第9章 神经网络控制

Chapter 9 Neural Networks-Based Control

内容提要

9.1 概述

- 神经网络是一种具有高度非线性的连续时间动力系统，它有着很强的自学习功能和对非线性系统的强大映射能力，已广泛应用于复杂对象的控制中。
- 神经网络所具有的大规模并行性、冗余性、容错性、本质的非线性及自组织、自学习、自适应能力，给不断面临挑战的控制理论带来生机。

9.1 概述

从控制角度来看，神经网络用于控制的优越性主要为：

（1）神经网络可是处理那些难以用模型或规则描述的对象；

（2）神经网络采用并行分布式信息处理方式，具有很强的容错性；

（3）神经网络在本质上是非线性系统，可以实现任意非线性映射。神经网络在非线性控制系统中具有很大的发展前途；

9.1 概述

（4）神经网络具有很强的信息综合能力，它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；

（5）神经网络的硬件实现愈趋方便。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，为神经网络在控制中的应用开辟了广阔的前景。

9.1 概述

神经网络控制所取得的进展为：

(1) 基于神经网络的系统辨识：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分类器、辨识器、控制器；

9.1 概述

(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；

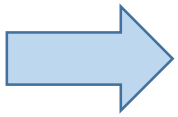
$$\begin{aligned}\text{min.}\quad & \dot{\theta}^{\mathrm{T}} W\dot{\theta}/2 + c^{\mathrm{T}}\dot{\theta} \\ \text{s.t.}\quad & J\dot{\theta} = b \\ & \zeta^{-} \leqslant \dot{\theta} \leqslant \zeta^{+}\end{aligned}$$

$$\begin{aligned}\text{minimize}\quad & \dot{\theta}^{\mathrm{T}} Q\dot{\theta}/2 + p^{\mathrm{T}}\dot{\theta} \\ \text{subject to}\quad & J_{\mathrm{E}}(\theta)\dot{\theta} = \dot{r}_{\mathrm{E}} \\ & A\dot{\theta} \leqslant b \\ & \xi^{-} \leqslant \dot{\theta} \leqslant \xi^{+}\end{aligned}$$

$$\begin{aligned}\text{minimize}\quad & \|\dot{\theta}\|_2^2/2 \\ \text{subject to}\quad & J_{\mathrm{E}}(\theta)\dot{\theta} = \dot{r}_{\mathrm{E}} \\ & J_{\mathrm{G}}(\theta)\dot{\theta} \leqslant b_{\mathrm{G}} \\ & \theta^{-} \leqslant \theta \leqslant \theta^{+} \\ & \dot{\theta}^{-} \leqslant \dot{\theta} \leqslant \dot{\theta}^{+}\end{aligned}$$

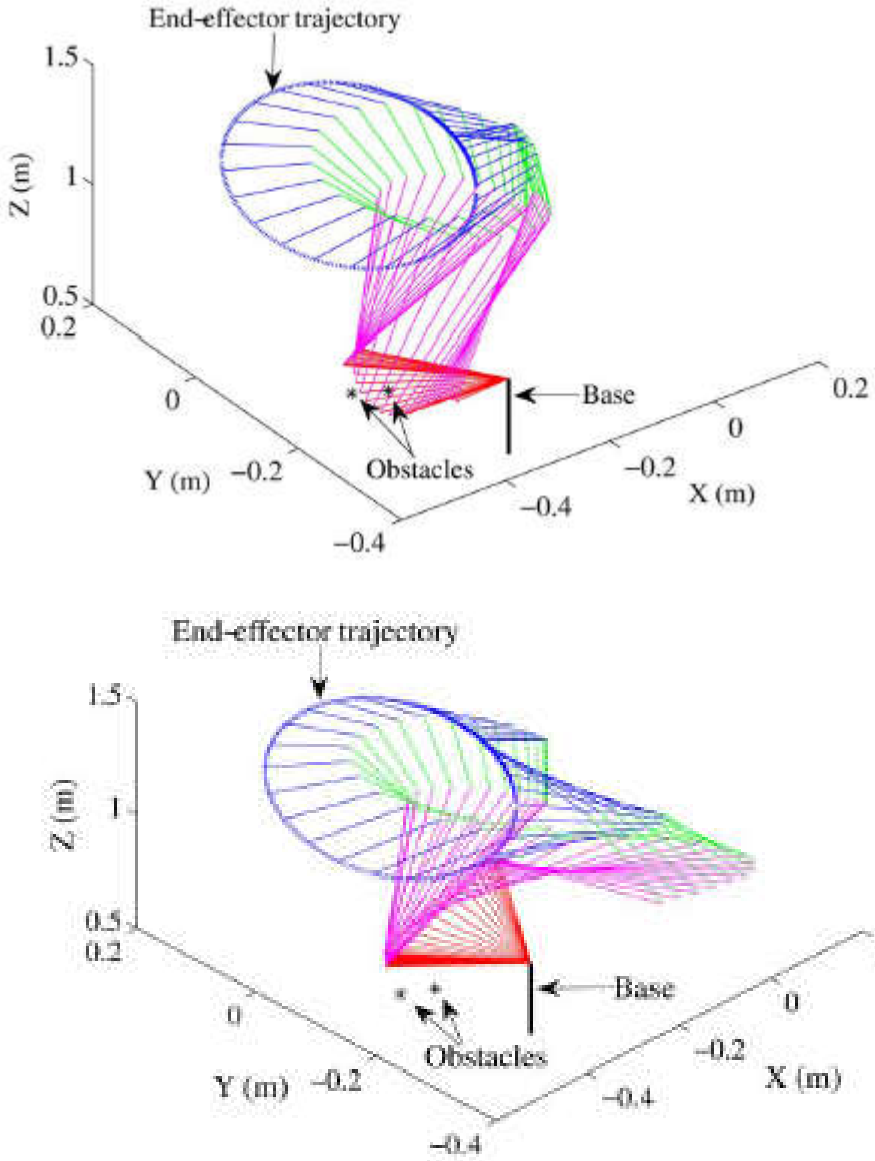
$$\begin{aligned}\min\ & \alpha_L\|\dot{\theta}_L\|_2^2/2 + \beta_L\|\dot{\theta}_L + c_L\|_2^2/2 \\ & + (1-\alpha_L-\beta_L)\|\dot{\theta}_L\|_\infty^2/2 \\ \text{s.t.}\ & J_L\dot{\theta}_L = \dot{r}_L \\ & \theta_L^{-} \leqslant \theta_L \leqslant \theta_L^{+} \\ & \dot{\theta}_L^{-} \leqslant \dot{\theta}_L \leqslant \dot{\theta}_L^{+} \\ & \text{with}\quad c_L = \lambda_L(\theta_L - \theta_L(0))\end{aligned}$$

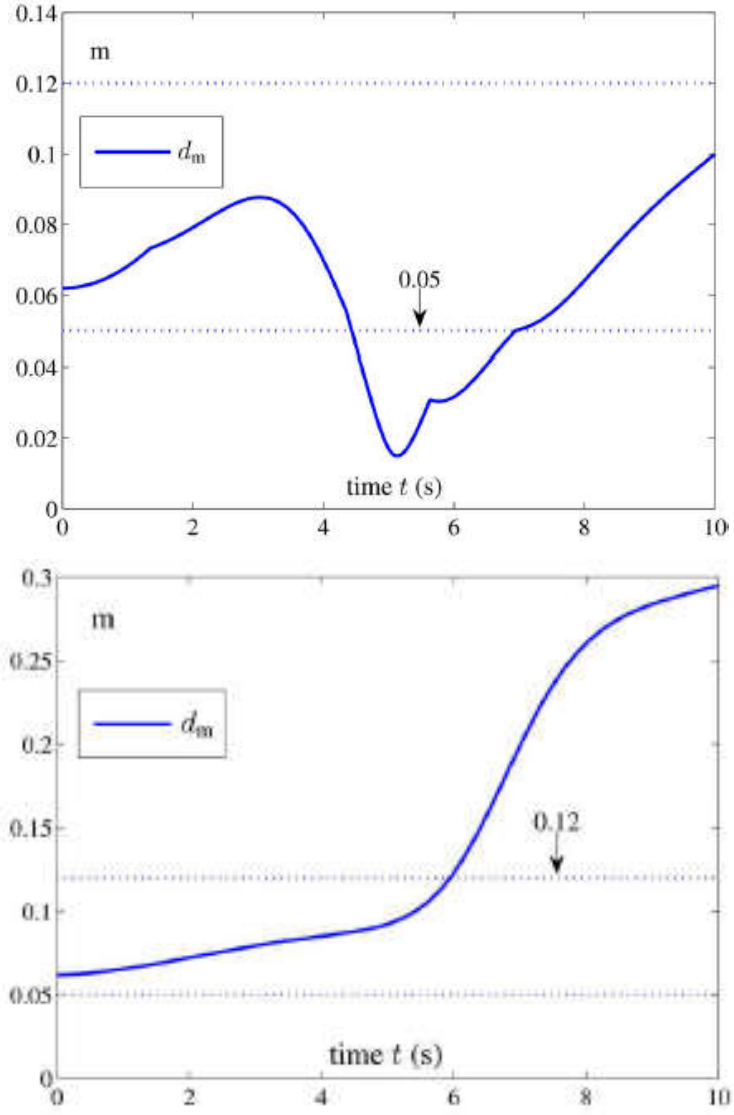
$$\begin{aligned}\text{minimize}\quad & (\ddot{\theta}+p)^{\mathrm{T}}(\ddot{\theta}+p)/2 \\ \text{subject to}\quad & J(\theta)\ddot{\theta} = \ddot{r}_a \\ & \theta^{-} \leqslant \theta \leqslant \theta^{+} \\ & \dot{\theta}^{-} \leqslant \dot{\theta} \leqslant \dot{\theta}^{+} \\ & \ddot{\theta}^{-} \leqslant \ddot{\theta} \leqslant \ddot{\theta}^{+} \\ \text{with}\quad & p = (\mu+\nu)\dot{\theta} + \mu\nu[\theta - \theta(0)]\end{aligned}$$



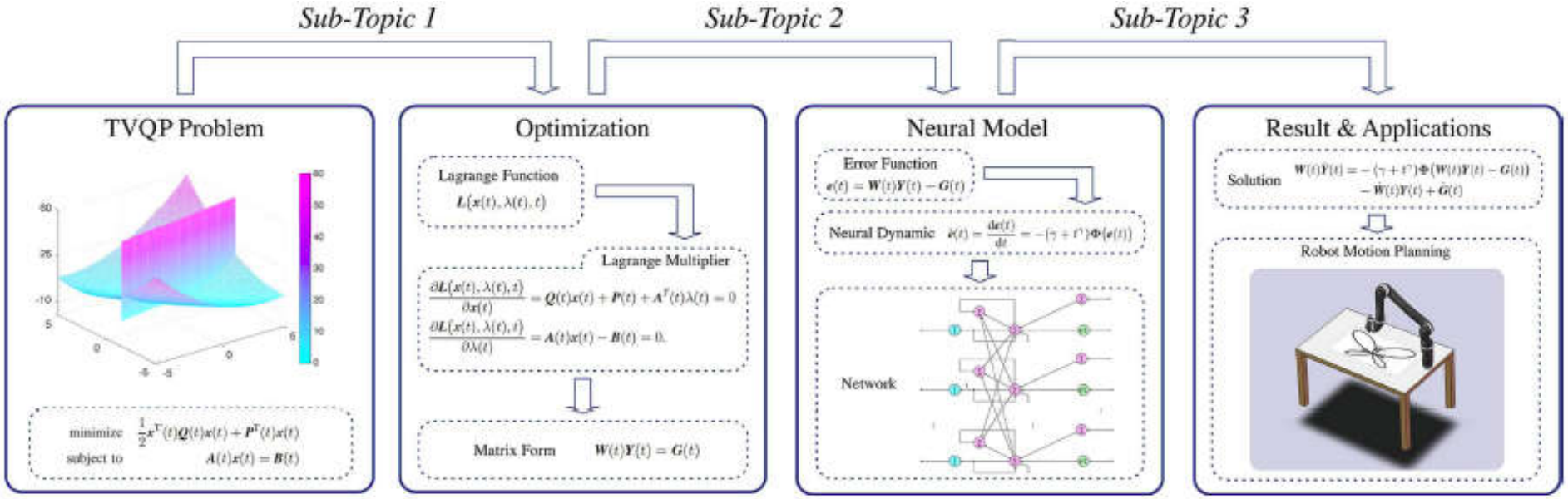
$$\begin{aligned}\text{minimize}\quad & x^{\mathrm{T}} W x/2 + p^{\mathrm{T}} x \\ \text{subject to}\quad & Cx = d \\ & \zeta^{-} \leqslant x \leqslant \zeta^{+} \\ \text{with}\quad & W = I \quad C = J(\theta) \quad d = \ddot{r}_a \\ & p = (\mu+\nu)\dot{\theta} + \mu\nu\,[\theta - \theta(0)]\end{aligned}$$

9.1 概述

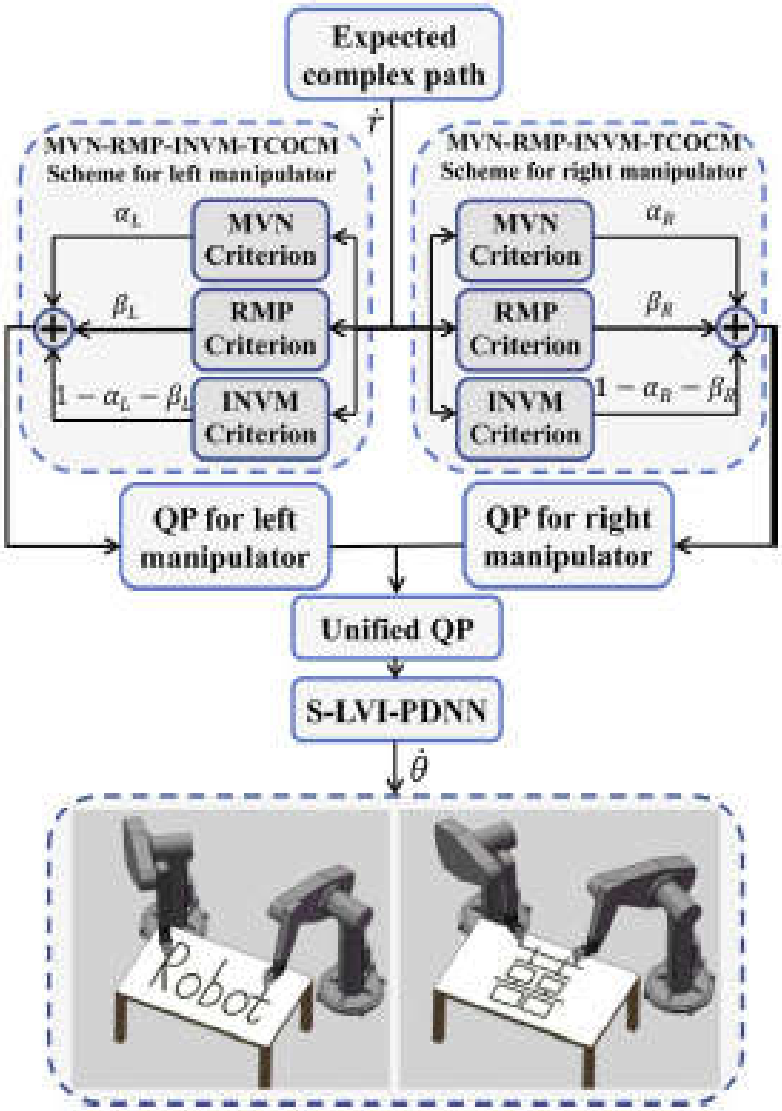

End-effector trajectory
Z (m)
Y (m)
X (m)
Base
Obstacles
End-effector trajectory
Z (m)
Y (m)
X (m)
Base
Obstacles


m
d_m
0.05
time t (s)
m
d_m
0.12
time t (s)

9.1 概述


Sub-Topic 1
Sub-Topic 2
Sub-Topic 3
TVQP Problem
minimize $\frac{1}{2}x^T(t)Q(t)x(t)+P^T(t)x(t)$
subject to $A(t)x(t)=B(t)$
Optimization
Lagrange Function
$L(x(t),\lambda(t),t)$
Lagrange Multiplier
$\frac{\partial L(x(t),\lambda(t),t)}{\partial x(t)}=Q(t)x(t)+P(t)+A^T(t)\lambda(t)=0$
$\frac{\partial L(x(t),\lambda(t),t)}{\partial \lambda(t)}=A(t)x(t)-B(t)=0.$
Matrix Form $W(t)Y(t)=G(t)$
Neural Model
Error Function
$e(t)=W(t)Y(t)-G(t)$
Neural Dynamic $\dot{e}(t)=\frac{de(t)}{dt}=-\{\gamma+\Gamma\}\Phi(e(t))$
Network
Result & Applications
Solution $\dot{W}(t)\dot{Y}(t)=-\{\gamma+\Gamma\}\Phi(W(t)Y(t)-G(t))-\dot{W}(t)Y(t)+\dot{G}(t)$
Robot Motion Planning

9.1 概述



$$\begin{aligned}\min\ & x_L^T Q_L x_L/2 + g_L^T x_L \\ \text{s.t. } & A_L x_L = b_L \\ & C_L x_L \leqslant d_L \\ & x_L^- \leqslant x_L \leqslant x_L^+\end{aligned} \qquad \begin{aligned}\min\ & x_R^T Q_R x_R/2 + g_R^T x_R \\ \text{s.t. } & A_R x_R = b_R \\ & C_R x_R \leqslant d_R \\ & x_R^- \leqslant x_R \leqslant x_R^+\end{aligned}$$

$$\begin{aligned}\min\ & z^T K z/2 + w^T z \\ \text{s.t. } & Gz = h \\ & Dz \leqslant e \\ & z^- \leqslant z \leqslant z^+\end{aligned}$$

$$K = \begin{bmatrix} Q_L & 0 \\ 0 & Q_R \end{bmatrix} \in R^{2(n+1)\times 2(n+1)}$$

$$z = \begin{bmatrix} x_L \\ x_R \end{bmatrix} \in R^{2(n+1)}, \quad w = \begin{bmatrix} g_L \\ g_R \end{bmatrix} \in R^{2(n+1)}$$

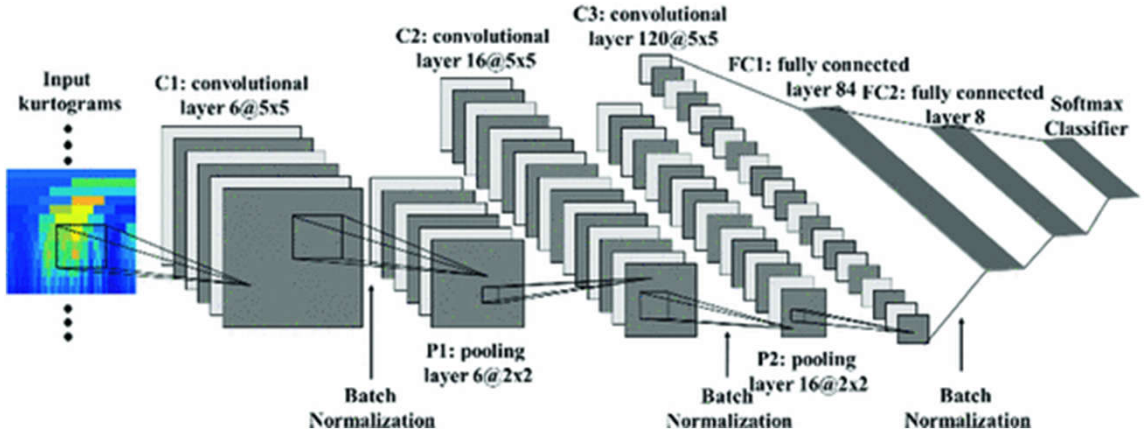
$$G = \begin{bmatrix} A_L & 0 \\ 0 & A_R \end{bmatrix} \in R^{2m\times 2(n+1)}, \quad h = \begin{bmatrix} b_L \\ b_R \end{bmatrix} \in R^{2m}$$

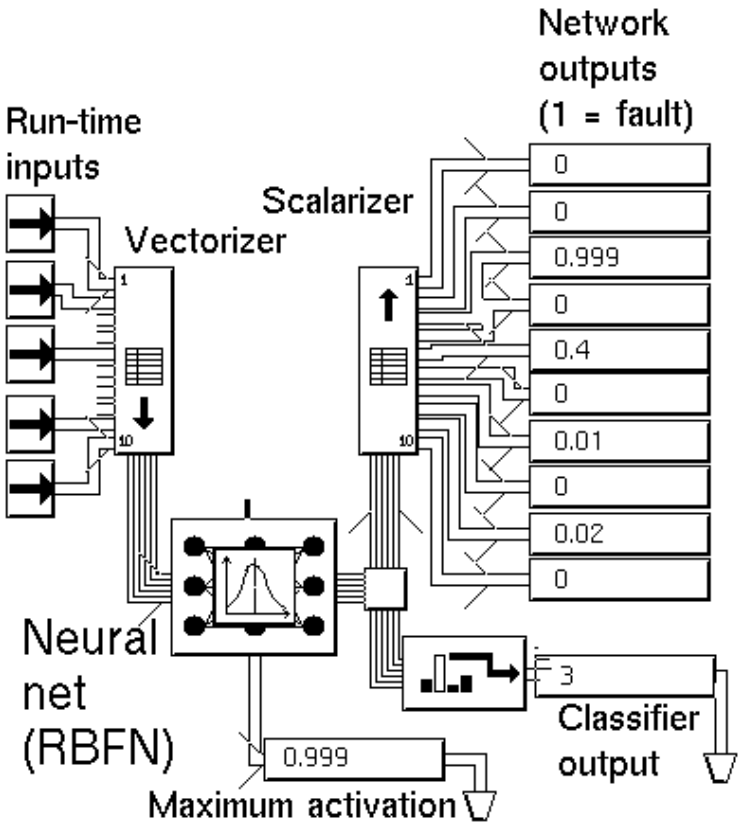
$$D = \begin{bmatrix} C_L & 0 \\ 0 & C_R \end{bmatrix} \in R^{4n\times 2(n+1)}, \quad e = 0 \in R^{4n}$$

$$z^- = \begin{bmatrix} x_L^- \\ x_R^- \end{bmatrix} \in R^{2(n+1)}, \quad z^+ = \begin{bmatrix} x_L^+ \\ x_R^+ \end{bmatrix} \in R^{2(n+1)}.$$

9.1 概述

(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。





9.1 概述

神经网络控制在理论和实践上，以下问题是研究的重点：

(1) 神经网络的稳定性与收敛性问题；

(2) 神经网络控制系统的稳定性与收敛性问题；

(3) 神经网络学习算法的实时性；

(4) 神经网络控制器和辨识器的模型和结构。

9.2 神经网络控制结构

神经网络控制的研究随着神经网络理论研究的不断深入而不断发展起来。根据神经网络在控制器中的作用不同，神经网络控制器可分为两类：

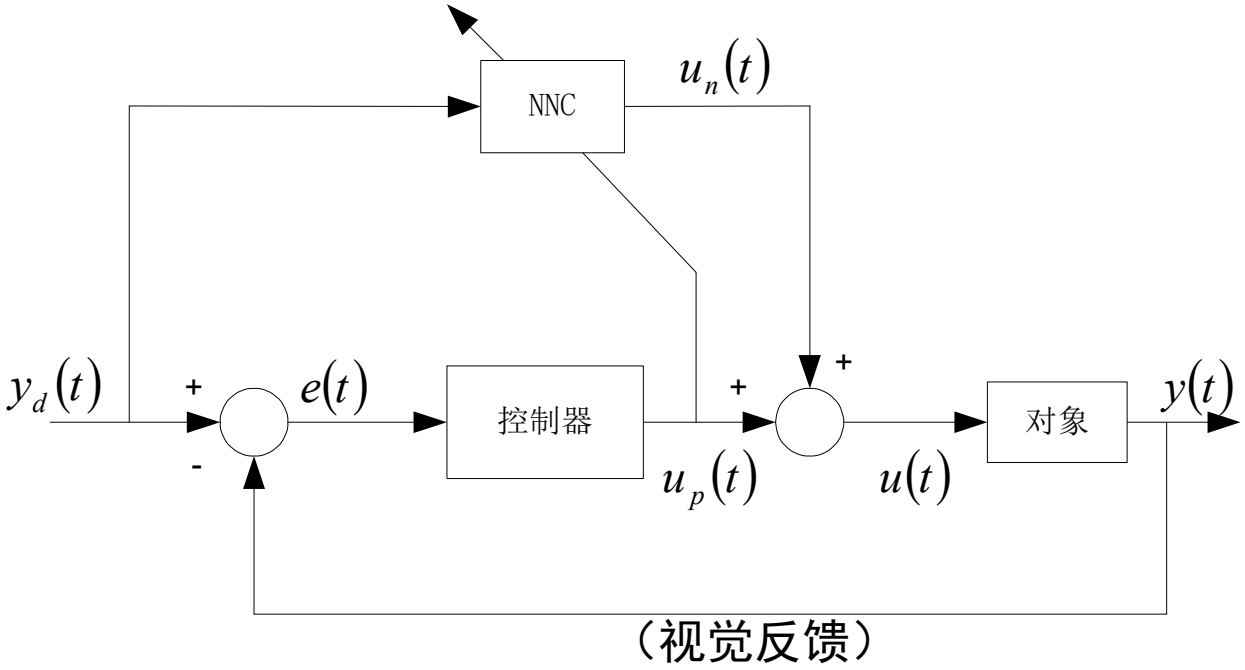
- 一类为神经控制，它是以神经网络为基础而形成的独立智能控制系统；
- 另一类为混合神经网络控制，它是指利用神经网络学习和优化能力来改善传统控制的智能控制方法，如自适应神经网络控制。

