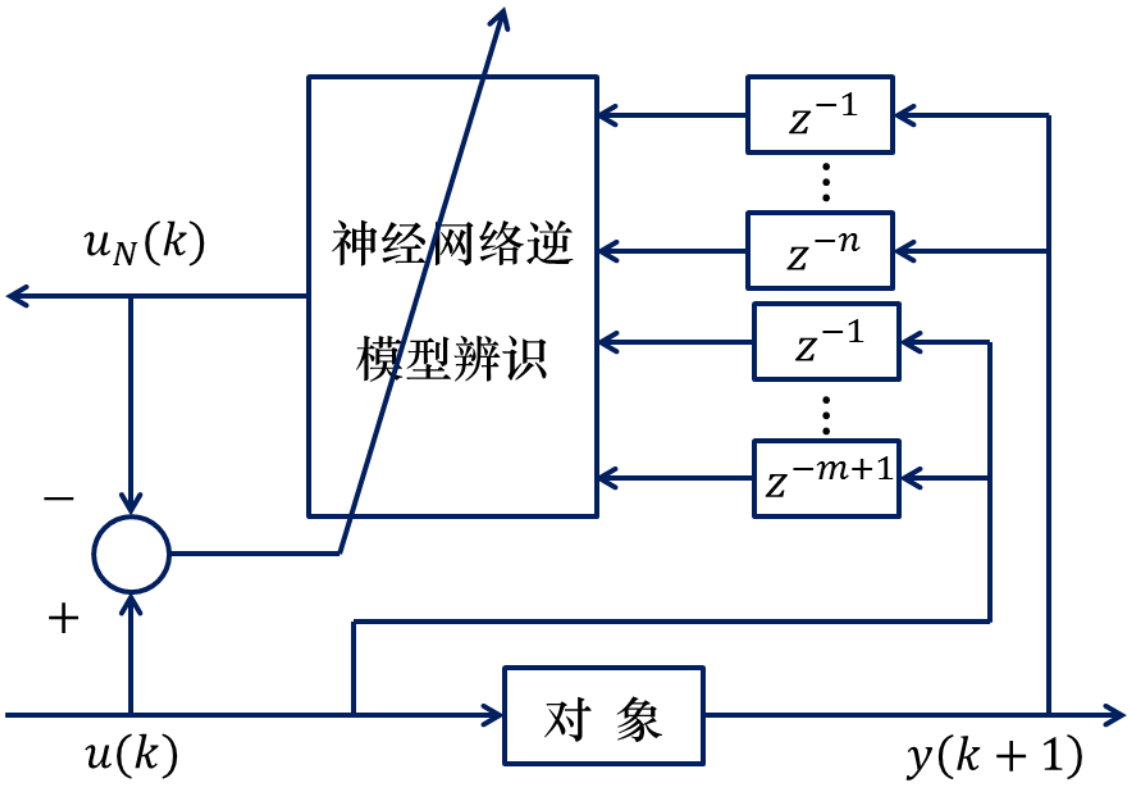
性能指标取为：

$$E(k)=\frac{1}{2}\sum_{p=1}^{P}\lambda_p(u(k-p)-u_N(k-p))^2$$

