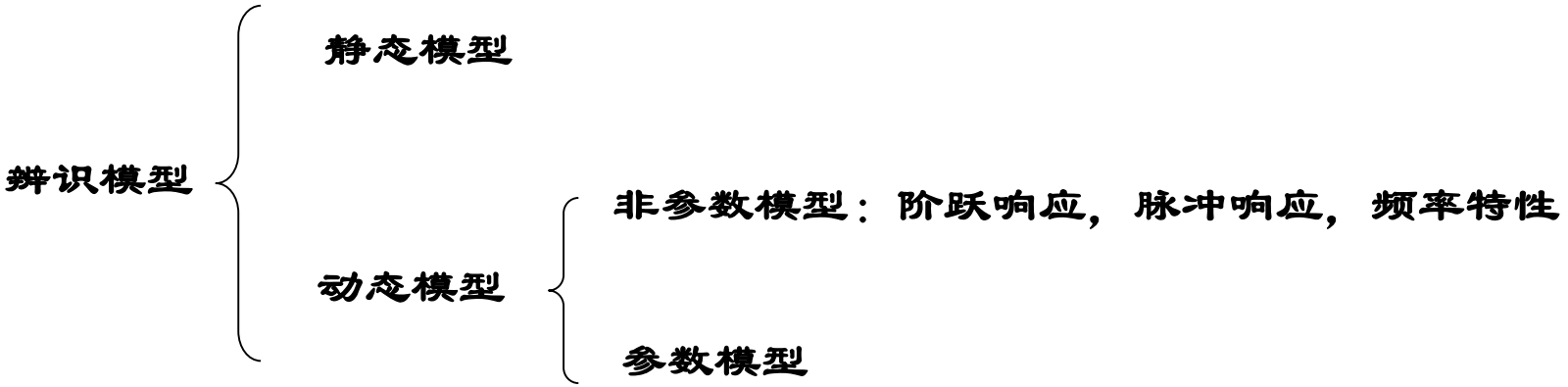
智能控制
第五章 神经控制

刘山
浙江大学控制科学与工程学院
2022/12/11

三、基于神经网络的系统建模与辨识

辨识三要素

- 辨识是在输入和输出数据的基础上，从一组给定的模型中，确定一个与所测系统等价的模型。
- 辨识三要素
  - 输入/输出数据：能够量测到的系统输入与输出。
  - 模型类：所考虑的系统的结构。
  - 等价原则：辨识的优化目标。



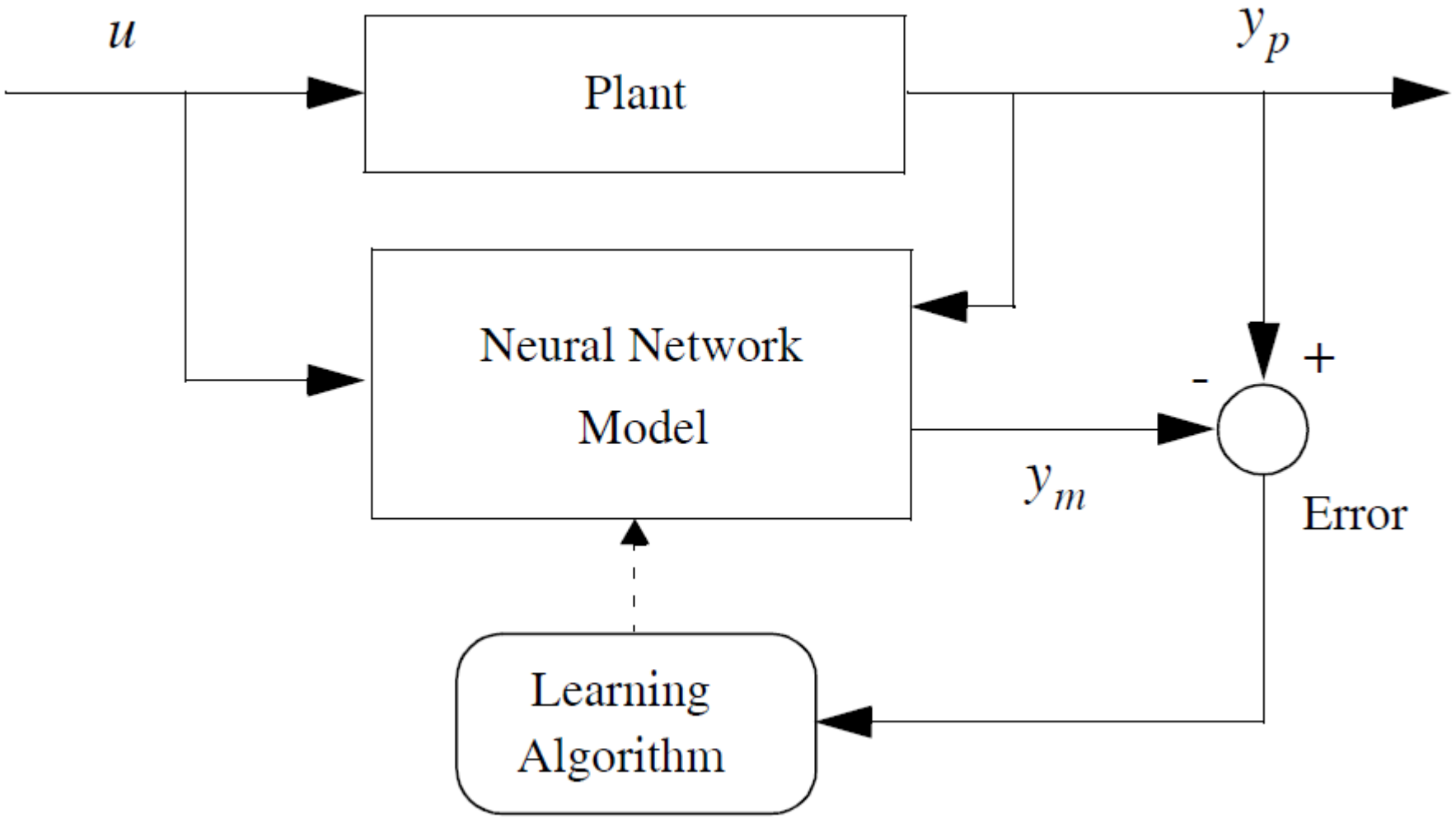
神经网络用于系统辨识

- 选择一个合适的神经网络模型来逼近实际的系统。
- 辨识准则为

$$\min_f \|e\| = \min_f \|\hat{z} - z\|$$

- $\hat{z}, z$分别是模型和系统对于系统输入的输出响应。

神经网络建模与辨识


u
Plant
y_p
Neural Network
Model
-
+
y_m
Error
Learning
Algorithm

神经网络建模需要考虑的问题

- （1）模型的选择
  - 存在精确性和复杂性的矛盾。例，多层网络模型的节点数或隐层数的选择。
- （2）输入信号的选择
  - 输入信号必须满足一定的条件
    - 在辨识时间内，输入信号必须是持续激励的，即输入信号必须充分激励系统的所有模态。从频谱观点看，输入信号的频谱必须足以覆盖系统的频谱。
    - 所谓输入信号的最优设计问题，即设计输入信号使给定问题的辨识精度最高，常用的输入信号有白噪声或伪随机信号。
- （3）误差准则的选择
  - 误差准则是用来衡量模型接近实际系统的标准，它通常表示为一个误差的泛函。记作

$$J(\theta)=\sum_{k=1}^{L} f\left[e(k)\right]$$

用得最多的是平方函数，即 $f\left[e(k)\right]=e^2(k)$

系统辨识的常用方法

- 系统辨识是一个优化问题。
- 一般辨识算法的基本原理，就是通过建立系统依赖于参数θ的模型，把辨识问题转化成对模型参数的估计问题。
- 针对线性系统或可线性化的系统，主要有三种方法
  - 最小二乘法
  - 梯度校正法
  - 极大似然法

基于神经网络的辨识的特点

1. 不要求建立实际系统的辨识格式。即可省去系统建模这一步，因为神经网络本质已作为一种辨识模型，其可调参数反映在网络内部的权值上。
2. 可以对本质非线性系统进行辨识。辨识是非算法式的，由神经网络本身体现。辨识的结果为网络外部特性拟合系统的输入/输出特性，网络的内部特性归纳隐含了由系统输入/输出数据决定的系统特性。
3. 辨识的收敛速度不依赖于待辨识系统的维数，只与神经网络本身及其所采用的学习算法有关，而传统的辨识算法随模型参数维数的增大变得很复杂。
4. 神经网络具有大量连接，其连接权的权值在辨识中对应于模型参数，通过调节这些参数可使网络输出逼近系统输出。
5. 神经网络作为实际系统的辨识模型，实际上也是系统的一个物理实现，可以用于在线控制。

神经网络用于系统辨识的一般结构

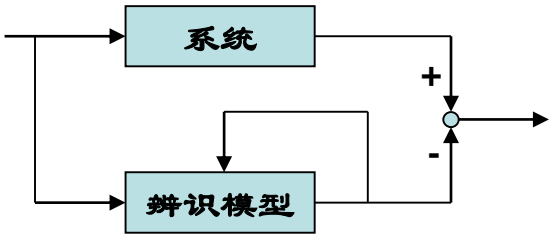
- 常利用多层静态网络进行系统辨识，要求预先给定系统的阶，即必须给出定阶的差分方程，典型的是NARMA模型。(非线性自回归滑动平均模型 **Nonlinear Auto Regressive Moving Average)**
- 一般可采用两种辨识模型
  - 并联模型：（动态反馈前向网络）

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

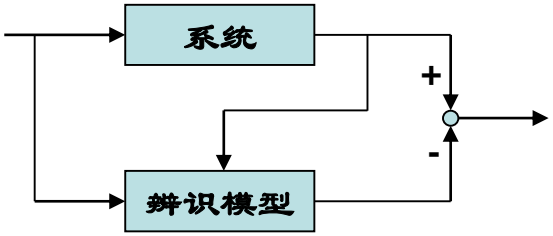
  - 串-并联模型：（静态前向网络）

$$\hat{y}_p(k+1) = f\left[y_p(k), y_p(k-1), \cdots, y_p(k-n+1); u(k), u(k-1), \cdots, u(k-m+1)\right]$$

- 一般，若并联模型可行，则优于串-并联模型，但学习算法需修改，比简单的基于误差的反向传播学习（**BP**）算法复杂。



并行辨识模式



串-并行辨识模式

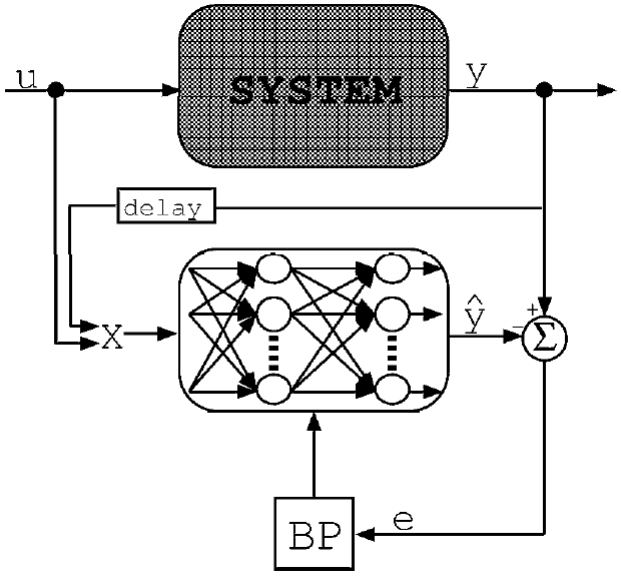
例：基于BP网络的系统辨识

- （1）结构设计
  - 采用串-并联辨识模式，考虑以下形式的单输入单输出非线性动态系统

$$\hat{y}_p(k+1) = f\left[y_p(k), y_p(k-1), \cdots, y_p(k-n+1); u(k), u(k-1), \cdots, u(k-m+1)\right]$$