目前神经网络控制器尚无统一的分类方法。综合目前的各种分类方法，可将神经网络控制的结构归为7类。

9.2 神经网络控制结构

（一）神经网络监督控制

通过对人工或传统控制器（如PID）进行学习，然后用神经网络控制器逐渐取代原有控制器的方法（Werbos1990），称为神经网络监督控制。神经网络监督控制的结构如下图：



9.2 神经网络控制结构

- 神经网络控制器实际上是一个前馈控制器，它建立的是被控对象的逆模型。
- 神经网络控制器通过对传统控制器的输出进行学习，在线调整网络的权值，使反馈控制输入$e(t)$或$u_p(t)$趋近于零，从而使神经网络控制器逐渐在控制作用中占据主导地位，最终取消反馈控制器的作用。
- 一旦系统出现干扰，反馈控制器重新起作用。这种前馈加反馈的监督控制方法，不仅可以确保控制系统的稳定性和鲁棒性，而且可有效地提高系统的精度和自适应能力。

9.2 神经网络控制结构

（二）神经网络直接逆控制

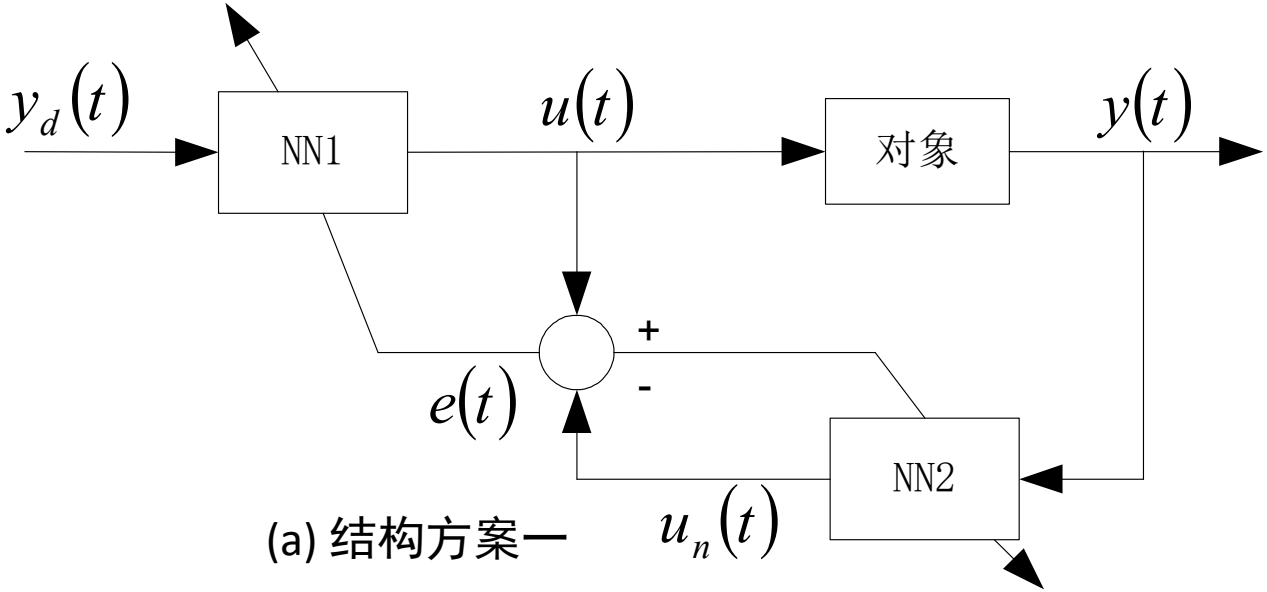
将被控对象的神经网络逆模型直接与被控对象串联起来，使期望输出yd(t)与对象实际输出y(t)之间的传递函数为1。从而再将此网络作为前馈控制器后，使被控对象的输出为期望输出。

显然，神经网络直接逆控制的可用性在相当程度上取决于逆模型的准确精度。由于缺乏反馈，简单连接的直接逆控制缺乏鲁棒性。为此，一般应使其具有在线学习能力，即作为逆模型的神经网络连接权能够在线调整。

9.2 神经网络控制结构

神经网络直接逆控制的两种结构方案：

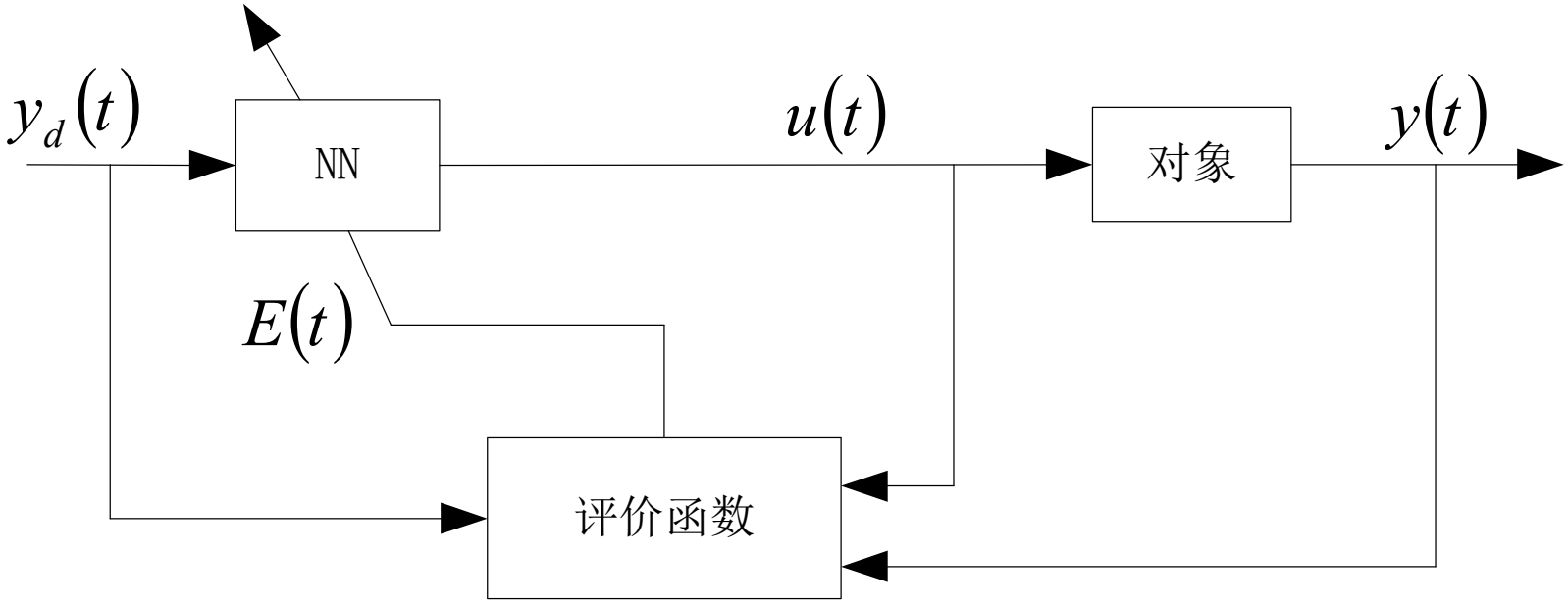
在方案一中，NN1和NN2具有完全相同的网络结构（逆模型），采用相同的学习算法，分别实现对象的逆。 该方法的可行性直接取决于逆模型辨识的准确程度，逆模型的连接权值必须在线修正，其成功的应用是机器人手臂的跟踪控制。



(a) 结构方案一

9.2 神经网络控制结构

在下图中，神经网络NN通过评价函数进行学习，实现对象的逆控制。



(b) 结构方案二

9.2 神经网络控制结构

（三）神经网络自适应控制

与传统自适应控制相同，神经网络自适应控制也分为神经网络自校正控制和神经网络模型参考自适应控制两种。

- 自校正控制根据对系统正向或逆模型的结果调节控制器内部参数，使系统满足给定的指标；
- 而在模型参考自适应控制中，闭环控制系统的期望性能由一个稳定的参考模型来描述。

9.2 神经网络控制结构

1　神经网络自校正控制

自校正控制系统是一种把参数的在线辨识与控制器的在线设计有机结合在一起的控制系统，并在设计辨识算法和控制算法时考虑了随机干扰的影响，因此，属于随机自适应控制系统。

神经网络自校正控制分为直接自校正控制和间接自校正控制。

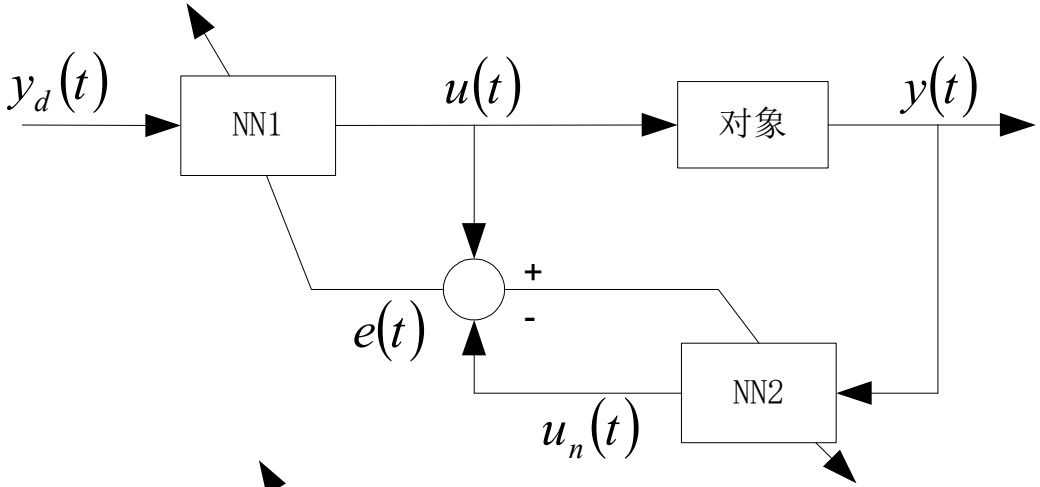
- 直接自校正控制也称为直接逆控制，同时使用神经网络控制器和神经网络估计器。
- 间接自校正控制使用常规控制器，神经网络估计器需要较高的建模精度。

9.2 神经网络控制结构

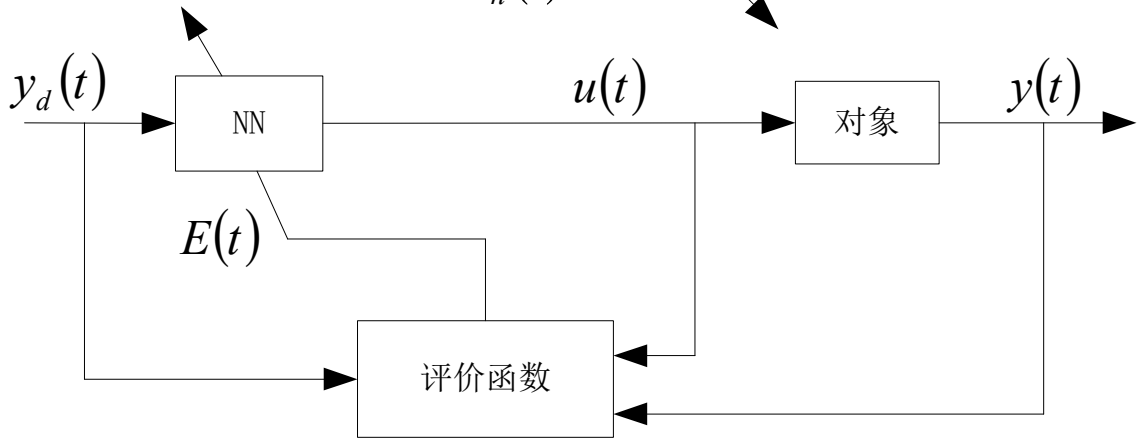
（1）神经网络直接自校正控制

在本质上和神经网络直接逆控制一样，其结构如图9-2所示。

(a) 结构方案一



(b) 结构方案二



9.2 神经网络控制结构

（2）神经网络间接自校正控制

其结构如图9-3所示。假设被控对象为如下单变量仿射非线性系统：

$$y(t)=f\left(y_t\right)+g\left(y_t\right)u(t)$$

若利用神经网络对非线性函数 $f\left(y_t\right)$ 和 $g\left(y_t\right)$ 进行逼近，得到 $\hat{f}\left(y_t\right)$ 和 $\hat{g}\left(y_t\right)$，则控制器为：

$$u\left(t\right)=\left[r(\mathrm{t})-\hat{f}\left(y_t\right)\right]/\hat{g}\left(y_t\right)$$

其中 $r\left(t\right)$ 为 t 时刻的期望输出值。

9.2 神经网络控制结构

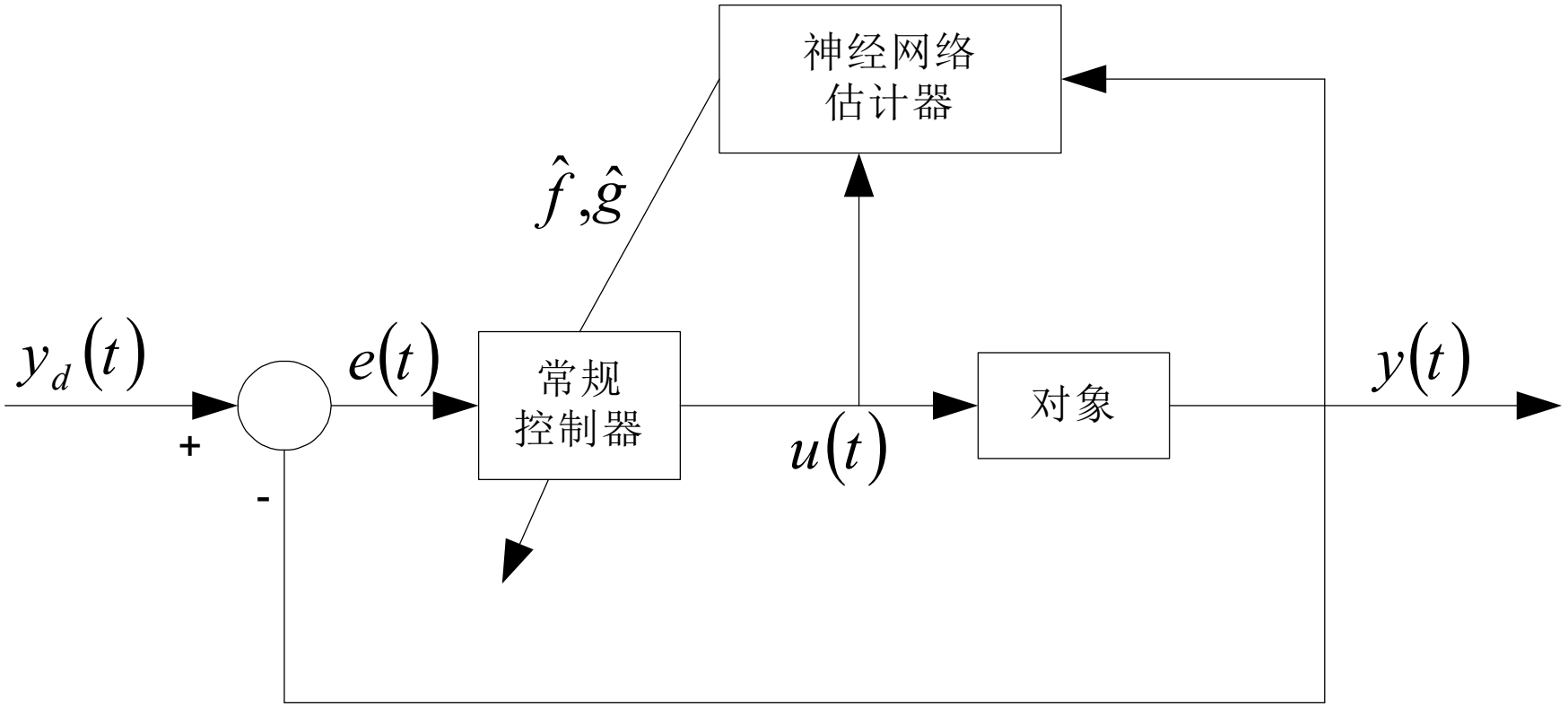


图9-3　神经网络间接自校正控制

9.2 神经网络控制结构

2 神经网络模型参考自适应控制

基于神经网络的非线性系统模型参考控制方案最早由Narendra等提出，分为直接模型参考自适应控制和间接模型参考自适应控制两种。

该方案将神经网络直接作为控制器，用系统实际输出和参考模型输出之间的误差来进行训练。因此，控制系统的作用是使被控系统的输出以最小的误差跟踪参考模型的输出。

9.2 神经网络控制结构

（1）直接模型参考自适应控制

如图9-4所示。神经网络控制器的作用是使被控对象与参考模型输出之差ec(t)=y(t)-y_m(t)→0或ec(t)的二次型最小。该方法需要知道对象的Jacobian信息$\frac{\partial y}{\partial u}$。

与正-逆建模中的方法类似，误差ec(t)的反向传播必须确知被控对象的数学模型，这给NNC的学习修正带来了许多问题。为解决这一问题，可采用间接模型参考控制。

9.2 神经网络控制结构

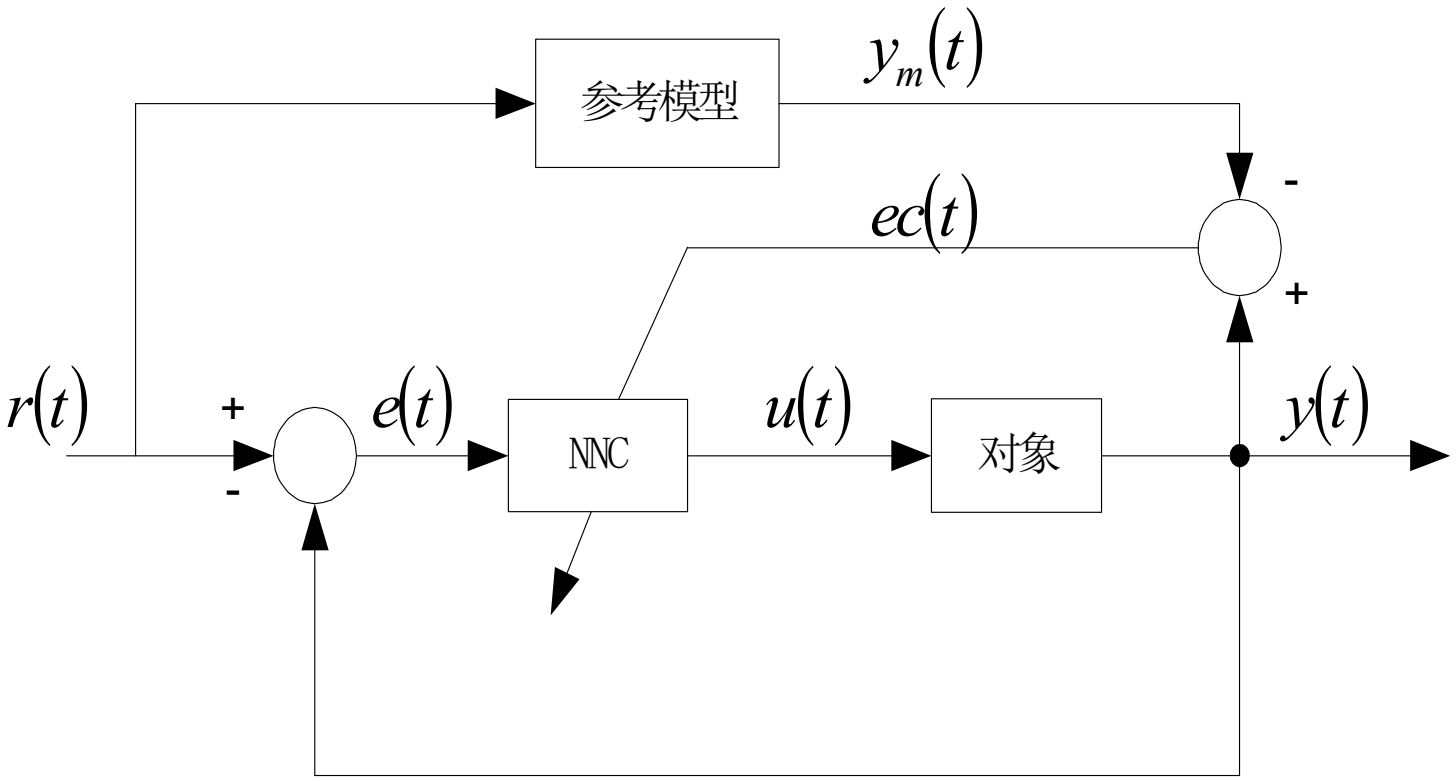


图9-4　神经网络直接模型参考自适应控制

9.2 神经网络控制结构

（2）间接模型参考自适应控制

如下图所示，神经网络辨识器NNI向神经网络控制器NNC提供对象的Jacobian信息，用于控制器NNC的学习。

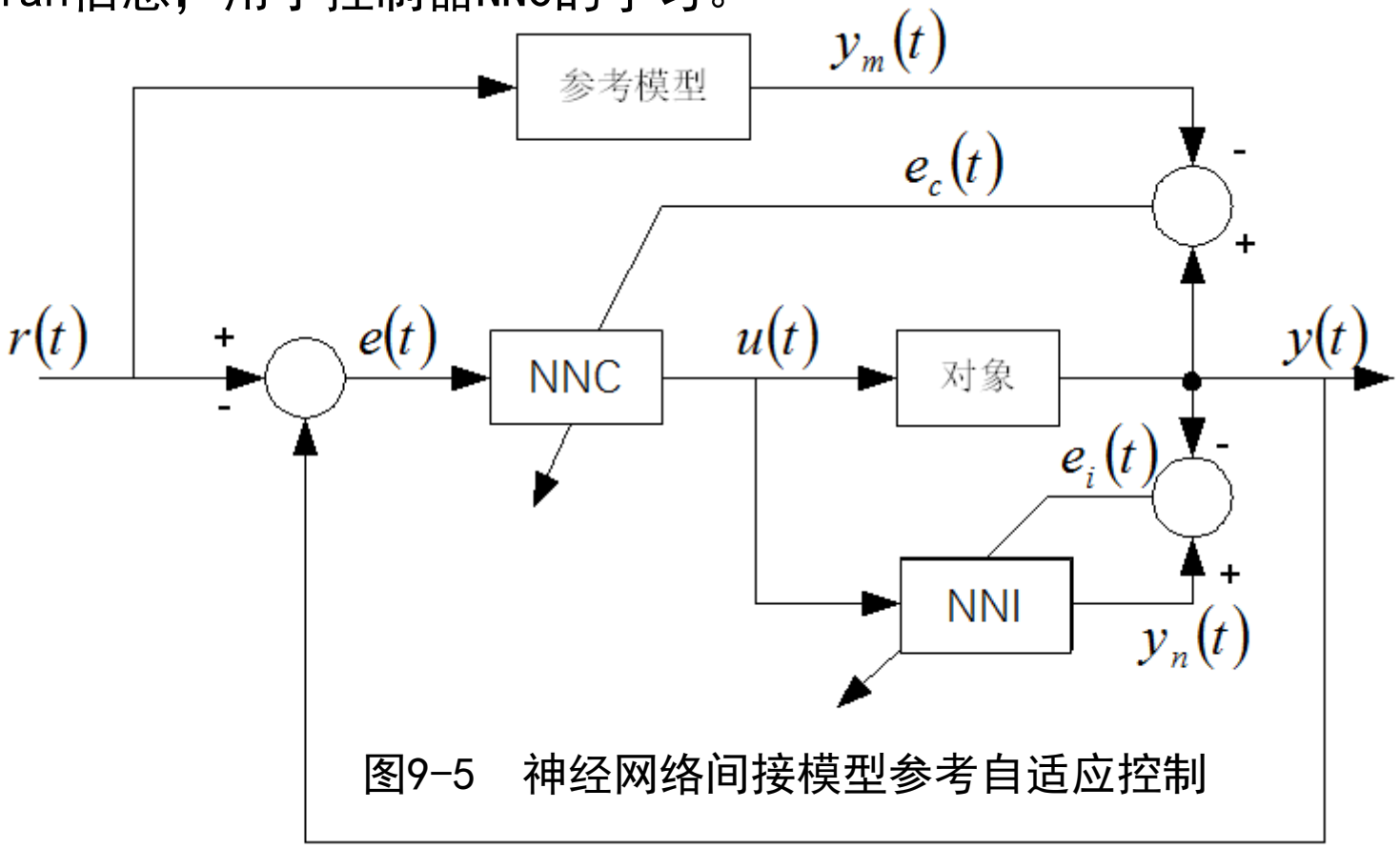


图9-5　神经网络间接模型参考自适应控制

9.2 神经网络控制结构

（四）神经网络内模控制

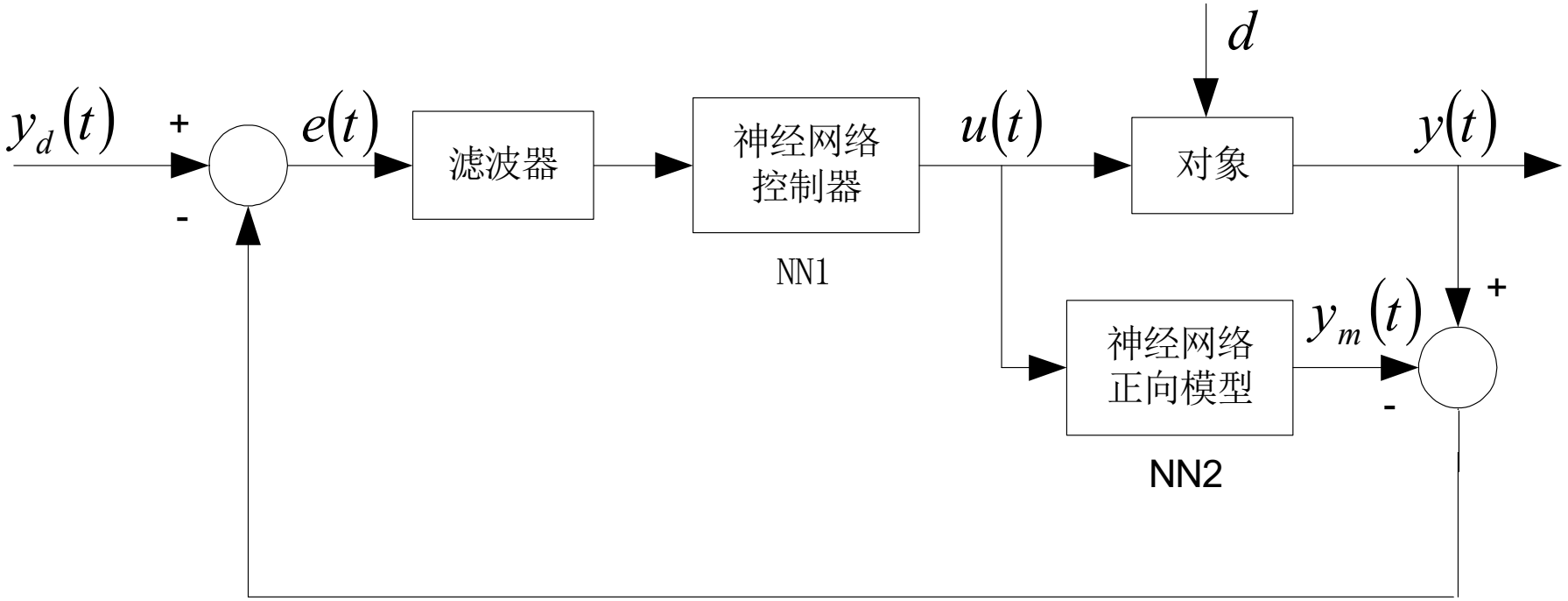
内模控制是一种采用系统对象的内部模型和反馈修正的预测控制，有较强鲁棒性，在线调整方便。经典的内模控制将被控系统的正向模型和逆模型直接加入反馈回路：