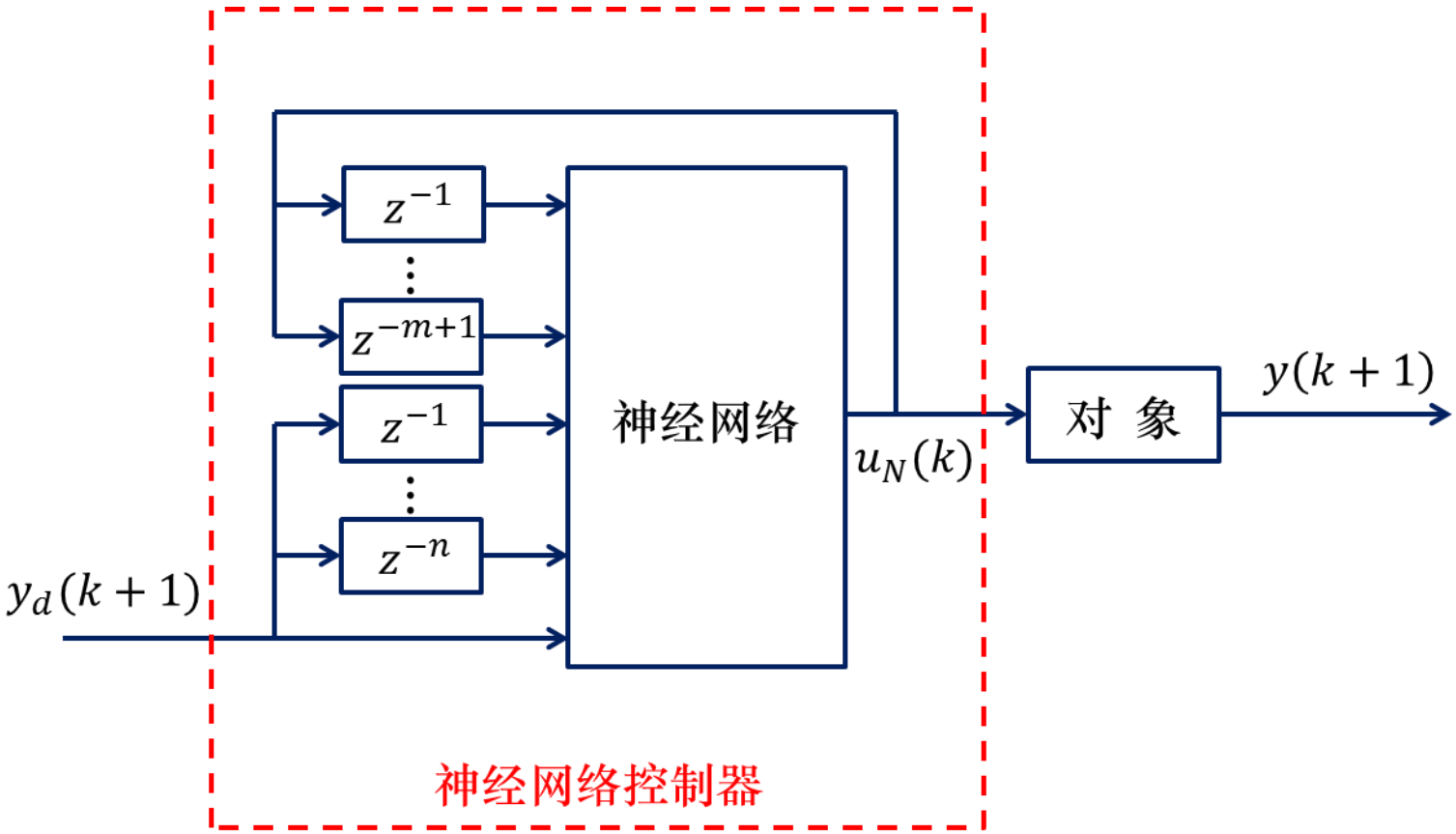
其中，$0\le\lambda_1\le\lambda_2\le\cdots\le\lambda_P\le 1$。