其中，m，n为输入、输出的阶次。

  - 选择包含有一个隐含层的三层前向网络，活化函数取Sigmoid函数.
  - 各层神经元节点数
    - 输入层：$n_I = n+m$
    - 输出层：$n_o = 1$
    - 隐含层：一般要求$n_H \geq n_I$，可采用尝试训练后再比较性能指标的方法决定。



例：基于BP网络的系统辨识

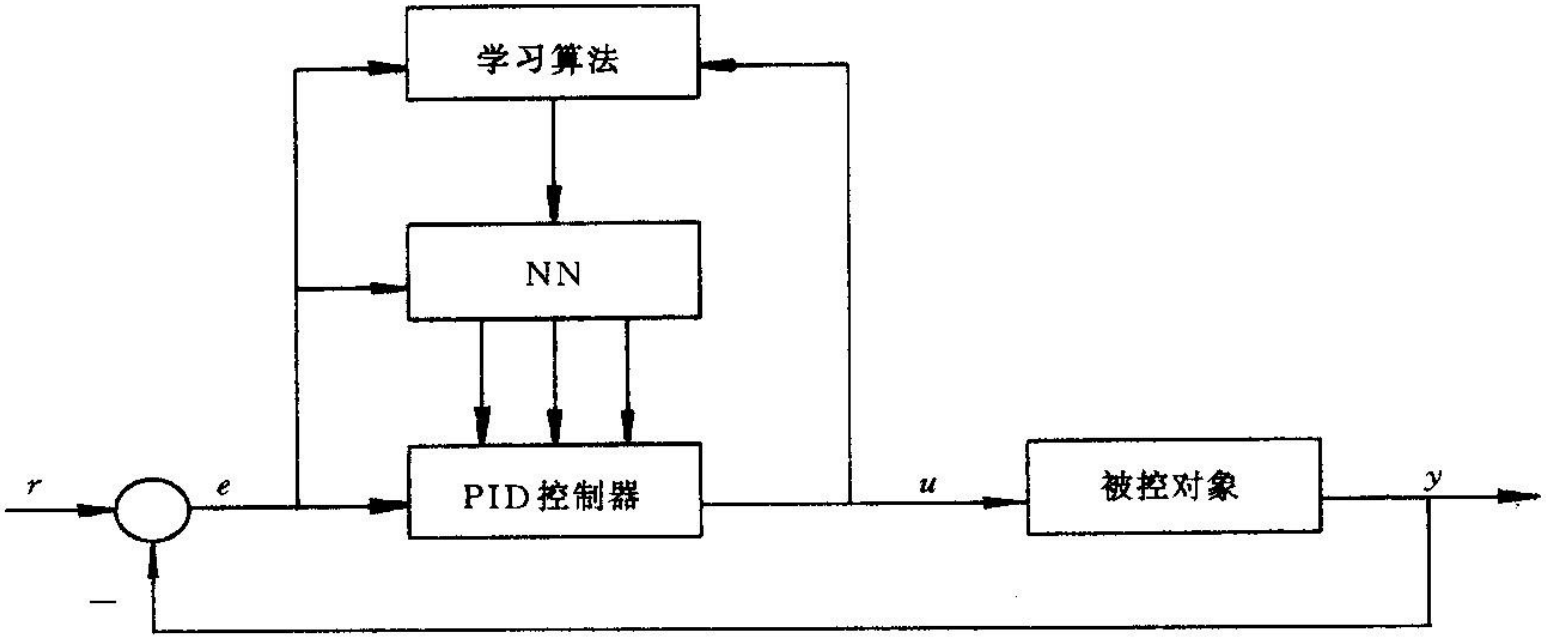
- （2）辨识算法
- a)取性能指标为 $J=\frac{1}{2}\sum_{k}\left[y(k)-\hat{y}(k)\right]^2$
- b)采用标准的BP算法（或改进的BP算法）修改网络权值。
- c)给定判定收敛的条件，如当所有网络连接权的变化充分小时，可判断辨识结束。

- 由于采用三层BP网络，结构比较简单，上述辨识算法的收敛性可由Lyapunov稳定性理论证明。

四、神经网络PID控制

基于BP网络的PID 控制

- 神经网络可以直接作为控制器，也可以作为控制器参数的调节器。
- 通过BP神经网络自身的学习，找到某一最优控制律下的*P*，*I*，*D*参数。



控制器组成

- 经典的PID控制器
  - 直接对被控对象进行闭环控制，并且K_P, K_I, K_D三个参数为在线整定；
  - 经典增量式数字PID 的控制算式为

$$u(k)=u(k-1)+K_p[e(k)-e(k-1)]+K_I e(k)+K_D[e(k)-2e(k-1)+e(k-2)]$$

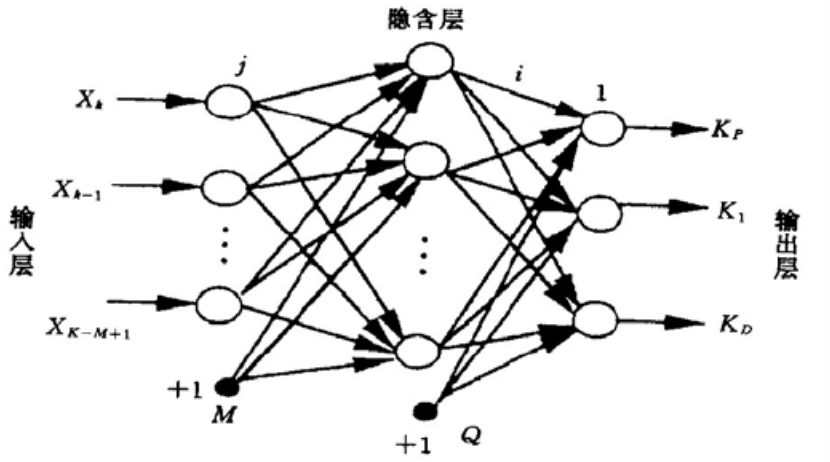
  - 因此，控制输出可表示为

$$u(k)=f[u(k-1),\ K_p,\ K_I,\ K_D,\ e(k),\ e(k-1),\ e(k-2)]$$

- 神经网络NN
  - 根据系统的运行状态，调节PID控制器的参数，以期达到某种性能指标的最优化。
  - 使输出层神经元的输出状态对应于PID控制器的三个可调参数K_P, K_I, K_D，通过神经网络的自学习、调整权系数，从而使其稳定状态对应于某种最优控制律下的PID控制器参数。

网络结构设计

- 用**BP** 神经网络通过训练和学习来找到一个最佳控制规律。
- 设**BP** 神经网络**NN** 是一个三层**BP** 网络，有***M*** 输入节点、***Q***个隐层节点、**3**个输出节点。
- 输入节点对应所选的系统运行状态量，如系统不同时刻的输入量和输出量等，必要时要进行归一化处理。
- 输出节点分别对应**PID** 控制器的三个可调参数K_P, K_I, K_D
- 由于K_P, K_I, K_D不能为负值，所以输出层神经元的激发函数取非负的**Sigmoid** 函数，而隐含层神经元的激发函数可取正负对称的**Sigmoid** 函数。



网络输入输出

- **BP** 神经网络输入层节点的输出为

$$\left.\begin{aligned} O_j^{(1)} &= x_{k-j} = e(k-j), \quad j = 0,1,\cdots,M-1 \\ O_M^{(1)} &\equiv 1 \end{aligned}\right\}$$

- 网络隐含层输入、输出为

$$\left.\begin{aligned} net_i^{(2)}(k) &= \sum_{j=0}^{M} \omega_{ij}^{(2)} O_j^{(1)}(k) \\ O_i^{(2)} &= f\left[net_i^{(2)}(k)\right], \qquad i = 0,1,\cdots,Q-1 \\ O_Q^{(2)}(k) &\equiv 1 \end{aligned}\right\}$$

- 网络输出层的输入、输出为

$$\left.\begin{aligned} net_l^{(3)}(k) &= \sum_{i=0}^{Q} \omega_{li}^{(3)} O_i^{(2)}(k) \\ O_l^{(3)} &= g\left[net_i^{(3)}(k)\right], \qquad l = 0,1,2 \\ K_p &= O_0^{(3)} \\ K_I &= O_1^{(3)} \\ K_D &= O_2^{(3)} \end{aligned}\right\}$$

权系数学习算法

- 取性能指标函数为 $$J=\frac{1}{2}[r(k+1)-y(k+1)]^2=\frac{1}{2}z^2(k+1)$$
- 依照最速下降法修正网络的权系数，按J对权系数的负梯度方向搜索调整，并附加一个使搜索快速收敛全局极小的惯性项，有

$$\Delta\omega_{li}^{(3)}(k+1)=-\eta\frac{\partial J}{\partial\omega_{li}^{(3)}}+\alpha\Delta\omega_{li}^{(3)}(k)$$

$$\frac{\partial J}{\partial\omega_{li}^{(3)}}=\frac{\partial J}{\partial y(k+1)}\cdot\frac{\partial y(k+1)}{\partial u(k)}\cdot\frac{\partial u(k)}{\partial O_l^{(3)}(k)}\cdot\frac{\partial O_l^{(3)}(k)}{\partial net_l^{(3)}(k)}\cdot\frac{\partial net_l^{(3)}(k)}{\partial\omega_{li}^{(3)}}$$

- 由于$\partial y(k+1)/\partial u(k)$未知，所以近似用符号函数$\mathrm{sgn}[\partial y(k+1)/\partial u(k)]$替代，由此带来的计算不精确的影响可以通过调整学习速率η来补偿。

权系数学习算法

- 由增量PID式

$$\left.\begin{aligned}\frac{\partial u(k)}{\partial O_0^{(3)}(k)} &= e(k)-e(k-1)\\ \frac{\partial u(k)}{\partial O_1^{(3)}(k)} &= e(k)\\ \frac{\partial u(k)}{\partial O_2^{(3)}(k)} &= e(k)-2e(k-1)+e(k-2)\end{aligned}\right\}$$

- BP 神经网络NN 输出层的权系数计算公式为

$$\left.\begin{aligned}\Delta\omega_{li}^{(3)}(k+1) &= \eta\delta_l^{(3)}O_i^{(2)}(k)+\alpha\Delta\omega_{li}^{(3)}(k)\\ \delta_l^{(3)} &= e(k+1)\,\mathrm{sgn}\left(\frac{\partial y(k+1)}{\partial u(k)}\right)\times\frac{\partial u(k)}{\partial O_l^{(3)}(k)}g'[net_l^{(3)}(k)]\\ l &= 0,1,2\end{aligned}\right\}$$

权系数学习算法

- 隐含层权系数的计算公式为

$$\left.\begin{aligned}\Delta\omega_{ij}^{(2)}(k+1)&=\eta\delta_i^{(2)}O_j^{(1)}(k)+\alpha\Delta\omega_{ij}^{(2)}(k)\\ \delta_i^{(2)}&=f'[net_i^{(2)}(k)]\sum_{l=0}^{2}\delta_l^{(3)}\omega_{li}^{(3)}(k)\\ i&=0,1,\cdots,Q-1\end{aligned}\right\}$$