- 系统的正向模型作为被控对象的近似模型，与实际对象并联，两者输出之差被用作反馈信号，该反馈信号又经过前向通道的滤波器（线性滤波器）及控制器进行处理。
- 控制器直接与系统的逆有关，通过引入滤波器来提高系统的鲁棒性。

应当注意，内模控制的应用仅限于开环稳定的系统。这一技术已广泛地应用于过程控制中，其中，Hunt和Sharbam等人(1990)实现了非线性系统的神经网络内模控制。

9.2 神经网络控制结构

下图为神经网络内模控制框图，被控对象的正向模型及控制器均由神经网络来实现，NN1实现对象的逆（控制器），NN2实现对象的逼近（正向模型）。



9.2 神经网络控制结构

（五）神经网络预测控制

- 预测控制又称为基于模型的控制，是70年代后期发展起来的新型计算机控制方法，该方法的特征是预测模型、滚动优化和反馈校正。
- 神经网络预测器建立了非线性被控对象的预测模型，并可在线进行学习修正。
- 利用此预测模型，通过设计优化性能指标，利用非线性优化器可求出优化的控制作用$u(t)$。

9.2 神经网络控制结构

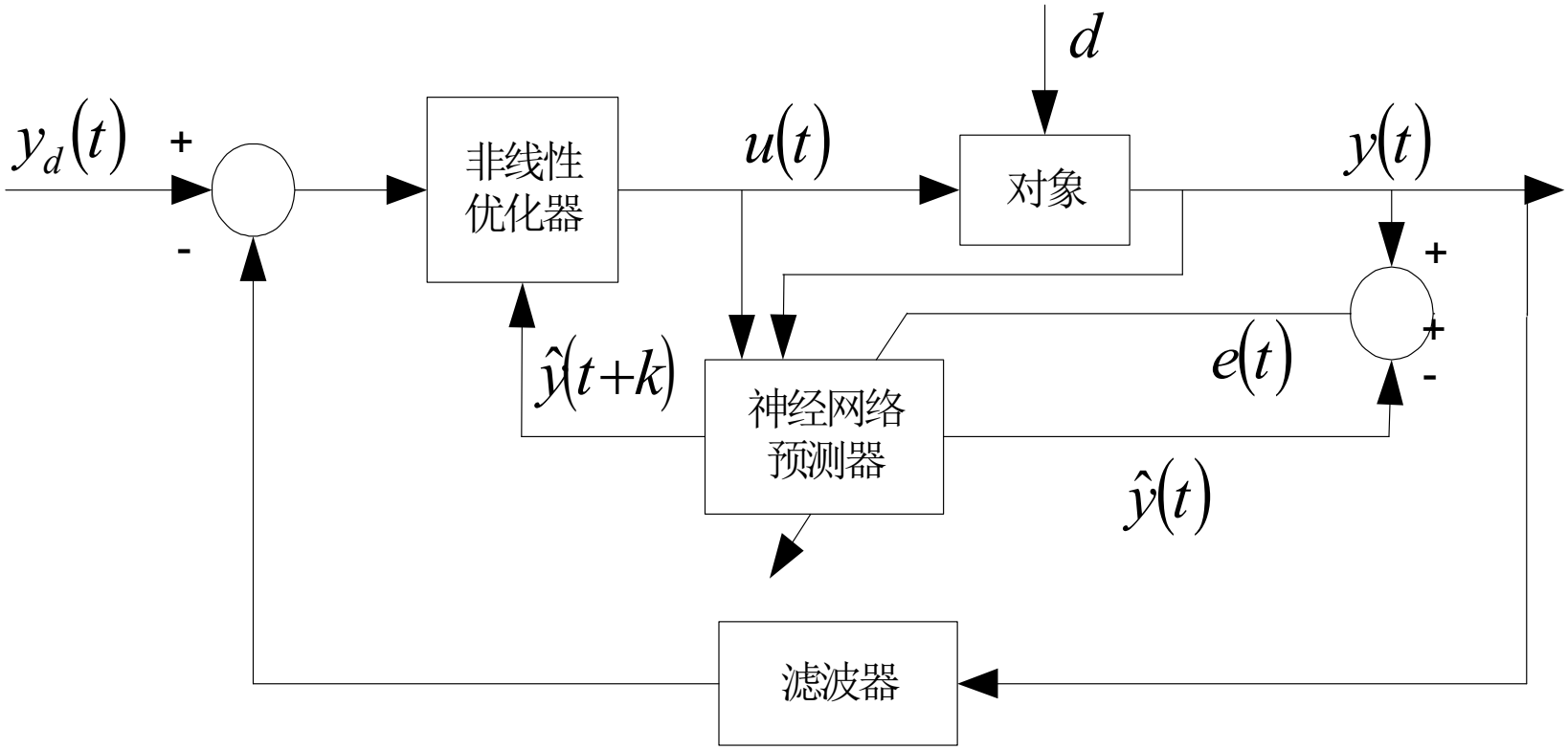


图9-7　神经网络预测控制

9.2 神经网络控制结构

- 预测模型根据系统的历史信息和选定的未来输入，预测系统未来的输出。
- 根据预测模型的输出，控制系统采用基于优化的控制策略对被控对象进行控制。与通常的最优控制策略不同，预测控制系统采用的是滚动式的有限时域优化策略，即优化过程不是通过离线计算一次得到，而是在线反复进行，所得到的只是全局次优解。但由于滚动实时变化，对模型时变、干扰和失配等影响能及时补偿，故在复杂工业环节中更为实际有效。
- 对于预测模型与被控对象实际输出间的偏差，采用反馈校正环节进行修正，即在每一采样时刻，用实测偏差对模型预测的未来输出进行校正，并按修正后的预测输出进行滚动优化，计算控制律。

9.2 神经网络控制结构

（六）神经网络自适应评判控制

神经网络自适应评判控制通常由两个网络组成（图9-8）：

- 自适应评判网络通过不断的奖励、惩罚等再励学习，使自己逐渐成为一个合格的“教师”，学习完成后，根据系统目前的状态和外部激励反馈信号$r(t)$产生一个内部再励信号$\hat{r}(t)$，以对目前的控制效果作出评价。
- 控制选择网络相当于一个在内部再励信号$\hat{r}(t)$指导下进行学习的多层前馈神经网络控制器，该网络在进行学习后，根据编码后的系统状态，在允许控制集中选择下一步的控制作用。

9.2 神经网络控制结构

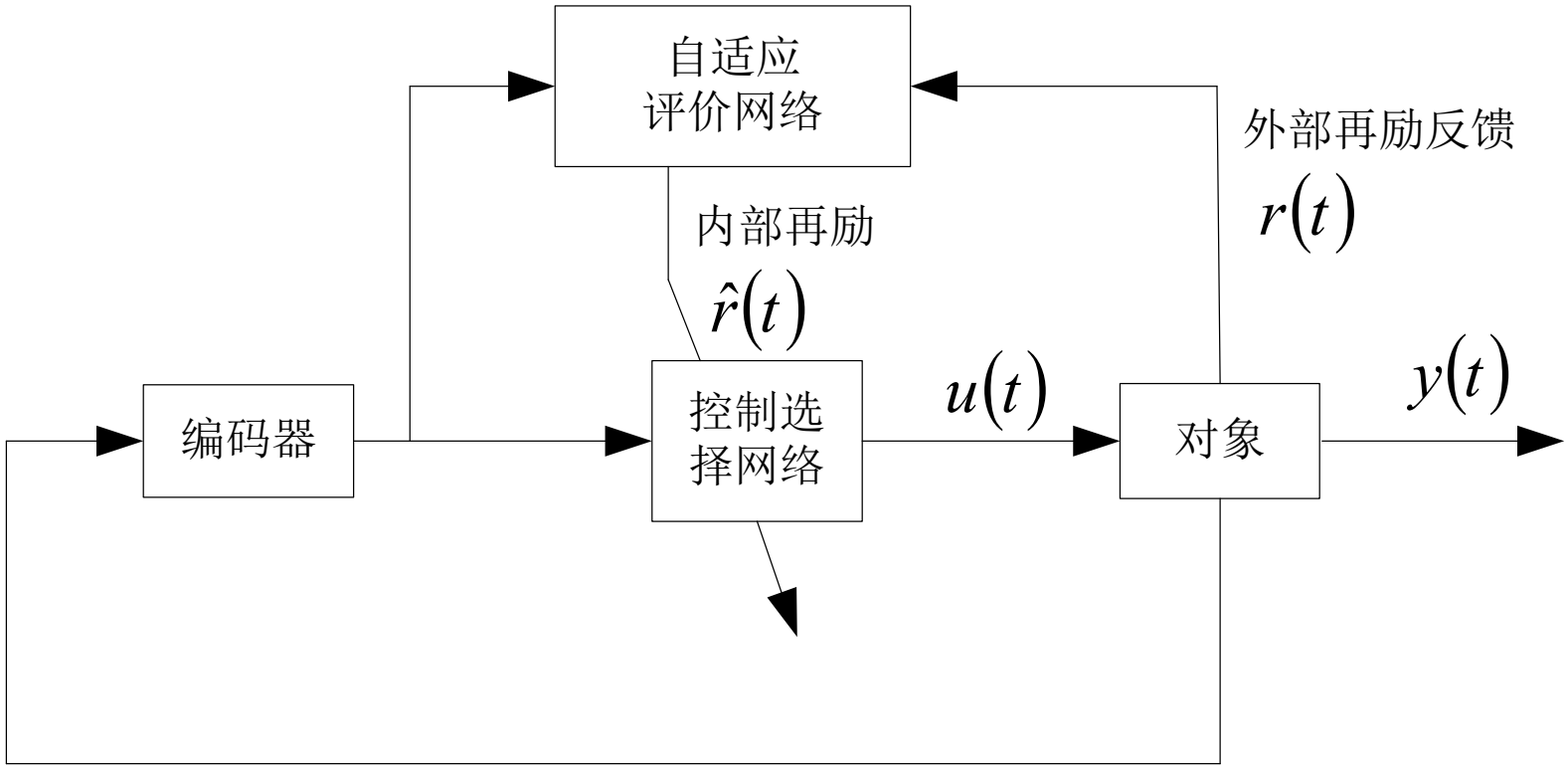


图9-8　神经网络自适应评判控制

9.2 神经网络控制结构

（七）神经网络混合控制

该控制方法是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。

由神经网络和模糊控制相结合构成模糊神经网络，由神经网络和专家系统相结合构成神经网络专家系统。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。

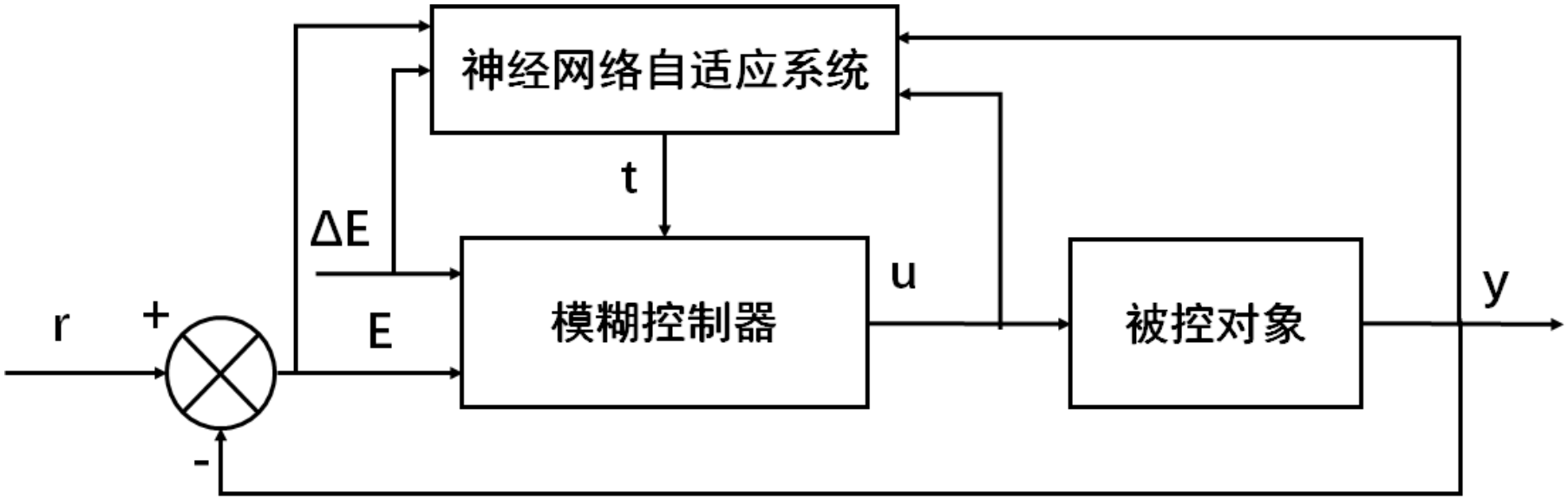
9.2 神经网络控制结构

模糊神经网络基本上有两种结构：

- 一种是神经网络作为模糊控制器的自适应机构，利用神经网络的自学习能力调整控制器的参数，改善控制系统性能；
- 另一种是直接采用神经网络实现模糊控制器，利用神经网络的联想记忆能力形成模糊决策规则，并利用神经网络的自学习能力自动调整网络连接权值，达到调整模糊决策规则的目的，改善控制性能。

下面简单介绍第一种结构。

9.2 神经网络控制结构



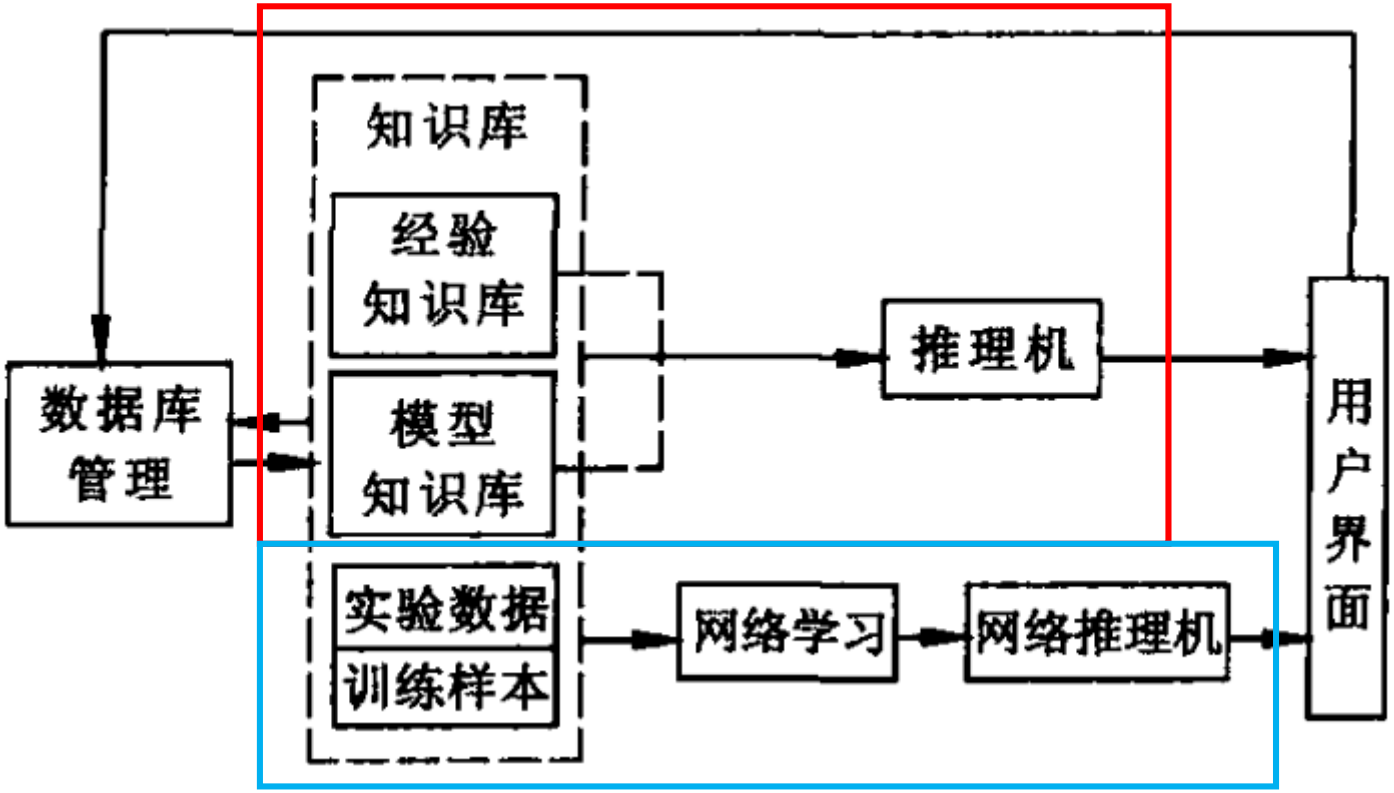
基于神经网络的自适应模糊控制系统

9.2 神经网络控制结构

在这个结构中，模糊控制器根据模糊控制规则（或模糊控制表）完成由论域$E\times\Delta E$到论域U的映射，实现在不确定性环境下的决策和对对象的控制。

但是，模糊控制器本身不具有学习能力。神经网络实现修改模糊规则或修改控制器的输入、输出比例系数的功能，神经网络对控制误差E、误差变化率ΔE以及控制性能进行综合后，向模糊控制器提供一个“教师”信息t，去调整模糊控制器的参数或规则。神经网络的自学习能力起到自适应机构的作用，指导模糊控制器完成对复杂的不确定对象的控制。

9.2 神经网络控制结构



9.3 单神经元网络控制

（一）单神经元自适应控制算法

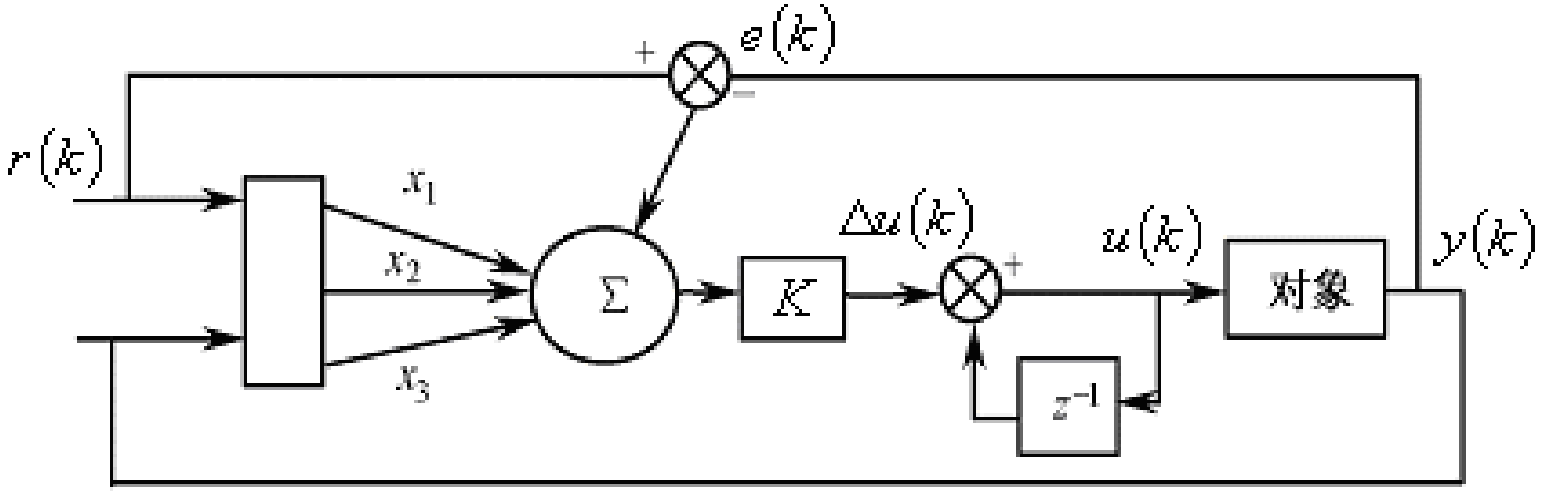
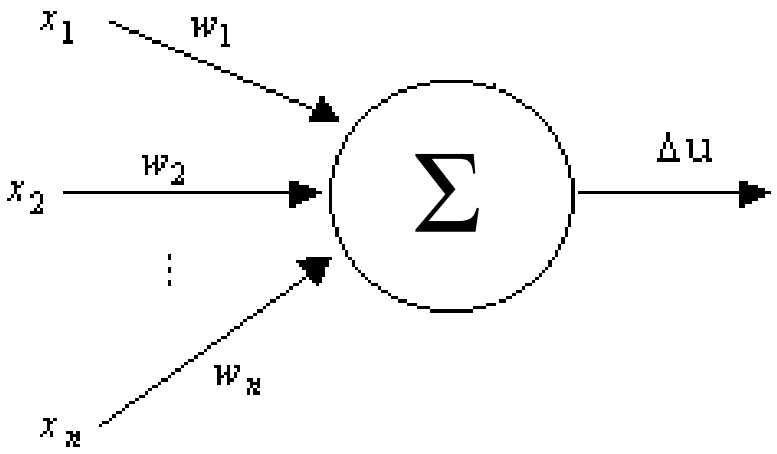


图9-9 单神经元自适应PID控制结构

9.3 单神经元网络控制

单神经元自适应控制器是通过对加权系数的调整来实现自适应、自组织功能，控制算法为

$$u(k) = u(k-1) + K\sum_{i=1}^{3} w_i(k)x_i(k)$$



单神经网络控制器

9.3 单神经元网络控制

如果权系数的调整按有监督的Hebb学习规则实现，即在学习算法中加入监督项$z(k)$，则神经网络权值学习算法为：

$$w_1(k) = w_1(k-1) + \eta z(k)u(k)x_1(k)$$
$$w_2(k) = w_2(k-1) + \eta z(k)u(k)x_2(k)$$
$$w_3(k) = w_3(k-1) + \eta z(k)u(k)x_3(k)$$

9.3 单神经元网络控制

式中，$z(k)=e(k)$

$$x_1(k)=e(k)$$

$$x_2(k)=e(k)-e(k-1)$$

$$x_3(k)=\Delta^2 e(k)=e(k)-2e(k-1)+e(k-2)$$

η为学习速率，K为神经元的比例系数，$K>0$，$\eta\in(0,1)$。

K值的选择非常重要。K越大，则快速性越好，但超调量大，甚至可能使系统不稳定。当被控对象时延增大时，K值必须减少，以保证系统稳定。K值选择过小，会使系统的快速性变差。