样本数量P决定了学习中所涉及到的离散时刻数；上式表示希望使网络的实际输出能同时在P个时刻上都尽可能接近系统的实际输入(导师信号)。

直接利用BP算法即可离线训练出网络的权值



建立逆模型时神经网络的训练示意图



直接逆模型网络控制器的控制系统结构示意图

（运行于静态参数环境）

引入在线学习机制调整神经网络控制器的权值，提高自适应能力。

例： 考虑被控系统

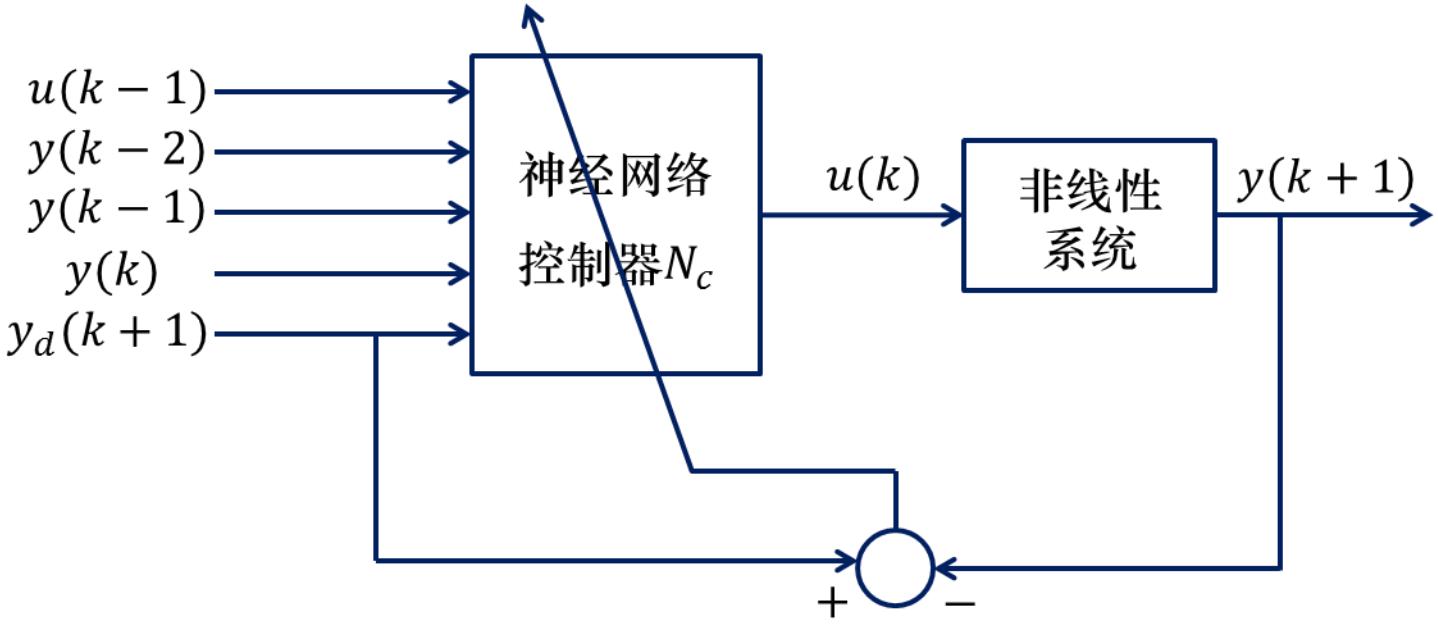
$$y(k+1)=\frac{y(k)y(k-1)y(k-2)u(k-1)(y(k-1)-1)+u(k)}{1+y^2(k-1)+y^2(k-2)}$$

假设动力学逆模型成立，为

$$u(k)=g[y(k+1),y(k),y(k-1),y(k-2),u(k-1)]$$