- 其中，由激发函数定义（输出层神经元的激发函数取非负的**Sigmoid**函数，而隐含层神经元的激发函数可取正负对称的**Sigmoid** 函数，即双曲正切函数。）

$$g(x)=\frac{1}{1-e^{-x}}$$

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

因此有：

$$g'[\bullet]=g(x)[1-g(x)]$$

$$f'[\bullet]=[1-f^2(x)]/2$$

基于BP神经网络的PID控制算法

1. 事先选定BP神经网络NN的结构，即选定输入层节点数M和隐含层节点数Q，并给出权系数的初值$w^{(2)}_{ij}(0)$, $w^{(3)}_{li}(0)$，选定学习速率η和平滑因子α，$k=1$；
2. 采样得到$r(k)$和$y(k)$，计算$e(k)=z(k)=r(k)-y(k)$；
3. 对$r(i)$, $y(i)$, $u(i-1)$, $e(i)$进行归一化处理，作为NN的输入；
4. 前向计算NN的各层神经元的输入和输出，NN输出层的输出即为PID控制器的三个可调参数$K_P(k)$, $K_I(k)$, $K_D(k)$；
5. 计算PID控制器的控制输出$u(k)$，参与控制和计算；
6. 计算修正输出层的权系数$w^{(3)}_{li}(k)$；
7. 计算修正隐含层的权系数$w^{(2)}_{ij}(k)$；
8. 置$k=k+1$，返回到“2”。

改进型BP网络PID控制

- 将神经网络用于控制器的设计或直接学习计算控制器的输出（控制量），一般都要用到系统的性质（如系统输出与系统输入的偏微分关系，即Jacobian）来计算权系数的修正量，然而，这些性质的定量值是不易获得的。

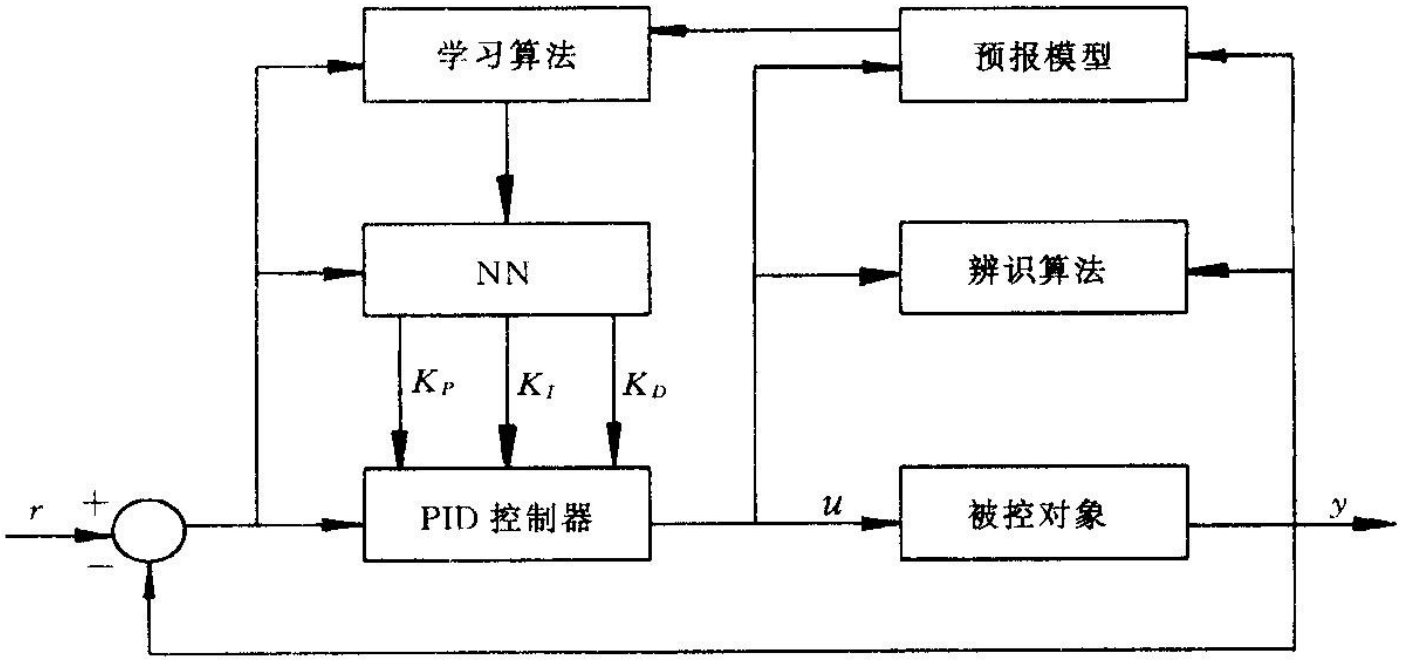
- 通常的做法是建立被控对象的预测数学模型，用该模型所计算的预测输出来取代预测处的实测值，以提高控制效果。

- 如BP网络PID控制中的$\partial y(k+1)/\partial u(k)$未知，也可由预测模型的估计值替代。

采用线性预测模型

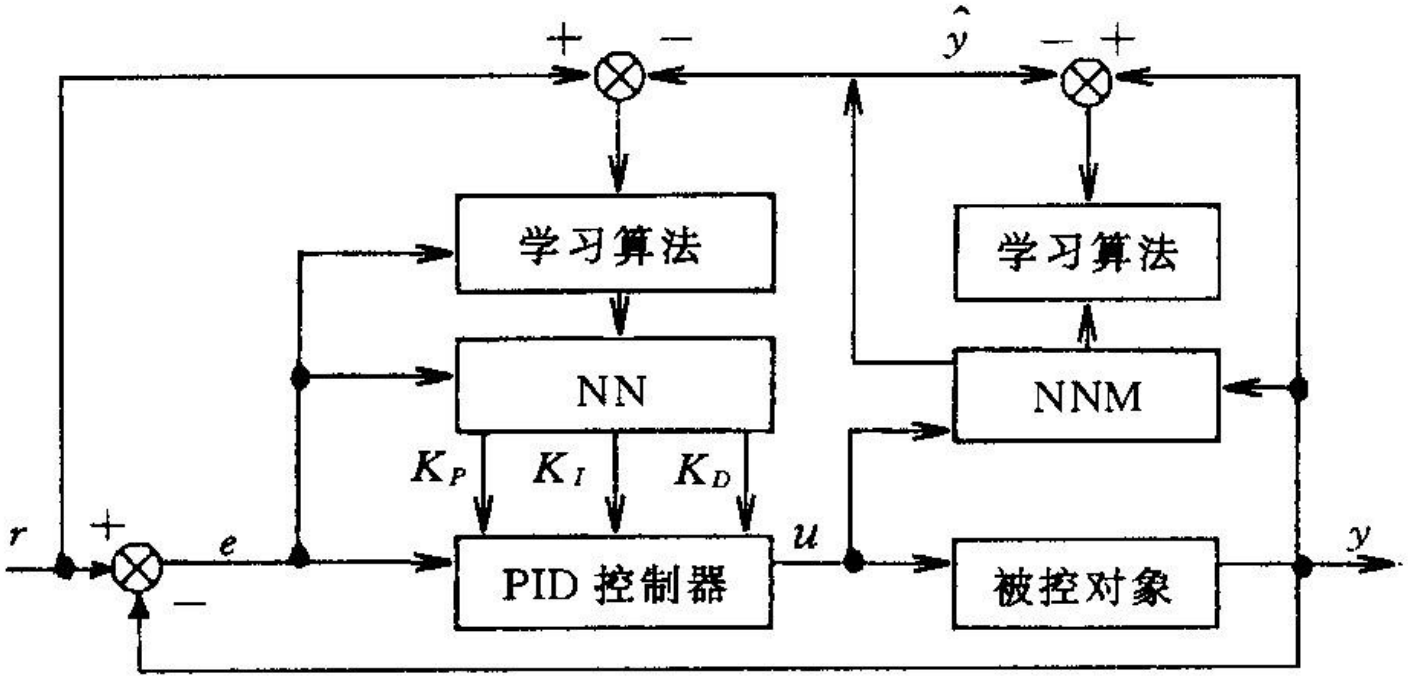
- 预报模型可用线性模型描述

$$A(z^{-1})y(k) = z^{-d}B(z^{-1})u(k) + v(k)$$



采用非线性预测模型

- 预报模型的非线性模型可以选择为三层**BP** 神经网络模型（**NNM**）



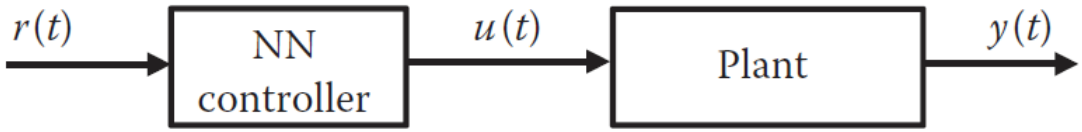
五、神经网络控制结构方案

神经控制的结构方案

- 基于自适应方式的神经网络控制系统
- 由神经网络单独构成的系统
- 基于常规控制原理的神经网络控制
  - 与其它类型的控制原理相结合，采用常规的控制理论设计控制器结构，用神经网络取代其中部分内容或进行决策处理。
- 神经网络智能控制
  - 将神经网络与人工智能、模糊逻辑等相结合。
- 神经网络优化控制
  - 由于神经网络可以表达成任何非线性函数，因此，使用神经网络能完成各种复杂的优化运算。
  - 可将神经网络用于控制器的优化设计。
  - 例：利用**Hopfield**网络求解广义预测控制中矩阵求逆问题。
  - 例：利用神经网络在线辨识对象的数学模型。
  - 例：利用神经网络设计控制器。

神经网络逆模型控制

- 采用受控系统的一个逆模型，神经网络与受控系统串接以便使系统在期望响应（网络输入）与受控系统输出间得到一个相同的映射。
- 网络(NN)直接作为前馈控制器，而且受控系统的输出等于期望输出。



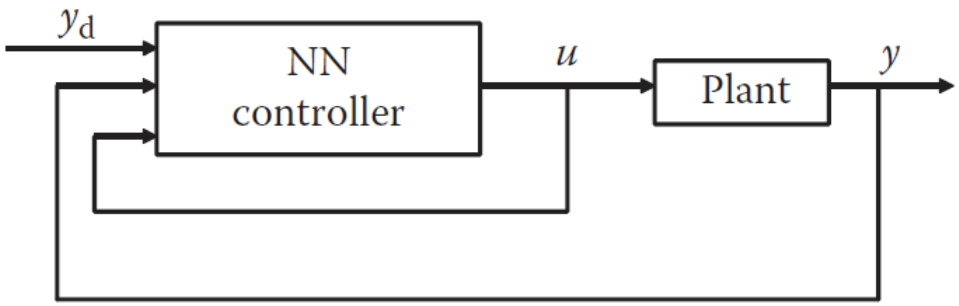
若系统为

$$y(k) = f(y(k-1), \ldots, y(k-n), u(k-1), u(k-2), \ldots, u(k-m))$$

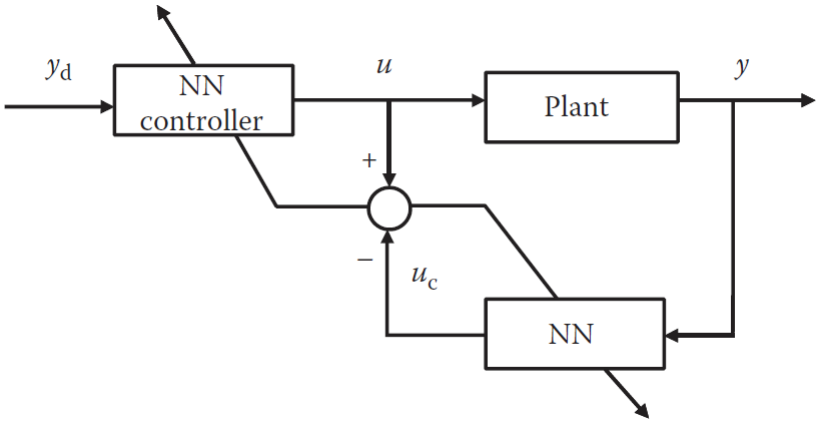
则欲训练的逆模型为

$$u(k) = \phi(y(k+1), y(k), \ldots, y(k-n+1), u(k-1), \ldots, u(k-m+1))$$

训练结束，在上式中用期望$r(k+1)$代替$y(k+1)$即得到逆控制器。



间接型NN在线逆模型控制



- 策略：
  - 系统运行过程中，在线应用多层前向网络学习从系统输出到系统输入的逆模型，一般采用BP算法调整网络权值。
  - NN控制器为结构与逆模型相同的网络，权值在线取逆模型的相同的权值。
- 缺陷：
  - 由于逆模型为静态网络，与真实的系统动态逆模型不一致，导致控制方案的稳定性常出问题。