9.3 单神经元网络控制

（二）仿真实例

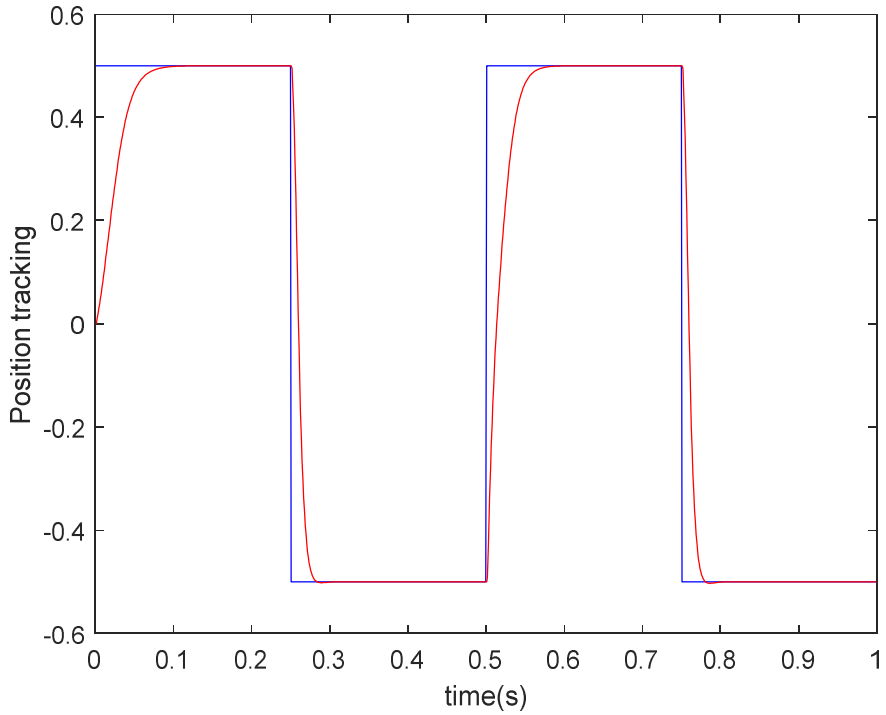
被控对象为

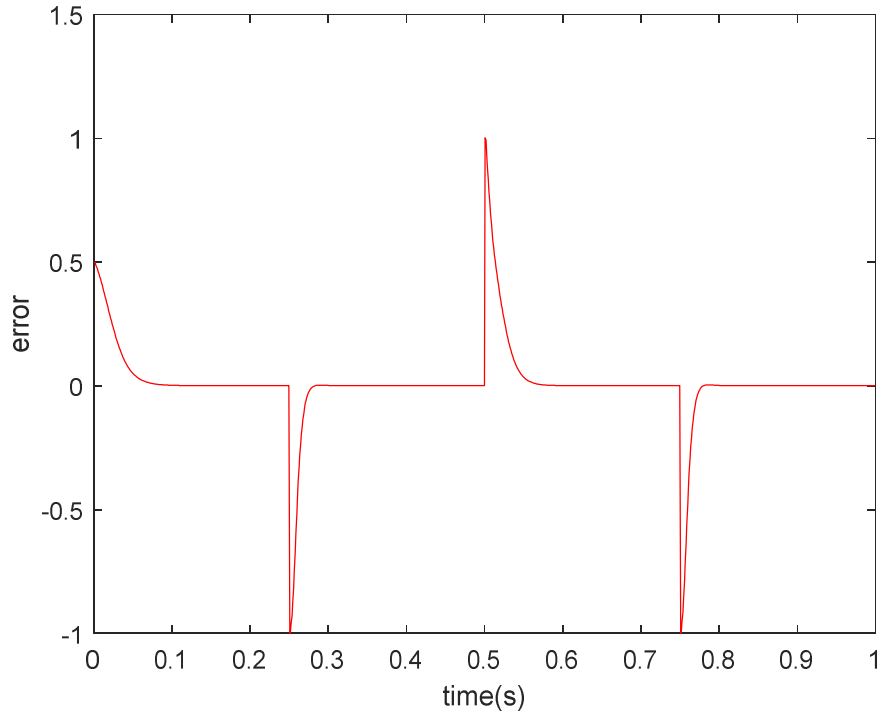
$$y(k) = 0.368y(k-1) + 0.26y(k-2) + 0.10u(k-1) + 0.632u(k-2)$$

输入指令为一方波信号 $r(k) = 0.5\mathrm{sgn}(\sin(4\pi t))$

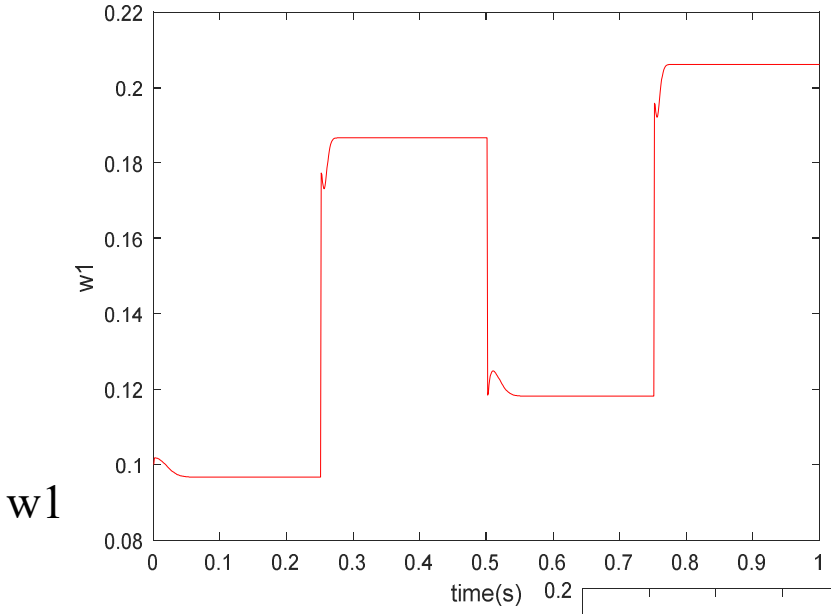
采样时间为1ms，采用单神经元自适应控制律进行控制。初始权值取W=[w1, w2, w3]=[0.1, 0.1, 0.1]，η=0.4，K=0.12。

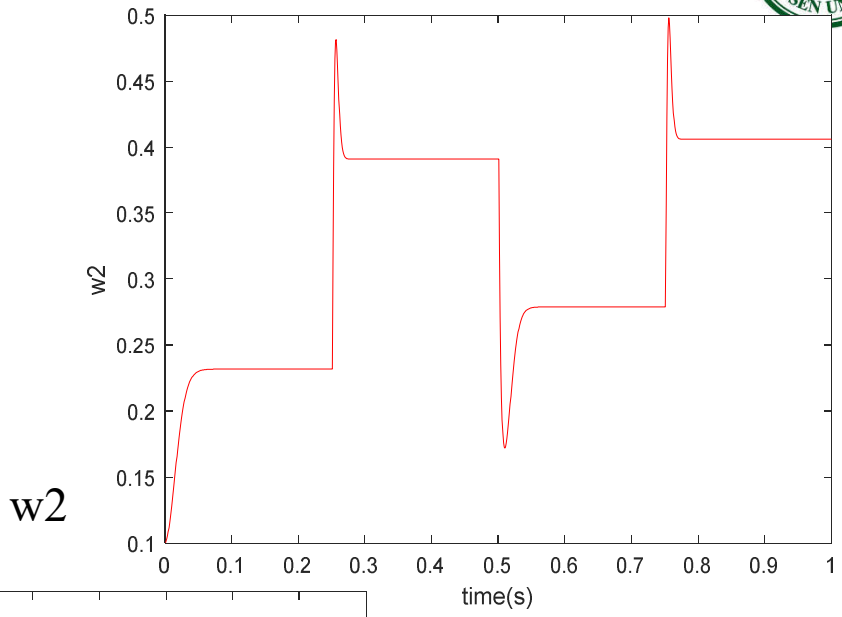
9.3 单神经元网络控制

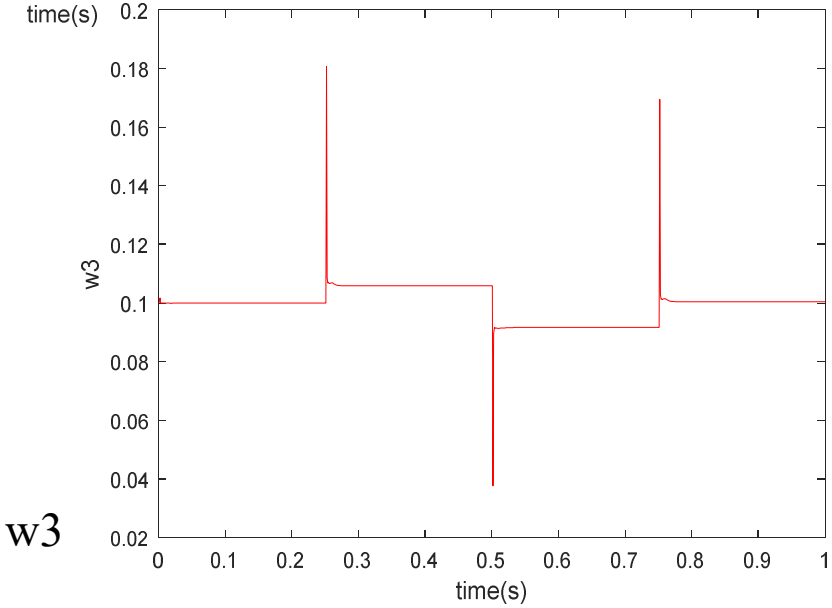

Position tracking
time(s)
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


error
time(s)
1.5
1
0.5
0
-0.5
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

9.3 单神经元网络控制


w1
0.22
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
time(s)

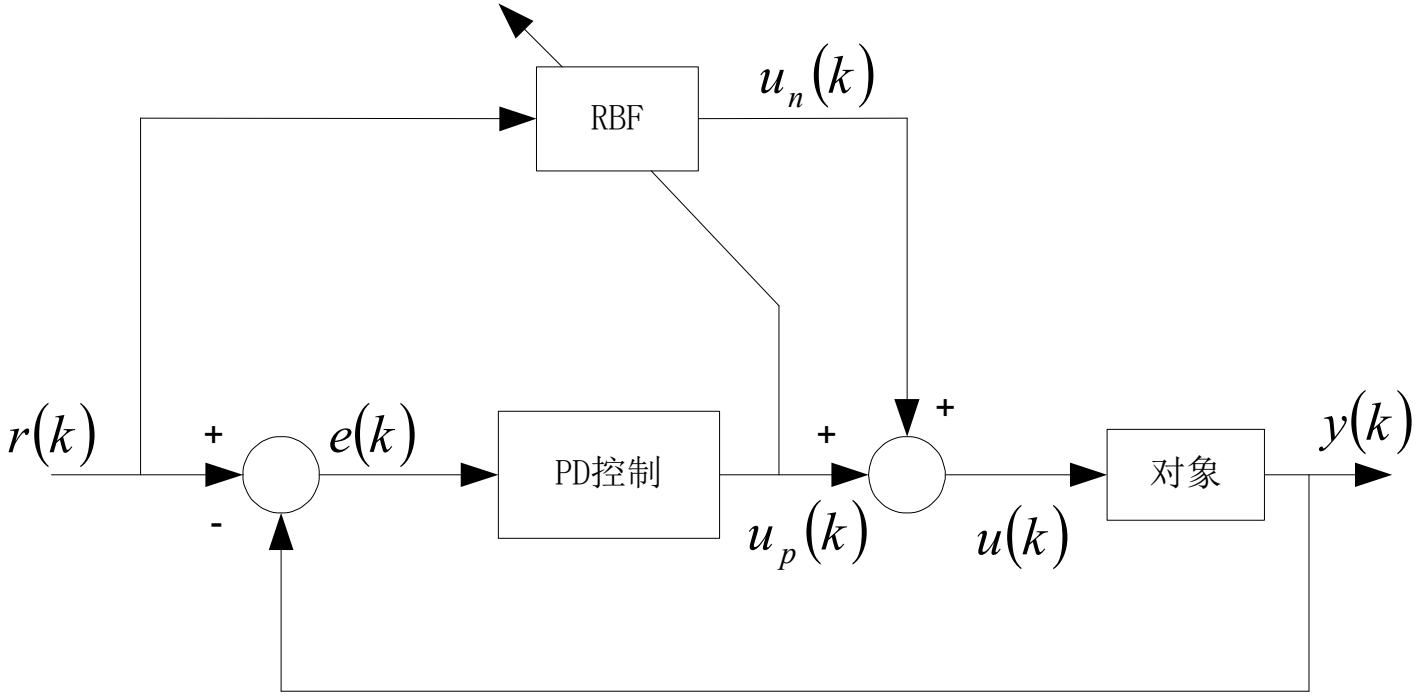

w2
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
time(s)


w3
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
time(s)

9.4 RBF网络监督控制

（一）RBF网络监督控制算法

基于RBF网络的监督控制系统结构如下图所示。



9.4 RBF网络监督控制

在RBF网络结构中，取网络的输入为$r(k)$，网络的径向基向量为$\mathbf{H}=[h_1, \cdots, h_m]^T$

h_j为高斯基函数：

$$h_j = \exp(-\frac{\|r(k)-\mathbf{C}_j\|^2}{2b_j^2})$$

网络的权向量为：

$$\mathbf{W}=[w_1, \cdots, w_m]^T$$

RBF网络的输出为：

$$u_n(k)=h_1w_1+\cdots h_jw_j\cdots+h_mw_m$$

其中m为RBF网络隐层神经元的个数。

9.4 RBF网络监督控制

控制律为：

$$u(k)=u_p(k)+u_n(k)$$

设神经网络调整的性能指标为：

$$E(k)=\frac{1}{2}(u_n(k)-u(k))^2$$

近似地取

$$\frac{\partial u_p(k)}{\partial w_j(k)}=\frac{\partial u_n(k)}{\partial w_j(k)}$$

由此所产生的不精确通过权值调节来补偿。

9.4 RBF网络监督控制

采用梯度下降法调整网络的权值：

$$\Delta w_j(k) = -\eta \frac{\partial E(k)}{\partial w_j(k)} = \eta\big(u_n(k) - u(k)\big)\frac{\partial u_p(k)}{\partial w_j(k)}$$
$$= \eta\big(u_n(k) - u(k)\big)h_j(k)$$

神经网络权值的调整过程为：

$$\mathbf{W}(k) = \mathbf{W}(k-1) + \Delta\mathbf{W}(k) + \alpha\big(\mathbf{W}(k-1) - \mathbf{W}(k-2)\big)$$

其中η为学习速率，α为动量因子。

9.4 RBF网络监督控制

（二）仿真实例

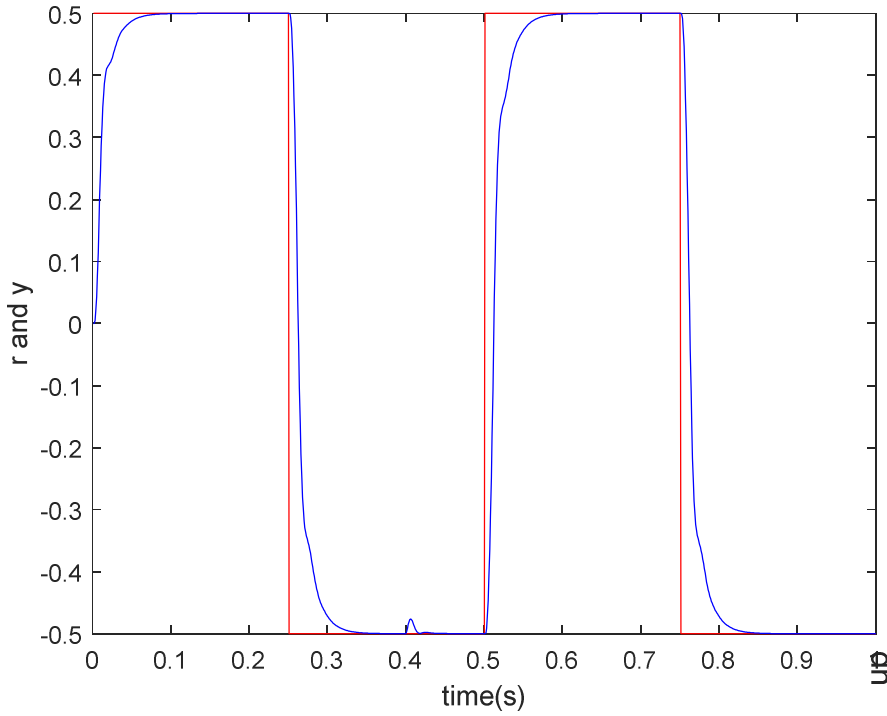
被控对象为：$G(s)=\dfrac{1000}{s^2+50s+2000}$

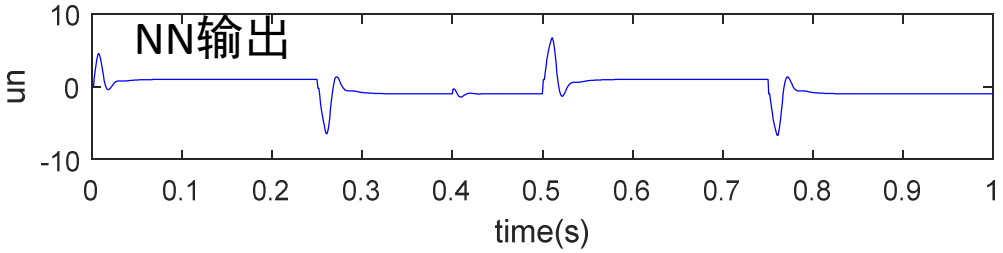
取采样时间为1ms，采用z变换进行离散化，得到

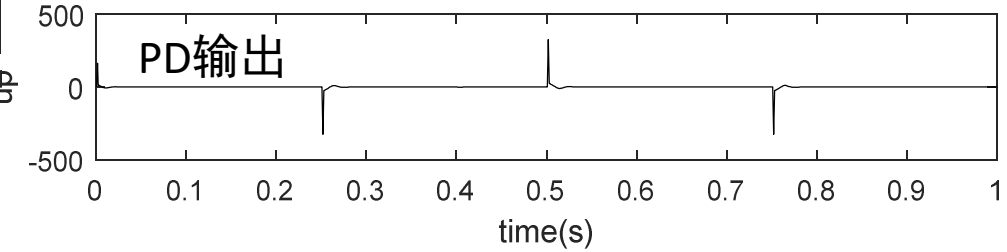
y(k)=-den(2)y(k-1)-den(3)y(k-2)+num(2)u(k-1)+num(3)u(k-2)

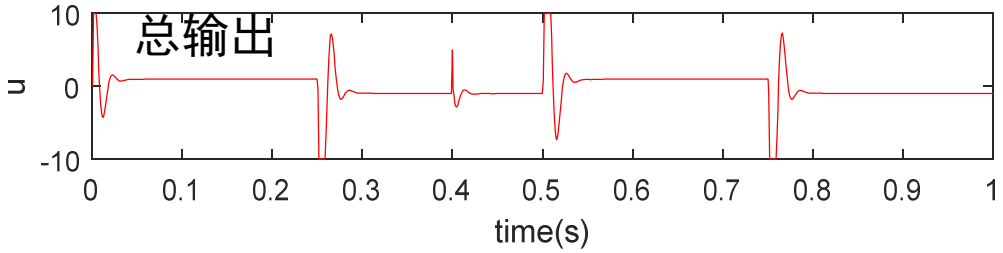
指令信号幅值为0.5，频率为2Hz的方波信号。取指令信号r(k)作为网络的输入，网络隐层神经元个数取m=4，网络结构为1-4-1，网络的初始权值W取0~1之间的随机值，高斯函数的参数值取C_j=[-2, -1, 1, 2]^T，B=[0.5, 0.5, 0.5, 0.5]^T。学习参数为η=0.3，α=0.05。

9.4 RBF网络监督控制









9.5 RBF网络自校正控制

（一）神经网络自校正控制原理

自校正控制有两种结构：直接型与间接型。

- 直接型自校正控制也称直接逆动态控制，是前馈控制。
- 间接自校正控制是一种由辨识器将对象参数进行在线估计，用调节器（或控制器）实现参数的自动整定相结合的自适应控制技术，可用于结构已知而参数未知但恒定的随机系统，也可用于结构已知而参数缓慢时变的随机系统。

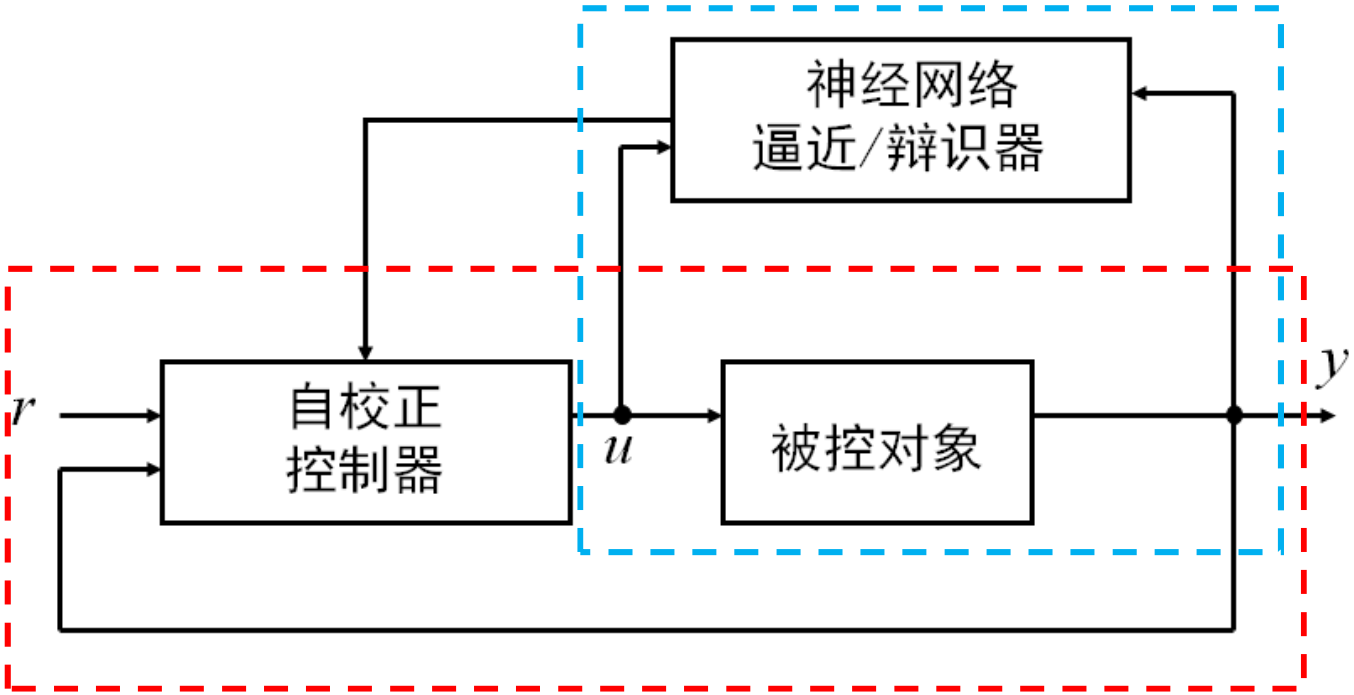
9.5 RBF网络自校正控制

神经间接自校正控制结构如下图所示，它由两个回路组成：

（1）自校正控制器与被控对象构成的反馈回路。

（2）神经网络辩识器的设计，以得到控制器的参数。

辩识器与自校正控制器的在线设计是自校正控制实现的关键。



9.5 RBF网络自校正控制

（二）自校正控制算法

考虑被控对象：

$$y(k+1)=g[y(k)]+\varphi[y(k)]u(k) \qquad (9.8)$$

其中u，y分别为对象的输入、输出，$\varphi[\bullet]$为非零函数。

若$g[\bullet]$，$\varphi[\bullet]$已知，根据“确定性等价原则”，控制器的控制算法为：

$$u(k)=\frac{-g[\bullet]}{\varphi[\bullet]}+\frac{r(k+1)}{\varphi[\bullet]}$$

将控制器代入（9.8），则系统的输出y(k)能精确地跟踪输入r(k)。

9.5 RBF网络自校正控制

若 $g[\bullet]$，$\varphi[\bullet]$ 未知，则通过在线训练神经网络辨识器，由辨识器结果 $Ng[\bullet]$、$N\varphi[\bullet]$来代替 $g[\bullet]$ 、$\varphi[\bullet]$，此时自校正控制器的控制算法为：

$$u(k)=\frac{-Ng[\bullet]}{N\varphi[\bullet]}+\frac{r(k+1)}{N\varphi[\bullet]}$$

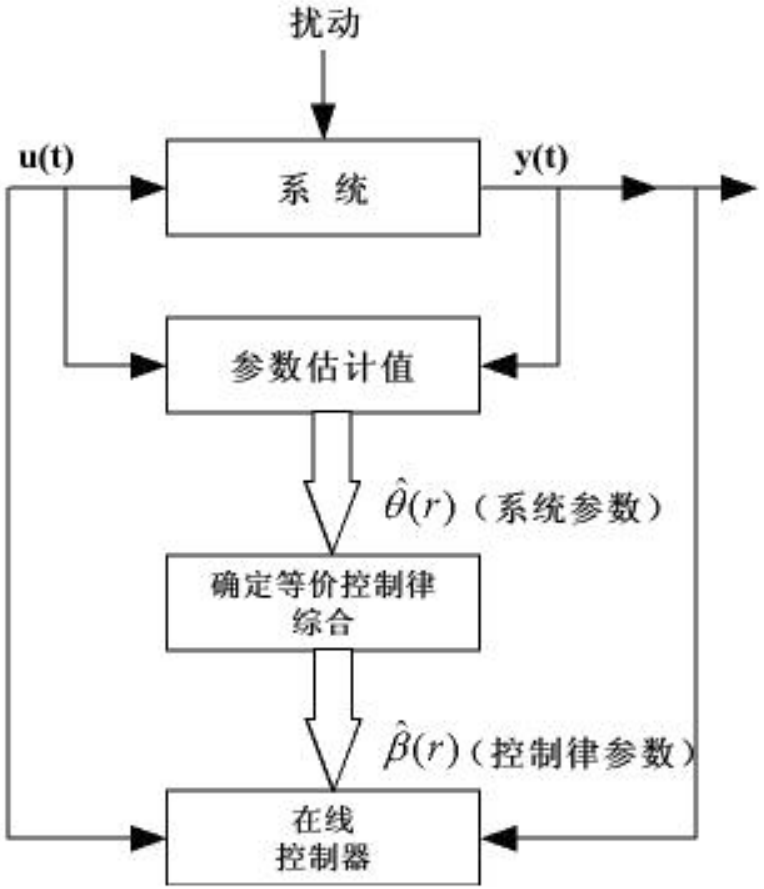
式中，$Ng[\bullet]$ 和 $N\varphi[\bullet]$分别为神经网络的输出。