试用直接神经网络控制器进行控制。

构造神经网络结构为5-25-12-1型；直接神经网络控制器的系统结构图为：



输出单元线性激励，其余层单元Sigmoid型激励；

控制性能指标：$E_p(k)=\frac{1}{2}\left(y_d(k)-y_p(k)\right)^2$

学习规则：

$$w_{ji}^{(r)}(k+1)=w_{ji}^{(r)}(k)+\eta\delta_{pj}^{(r)}o_{pi}^{(r)}$$

$$\delta_{pj}^{(L)}=(y_d(k)-y_p(k))\frac{dy_p(k)}{du_p(k-1)}$$

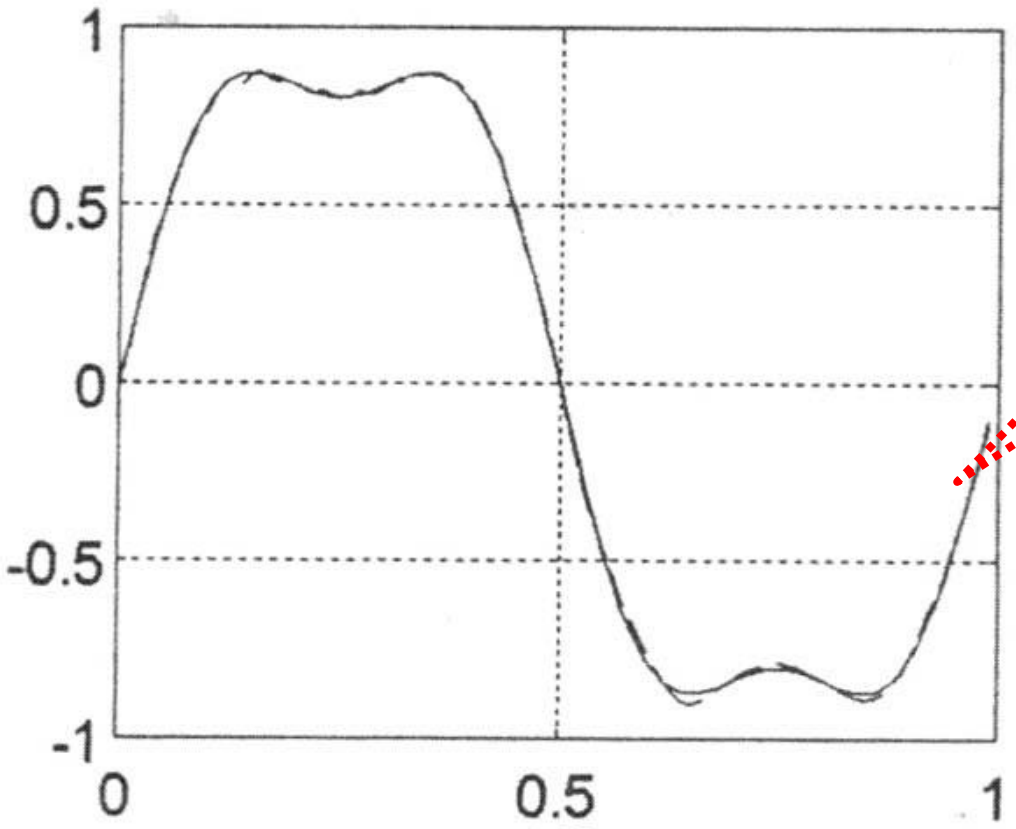
$$=(y_d(k)-y_p(k))\frac{1}{1+y_p^2(k-2)+y_p^2(k-3)}$$