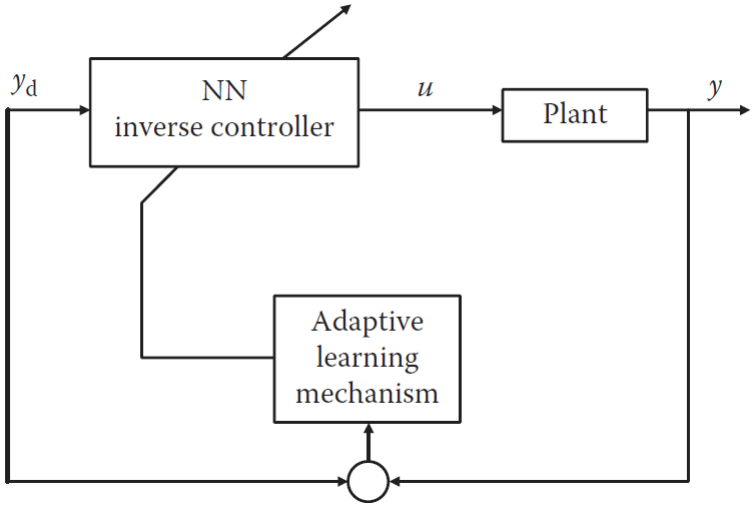
直接型NN在线逆模型控制

- 策略：
  - 在线通过期望输出与系统输出的误差直接调整逆模型网络的权值。
  - 若$e=y_d-y$，$P_i(\mathbf{u})$系统的第i个输出变量，通过反向传播方法，得到

$$\Delta w_{ji}^{(k)} = \mu \delta_j u_i^{(k)}$$

$$\delta_j^{(k-1)} = f'(\text{net}) \sum_i \delta_i^{\text{p}} \frac{\partial P_i}{\partial u_j}$$

$$\delta_i^{\text{p}} = y_{d_i} - y_i$$



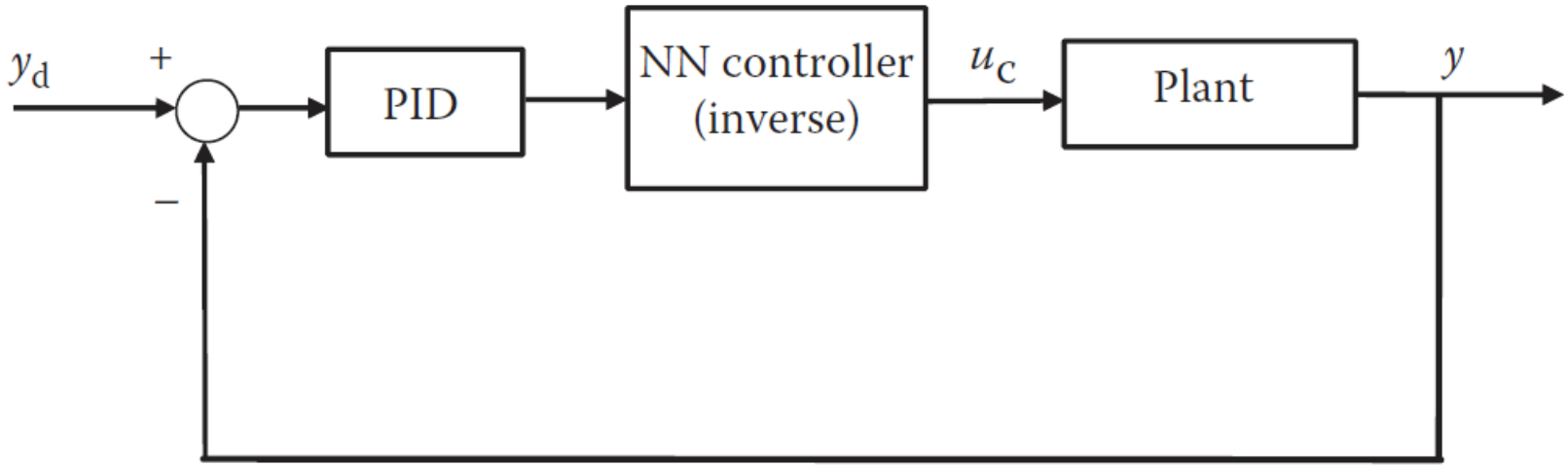
  - 由于一般系统的Jacobian未知，可采用前述的符号法或如下的测试运行获得：

$$\frac{\partial P_i}{\partial u_j} \approx \frac{P_i(\mathbf{u}+\delta u_j) - P_i(\mathbf{u})}{\delta u_j}$$