9.5 RBF网络自校正控制

当设计控制器时，最新估计出的参数值被当作真实值应用在控制规律中。即在设计中没有考虑估计的不确定性，这被称之为确定性等价准则。

基于确定性等价准则的自适应控制器的参数估计和控制器设计是分离的，即在设计控制器时忽略了参数估计的不确定性。因此，任何参数估计算法都可以和任意控制律相结合，产生出相应的确定性等价自适应控制器。

确定性等价控制结构框图如右图所示。



9.5 RBF网络自校正控制

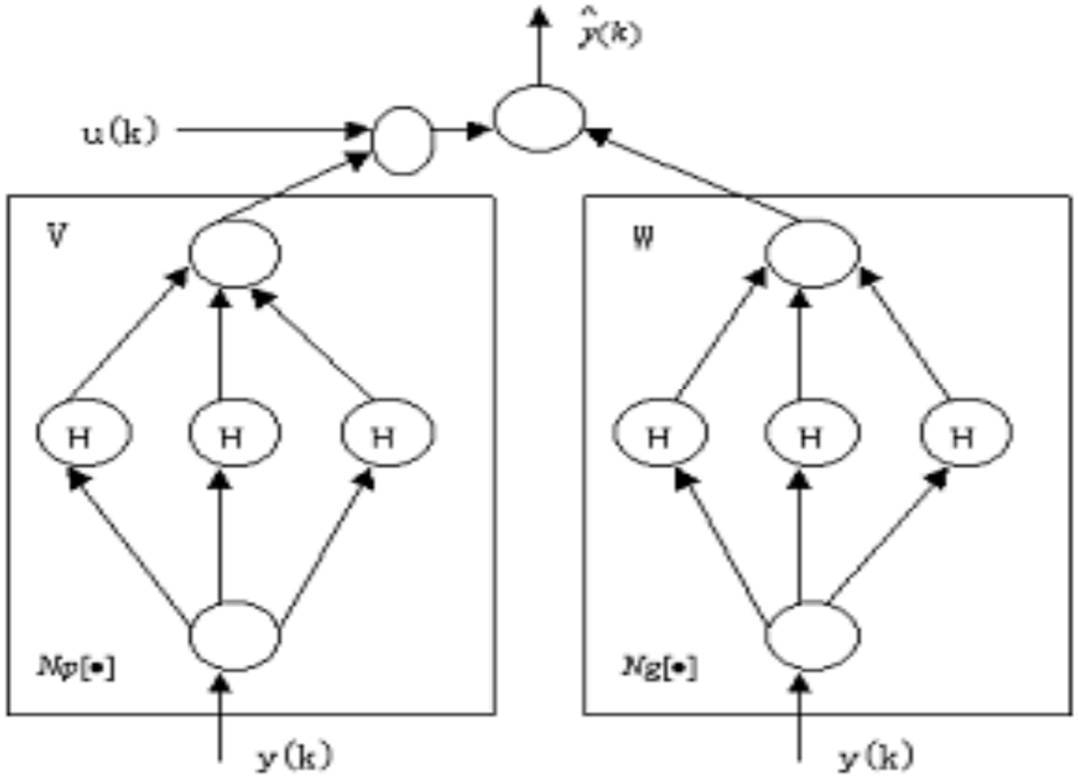
其最大的优点就是计算简单，当系统参数估计的精确性要求可以忽略不计时，确定性等价准则是设计自适应控制算法的最佳选择。几乎所有的传统自适应控制算法都是而且都是基于确定性等价准则的。

然而，大多数自适应控制规律都是系统参数估计值的函数，估计精度的高低会限制影响这些自适应控制器的控制效果。在工业过程控制应用中，系统往往存在较大的不确定性，系统在快速调节过程中(如控制器启动和参数变化时)，过渡过程会出现大超调。大超调和系统状态的震荡会导致机械设备过早的磨损和报废。当估计精度过差时甚至会导致系统失稳。这一缺点限制了基于确定性等价准则的自适应控制算法在实际工程中的应用。

9.5 RBF网络自校正控制

（三）RBF网络自校正控制算法

采用两个RBF网络分别实现未知项g[·]、φ[·]的辨识。RBF网络辨识器的结构如下图所示，**W**和**V**分别为两个神经网络的权值向量。



9.5 RBF网络自校正控制

在RBF网络结构中，取网络的输入为y(k)，网络的径向基向量为 $\mathbf{H}=[h_1, \cdots, h_m]^T$，$h_j$为高斯基函数：

$$h_j = \exp(-\frac{\|y(k)-\mathbf{C}_j\|^2}{2b_j^2})$$

其中 $j=1,\cdots,m$。 b_j为节点 j 的基宽度参数，$b_j>0$，C_j为网络第 j个结点的中心矢量，

$$\mathbf{C}_j = [c_{11},\cdots,c_{1m}] \qquad \mathbf{B} = [b_1,\cdots,b_m]^T$$

9.5 RBF网络自校正控制

网络的权向量为：$\mathbf{W}=[w_1,\ \cdots,\ w_m]^{\mathrm{T}}$

$$\mathbf{V}=[v_1,\ \cdots,\ v_m]^{\mathrm{T}}$$

两个RBF网络的输出分别为：

$$Ng(k)=h_1w_1+\cdots h_jw_j\cdots+h_mw_m$$

$$N\varphi(k)=h_1v_1+\cdots h_jv_j\cdots+h_mv_m$$

其中m为RBF网络隐层神经元的个数。

辨识后，对象的输出估计值为：

$$\hat{y}(k)=Ng\left[y(k-1);W(k)\right]+N\varphi\left[y(k-1);V(k)\right]u(k-1)$$

9.5 RBF网络自校正控制

设神经网络调整的性能指标为：

$$E(k)=\frac{1}{2}(y(k)-y_m(k))^2$$

采用梯度下降法调整网络的权值：

$$\Delta w_j(k)=-\eta_w\frac{\partial E(k)}{\partial w_j(k)}=\eta_w(y(k)-y_m(k))h_j(k)$$

$$\Delta v_j(k)=-\eta_v\frac{\partial E(k)}{\partial v_j(k)}=\eta_v(y(k)-y_m(k))h_j(k)$$

其中η_w和η_v为学习速率，α为动量因子。

神经网络权值的调整过程为：

$$\mathbf{W}(k)=\mathbf{W}(k-1)+\Delta\mathbf{W}(k)+\alpha(\mathbf{W}(k-1)-\mathbf{W}(k-2))$$

$$\mathbf{V}(k)=\mathbf{V}(k-1)+\Delta\mathbf{V}(k)+\alpha(\mathbf{V}(k-1)-\mathbf{V}(k-2))$$

9.5 RBF网络自校正控制

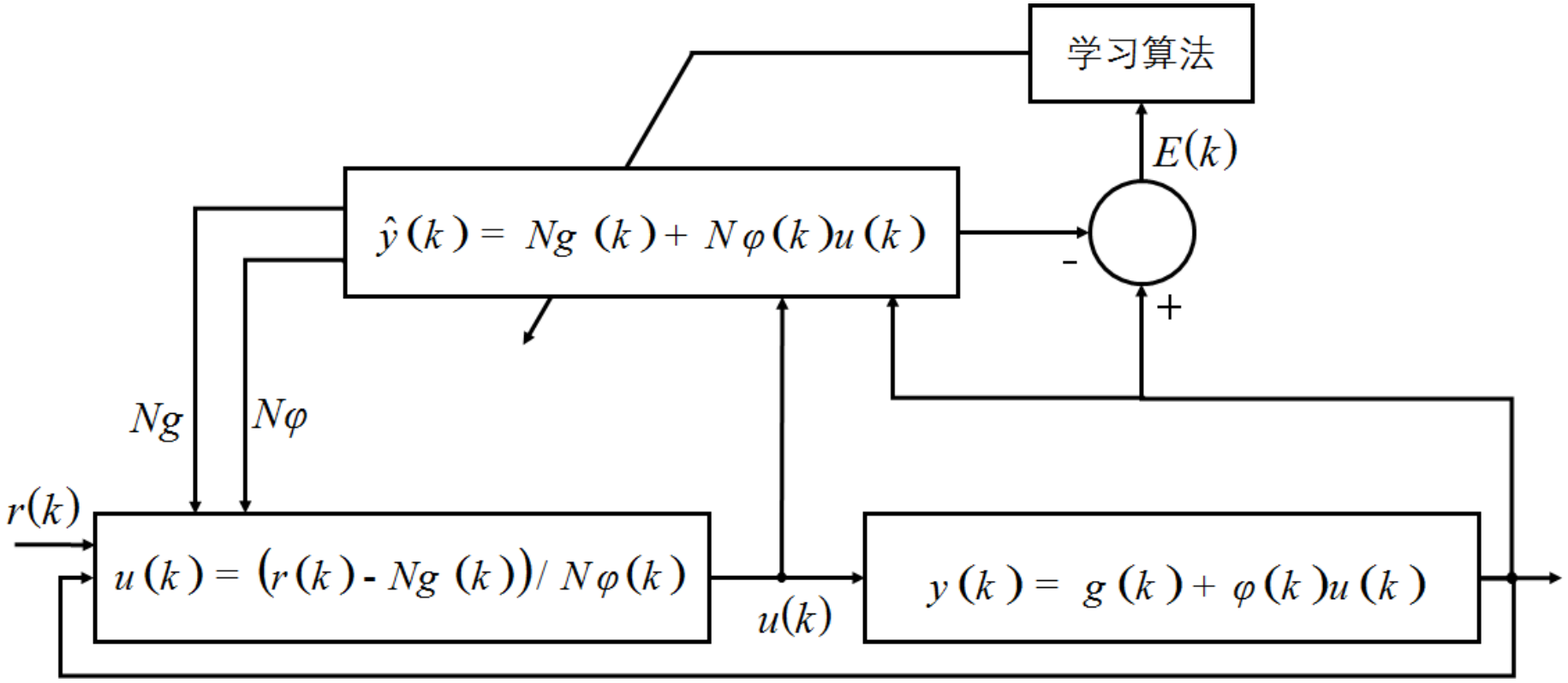


图9-19 神经网络自校正控制框图

9.5 RBF网络自校正控制

（四）仿真实例

被控对象为：

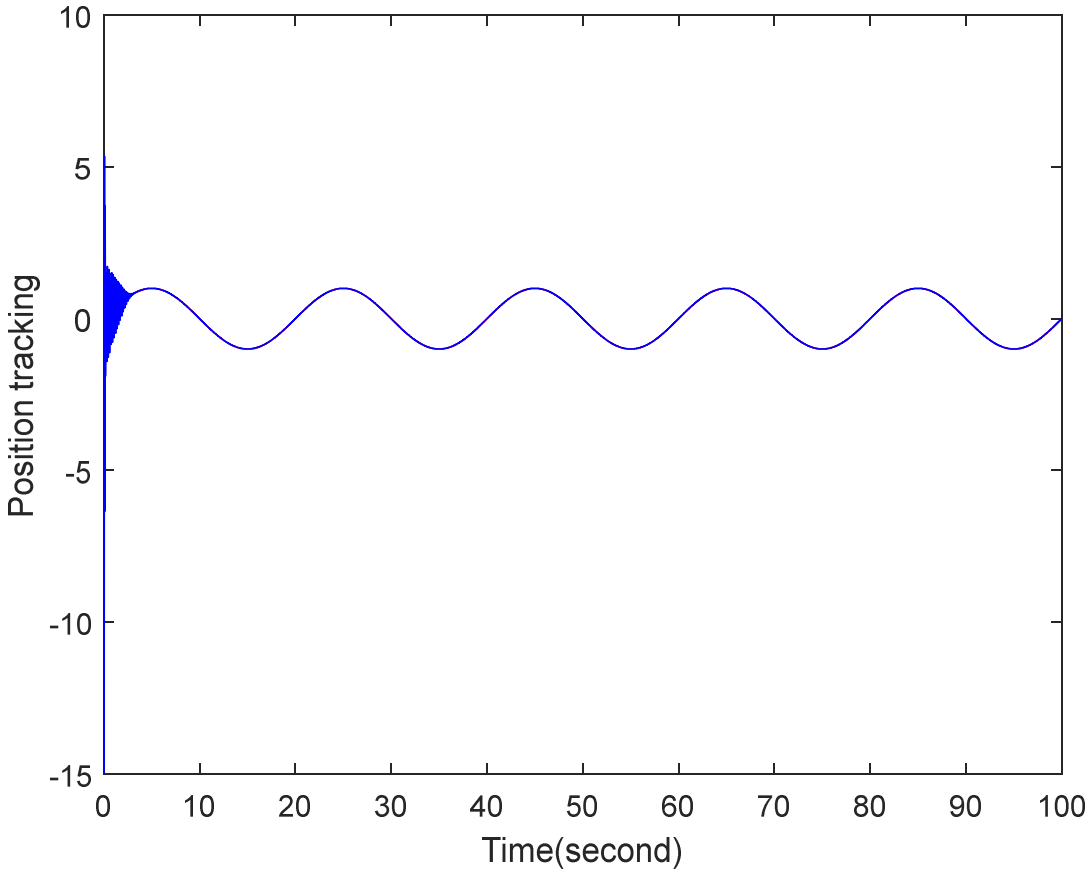
$$y(k) = 0.8\sin(y(k-1)) + 15u(k-1)$$

其中

$$g[y(k)] = 0.8\sin(y(k-1)) \qquad \varphi[y(k)] = 15$$

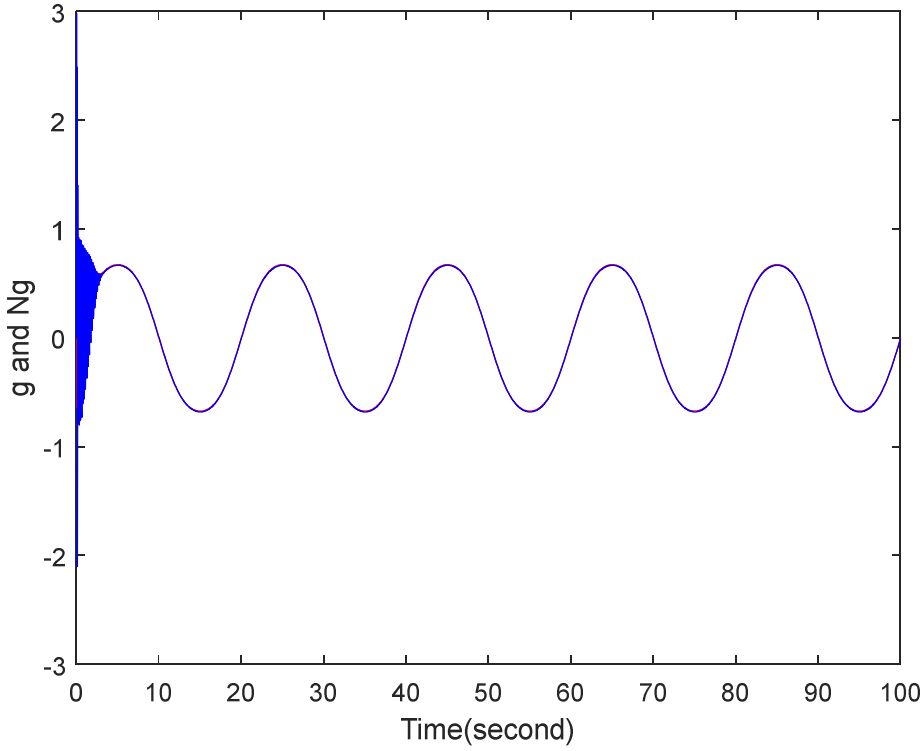
输入信号为正弦信号r(t)=sin(0.1πt)。取y(k)作为网络的输入，网络隐层神经元个数取m=6，神经网络结构为1-6-1，网络的初始权值$\mathbf{W}=[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]^T$，$\mathbf{V}=[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]^T$，高斯函数的初始值取$\boldsymbol{C}_j=[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]^T$，$\mathbf{B}=[5, 5, 5, 5, 5, 5]^T$。权值学习参数为η1=0.15，η2=0.5，α=0.05。RBF网络自校正控制程序为chap9_3.m。

9.5 RBF网络自校正控制

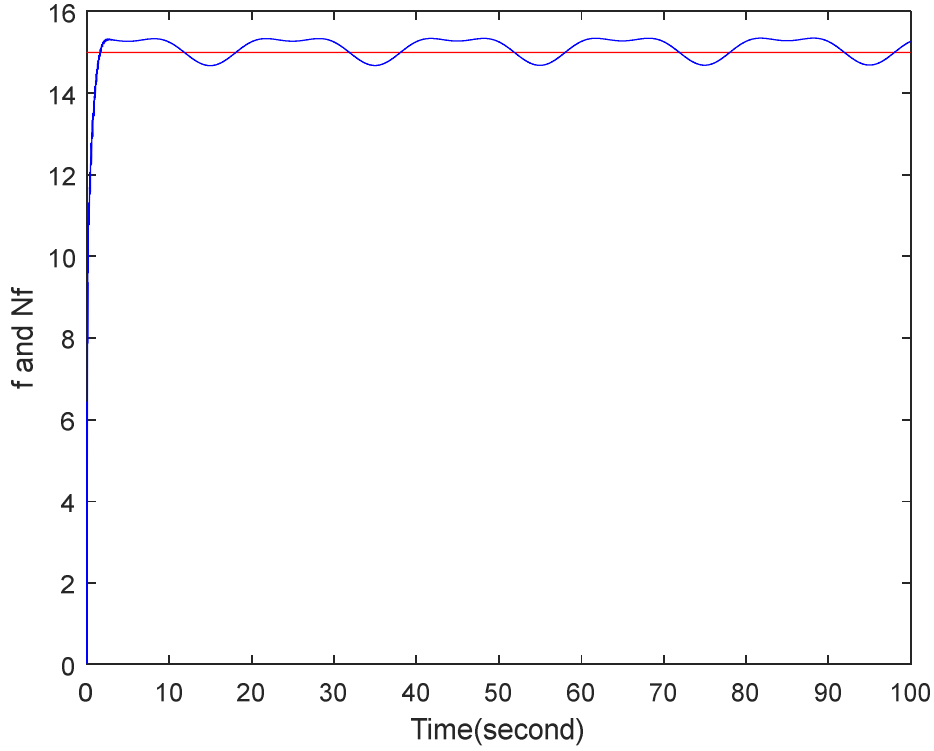


位置跟踪结果

9.5 RBF网络自校正控制



g[y(k)]的逼近结果



φ[y(k)]的逼近结果

9.6 基于RBF网络直接模型参考自适应控制

（一）基于RBF网络的控制器设计

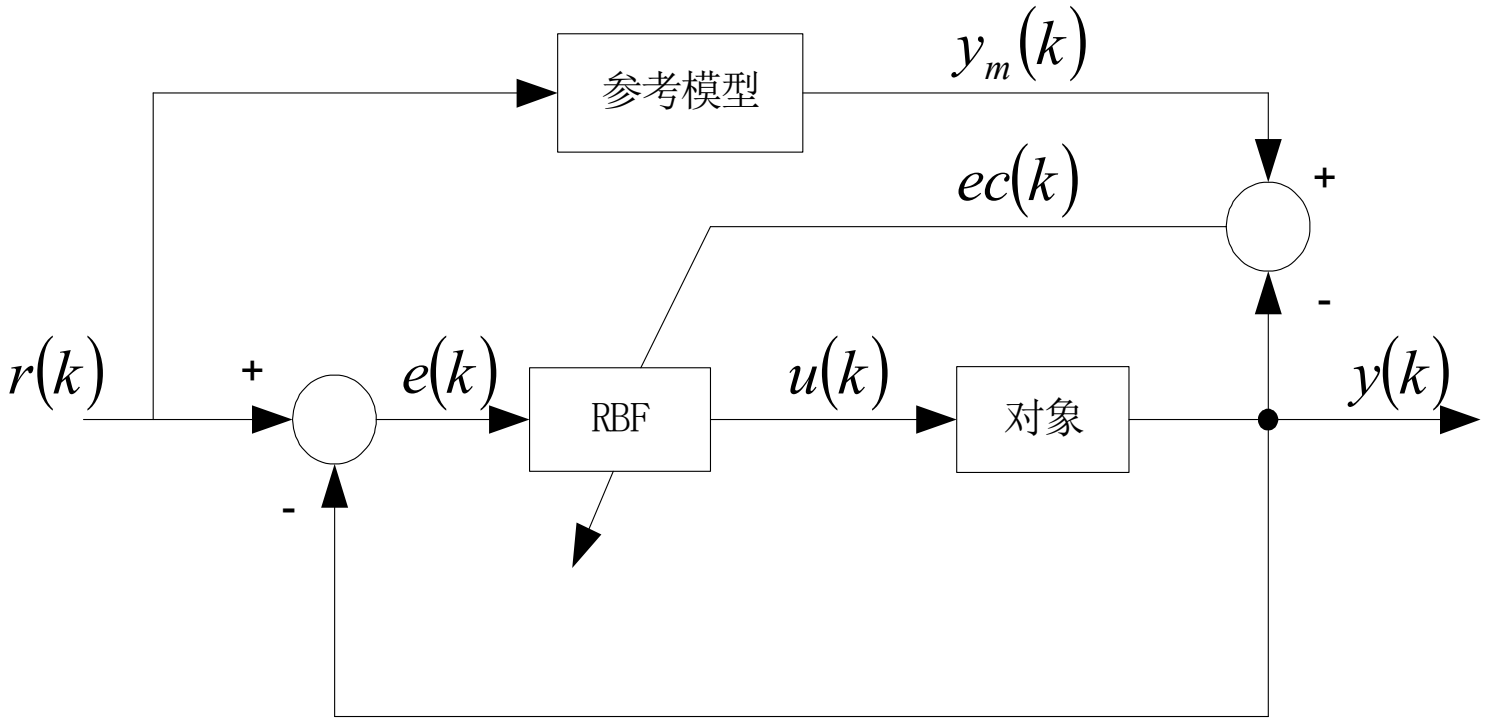


图9-23 基于RBF网络的直接模型参考自适应控制

9.6 基于RBF网络直接模型参考自适应控制

设参考模型输出为$y_m(k)$，控制系统要求对象的输出$y(k)$能够跟踪参考模型的输出$y_m(k)$。

则跟踪误差为：$ec(k)=y_m(k)-y(k)$

控制目标函数为：$E(k)=\frac{1}{2}ec(\mathrm{k})^2$

控制器为RBF网络的输出：

$$u(k)=h_1w_1+\cdots h_jw_j\cdots+h_mw_m \qquad (9.22)$$

其中m为RBF网络隐层神经元的个数，w_j为第j个网络隐层神经元与输出层之间的连接权，h_j为第个j隐层神经元的输出。

9.6 基于RBF网络直接模型参考自适应控制

在RBF网络结构中，$\mathbf{X}=[x_1,\cdots,x_n]^T$ 为网络的输入向量。RBF网络的径向基向量为 $\mathbf{H}=[h_1,\ \cdots,\ h_m]^T$，$h_j$为高斯基函数：

$$h_j=\exp(-\frac{\|\mathbf{X}-\mathbf{C}_j\|^2}{2b_j^2})$$

其中$j=1,...,m$，b_j为节点j的基宽度参数，$C_j>0$，C_j为网络第j个结点的中心矢量，

$$\mathbf{C}_j=[c_{j1},\cdots,c_{ji},\cdots,c_{jn}] \qquad \mathbf{B}=[b_1,\cdots,b_m]^T$$

9.6 基于RBF网络直接模型参考自适应控制

网络的权向量为：

$$\boldsymbol{W}=\left[w_1,\quad \cdots,\quad w_m\right]^{\mathrm{T}}$$

按梯度下降法及链式法则，可得权值的学习算法如下：

$$\Delta w_j(\mathrm{k})=-\eta\frac{\partial E(k)}{\partial w}=\eta ec(\mathrm{k})\frac{\partial y(k)}{\partial u(k)}h_j$$

$$w_j(\mathrm{k})=w_j(\mathrm{k}-1)+\Delta w_j(\mathrm{k})+\alpha\Delta w_j(k) \qquad (9.25)$$

其中η为学习速率，α为动量因子。

9.6 基于RBF网络直接模型参考自适应控制

同理，可得RBF网络隐层神经元的高斯函数的中心参数及基宽的学习算法如下：

$$\Delta b_j(k) = -\eta \frac{\partial E(k)}{\partial b_j} = \eta ec(k) \frac{\partial y(\mathrm{k})}{\partial u(k)} \frac{\partial u(\mathrm{k})}{\partial b_j} = \eta ec(k) \frac{\partial y(\mathrm{k})}{\partial u(k)} w_j h_j \frac{\left\| \mathbf{x} - c_{ij} \right\|^2}{b_j^{\,3}}$$

$$b_j(\mathrm{k}) = b_j(\mathrm{k}-1) + \eta \Delta b_j(k) + \alpha (b_j(\mathrm{k}-1) - b_j(\mathrm{k}-2))$$

$$\Delta c_{ij}(k) = -\eta \frac{\partial E(k)}{\partial c_{ij}} = \eta ec(k) \frac{\partial y(\mathrm{k})}{\partial u(k)} \frac{\partial u(\mathrm{k})}{\partial c_{ij}} = \eta ec(k) \frac{\partial y(\mathrm{k})}{\partial u(k)} w_j h_j \frac{x_i - c_{ij}}{b_j^{\,2}}$$

$$c_{ij}(\mathrm{k}) = c_{ij}(\mathrm{k}-1) + \eta \Delta c_{ij}(k) + \alpha (\mathrm{c}_{ij}(\mathrm{k}-1) - \mathrm{c}_{ij}(\mathrm{k}-2))$$