$$\delta_{pj}^{(r)}=o_{pj}^{(r)}\left(1-o_{pj}^{(r)}\right)\sum_l\delta_{pl}^{(r+1)}w_{lj}^{(r+1)}$$

取：$\eta=0.05$；

期望输出：$y_d(k)=\sin\frac{2\pi k}{100}+0.2\sin\frac{6\pi k}{100}$；

仿真结果：



100个周期的学习后，均方误差小于0.005

Jacobian矩阵的替代方法：

(1) 摄动法

用$\frac{\Delta y}{\Delta u}$代替$\frac{\partial y}{\partial u}$，采样周期短时可行。

(2) 符号函数法

用$sgn\left(\frac{y}{u}\right)$代替$\frac{\partial y}{\partial u}$，简单实用。

对大多数系统，输出随输入变化的趋势易于知道。

(3) 前向神经网络建模仿真法

利用另一神经网络、以仿真方式对系统进行正向建模，得到系统的Jacobian矩阵信息。

(4) 多神经网络自学习控制法

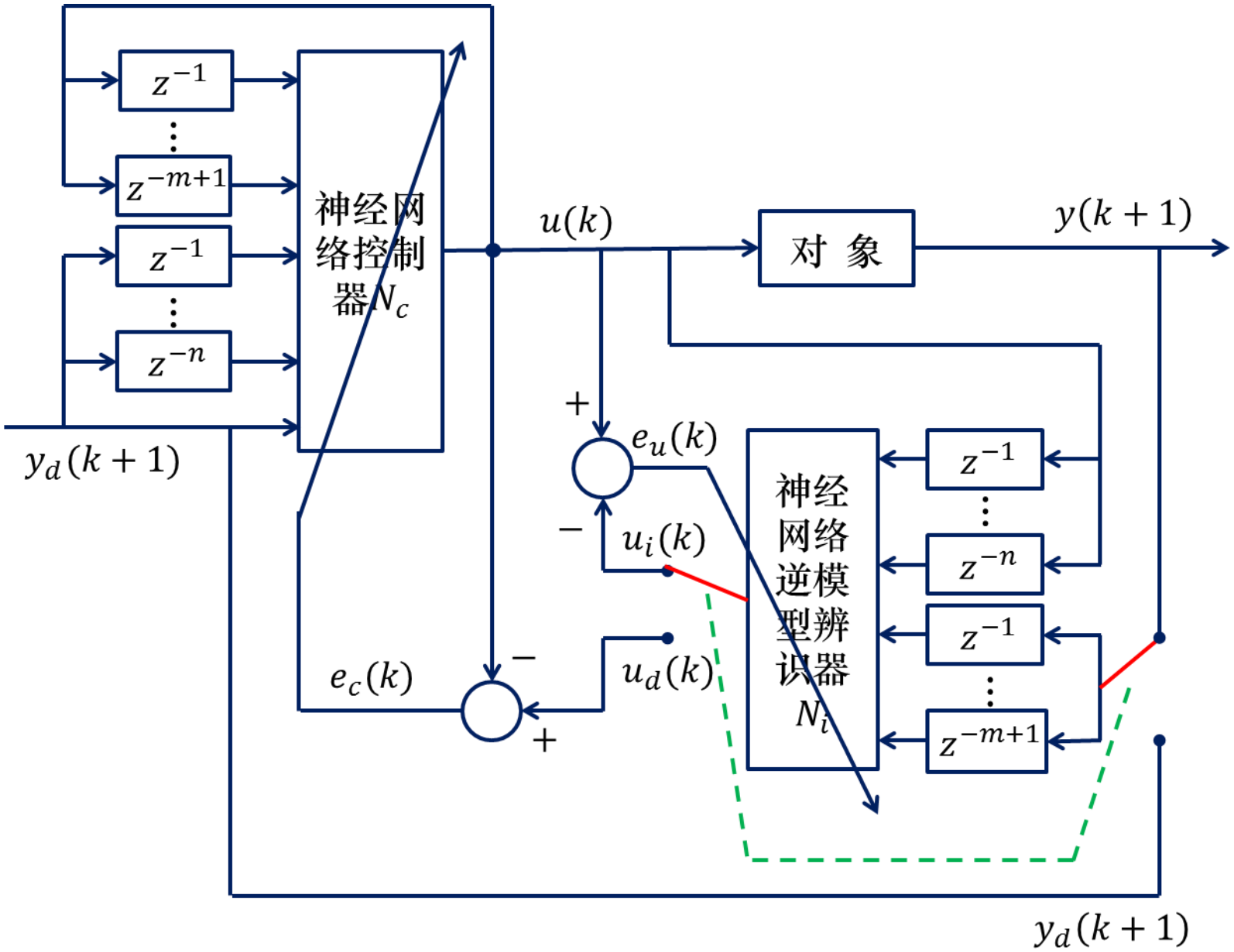
利用另一神经网络(辨识器)在线建立系统的逆模型，并利用期望输出产生期望的控制信号作为导师信号，实现神经网络控制器的有导师学习。