- 缺陷：
  - 由于在线运行，系统缺乏持续激励，网络输入的适应范围受限。
  - 控制方案的抗干扰性弱。

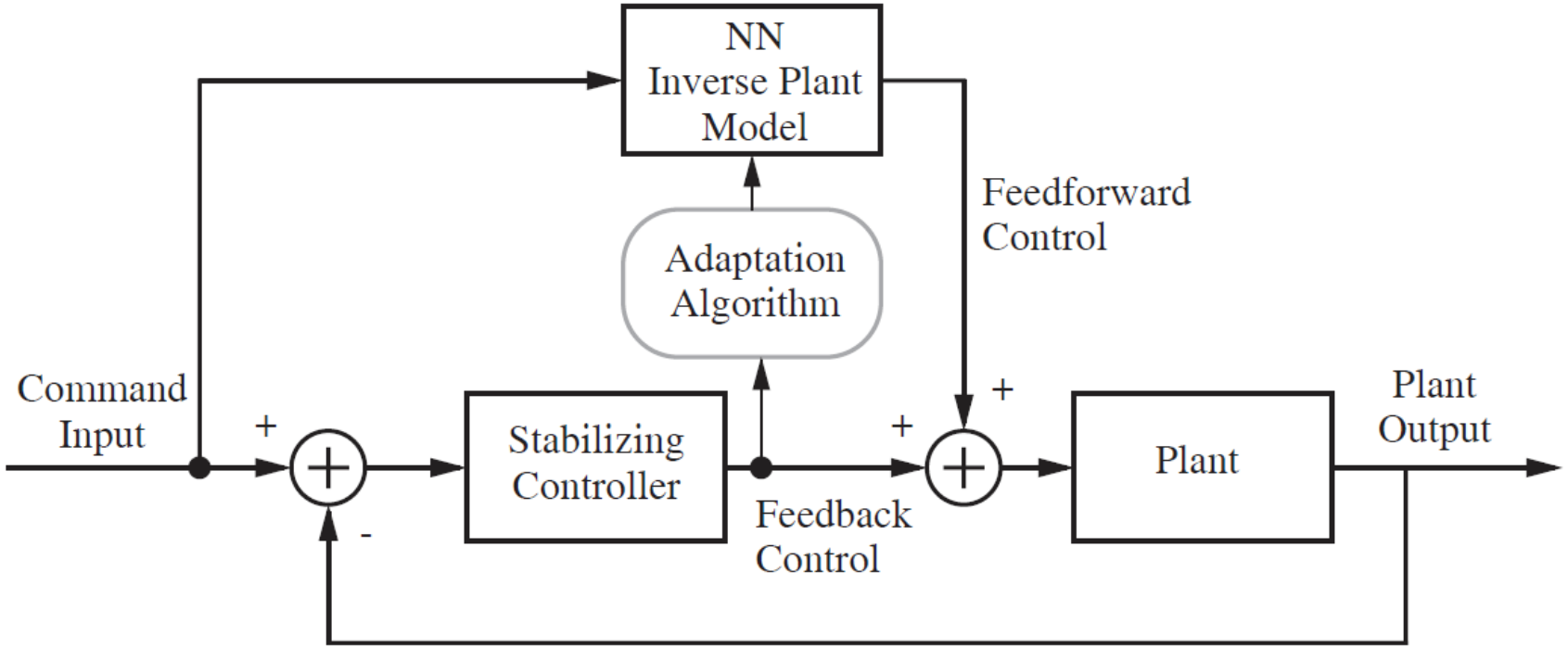
神经网络逆模型控制

- 结合**PID**控制，增加系统的鲁棒性和抗干扰能力



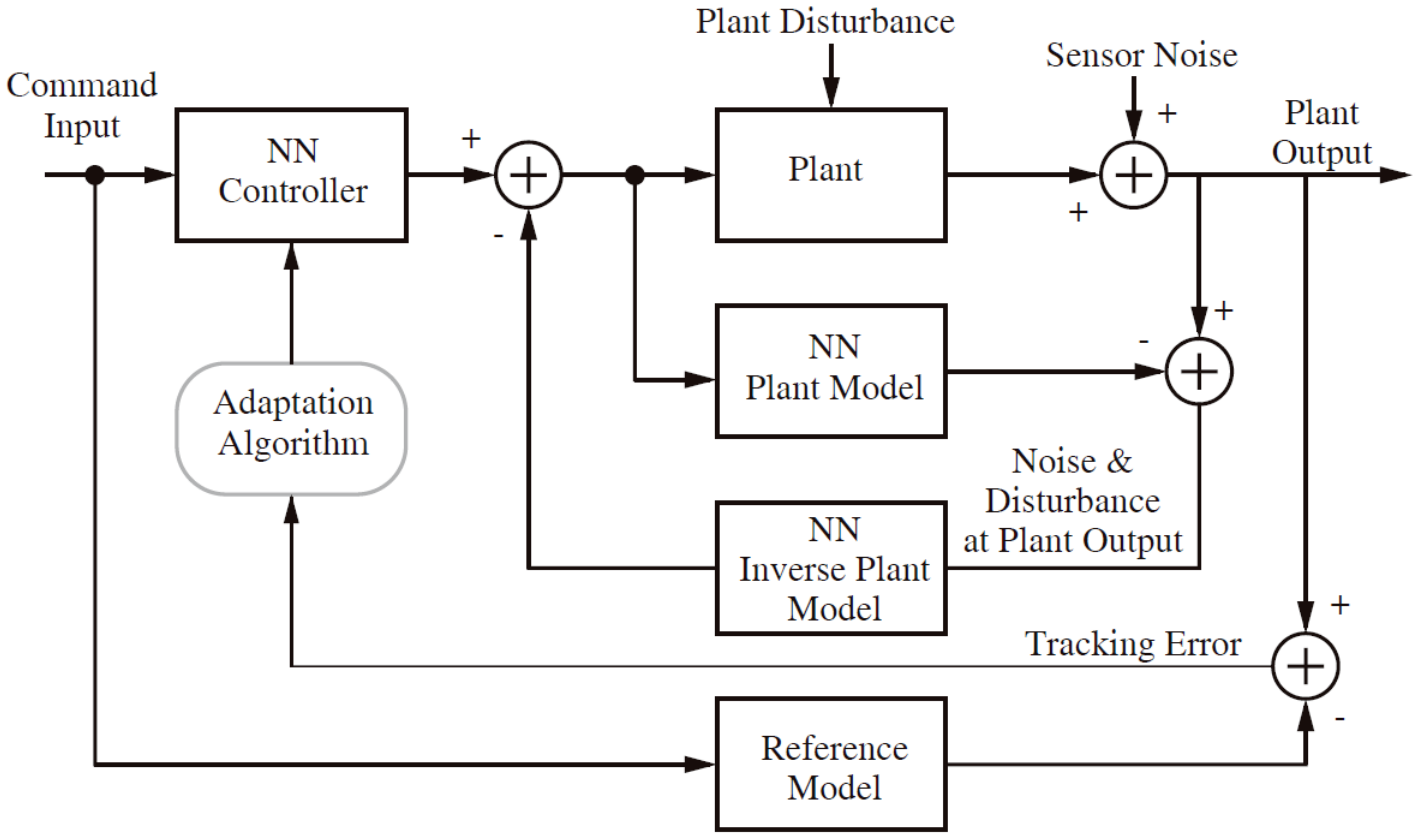
神经网络逆模型控制

- 在网络未充分学习的前提下，系统运行过程从一个稳定的控制开始。



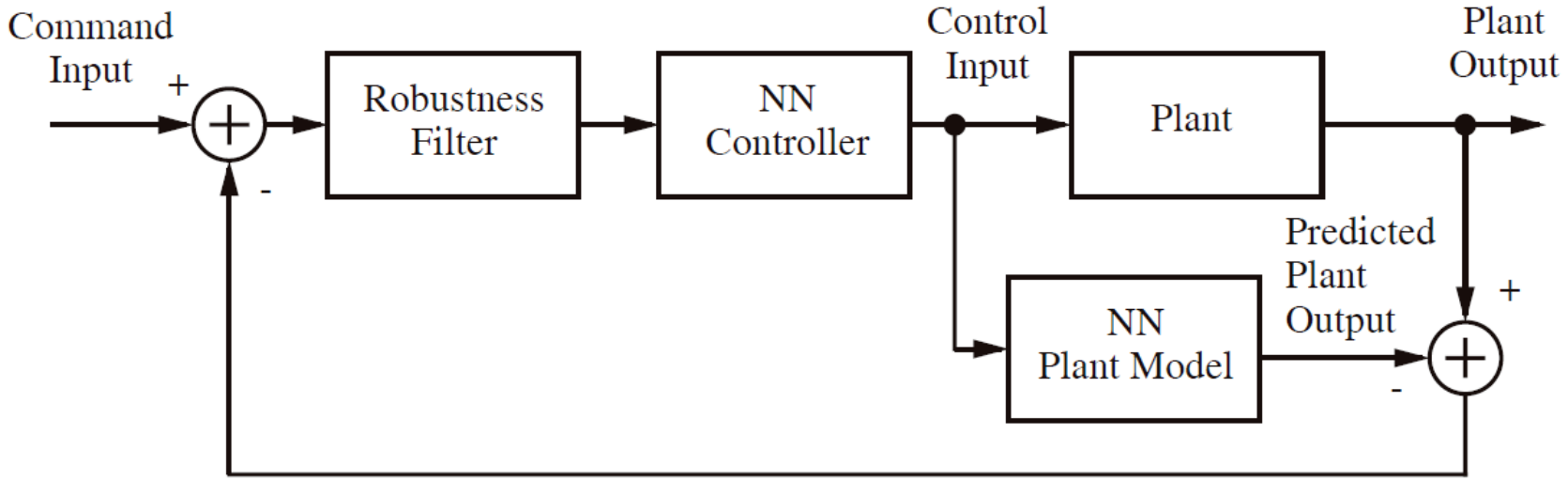
神经网络自适应逆模控制

- 增加一个逆系统模型，目的是消除系统干扰和测量噪声。



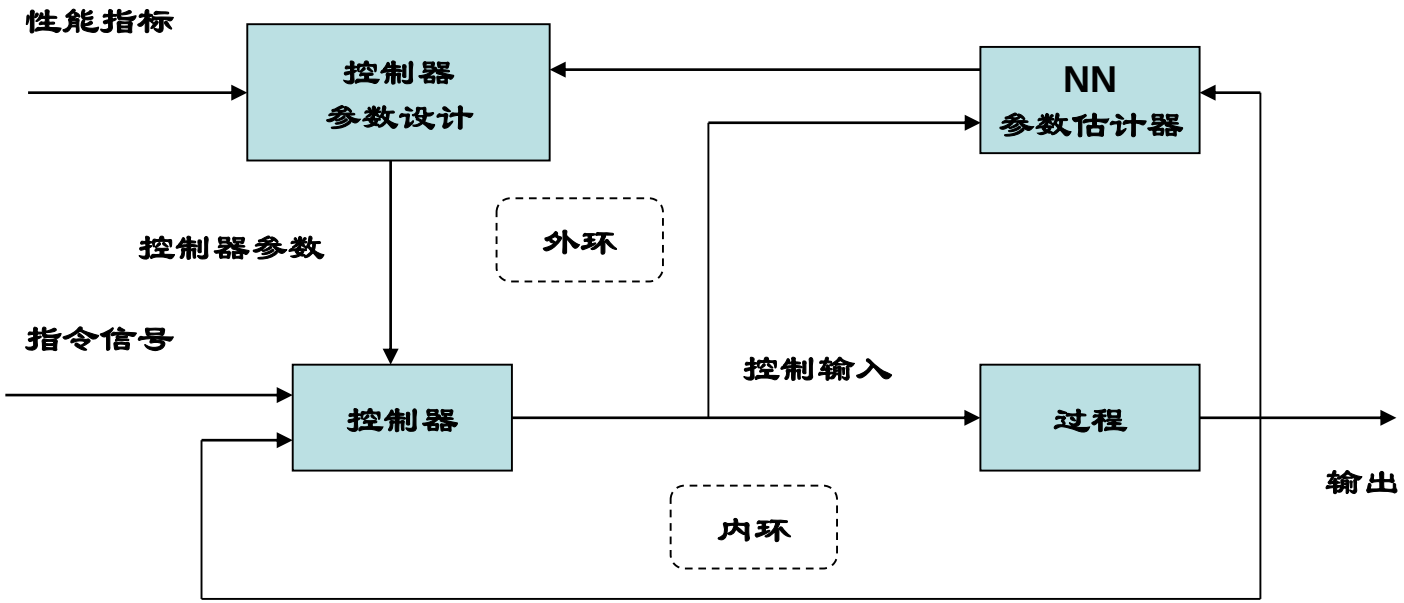
神经网络内模控制

- 系统模型(NN Plant Model)与实际系统并行设置。
- 反馈信号由系统输出与模型输出间的差得到，然后由NN Controller（在正向控制通道上一个具有逆模型的NN控制器）进行处理；
- NN控制器与系统的逆有关。