9.6 基于RBF网络直接模型参考自适应控制

在学习算法中，$\frac{\partial y(\mathrm{k})}{\partial u(k)}$称为*Jacobian*信息，表示系统的输出对控制输入的敏感性，其值可由神经网络辨识而得。在神经网络算法中，对$\frac{\partial y(\mathrm{k})}{\partial u(k)}$值的精确度要求不是很高，不精确部分可通过网络参数及权值的调整来修正，关键是其符号，因此可用$\frac{\partial y(\mathrm{k})}{\partial u(k)}$的正负号来代替，这样可使算法更加简单。

9.6 基于RBF网络直接模型参考自适应控制

（二）仿真实例

被控对象为一非线性模型：

$$y(k)=(-0.10y(k-1)+u(k-1))/(1+y(k-1)^2)$$

取采样周期为 $ts=1ms$，参考模型为

$$y_m(k)=0.6y_m(k-1)+r(k)$$

其中 $r(k)$ 为正弦信号 $r(k)=0.50\sin(2\pi k\times ts)$

采用RBF网络进行控制，取r(k)， ec(k)和y(k)作为RBF网络的输入，网络采用3-6-1结构。取学习速率η=0.35，动量因子α=0.05。

9.6 基于RBF网络直接模型参考自适应控制

采用控制律(9.22)和权值调整算法式(9.25)，根据网络输入值的范围，高斯基函数参数的初始值取

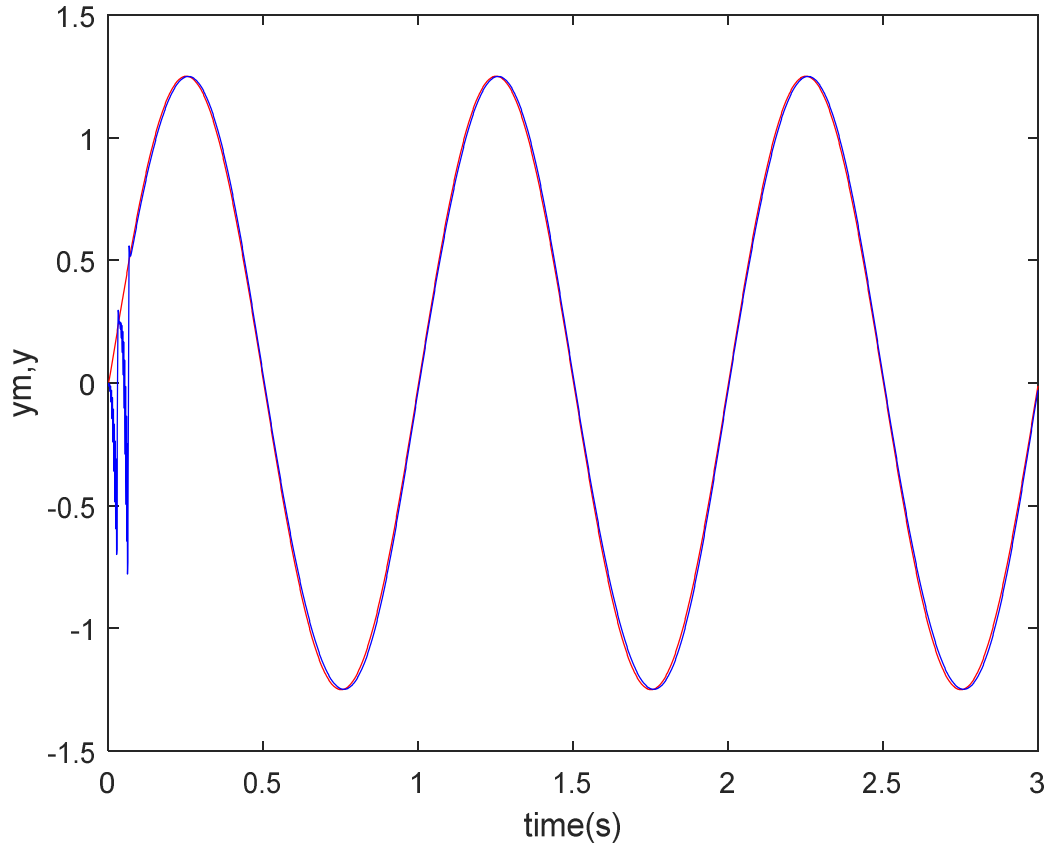
$$C=\begin{bmatrix} -3 & -2 & -1 & 1 & 2 & 3 \\ -3 & -2 & -1 & 1 & 2 & 3 \\ -3 & -2 & -1 & 1 & 2 & 3 \end{bmatrix} \qquad B=[2,2,2,2,2,2]^T$$

首先，网络的初始权值取[-1, +1]之间的随机值。在Matlab仿真中采用$\text{sign}\left(\frac{\Delta y(k)}{\Delta u(k)}\right)$近似实现$\frac{\Delta y(k)}{\Delta u(k)}$。

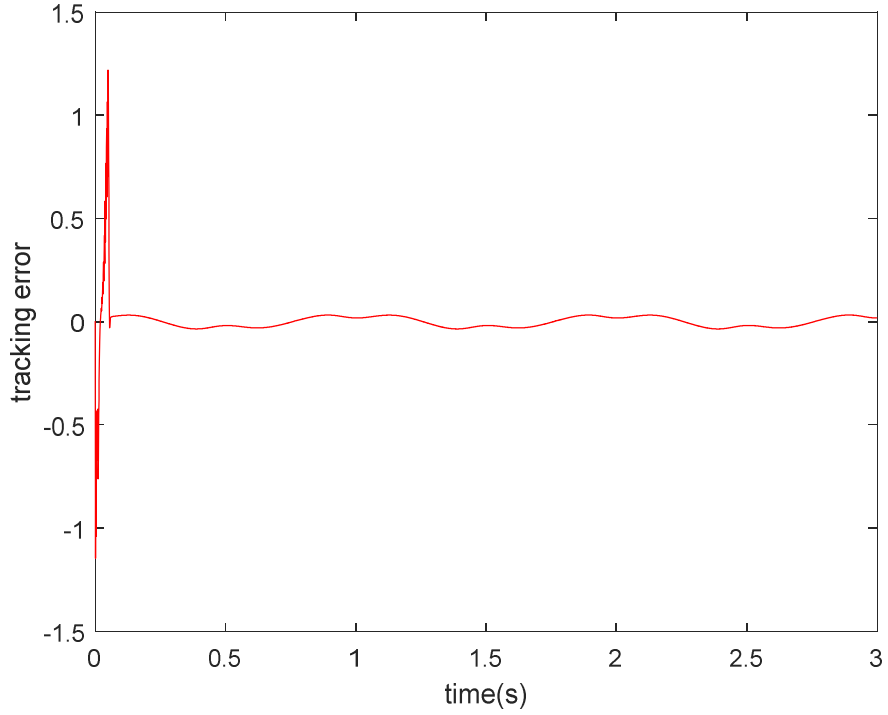
```
dyu(k)=sign((y(k)-y_1)/(u(k)-u_1));
```

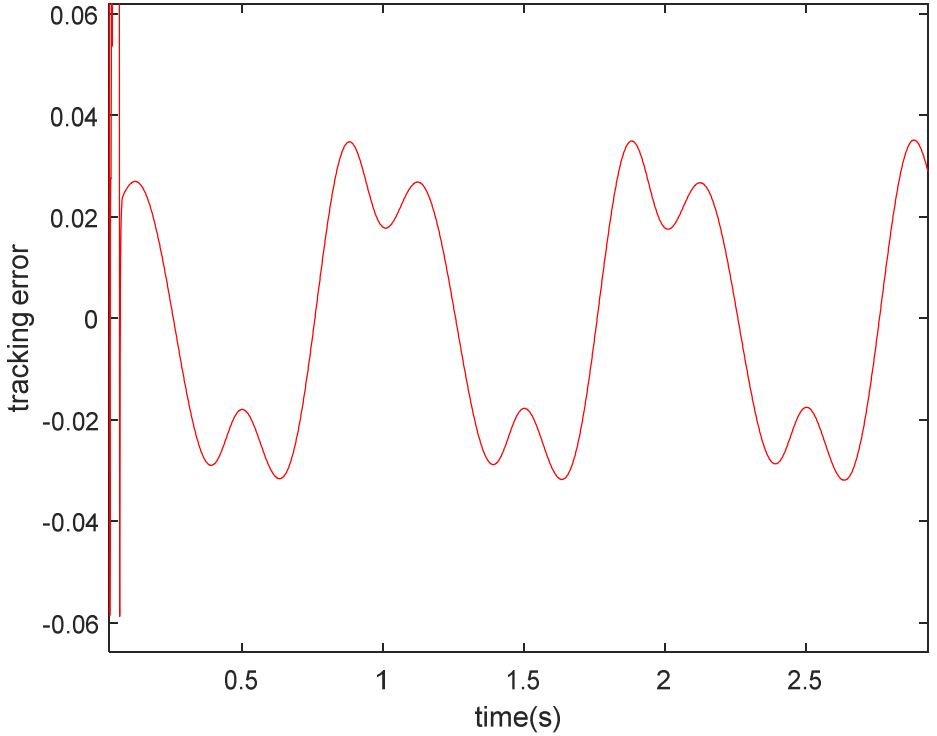
9.6 基于RBF网络直接模型参考自适应控制

仿真中，取M=1时为调整b和c，M=2时为不调整b和c。



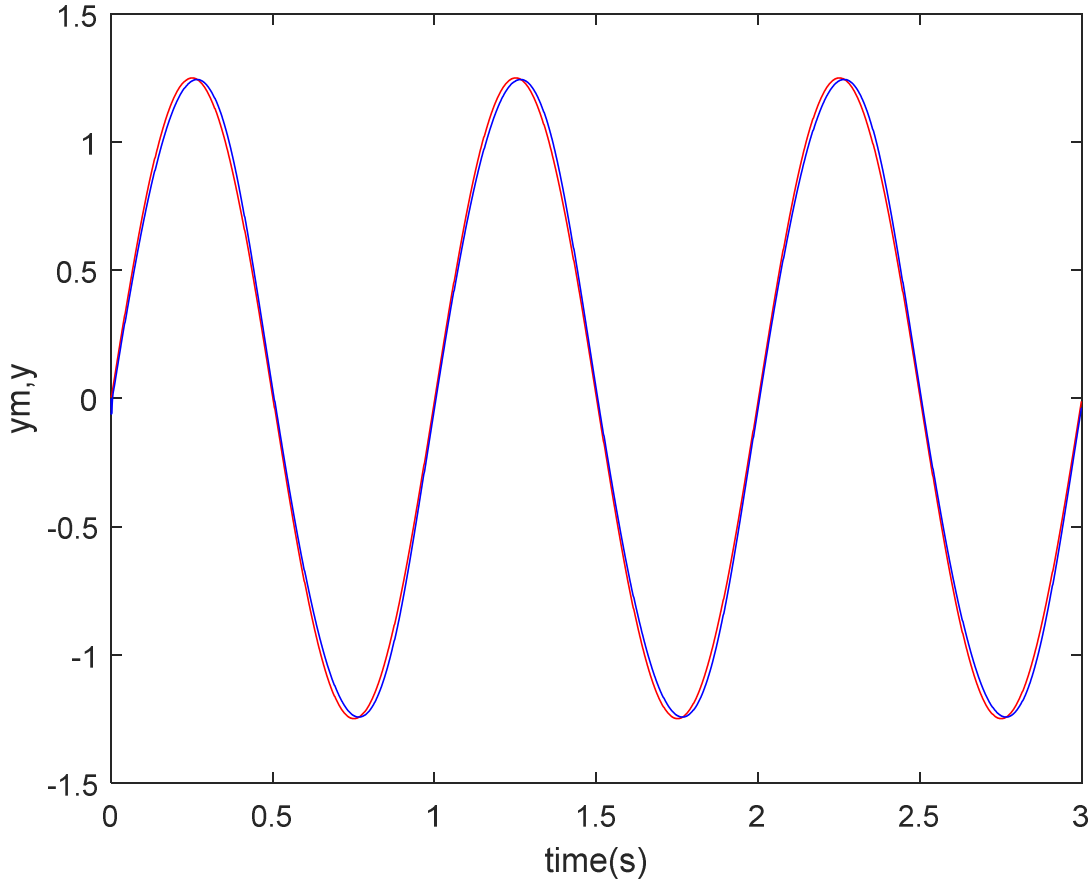
9.6 基于RBF网络直接模型参考自适应控制



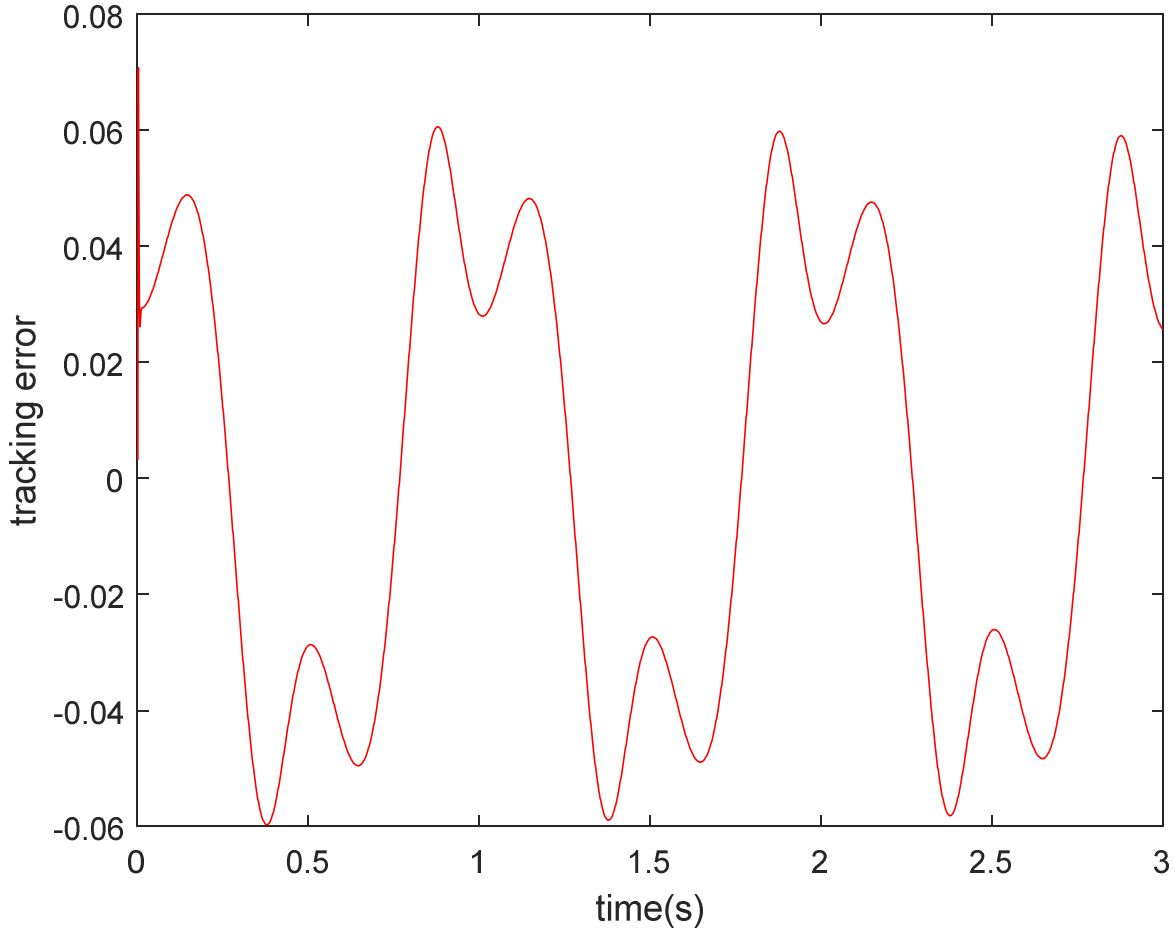


9.6 基于RBF网络直接模型参考自适应控制

仿真时间为3s，取M=1，仿真结果如下图所示。由仿真分析可见，采用调整b和c的方法可得到更好的效果。



9.6 基于RBF网络直接模型参考自适应控制



9.6 基于RBF网络直接模型参考自适应控制

```
if M==1
    d_b=0*b;
    for j=1:1:6
        d_b(j)=xite*ec(k)*w(j)*h(j)*(b(j)^-
3)*norm(x-c(:,j))^2*dyu(k);
    end
    b=b_1+d_b+alfa*(b_1-b_2);

    d_c=0*c;
    for j=1:1:6
      for i=1:1:3
        d_c(i,j)=xite*ec(k)*w(j)*h(j)*(x(i)-
c(i,j))*(b(j)^-2)*dyu(k);
      end
    end
    c=c_1+d_c+alfa*(c_1-c_2);
elseif M==2
        b=b_1;
        c=c_1;
end
```

9.6 基于RBF网络直接模型参考自适应控制

所设计的控制律采用Hebb学习规则或梯度下降法来设计权值调节律，而Hebb学习规则和梯度下降法只能保证系统在理想值附近有效，不能保证全局收敛，因此闭环系统的稳定性得不到保障。因此，在控制系统设计中，不建议采用上述方法。

针对这一问题，人们提出了在线自适应神经网络方法，它基于Lyapunov稳定性理论获得权值自适应律，闭环系统的稳定性得到保障，最近十多年这种方法被广泛采用。以下两节介绍两种针对机械手的在线自适应神经网络控制方法。

- 研究的目的和意义：稳定性是自动控制系统正常工作的必要条件，是一个重要特征。

- 要求：在受到外界扰动后，虽然其原平衡状态被打破，但在扰动消失后，仍然能恢复到原来的平衡状态，或者趋于另一平衡状态继续工作。

- 稳定性：系统在受到小的外界扰动后，系统状态方程解的收敛性，而与输入作用无关。

- 经典控制理论稳定性判别方法：代数判据（劳斯判据），奈奎斯特判据，对数判据，根轨迹判据

- 非线性系统：相平面法(适用于一，二阶非线性系统)

- 1892年，俄国学者李亚普诺夫(Lyapunov)提出的稳定性定理采用了状态向量来描述，适用于单变量，线性，非线性，定常，时变，多变量等系统。

- 应用：自适应控制，最优控制，非线性控制等。

➢李亚普诺夫第一法（间接法）

通过求解系统微分方程（或状态方程），然后根据解的特性来判断系统稳定性的方法，适用于线性定常、线性时变及可线性化的非线性系统，其基本思路与分析方法和经典理论相一致。

➢李亚普诺夫第二法（直接法）

不求解系统的状态方程，直接判断系统稳定性，利用经验和技巧构造一个类似于能量函数的Lyapunov函数$V(x)$，然后根据$\dot{V}(x)$的符号性质判断系统稳定性

-----对任何系统都适用。

从能量的观点出发得来的，基本思想建立在古典力学-振动系统中一个直观的物理事实上：任何物理系统的运动都要消耗能量，并且能量总是大于零的。

若系统贮存的能量（含动能与位能）随时间推移而衰减，系统迟早会到达平衡状。

对于一个不受外部作用的系统，如果系统的能量随时间而最终消失，那么这个系统是渐近稳定的。反之，如果系统不断从外界吸收能量，储能越来越大，那么这个平衡状态是不稳定的。

若能量在运动过程中不增不减，则称为李亚普诺夫意义下的稳定。

李亚普诺夫第二法的基本思想

- 但由于系统的形式是多种多样的，实际系统的能量函数表达式相当难找，不可能找到一种能量函数的统一表达形式。因此李亚普诺夫引入了广义（或虚构）能量函数，称之为李亚普诺夫函数。它与x1, ···, xn及t有关，是一个标量函数，记以$V(x,t)$；若不显含t，则记以$V(x)$。依据系统的状态方程考察能量函数在运动过程中的变换规律。

- 考虑到能量总大于零，故为正定函数。能量衰减特性用$\dot{V}(x,t)$或$\dot{V}(x)$表示。
- 利用$V(x,t)$和$\dot{V}(x,t)$的符号特征，判断平衡状态稳定性。由于$V(x)$是表示能量的函数，所以$V(x)>0$。这样就可以根据$\dot{V}(x)$的定号性来判断系统的稳定性。显然，若$V(x)>0$，并且$\dot{V}(x)<0$，则系统就是渐近稳定的。

<李雅普诺夫稳定性第二定理>

- 实践表明，对于大多数系统，可先尝试用二次型函数x^TPx作为李亚普诺夫函数。

李亚普诺夫第二法稳定性判据

设系统的状态方程为

$$\dot{x} = f(x)$$

平衡状态为$x_e=0$，满足$f(x_e)=0$.

如果存在一个标量函数$V(x)$满足

1） $V(x)$对所有x都具有连续的一阶偏导数。

2） $V(x)$是正定的，即当 $x = 0, V(x) = 0; x \neq 0, V(x) > 0$

可以根据$V(x)$对时间的导数判断系统的稳定性。

李亚普诺夫第二法稳定性判据

① 若$\dot{V}(\boldsymbol{x})$为半负定，那么平衡状态x_e为李亚普诺夫意义下稳定。稳定判据

② 若 $\dot{V}(\boldsymbol{x})$为负定，或者虽然$\dot{V}(\boldsymbol{x})$ 为半负定，但对任意初始状态$\boldsymbol{x}(t_0)\neq \boldsymbol{0}$ 来说，除去x=0外，对$\boldsymbol{x}\neq \boldsymbol{0}$ ，$\dot{V}(\boldsymbol{x})$不恒为零。原点平衡状态为渐近稳定。如果有$\|\boldsymbol{x}\|\to\infty$时，$V(\boldsymbol{x})\to\infty$ 则系统是大范围渐近稳定。

③ 若 $\dot{V}(\boldsymbol{x})$ 为正定，那么平衡状态是不稳定的

9.7 一种简单的RBF网络自适应控制

（一）问题描述

考虑一种简单的动力学系统：

$$\ddot{\theta} = f\left(\theta, \dot{\theta}\right) + u \tag{9.30}$$

其中θ为转动角度，u为控制输入。
写成状态方程形式为:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f\left(x\right) + u \end{aligned} \tag{9.31}$$

其中 $f(x)$为未知。

位置指令为x_d，则误差及其导数为

$$e = x_1 - x_d \qquad \dot{e} = x_2 - \dot{x}_d$$

9.7 一种简单的RBF网络自适应控制

定义误差函数为

$$s = \lambda e + \dot{e} \qquad \lambda > 0 \tag{9.32}$$

则

$$\dot{s} = \lambda \dot{e} + \ddot{e} = \lambda \dot{e} + \dot{x}_2 - \ddot{x}_{\mathrm{d}} = \lambda \dot{e} + f\left(x\right) + u - \ddot{x}_{\mathrm{d}}$$

由式（9.32）可见，如果$s \to 0$ ，则$e \to 0$ 且$\dot{e} \to 0$ 。

9.7 一种简单的RBF网络自适应控制

（二）RBF网络原理

由于RBF网络具有万能逼近特性，采用RBF神经网络逼近$f(x)$，网络算法为：

$$h_j = \exp\left(\frac{\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|^2}{2b_j^2}\right) \tag{9.33}$$

$$f = \boldsymbol{W}^{*\mathrm{T}}\boldsymbol{h}(\boldsymbol{x}) + \varepsilon \tag{9.34}$$

其中 x为网络的输入，j 为网络隐含层第 j个节点，$\boldsymbol{h}=\left[h_j\right]^{\mathrm{T}}$为网络的高斯基函数输出，$\boldsymbol{W}^*$为网络的理想权值，ε为网络的逼近误差，$\varepsilon \le \varepsilon_{\mathrm{N}}$。

网络输入取 $\boldsymbol{x}=[x_1 \quad x_2]^{\mathrm{T}}$,则网络输出为：

$$\hat{f}(x) = \hat{\boldsymbol{W}}^{\mathrm{T}}\boldsymbol{h}(x) \tag{9.35}$$

9.7 一种简单的RBF网络自适应控制

（三）控制算法设计与分析

由于

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

定义Lyapunov函数为

$$V=\frac{1}{2}s^2+\frac{1}{2\gamma}\tilde{\boldsymbol{W}}^{\mathrm{T}}\tilde{\boldsymbol{W}} \tag{9.36}$$

其中 $\gamma>0$，$\tilde{\boldsymbol{W}}=\hat{\boldsymbol{W}}-\boldsymbol{W}^{*}$。

则

$$\dot{V}=s\dot{s}+\frac{1}{\gamma}\tilde{W}^{\mathrm{T}}\dot{\hat{W}}=s\left(c\dot{e}+f(x)+u-\ddot{x}_{\mathrm{d}}\right)+\frac{1}{\gamma}\tilde{\boldsymbol{W}}^{\mathrm{T}}\dot{\hat{\boldsymbol{W}}}$$

设计控制律为

$$u=-c\dot{e}-\hat{f}(x)+\ddot{x}_{\mathrm{d}}-\eta\,\mathrm{sgn}(s) \tag{9.37}$$

9.7 一种简单的RBF网络自适应控制

则

$$\dot{V}=s\left(f(x)-\hat{f}(x)-\eta\,\mathrm{sgn}(s)\right)+\frac{1}{\gamma}\tilde{\boldsymbol{W}}^{\mathrm{T}}\dot{\hat{\boldsymbol{W}}}$$

$$=s\left(-\tilde{\boldsymbol{W}}^{\mathrm{T}}\boldsymbol{h}(\boldsymbol{x})+\varepsilon-\eta\,\mathrm{sgn}(s)\right)+\frac{1}{\gamma}\tilde{\boldsymbol{W}}^{\mathrm{T}}\dot{\hat{\boldsymbol{W}}}$$

$$=\varepsilon s-\eta|s|+\tilde{W}^{\mathrm{T}}\left(\frac{1}{\gamma}\dot{\hat{W}}-s\boldsymbol{h}(\boldsymbol{x})\right)$$