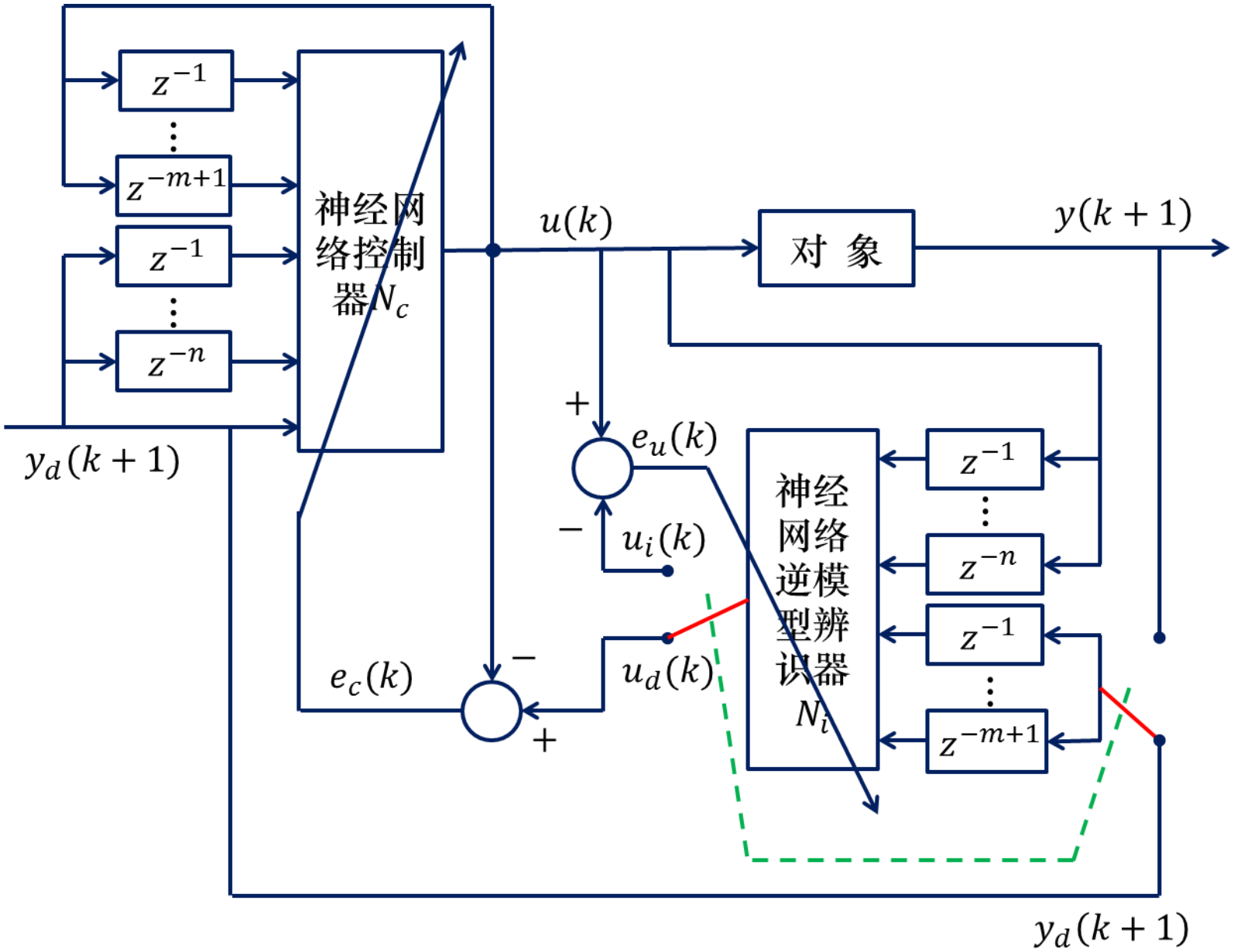
多神经网络自学习控制法基本思想：

利用未知系统的逆模型和系统的期望输出$y_d(k+1)$构造一个期望的控制量$u_d(k)$，以解决神经控制器N_c在系统模型未知情况下学习的无导师问题。

原理（结构图见下页）：

先将神经网络辨识器N_I进行离线训练，建立未知对象的大概模型，然后连入系统、并和神经网络控制器N_c一块进行在线训练；在每个采样周期内，首先将开关合上，利用误差信息$e_u(k)$对N_I的权系数学习并调整一次，然后将开关拉下利用误差信息$e_c(k)$对N_c的权系数学习并调整一次；如此往复不断，直到系统的实际输出$y(k+1)$能以期望精度跟踪期望输出$y_d(k+1)$为止。


z^{-1}
z^{-m+1}
z^{-1}
z^{-n}
神经网络控制器N_c
$y_d(k+1)$
$u(k)$
对 象
$y(k+1)$
+
$e_u(k)$
−
$u_i(k)$
神经网络逆模型辨识器N_i
z^{-1}
z^{-n}
z^{-1}
z^{-m+1}
$e_c(k)$
−
+
$u_d(k)$
$y_d(k+1)$


z^{-1}
z^{-m+1}
z^{-1}
z^{-n}
神经网络控制器N_c
$y_d(k+1)$
$u(k)$
对 象
$y(k+1)$
+
$e_u(k)$
−
$u_i(k)$
神经网络逆模型辨识器N_i
z^{-1}
z^{-n}
z^{-1}
z^{-m+1}
$e_c(k)$
−
+
$u_d(k)$
$y_d(k+1)$

学习规则（结合结构图）：

N_I（用作辨识器建立未知对象的逆模型）：

$$e_u(k) = u(k) - u_I(k)$$

$$\delta_{pj}^{I(L)} = e_{uj}(k)$$

$$\delta_{pj}^{I(r)} = o_{pj}^{I(r)}\left(1 - o_{pj}^{I(r)}\right)\sum_l \delta_{pl}^{I(r+1)} w_{lj}^{I(r+1)}$$

$$\Delta w_{jl}^{I(r)}(k) = \eta_I \delta_{pj}^{I(r)} o_{pl}^{I(r-1)} + \alpha_I \Delta w_{jl}^{I(r)}(k-1)$$

其中，$e_{uj}(k)$为矢量$e_u(k)$的第j个分量。

N_c（辨识器N_I提供期望的控制量）：

$$e_c(k) = u_d(k) - u(k)$$

$$\delta_{pj}^{c(L)} = e_{cj}(k)$$

$$\delta_{pj}^{c(r)} = o_{pj}^{c(r)}\left(1 - o_{pj}^{c(r)}\right)\sum_l \delta_{pl}^{c(r+1)} w_{lj}^{c(r+1)}$$

$$\Delta w_{jl}^{c(r)}(k) = \eta_c \delta_{pj}^{c(r)} o_{pl}^{c(r-1)} + \alpha_c \Delta w_{jl}^{c(r)}(k-1)$$