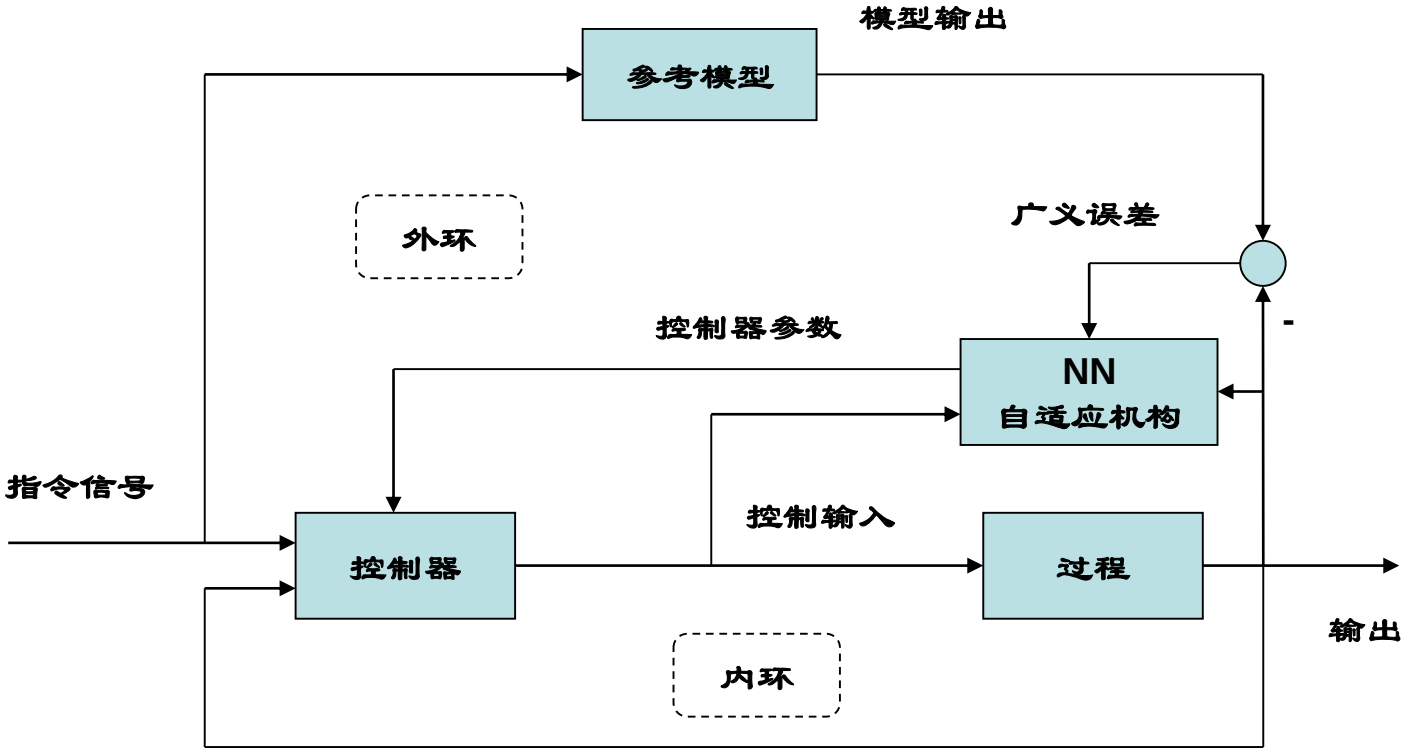
神经网络自适应控制

- NN自校正控制



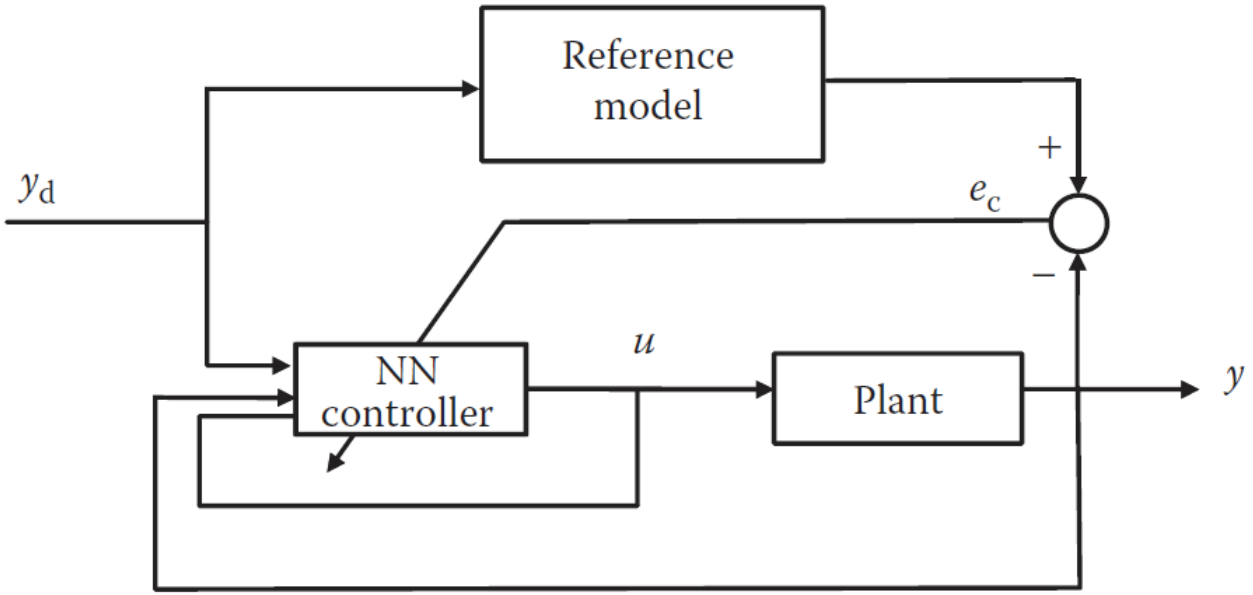
神经网络自适应控制

- NN模型参考自适应控制



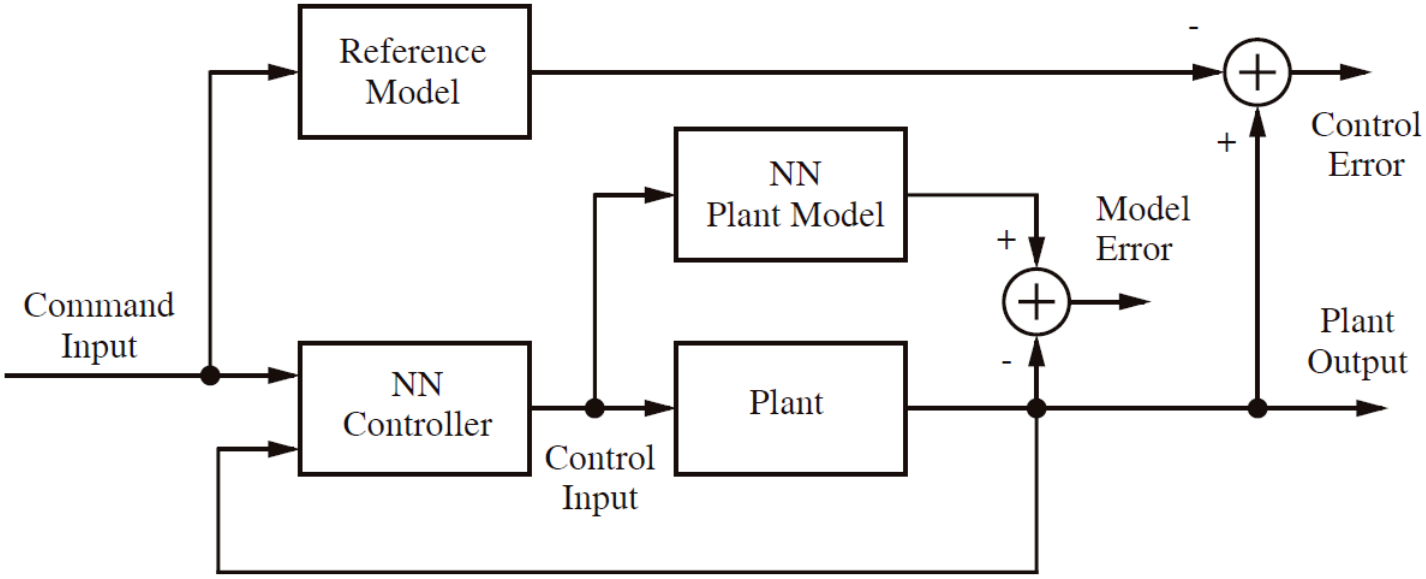
直接NN参考自适应控制

- NN在参考自适应控制中直接作为控制器



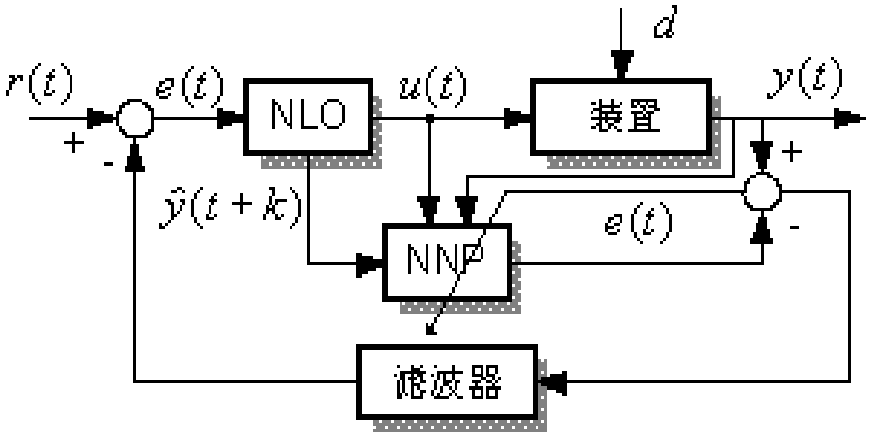
间接NN参考自适应控制

- NN系统模型先通过系统的历史数据离线训练好，并在在线运行过程中自适应调整。
- NN控制器先通过已训练好的NN系统模型进行仿真运行训练。
- 在线运行时，NN控制器通过参考模型输出与系统输出的误差在线调整。同时，自适应运行的NN系统模型针对控制器输入给出预测输出，并可根据预测误差调整NN控制器参数。