取 $\eta>|\varepsilon|_{\max}$，自适应律为

$$\dot{\hat{\boldsymbol{W}}}=\gamma s\boldsymbol{h}(\boldsymbol{x}) \qquad (9.38)$$

则 $\dot{V}=\varepsilon s-\eta|s|\le 0$

由于$V\ge 0$，$\dot{V}\le 0$，则当t→∞时，V有界，从而$\tilde{W}$有界。

可见， 控制律中的鲁棒项ηsgn(s)的作用是克服神经网络的逼近误差，以保证系统稳定。

9.7 一种简单的RBF网络自适应控制

（四）仿真实例

考虑如下被控对象

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = f\left(x\right) + u$$

式中，$f\left(x\right) = 10x_1x_2$

位置指令为$x_{\mathrm{d}} = \sin t$，控制律采用式（9.37），自适应律采用式（9.38），取$\gamma = 500$，$\eta = 0.50$。根据网络输入x1 和x2 的实际范围来设计高斯基函数的参数，参数ci和bi取值分别为[-2 -1 0 1 2] 和3.0。网络权值中各个元素的初始值取0.10。

9.7 一种简单的RBF网络自适应控制

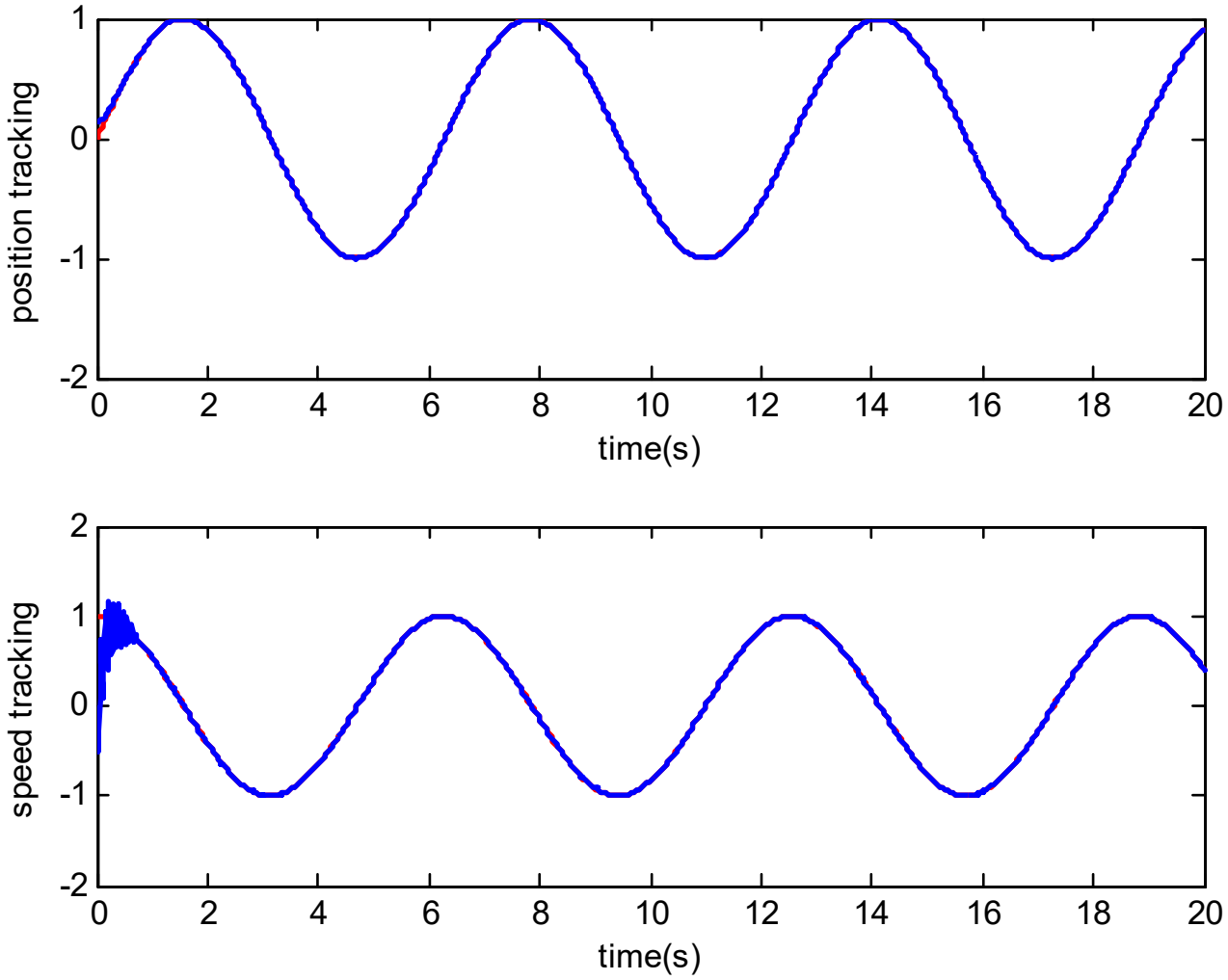


图9.25 位置和速度跟踪

9.7 一种简单的RBF网络自适应控制

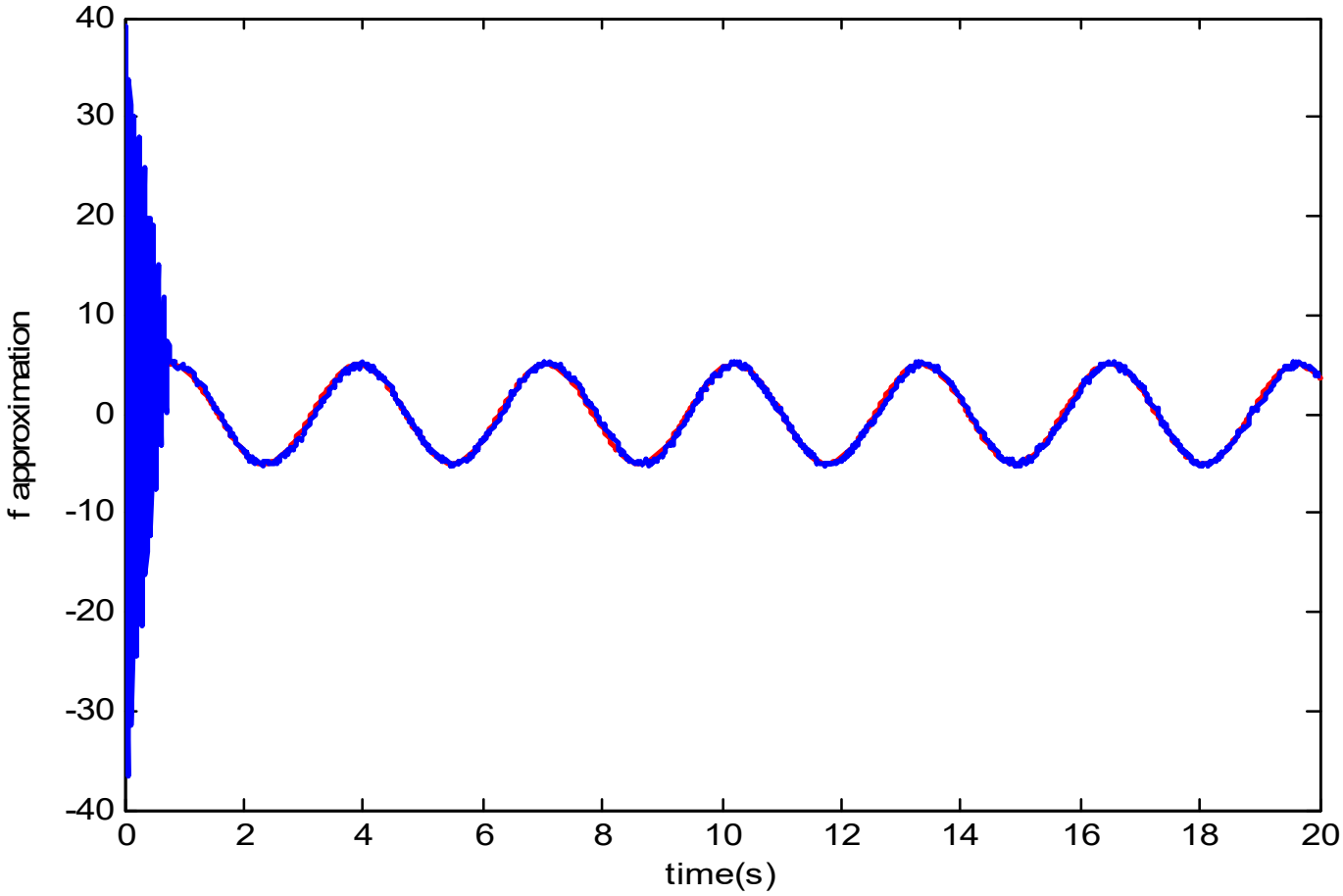


图9.26 f(x)及逼近

9.8 基于不确定逼近的RBF网络自适应控制

研究基于模型不确定逼近的RBF网络机器人自适应控制的设计方法：

（一）问题的提出

设n关节机械手的动力学方程为：

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau} + \mathbf{d} \qquad (9.39)$$

其中 $\mathbf{D}(\mathbf{q})$为n×n阶惯性/质量矩阵，$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ 为n×n阶离心力和哥氏力矩阵，$\mathbf{G}(\mathbf{q})$为n×1 阶重力向量。

如果模型建模精确，且d=0，则控制律可设计为：

$$\boldsymbol{\tau} = \mathbf{D}(\mathbf{q})\left(\ddot{\mathbf{q}}_d - \mathbf{k}_v\dot{\mathbf{e}} - \mathbf{k}_p\mathbf{e}\right) + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) \qquad (9.40)$$

9.8 基于不确定逼近的RBF网络自适应控制

将控制律式（9.40）代入式（9.39）中，得到稳定的闭环系统为

$$\ddot{\mathbf{e}}+\mathbf{k}_v\dot{\mathbf{e}}+\mathbf{k}_p\mathbf{e}=0 \tag{9.41}$$

其中$\boldsymbol{q}_d$为理想的角度，$\mathbf{e}=\mathbf{q}-\mathbf{q}_d$，$\dot{\mathbf{e}}=\dot{\mathbf{q}}-\dot{\mathbf{q}}_d$。

在实际工程中，对象的实际模型很难得到，即无法得到精确的$\mathbf{D}(\mathbf{q})$、$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$、$\mathbf{G}(\mathbf{q})$，只能建立理想的名义模型。

将机器人名义模型表示为$\mathbf{D}_0(\mathbf{q})$，$\mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})$，$\mathbf{G}_0(\mathbf{q})$，针对名义模型的控制律设计为：

$$\boldsymbol{\tau}=\mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{q}}_d-\mathbf{k}_v\dot{\mathbf{e}}-\mathbf{k}_p\mathbf{e}\right)+\mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}_0(\mathbf{q}) \tag{9.42}$$

将控制律式（9.42）代入式（9.39）中，得到

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}(\mathbf{q})=\mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{q}}_d-\mathbf{k}_v\dot{\mathbf{e}}-\mathbf{k}_p\mathbf{e}\right)+\mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}_0(\mathbf{q})+\mathbf{d} \tag{9.43}$$

取 $\Delta\mathbf{D}=\mathbf{D}_0-\mathbf{D}$　$\Delta\mathbf{C}=\mathbf{C}_0-\mathbf{C}$　$\Delta\mathbf{G}=\mathbf{G}_0-\mathbf{G}$

则

$$\ddot{\mathbf{e}}+\mathbf{k}_v\dot{\mathbf{e}}+\mathbf{k}_p\mathbf{e}=\mathbf{D}_0^{-1}\left(\Delta\mathbf{D}\ddot{\mathbf{q}}+\Delta\mathbf{C}\dot{\mathbf{q}}+\Delta\mathbf{G}+d\right) \tag{9.44}$$

9.8 基于不确定逼近的RBF网络自适应控制

由式（9.44）可见，由于模型建模的不精确会导致控制性能的下降。因此，需要对建模不精确部分进行逼近。

式（9.44）中，取建模不精确部分 $f=\mathbf{D}_0^{-1}\left(\Delta\mathbf{D}\ddot{\mathbf{q}}+\Delta\mathbf{C}\dot{\mathbf{q}}+\Delta\mathbf{G}+\mathbf{d}\right)$ 则可将式（9.44）转化为如下误差状态方程

$$\dot{x}=Ax+Bf \tag{9.45}$$

式中，$A=\begin{pmatrix}0 & I\\ -k_p & -k_v\end{pmatrix}$，$B=\begin{pmatrix}0\\ I\end{pmatrix}$，I为单位阵。

假设模型不确定项f为已知，则可设计修正的控制律为：

$$\boldsymbol{\tau}=\mathbf{D}_0\left(\mathbf{q}\right)\left(\ddot{\mathbf{q}}_{\mathrm{d}}-\mathbf{k}_{\mathrm{v}}\dot{\mathbf{e}}-\mathbf{k}_{\mathrm{p}}\mathbf{e}\right)+\mathbf{C}_0\left(\mathbf{q},\dot{\mathbf{q}}\right)\dot{\mathbf{q}}+\mathbf{G}_0\left(\mathbf{q}\right)-D_0(q)f \tag{9.46}$$

将控制律（9.46）代入式（9.39）中，则可得到稳定的闭环系统式（9.41）。在实际工程中，模型不确定项f为未知，为此，需要对不确定项f进行逼近，从而在控制律中实现对不确定项f的补偿。

9.8 基于不确定逼近的RBF网络自适应控制

（二）模型不确定部分的RBF网络逼近

采用RBF网络对不确定项 f 进行自适应逼近。

RBF网络算法为：$\phi_i = g\left(\left\|\mathbf{x}-\mathbf{c}_i\right\|^2 / b_i^2\right) \quad i=1,2\cdots,n$

$$\mathbf{y}=\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})$$

其中x为网络的输入信号，y为网络的输出信号，$\boldsymbol{\varphi}=[\phi_1,\phi_2,\cdots,\phi_n]$ 为高斯基函数的输出，$\boldsymbol{\theta}$为神经网络权值。

已经证明，在下述假设条件下，RBF网络针对连续函数在紧集范围内具有任意精度的逼近能力。

9.8 基于不确定逼近的RBF网络自适应控制

假设：

（1）神经网络输出 $\hat{f}(x,\theta)$ 为连续；

（2）存在理想逼近的神经网络输出 $\hat{f}(x,\theta^*)$，针对一个非常小的正数ε_0，有

$$\max \left\| \hat{\mathbf{f}}\left(\mathbf{x},\boldsymbol{\theta}^*\right) - \mathbf{f}\left(\mathbf{x}\right) \right\| \le \varepsilon_0$$

误差状态方程式（9.45）可写为

$$\dot{x} = Ax + B\left\{\hat{f}(x,\theta^*) + [f(x) - \hat{f}(x,\theta^*)]\right\} \tag{9.47}$$

其中，

$$\boldsymbol{\theta}^* = \arg \min_{\theta \in \beta(M_\theta)} \left\{ \sup_{x \in \varphi(M_x)} \left\| \mathbf{f}\left(\mathbf{x}\right) - \hat{\mathbf{f}}\left(\mathbf{x},\boldsymbol{\theta}\right) \right\| \right\}$$

$\boldsymbol{\theta}^*$为$n \times n$阶矩阵且有界，表示对$f(x)$最佳逼近的神经网络权值。

9.8 基于不确定逼近的RBF网络自适应控制

式(9.47)可写为

$$\dot{x} = Ax + B\left\{\widehat{f}(x,\theta^*) + \eta\right\} \tag{9.48}$$

η为理想神经网络的逼近误差，即

$$\boldsymbol{\eta} = \mathbf{f}\left(\mathbf{x}\right) - \hat{\mathbf{f}}\left(\mathbf{x},\boldsymbol{\theta}^*\right) \tag{9.49}$$

逼近误差η为有界，假设其界为η_0，即

$$\boldsymbol{\eta}_0 = \sup\left\|\mathbf{f}\left(\mathbf{x}\right) - \hat{\mathbf{f}}\left(\mathbf{x},\boldsymbol{\theta}^*\right)\right\| \tag{9.50}$$

神经网络输出$\widehat{f}(\bullet)$的最佳估计值为

$$\hat{\mathbf{f}}\left(\mathbf{x},\boldsymbol{\theta}^*\right) = \boldsymbol{\theta}^{*\mathrm{T}}\boldsymbol{\varphi}\left(\mathbf{x}\right) \tag{9.51}$$

则式(9.48)可写为

$$\dot{x} = Ax + B\left\{\theta^{*T}\varphi(x) + \eta\right\} \tag{9.52}$$

9.8 基于不确定逼近的RBF网络自适应控制

（三）控制器的设计及分析

控制器设计为

$$\boldsymbol{\tau} = \boldsymbol{\tau}_1 + \boldsymbol{\tau}_2 \tag{9.53}$$

其中

$$\boldsymbol{\tau}_1 = \mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{q}}_d - \mathbf{k}_v\dot{\mathbf{e}} - \mathbf{k}_p\mathbf{e}\right) + \mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}_0(\mathbf{q}) \tag{9.54}$$

$$\boldsymbol{\tau}_2 = -\mathbf{D}_0(\mathbf{q})\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta}) \tag{9.55}$$

其中$\hat{\boldsymbol{\theta}}$ 为 $\boldsymbol{\theta}^*$的估计值 $\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta}) = \hat{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})$

将控制律式（9.53）代入式（9.39）中，有

$$\begin{aligned}&\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q})\\ &= \mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{q}}_d - \mathbf{k}_v\dot{\mathbf{e}} - \mathbf{k}_p\mathbf{e}\right) + \mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}_0(\mathbf{q}) - \mathbf{D}_0(\mathbf{q})\hat{\mathbf{f}} + \mathbf{d}\end{aligned}$$

9.8 基于不确定逼近的RBF网络自适应控制

将 $\mathbf{D}_0(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}_0(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}_0(\mathbf{q})$ 分别减去上式左右两边，得：

$$\Delta\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\Delta\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\Delta\mathbf{G}(\mathbf{q})+\mathbf{d}$$

$$=\mathbf{D}_0(\mathbf{q})\ddot{\mathbf{q}}-\mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{q}}_d-\mathbf{k}_v\dot{\mathbf{e}}-\mathbf{k}_p\mathbf{e}\right)+\mathbf{D}_0(\mathbf{q})\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})$$

$$\Delta\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\Delta\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\Delta\mathbf{G}(\mathbf{q})+\mathbf{d}=\mathbf{D}_0(\mathbf{q})\left(\ddot{\mathbf{e}}+\mathbf{k}_v\dot{\mathbf{e}}+\mathbf{k}_p\mathbf{e}+\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})\right)$$

$$\ddot{\mathbf{e}}+\mathbf{k}_v\dot{\mathbf{e}}+\mathbf{k}_p\mathbf{e}+\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})=\mathbf{D}_0^{-1}(\mathbf{q})\left(\Delta\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\Delta\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\Delta\mathbf{G}(\mathbf{q})+\mathbf{d}\right)$$

$$\ddot{\mathbf{e}}+\mathbf{k}_v\dot{\mathbf{e}}+\mathbf{k}_p\mathbf{e}+\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})=\mathbf{f}(\mathbf{x})$$

式中，$\mathbf{f}(\mathbf{x})=\mathbf{D}_0^{-1}\left(\Delta\mathbf{D}\ddot{\mathbf{q}}+\Delta\mathbf{C}\dot{\mathbf{q}}+\Delta\mathbf{G}+\mathbf{d}\right)$

上式可写为：$\dot{\mathbf{x}}=\mathbf{A}\mathbf{x}+\mathbf{B}\left\{\mathbf{f}(\mathbf{x})-\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})\right\}$

其中 $\mathbf{A}=\begin{pmatrix}0 & \mathbf{I}\\ -\mathbf{k}_p & -\mathbf{k}_v\end{pmatrix}$ $\mathbf{B}=\begin{pmatrix}0\\ \mathbf{I}\end{pmatrix}$

9.8 基于不确定逼近的RBF网络自适应控制

由于 $$\mathbf{f}(\mathbf{x})-\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})=\mathbf{f}(\mathbf{x})-\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta}^*)+\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta}^*)-\hat{\mathbf{f}}(\mathbf{x},\boldsymbol{\theta})$$

$$=\boldsymbol{\eta}+\boldsymbol{\theta}^{*\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})-\hat{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})=\boldsymbol{\eta}+\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})$$

其中 $\tilde{\boldsymbol{\theta}}=\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}^*$

则

$$\dot{\mathbf{x}}=\mathbf{A}\mathbf{x}+\mathbf{B}\left(\boldsymbol{\eta}+\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})\right) \tag{9.56}$$

定义Lyapunov函数为

$$V=\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{x}+\frac{1}{2\gamma}\left\|\tilde{\boldsymbol{\theta}}\right\|^2 \tag{9.57}$$

其中 $\gamma>0$

存在正定矩P和Q，并满足如下Lyapunov方程

$$\mathbf{P}\mathbf{A}+\mathbf{A}^{\mathrm{T}}\mathbf{P}=-\mathbf{Q} \tag{9.58}$$

9.8 基于不确定逼近的RBF网络自适应控制

定义 $\|\mathbf{R}\|^2=\sum_{i,j}\left|r_{ij}\right|^2=\mathrm{tr}\left(\mathbf{R}\mathbf{R}^{\mathrm{T}}\right)=\mathrm{tr}\left(\mathbf{R}^{\mathrm{T}}\mathbf{R}\right)$

其中 $\mathrm{tr}(\cdot)$ 为矩阵的迹，则 $\left\|\tilde{\boldsymbol{\theta}}\right\|_F^2=\mathrm{tr}\left(\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)$

$$\dot{V}=\frac{1}{2}\left[\mathbf{x}^{\mathrm{T}}\mathbf{P}\dot{\mathbf{x}}+\dot{\mathbf{x}}^{\mathrm{T}}\mathbf{P}\mathbf{x}\right]+\frac{1}{\gamma}\mathrm{tr}\left(\dot{\tilde{\boldsymbol{\theta}}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)$$

$$=\frac{1}{2}\left[\mathbf{x}^{\mathrm{T}}\mathbf{P}\left(\mathbf{A}\mathbf{x}+\mathbf{B}\left(\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})+\boldsymbol{\eta}\right)\right)+\left(\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}+\left(\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})+\boldsymbol{\eta}\right)^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\right)\mathbf{P}\mathbf{x}\right]+\frac{1}{\gamma}\mathrm{tr}\left(\dot{\tilde{\boldsymbol{\theta}}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)$$

$$=\frac{1}{2}\left[\mathbf{x}^{\mathrm{T}}\left(\mathbf{P}\mathbf{A}+\mathbf{A}^{\mathrm{T}}\mathbf{P}\right)\mathbf{x}+\left(\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{B}\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})+\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{B}\boldsymbol{\eta}+\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}+\boldsymbol{\eta}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}\right)\right]+\frac{1}{\gamma}\mathrm{tr}\left(\dot{\tilde{\boldsymbol{\theta}}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)$$

$$=-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x}+\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}+\boldsymbol{\eta}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}+\frac{1}{\gamma}\mathrm{tr}\left(\dot{\tilde{\boldsymbol{\theta}}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)\qquad(9.59)$$

其中 $\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{B}\tilde{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})=\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}$　$\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{B}\boldsymbol{\eta}=\boldsymbol{\eta}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}$　$P^T=P$

9.8 基于不确定逼近的RBF网络自适应控制

对于n关节机械手动力学方程，由于

$$\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}=\mathrm{tr}\left[\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}\right] \tag{9.60}$$

则

$$\dot{V}=-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x}+\frac{1}{\gamma}\mathrm{tr}\left(-\gamma\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})\tilde{\boldsymbol{\theta}}+\dot{\tilde{\boldsymbol{\theta}}}^{\mathrm{T}}\tilde{\boldsymbol{\theta}}\right)+\boldsymbol{\eta}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x} \tag{9.61}$$

可以采用以下两种自适应律设计方法。

取自适应律为：$\dot{\hat{\boldsymbol{\theta}}}^{\mathrm{T}}=\gamma\mathbf{B}^{\mathrm{T}}\mathbf{P}\mathbf{x}\boldsymbol{\varphi}^{\mathrm{T}}(\mathbf{x})$

即

$$\dot{\hat{\boldsymbol{\theta}}}=\gamma\boldsymbol{\varphi}(\mathbf{x})\mathbf{x}^{\mathrm{T}}\mathbf{P}\mathbf{B} \tag{9.62}$$

则

$$\dot{V}=-\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{x}$$

由于Q>0，η是最小逼近误差，通过设计神经网络可使η充分小，并满足 $\left|\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{x}\right|\le\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}$，从而使得 $\dot{V}\le 0$，闭环系统稳定。

9.8 基于不确定逼近的RBF网络自适应控制

（四）仿真实例

选二关节机器人系统（不考虑摩擦力），其动力学模型为：

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}(\mathbf{q})=\boldsymbol{\tau}+\mathbf{d}$$

其中 $\mathbf{D}(\mathbf{q})=\begin{bmatrix} v+q_{01}+2\gamma\cos(q_2) & q_{01}+q_{02}\cos(q_2) \\ q_{01}+q_{02}\cos(q_2) & q_{01} \end{bmatrix}$

$$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})=\begin{bmatrix} -q_{02}\dot{q}_2\sin(q_2) & -q_{02}(\dot{q}_1+\dot{q}_2)\sin(q_2) \\ q_{02}\dot{q}_1\sin(q_2) & 0 \end{bmatrix}$$

$$\mathbf{G}(\mathbf{q})=\begin{bmatrix} 15g\cos q_1+8.75g\cos(q_1+q_2) \\ 8.75g\cos(q_1+q_2) \end{bmatrix}$$