其中，$e_{cj}(k)$为矢量$e_c(k)$的第j个分量；

η_I, η_c分别为两个网络的学习因子；α_I, α_c为动量因子。

例： 对上例中的系统，试用多神经网络自学习控制方法来设计此系统的非线性控制器，要求系统的期望输出为：

$$y_d(k) = \sin\frac{2\pi k}{100}, k = 0,1,\cdots,100$$

解：

假定系统的逆动力学存在，即

$$u(k) = g(y(k+1), y(k), y(k-1), y(k-2), u(k-1))$$

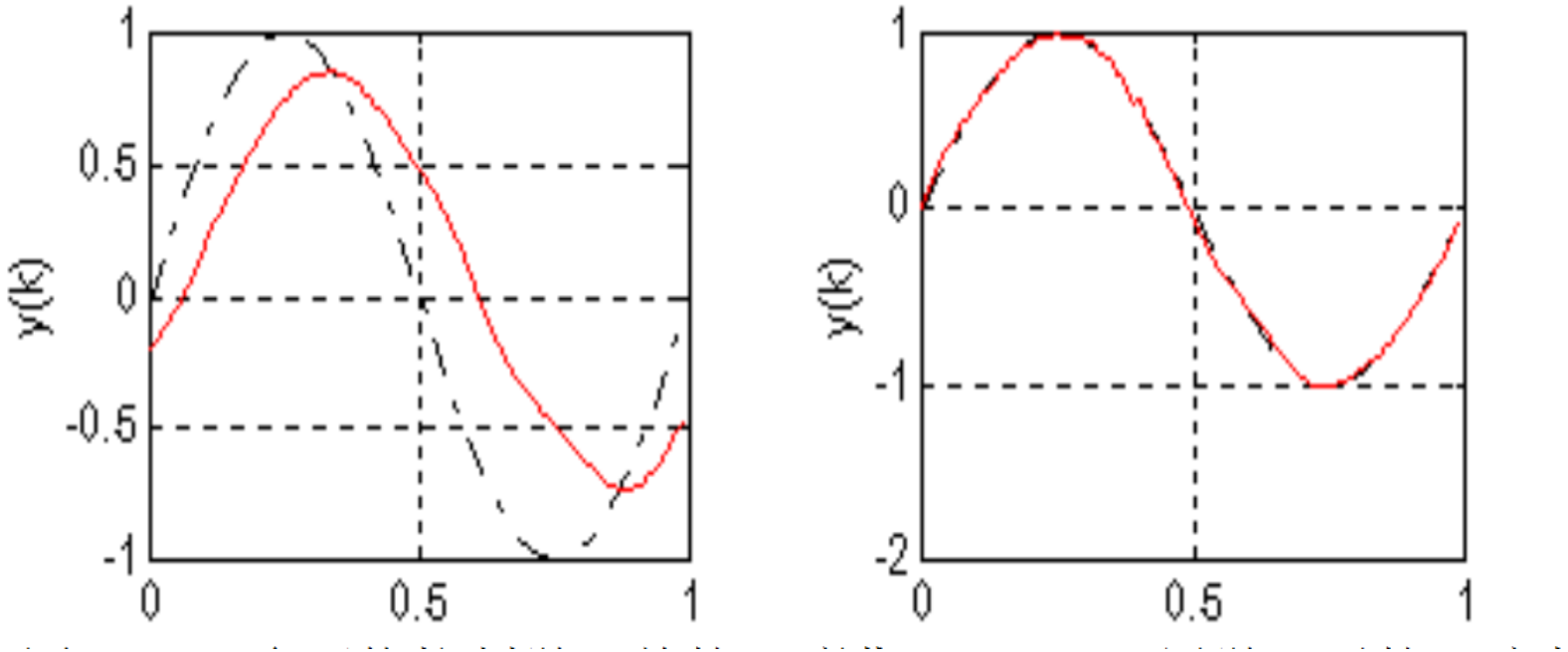
两个前向网络非别选择为

$$N_i: \ 5\text{-}25\text{-}12\text{-}1$$

$$N_c: \ 6\text{-}20\text{-}10\text{-}1$$

输出神经元均为线性激励函数。

根据前面的工作原理和学习算法，可以得到仿真结果为



100个周期的学习效果，误差<1%

多神经网络自学习控制特点概括：

① 边控制边辨识。利用实际系统的输入输出信息更新网络辨识器的权系数，不断提高对被控对象的准确识别；利用期望输出y_d、经神经网络逆模型得到期望的控制量，使得基于Delta学习规则的神经网络控制器的权系数调节得以实现。

② 整个系统具有在线辨识、实时控制的能力，能满足环境或系统参数变化情况下的控制性能，具备良好的自适应、自学习能力。

③ 缺点是要求被控对象的动力学特性是可逆的。

发展历史

- 1963年Widrow和Smith提出的动平衡器(broom balancer)：第一个神经网络控制器；由自适应线性单元Adline构成，用于学习并产生一条开关曲线，控制器的左右是离散的，可产生二元控制作用