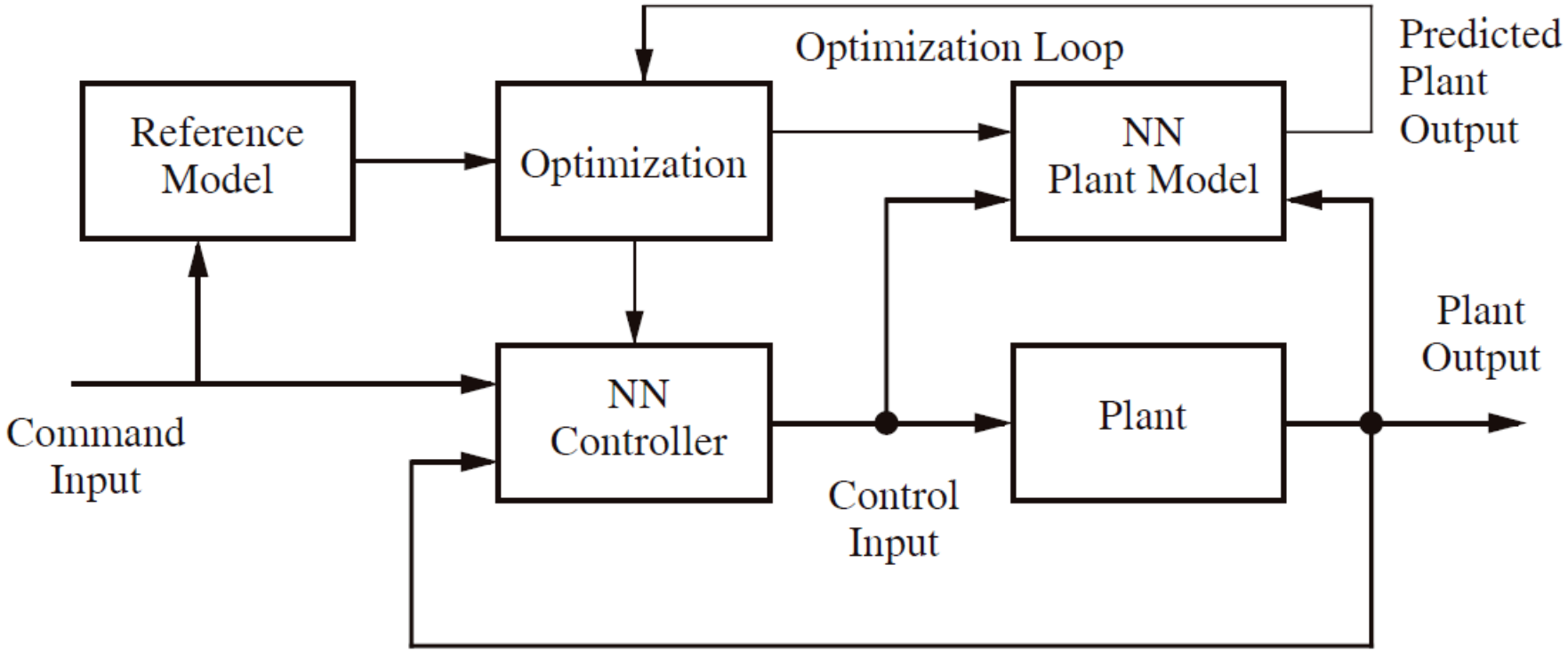
神经网络预测控制

- 预测控制具有预测模型、滚动优化和反馈校正等特点。
- 神经网络预测器**NNP**为一神经网络模型。
- 神经网络**NLO**为一非线性优化器。



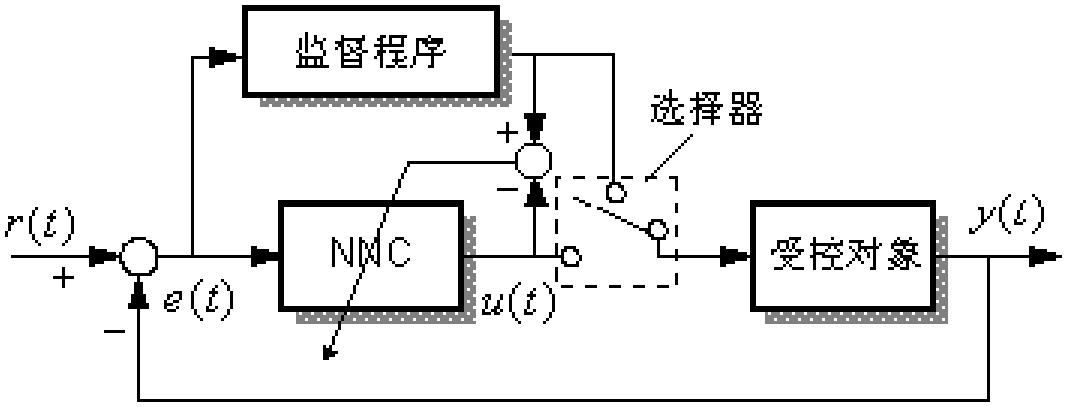
神经网络预测控制

- 另一种策略是：两个神经网络分别作为预测控制中的系统预测模型和控制器。



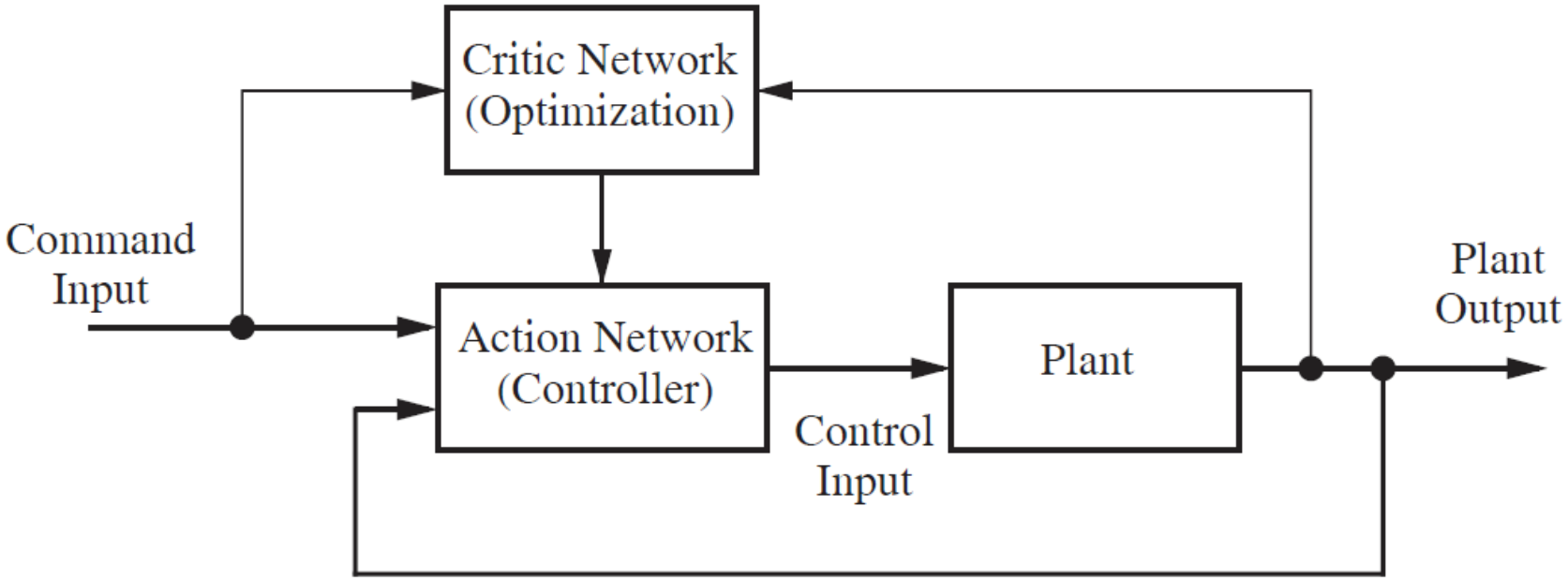
神经网络监督控制

- （1）选择网络的输入。
  - 通过传感器和传感信息处理，调用必要的和有用的控制信息。
- （2）选择和构造一个神经网络作为控制器。
  - 构造神经网络，选择NN类型、结构参数和学习算法等。
- （3）采用参考控制器（监督程序）训练网络。
  - 训练NN控制器，实现输入和输出间的映射，以便进行正确的控制。在训练过程中，可采用线性律、反馈线性化或解耦变换的非线性反馈作为导师（监督程序）来训练NN控制器。



神经网络监督控制

- 一个神经网络作为逆模控制器，另一个神经网络进行控制性能评价，给出控制预测并调整逆模控制器参数。



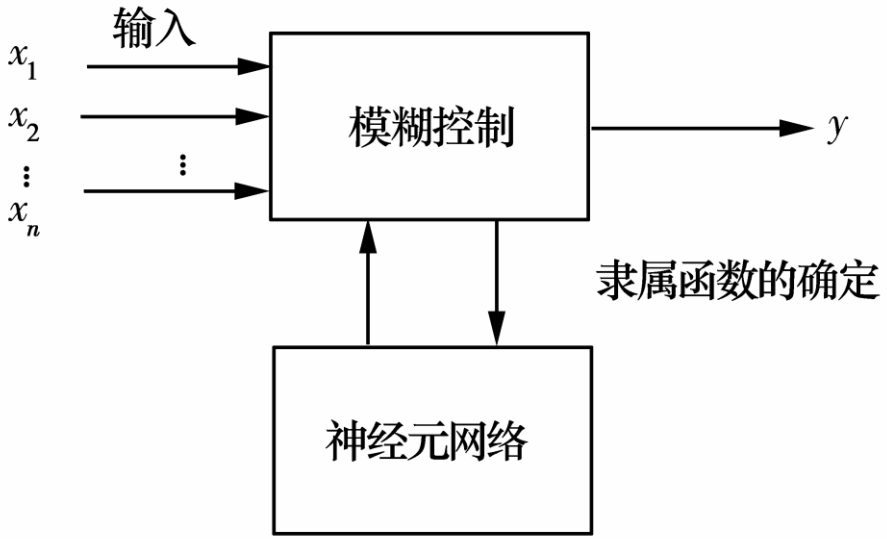
六、模糊神经网络

模糊控制与神经网络相结合

- 模糊系统是仿效人的模糊逻辑思维方法，明确说明系统在工作过程中允许定性知识的存在。
- 神经网络在计算处理信息的过程中所表现出的学习能力和容错性来自于其网络自身的结构特点。
- 模糊神经网络是模糊逻辑推理与神经网络有机结合的产物
  - 模糊逻辑推理的结构性知识表达能力
  - 神经网络的自学习能力

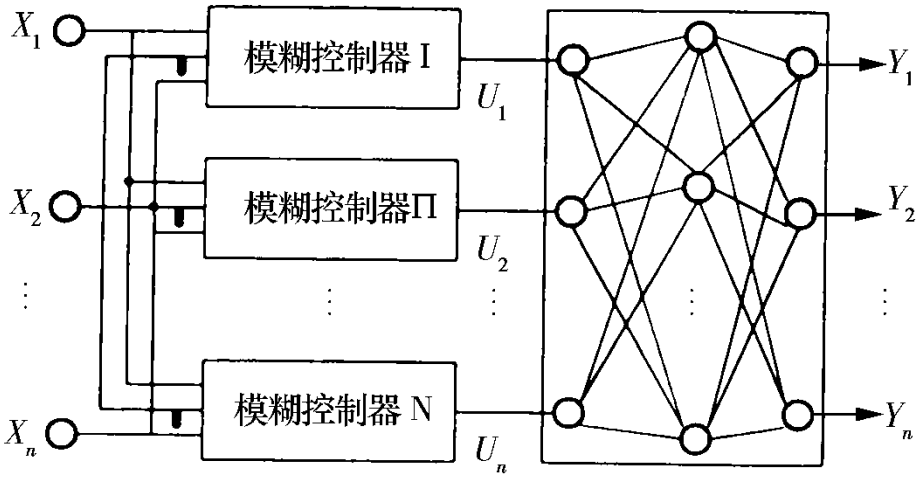
结合方式-神经网络模糊模型

- 以模糊控制为主体，应用神经网络，实现模糊控制的决策过程。以模糊控制方法为“样本”，对神经网络进行离线训练学习。
  - “样本”就是学习的“教师”。
  - 所有样本学习完以后，神经网络就是一个模糊规则表。
  - 具有自学习、自适应功能。



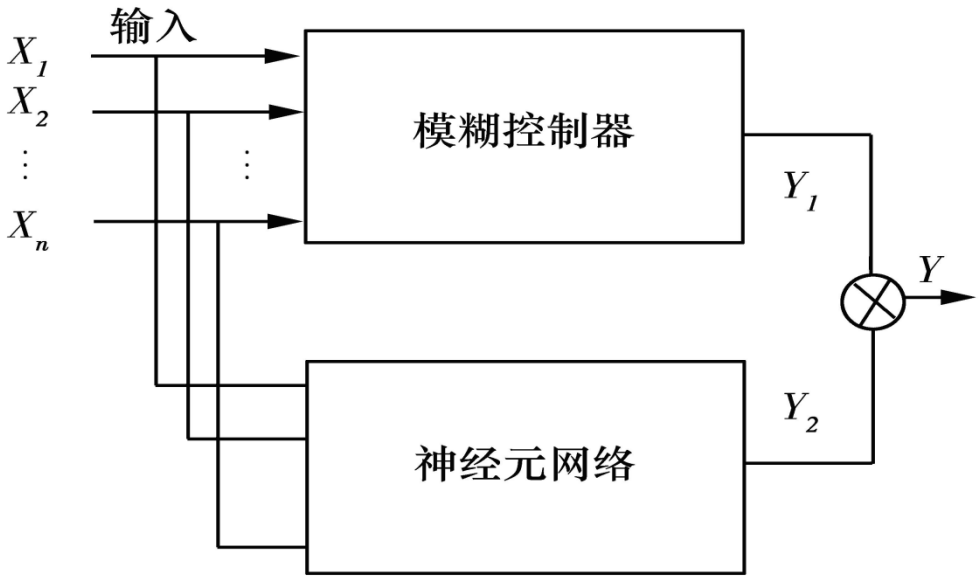
结合方式-模糊神经网络模型

- 以神经网络为主体，将输入空间分割成若干不同型式的模糊推论组合，对系统先进行模糊逻辑判断，以模糊控制器输出作为神经元网络的输入。
  - 神经网络具有自学习能力，使整个系统在面对相同控制局势时具有更好的控制特性。



结合方式-神经与模糊模型

- 根据输入量的不同性质分别由神经网络与模糊控制直接处理输入信息，并作用于控制对象，发挥各自的控制特点。

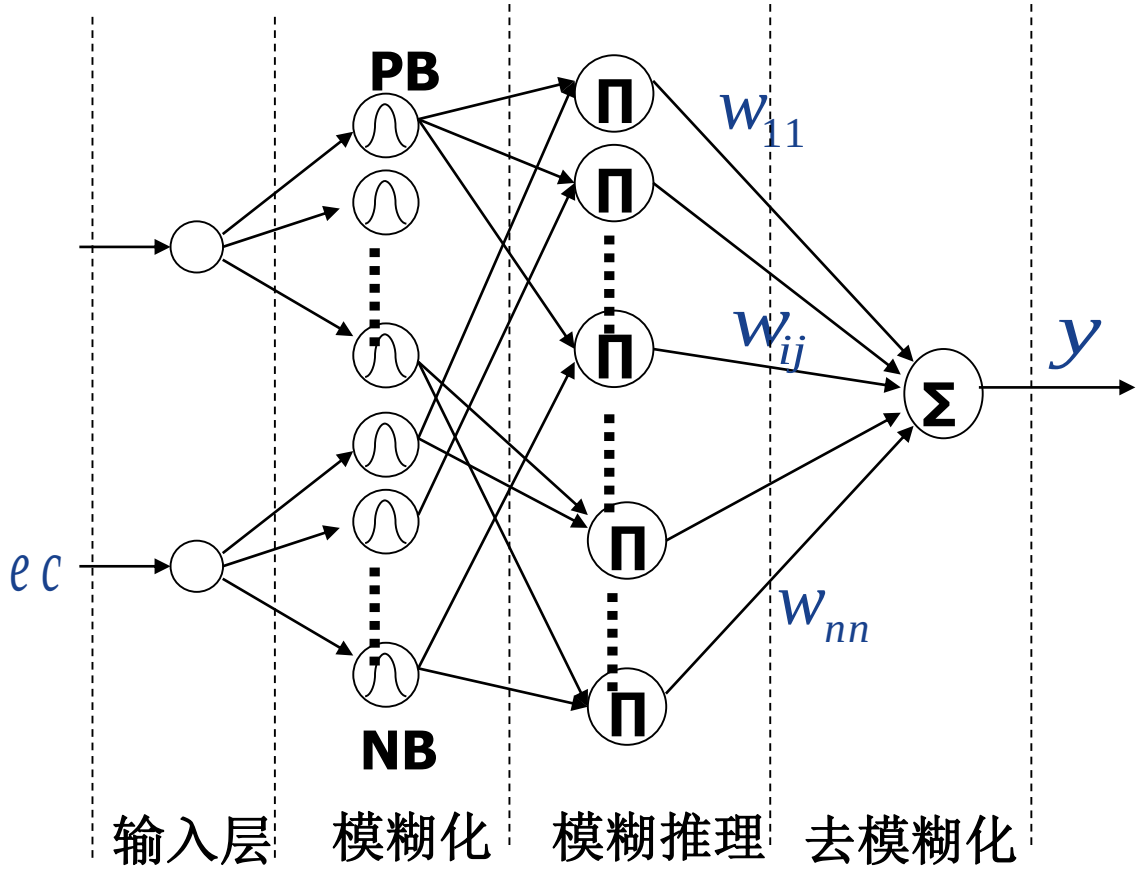


模糊神经网络

- 模糊推理神经网络：利用神经网络来实现模糊推理。
  - 使网络的结构和权值具有明确的物理意义，网络的结构设计和权值的初始化都有了理论的根据，避免了网络陷入局部最优。
  - 可利用神经网络的学习能力来调整模糊控制的控制规则和模糊化的方式，使模糊控制具有了一定的自适应能力。
  - 模糊神经网络将定性的知识表达和定量的数值运算很好地结合了起来，具有很好的控制效果。

模糊神经网络

- 模糊推理神经网络结构

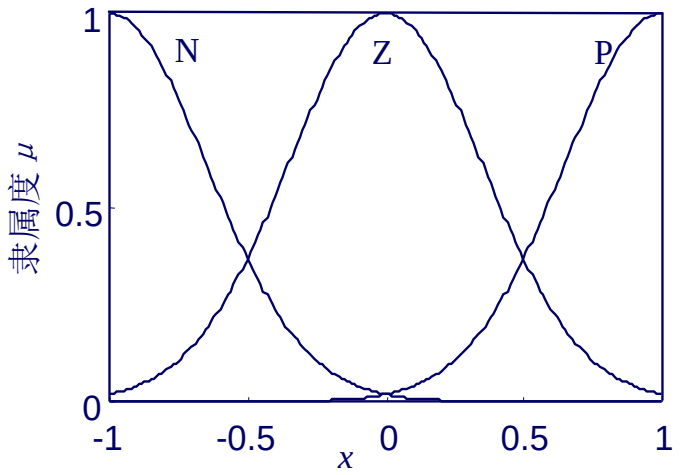


模糊神经网络

- 第1层（输入层）：将输入引入网络（例：系统误差，误差变化率）

$$out_1^{(1)} = in_1^{(1)} = e, \quad out_2^{(1)} = in_2^{(1)} = ec$$

- 第2层（模糊化层）：对输入进行模糊化。例：假设在每个输入论域上定义3个模糊语言集{N，Z，P}={“负”，“零”，“正”}，隶属函数采用高斯基函数，与{N，Z，P}对应的中心值分别为{-1，0，1}，宽度为{0.5，0.5，0.5}. 隶属函数的形状与分布如图所示。



$$in_{ij}^{(2)} = out_i^{(1)}$$

$$out_{ij}^{(2)} = \mu_{A_{ij}}\left(in_{ij}^{(2)}\right) = \mu_{A_{ij}}\left(out_i^{(1)}\right) = e^{-\frac{\left(out_i^{(1)} - a_{ij}\right)^2}{b_{ij}^{\ 2}}}$$

即为求相似度

模糊神经网络

- 第3层（模糊推理）：进行模糊推理。（与模糊规则有关，模糊推理步骤在此处可以进行分解，例：在此步仅进行求取激励强度计算，如“and”操作，在网络中采用乘法运算.）

$$out_{ij}^{(3)} = in_{ij}^{(3)} = out_{1i}^{(2)} \cdot out_{2j}^{(2)}$$

- 第4层（解模糊化）：例：采用重心法。

$$in^{(4)} = \sum_{i,j=1}^{3} \left(out_{ij}^{(3)} \cdot w_{ij} \right)$$

$$y = out^{(4)} = in^{(4)} \Big/ \sum_{i,j=1}^{3} out_{ij}^{(3)}$$

w_{ij}为网络的权值，其物理意义是各控制规则的输出对应的语言集的中心值。

模糊神经网络

- 网络学习算法
  - 例：学习采用**BP**算法，定义目标函数

$$J = \frac{1}{2}(r - y)^2 = \frac{1}{2}e^2$$

  - 则学习算法

$$w_{ij}(t+1) = w_{ij}(t) - \eta_w \frac{\partial J}{\partial w_{ij}(t)}$$

$$a_{ij}(t+1) = a_{ij}(t) - \eta_a \frac{\partial J}{\partial a_{ij}(t)}$$

$$b_{ij}(t+1) = b_{ij}(t) - \eta_b \frac{\partial J}{\partial b_{ij}(t)}$$

七、神经控制的应用

基于神经网络的板带材轧制过程自适应厚度控制

- 板带材的轧制过程是一个复杂的非线性过程。
- 厚度自动控制(**Automatic Gauge Control, AGC**)是轧制过程自动化的重要组成部分。
- 传统的**AGC**技术是从轧制过程所发生的基本现象出发，把轧机的弹性变形与轧件的塑性变形均近似为线性变形，并设计出线性控制器，但对于一些无法忽略的非线性和不确定性因素，不得不采取各种改进或补偿措施。
- 由于神经网络具有极强的处理非线性及不确定性问题的能力，神经网络用于轧制过程，能改善控制性能。

轧制过程的数学模型

- 轧制过程的基本数学模型
  - 轧机弹跳方程 $h = S_0 + \frac{F}{M} + \Sigma$
  - 轧制力方程 $F = \varphi(h, H, T, t, f, B, R)$

h —— 出口厚度；
S_0 —— 空载辊缝；
F —— 轧制力；
M —— 轧机模数；
Σ —— 补偿量(包括油膜补偿、热膨胀补偿等)；
H —— 入口厚度；
T、t —— 机架前、后张力；
f —— 轧辊与轧件之间的摩擦因数；
B —— 带材宽度；
R —— 轧辊半径；
φ —— 非线性函数，因建模方法不同而呈现不同形式

- 由于系统的非线性及不确定性，表示成一般的非线性模型

$$h(k) = \psi(h(k\text{-}1), H(k), H(k\text{-}1), S_0(k\text{-}1))$$

k —— 当前采样时刻

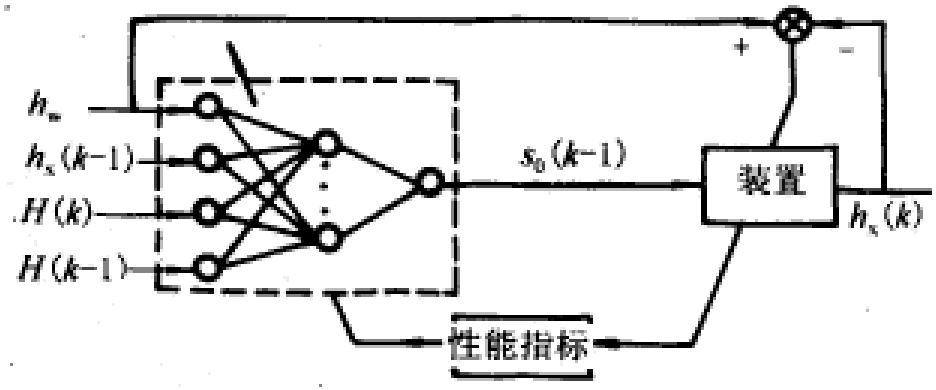
基于神经网络的自适应厚度控制(NNA-AGC)方案

- **AGC的控制手段是调节空载辊缝S_0，采用直接神经网络控制，由轧制模型和轧制过程的时滞性，可知前馈神经网络控制器可表示为**

$$S_0(k\text{-}1)=N(h(k\text{-}1),\ H(k),\ H(k\text{-}1),\ h_{ref}(k))$$

h_{ref} —— 期望的出口厚度

- **NNA-AGC(neural networks based adaptive AGC)控制方案**



网络设计及训练算法

- 适于过程控制的输出层网络激励函数

$$g(x)=k_v\left(\frac{1}{1+e^{-x}}-0.5\right)$$

k_v ——由所需要的控制信号幅度所决定

- 隐层的网络激励函数为 $f(x)=\frac{1}{1+e^{-x}}$

- **NNA-AGC系统中的网络学习训练用的偏差函数可定义为**

$$E_k=\frac{1}{2}\left(h_{ref}(k)-h(k)\right)^2$$

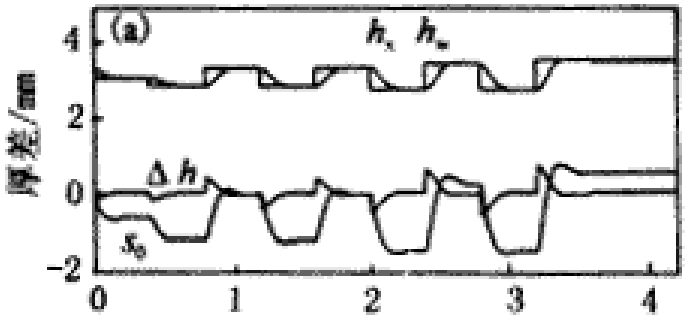
- **为加快学习速度，采用惯性BP学习算法**

$$w_{ij}(k+1)=w_{ij}(k)-\eta\frac{\partial E_k}{\partial w_{ij}}+\beta\cdot\Delta w_{ij}(k)$$

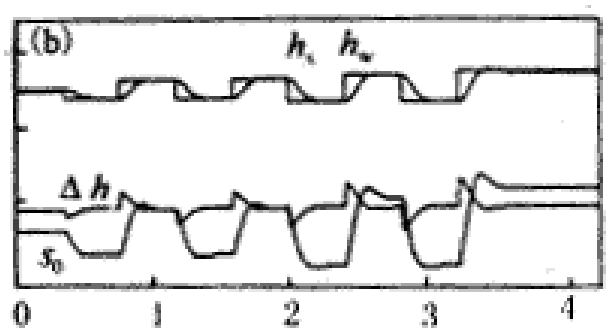
计算时要需要利用$\partial h(k)/\partial S_0(k\text{-}1)$，由于该值无法直接计算，因此，在实际控制中，根据轧制过程模型的性质，用符号$\mathrm{sgn}(\partial h(k)/\partial S_0(k\text{-}1))$代替$\partial h(k)/\partial S_0(k\text{-}1)$

NNA-AGC系统跟随性能

- 入口厚度基准值H_{REF} = 5.0 mm
- 学习率η = 0.1
- 惯性因数β = 0.05
- 期望的出口厚度为方波（仿真结果）

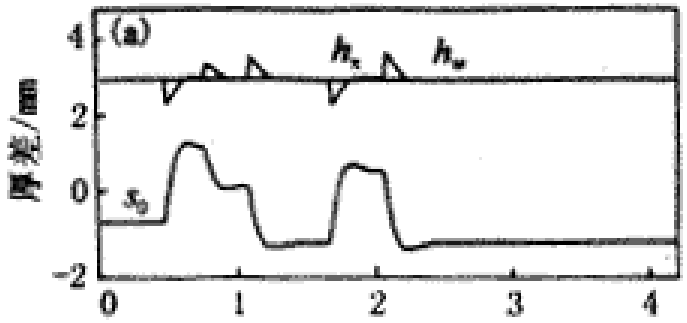


(a) 一般BP算法

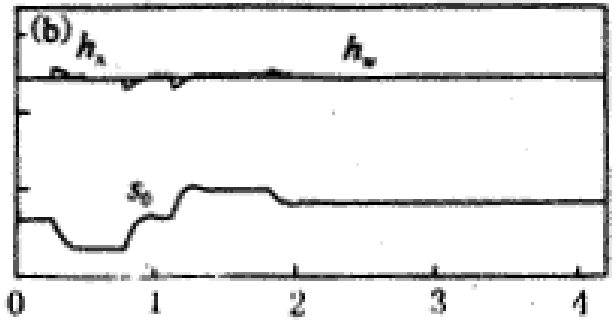


(b) 改进算法

NNA-AGC系统抗扰性能



(a)入口厚度波动时的抗扰调节过程



(b)轧件塑性系数波动时的抗扰调节过程

神经网络控制特点

- 新的神经网络控制结构不断出现；
- 每种神经网络控制结构各有优缺点；
  - 例，模型参考自适应不能保证稳定性；
  - 例，逆模控制技术需要系统的逆存在且稳定；
- 神经网络本身的泛化能力（外推的能力）有限，特定的神经网络控制结构一般均只能应用于具有特殊属性的控制对象类型；
- 神经网络控制需要训练权值，因此，需要大量样本数据，控制对象需要充分激励。

神经网络控制学习方法

- 经典神经网络控制多采用**BP**算法
- 理论性强的方法多采用**Lyapunov**技术
- 结合模糊逻辑的方法
- 采用计算智能的学习方法