- 20世纪70年代，Albus提出一种模仿小脑如何控制机体运动机理的控制模型，成为小脑模型关节控制器，简称CMAC，并将其用于机械手的运动控制
- 20世纪80年代，CMAC应用于高度非线性的化工过程及机械手的建模与控制

早期发展历史

- 20世纪80年代研究了再励学习和自适应评判方案，集成和扩展了神经网络学习算法，递归神经网络也用于优化控制方案以及对象的建模和估计

- 20世纪80年代，美国物理学家Hopefield提出了Hopefield人工神经网络（1982）
- 1986年Rumlhart和McClelland于1986年提出了多层神经网络的反向传播（Back propagation）学习算法，简称BP算法

早期发展历史

- 1989年Cybenko等从理论上证明对于任何闭区间的连续函数都可以用含一个隐层的多层前向神经网络来逼近

- 基于多层前向神经网络的反传学习，提出了多种神经网络建模和控制结构；神经网络建模主要分为间接学习、一般学习、特殊学习三种结构[Psaltis, et al., 1988]；神经网络控制的主要结构[Hunt et a1.,1992]分为模型参考结构、内模控制结构、预测控制结构、最优控制结构、前馈网络控制结构、自校正控制结构[Chen, 1990]等

早期发展历史

- 改进了多层前向神经网络的学习算法，加快训练速度。主要改进方法包括引入动量项、变尺度法、变步长法、共轭梯度法、随机逼近算法、扩展卡尔曼滤波算法、模糊逻辑、进化算法和变结构等

- 20世纪90年代是神经网络控制理论与应用蓬勃发展的时期
- 1991年Sanner和Slotine及Polycarpou等提出了稳定的神经网络自适应控制；这方面早期的研究大多基于线性参数化的神经网络，如径向基函数神经网络、CMAC神经网络、B样条网络，小波神经网络和某类模糊神经网络等

早期发展历史

- 基于神经网络和模糊逻辑系统对非线性函数逼近的等效性，提出了许多神经模糊((Neuro fuzzy)自适应控制方法 [Brown &Harris,1994; Spooner & Passino, 1996,1997]

- 神经网络与变结构控制

a)利用神经元网络的优化特性实施变结构控制[Li & Zeng, 1992] ;

b)在神经元控制中引入变结构控制，克服神经元网络学习陷入局部极小 [Makki & Siy, 1992];

c)在神经元网络控制的基础上，利用变结构控制保证系统的全局稳定性 [Sanner & Slotine, 1991]

神经网络控制局限

- 神经网络本身的研究，如网络结构未有根本突破，专门适用于控制问题的动态神经网络仍有待于进一步发展
- 神经网络泛化能力不足，制约了控制系统鲁棒性
- 网络本身黑箱式的知识表达方式，使其不能利用初始经验进行学习，易于陷入局部极小值
- 分布式并行计算的潜力还有待于硬件技术的进步
- 神经网络虽有很强的学习能力，但就其本质来说，一般只能模拟人类低层次的感知智能，仅有较低的智能

参考书目

Simon S Haykin. Neural networks a comprehensive foundation. Prentice Hall (2008)

G. P. Liu et. al. Nonlinear Identification and Control A Neural Network Approach. Springer-Verlag London (2001)

孙增圻等. 智能控制理论与技术.清华大学出版社,2004

何玉彬等. 神经网络控制技术及应用.科学出版社,2000

张乃尧等. 神经网络与模糊控制.清华大学出版社,1996

史忠科. 神经网络控制理论. 西北工业大学出版社,2000

徐丽娜等. 神经网络控制,电子工业出版社, 2003

喻宗泉等.神经网络控制.西安电子科技大学出版社， 2009

思考

- 人工神经元激活函数的作用是什么？
- 人工神经网络的学习规则有哪些？各有何特征？
- 如何有效地训练一个神经网络？
- BP学习算法学习率的考虑因素？调整方式？
- 常用传统辨识算法有哪些？神经网络辨识相对于传统算法有何优势？
- 完成系统辨识需具备哪些条件？辨识采用什么准则？辨识系统的输入信号一般应具备什么特征？如何设计？
- 系统可辨识性与可控性、客观性的关系？只用输入输出信号能否辨识系统内部模型的全部参数？
- 何如合理选择辨识模型，一般应考虑哪些因素？
- 直接逆控制方法优缺点？