$v=13.33$

$q_{01}=8.98$

$q_{02}=8.75$

$g=9.8$

9.8 基于不确定逼近的RBF网络自适应控制

误差扰动为：

$$d_1 = 2,\ \ d_2 = 3,\ \ d_3 = 6 \qquad \omega = d_1 + d_2 \|\mathbf{e}\| + d_3 \|\dot{\mathbf{e}}\|$$

位置指令为：
$$\begin{cases} q_{1\mathrm{d}} = 1 + 0.2\sin(0.5\pi t) \\ q_{2\mathrm{d}} = 1 - 0.2\cos(0.5\pi t) \end{cases}$$

被控对象的初值为 $[q_1 \quad q_2 \quad q_3 \quad q_4]^{\mathrm{T}} = [0.6 \quad 0.3 \quad 0.5 \quad 0.5]^{\mathrm{T}}$

控制参数取：

$$\mathbf{Q} = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{bmatrix} \quad \alpha = 3 \quad k_p = \begin{bmatrix} \alpha^2 & 0 \\ 0 & \alpha^2 \end{bmatrix} \quad k_v = \begin{bmatrix} 2\alpha & 0 \\ 0 & 2\alpha \end{bmatrix}$$

根据RBF网络输入$x=(e, \dot{e})^{\mathrm{T}}$的范围，$\boldsymbol{c}_i$=[-2, -1, 0, 1, 2]和$b_i$=3。

9.8 基于不确定逼近的RBF网络自适应控制

采用控制律式(9.53)和自适应律式(9.62)，采用Simulink和S函数进行控制系统的设计，仿真结果如图9-27至图9-30所示。

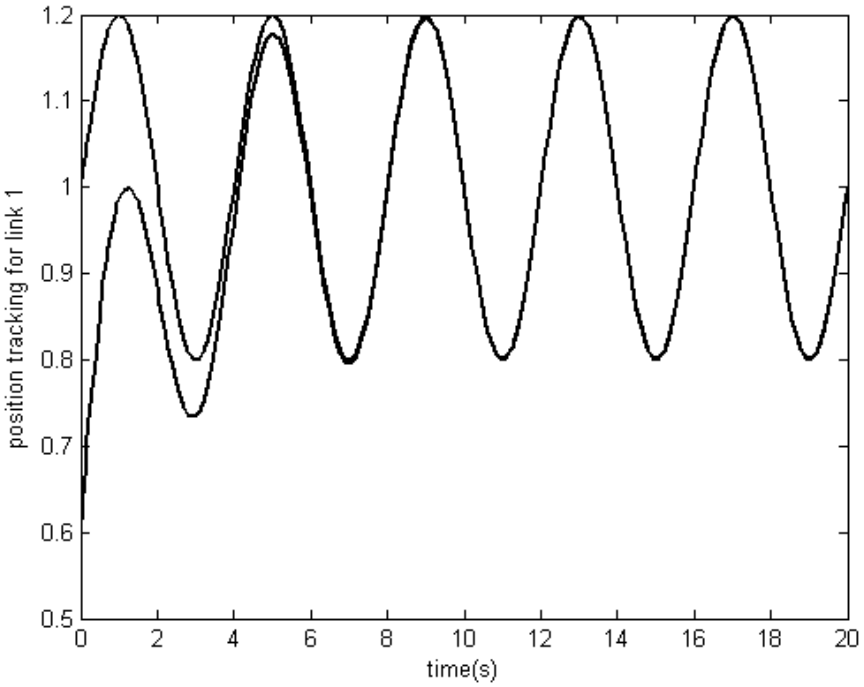


图9-27 关节1的位置跟踪

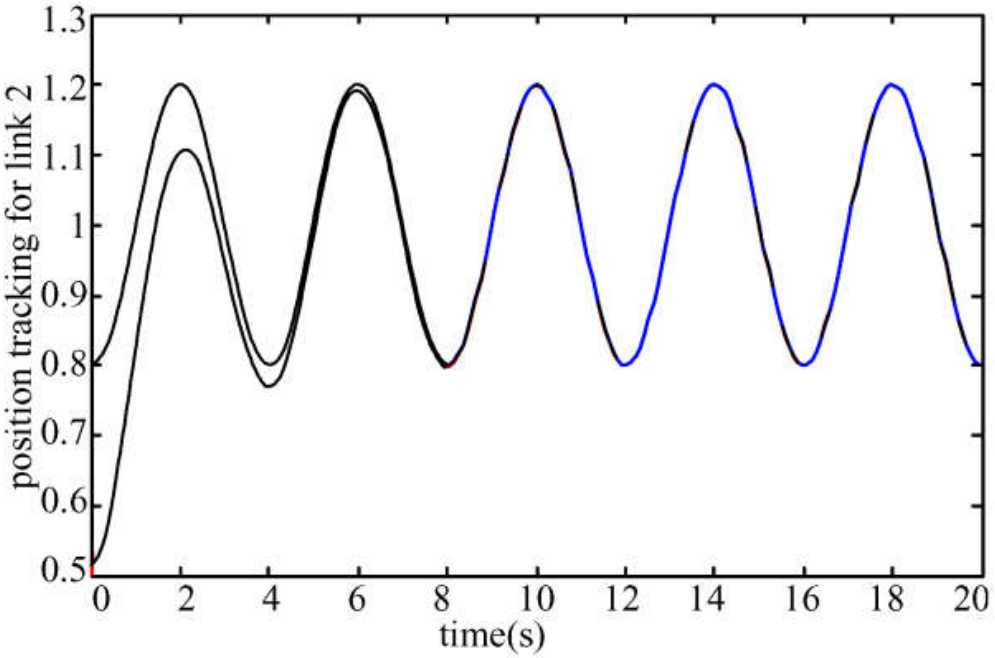


图9-28 关节2的位置跟踪

9.8 基于不确定逼近的RBF网络自适应控制

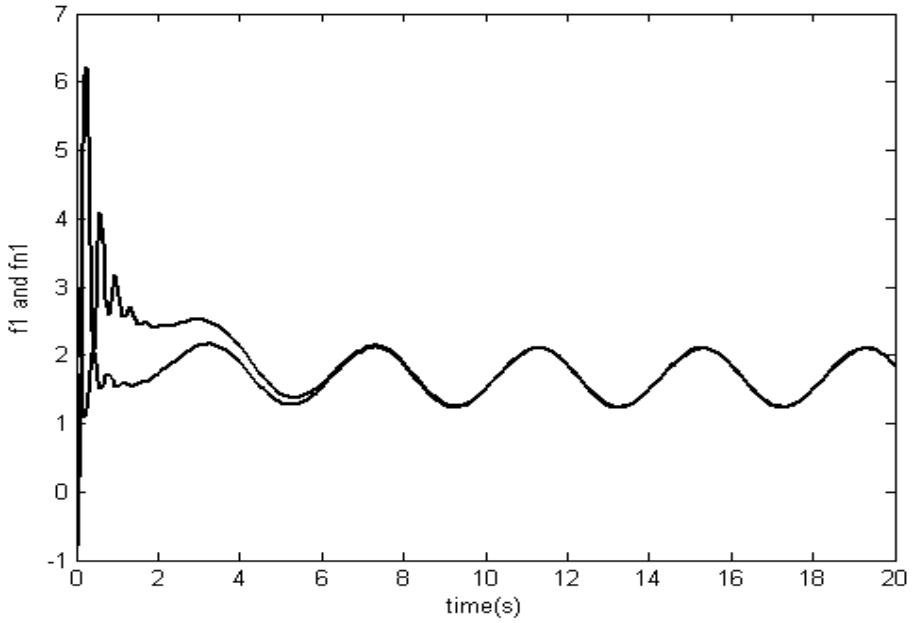


图9-29关节1的模型不确定项及其估计

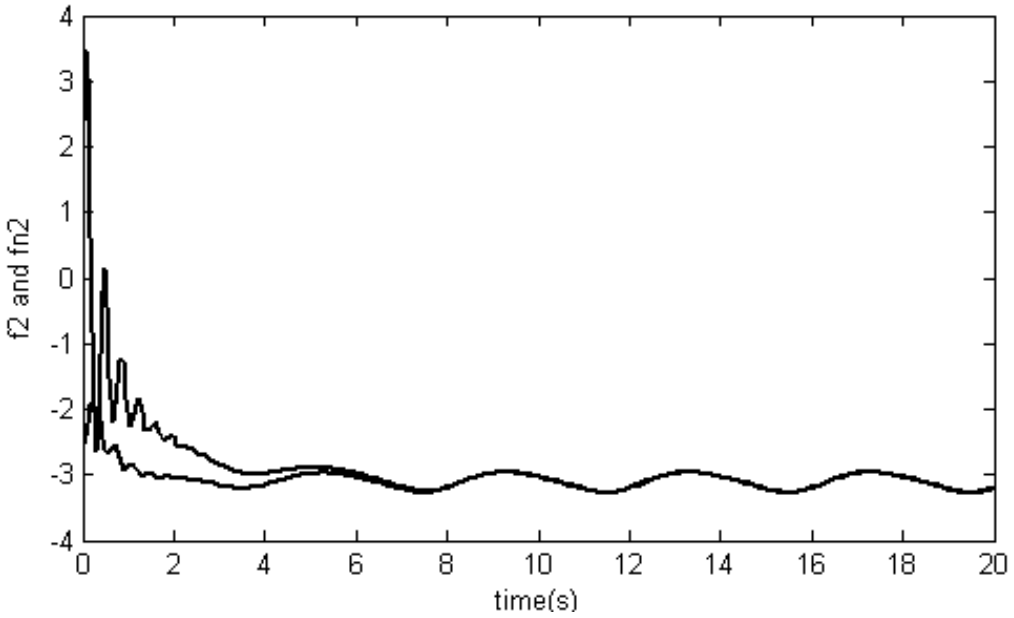


图9-30 关节2的模型不确定项及其估计

9.9 基于模型整体逼近的机器人RBF网络自适应控制

研究基于模型整体逼近的机器人RBF网络自适应控制设计方法。

（一）问题的提出

设n关节机械手方程为：

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}(\mathbf{q})+\mathbf{F}(\dot{\mathbf{q}})+\boldsymbol{\tau}_d=\boldsymbol{\tau}$$

其中 $\mathbf{D}(\mathbf{q})$ 为$n\times n$阶正定惯性矩阵，$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ 为$n\times n$阶惯性矩阵，$\mathrm{G}(\mathrm{q})$为$n\times 1$阶惯性向量， $\mathbf{F}(\dot{\mathbf{q}})$为摩擦力，$\tau_d$为未知外加干扰，$\tau$为控制输入。

9.9 基于模型整体逼近的机器人RBF网络自适应控制

跟踪误差为：　$\mathbf{e}(t)=\mathbf{q}_{\mathrm{d}}(t)-\mathbf{q}(t)$

定义误差函数为：

$$\mathbf{r}=\dot{\mathbf{e}}+\boldsymbol{\Lambda}\mathbf{e} \tag{9.65}$$

其中　$\boldsymbol{\Lambda}=\boldsymbol{\Lambda}^{\mathrm{T}}>0$ ，则　$\dot{\mathbf{q}}=-\mathbf{r}+\dot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\mathbf{e}$ ⇒ $\dot{\mathbf{r}}=\ddot{\mathbf{q}}_{\mathrm{d}}-\ddot{\mathbf{q}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}$

$$\begin{aligned}
\mathbf{D}\dot{\mathbf{r}} &= \mathbf{D}\left(\ddot{\mathbf{q}}_{\mathrm{d}}-\ddot{\mathbf{q}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}\right)=\mathbf{D}\left(\ddot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}\right)-\mathbf{D}\ddot{\mathbf{q}} \\
&= \mathbf{D}\left(\ddot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}\right)+\mathbf{C}\dot{\mathbf{q}}+\mathbf{G}+\mathbf{F}+\boldsymbol{\tau}_{\mathrm{d}}-\boldsymbol{\tau} \\
&= \mathbf{D}\left(\ddot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}\right)-\mathbf{C}\mathbf{r}+\mathbf{C}\left(\dot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\mathbf{e}\right)+\mathbf{G}+\mathbf{F}+\boldsymbol{\tau}_{\mathrm{d}}-\boldsymbol{\tau} \\
&= -\mathbf{C}\mathbf{r}-\boldsymbol{\tau}+\mathbf{f}+\boldsymbol{\tau}_{\mathrm{d}}
\end{aligned} \tag{9.66}$$

⇒ $\mathbf{D}\dot{\mathbf{r}}=-\mathbf{C}\mathbf{r}-\boldsymbol{\tau}+\mathbf{f}+\boldsymbol{\tau}_{\mathrm{d}}$

其中　$\mathbf{f}=\mathbf{D}\left(\ddot{\mathbf{q}}_{\mathrm{d}}+\boldsymbol{\Lambda}\dot{\mathbf{e}}\right)+\mathbf{C}\left(\mathbf{q}_{\mathrm{d}}+\boldsymbol{\Lambda}\mathbf{e}\right)+\mathbf{G}+\mathbf{F}$ ⇐ 模型不确定

9.9 基于模型整体逼近的机器人RBF网络自适应控制

在实际工程中，模型不确定项 f 为未知，为此，需要对不确定项 f 进行逼近。采用RBF网络逼近 $f(x)$，根据 $f(x)$ 的表达式，网络输入取 $\mathbf{x}=\begin{bmatrix}\mathbf{e}^{\mathrm{T}} & \dot{\mathbf{e}}^{\mathrm{T}} & \mathbf{q}_{\mathrm{d}}^{\mathrm{T}} & \dot{\mathbf{q}}_{\mathrm{d}}^{\mathrm{T}} & \ddot{\mathbf{q}}_{\mathrm{d}}^{\mathrm{T}}\end{bmatrix}$

设计控制律为：

$$\boldsymbol{\tau}=\hat{\mathbf{f}}+\mathbf{K}_{\mathrm{v}}\mathbf{r} \tag{9.67}$$

其中 $\hat{\mathbf{f}}(\mathbf{x})$ 为针对 f 进行逼近的RBF网络输出值。

将控制律式（9.67）代入式（9.66），得：

$$\begin{aligned}\mathbf{D}\dot{\mathbf{r}} &= -\mathbf{C}\mathbf{r}-\hat{\mathbf{f}}-\mathbf{K}_{\mathrm{v}}\mathbf{r}+\mathbf{f}+\boldsymbol{\tau}_{\mathrm{d}} \\ &= -(\mathbf{K}_{\mathrm{v}}+\mathbf{C})\mathbf{r}+\tilde{\mathbf{f}}+\boldsymbol{\tau}_{\mathrm{d}} = -(\mathbf{K}_{\mathrm{v}}+\mathbf{C})\mathbf{r}+\boldsymbol{\varsigma}_0\end{aligned} \tag{9.68}$$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

如果定义Lyapunov函数 $L=\frac{1}{2}\mathbf{r}^{\mathrm{T}}\mathbf{D}\mathbf{r}$

则 $$\dot{L}=\mathbf{r}^{\mathrm{T}}\mathbf{D}\dot{\mathbf{r}}+\frac{1}{2}\mathbf{r}^{\mathrm{T}}\dot{\mathbf{D}}\mathbf{r}=-\mathbf{r}^{\mathrm{T}}\mathbf{K}_{\mathrm{v}}\mathbf{r}+\frac{1}{2}\mathbf{r}^{\mathrm{T}}\left(\dot{\mathbf{D}}-2\mathbf{C}\right)\mathbf{r}+\mathbf{r}^{\mathrm{T}}\boldsymbol{\varsigma}_0$$

$$\dot{L}=\mathbf{r}^{\mathrm{T}}\boldsymbol{\varsigma}_0-\mathbf{r}^{\mathrm{T}}\mathbf{K}_{\mathrm{v}}\mathbf{r}$$

这说明在$\mathbf{K}_{\mathrm{v}}$固定条件下，控制系统的稳定依赖于$\boldsymbol{\varsigma}_0$，即$\hat{\mathbf{f}}$对$\mathbf{f}$的逼近精度及干扰$\boldsymbol{\tau}_{\mathrm{d}}$的大小。

采用RBF网络对不确定项 $\mathbf{f}$ 进行逼近。理想的RBF网络算法为：

$$\phi_i=g\left(\left\|\mathbf{x}-\mathbf{c}_i\right\|^2/\sigma_i^2\right)\quad i=1,2,\cdots,n$$

$$y=\mathbf{W}^{*T}\boldsymbol{\varphi}\left(\mathbf{x}\right)\qquad f(x)=w^{*T}\boldsymbol{\varphi}\left(\mathbf{x}\right)+\varepsilon$$

其中x为网络的输入信号，$\boldsymbol{\varphi}=\left[f_1\ f_2\ \cdots\ f_n\right]$　理想的权值　逼近误差

9.9 基于模型整体逼近的机器人RBF网络自适应控制

（二）针对 $f(x)$ 进行逼近的控制

1．控制器的设计

采用RBF网络逼近，则RBF神经网络的输出为：

$$\hat{\mathbf{f}}(\mathbf{x})=\hat{\mathbf{W}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x}) \tag{9.69}$$

取 $\tilde{\mathbf{W}}=\mathbf{W}-\hat{\mathbf{W}}$ $\|\mathbf{W}\|_{\mathrm{F}}\le W_{\max}$

设计控制律为：

$$\boldsymbol{\tau}=\hat{\mathbf{f}}(\mathbf{x})+\mathbf{K}_{\mathrm{v}}\mathbf{r}-\mathbf{v} \tag{9.70}$$

其中 v 为用于克服神经网络逼近误差ε的鲁棒项。

将控制律式（9.70）代入式（9.66），得：

$$\mathbf{D}\dot{\mathbf{r}}=-(\mathbf{K}_{\mathrm{v}}+\mathbf{C})\mathbf{r}+\tilde{\mathbf{W}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})+(\boldsymbol{\varepsilon}+\boldsymbol{\tau}_{\mathrm{d}})+\mathbf{v}=-(\mathbf{K}_{\mathrm{v}}+\mathbf{C})\mathbf{r}+\varsigma_1 \tag{9.71}$$

其中 $\varsigma_1=\tilde{\mathbf{W}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x})+(\boldsymbol{\varepsilon}+\boldsymbol{\tau}_{\mathrm{d}})+\mathbf{v}$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

将鲁棒项v设计为：

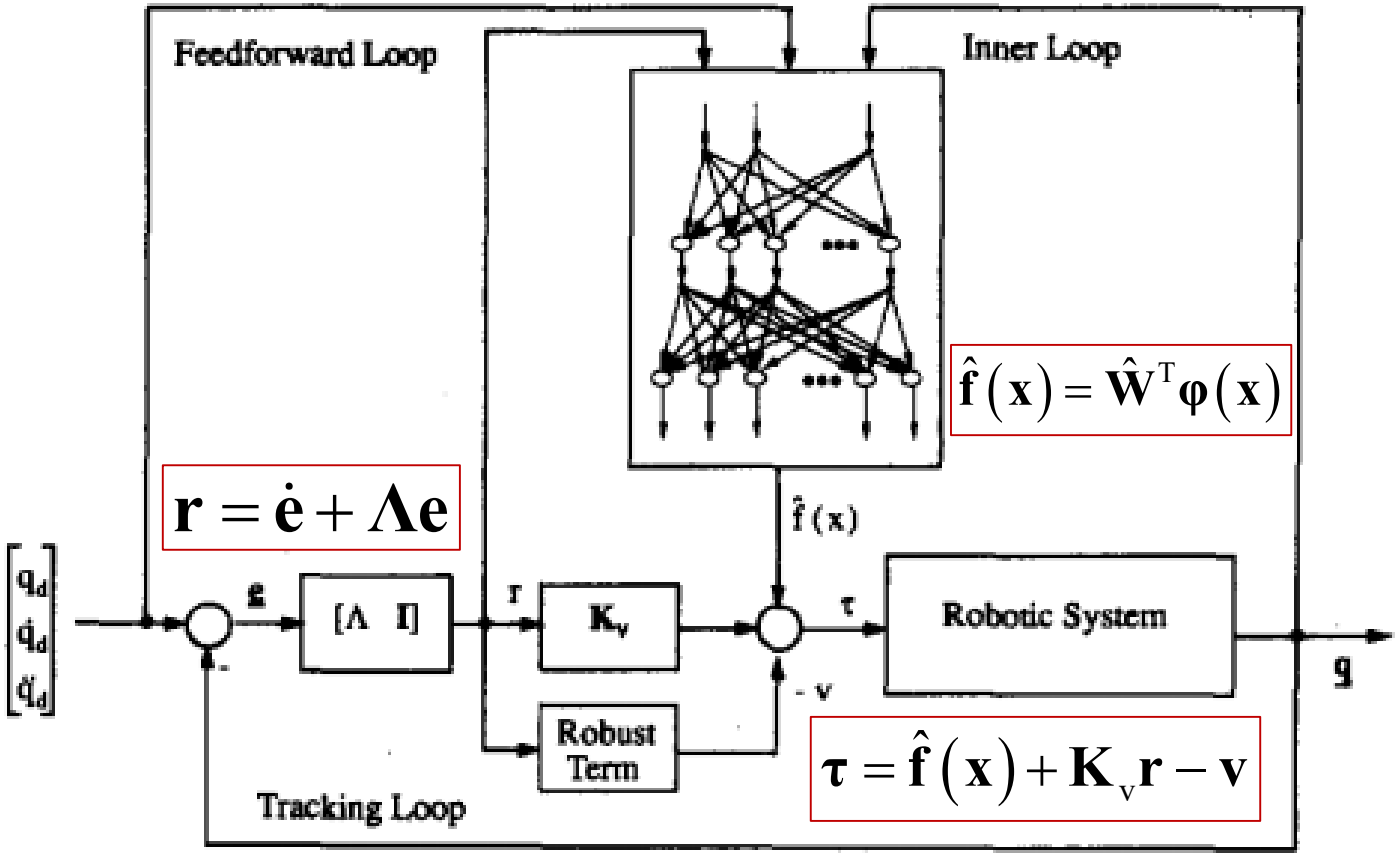
$$\mathbf{v} = -\left(\varepsilon_{\mathrm{N}} + b_{\mathrm{d}} + \eta_0\right)\mathrm{sgn}\left(\mathbf{r}\right) \qquad (9.72)$$

其中 $\|\boldsymbol{\varepsilon}\| \le e_{\mathrm{N}}$ $\|\boldsymbol{\tau}_{\mathrm{d}}\| \le b_{\mathrm{d}}$ $h_0 > 0$

```
v=-(epN+bd)*sign(r);
```

$$\mathbf{v} = -\left(\varepsilon_{\mathrm{N}} + b_{\mathrm{d}}\right)\mathrm{sgn}\left(\mathbf{r}\right)$$

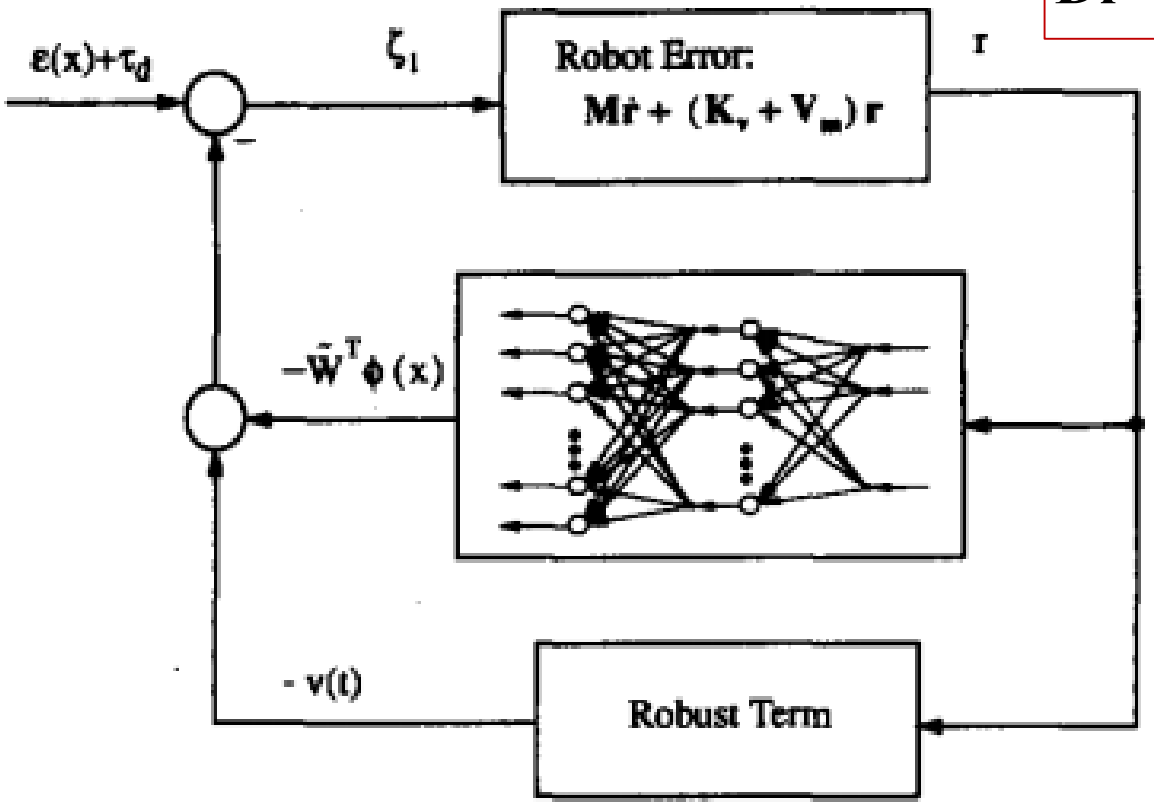
9.9 基于模型整体逼近的机器人RBF网络自适应控制



9.9 基于模型整体逼近的机器人RBF网络自适应控制

$$\varsigma_1 = \tilde{\mathbf{W}}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{x}) + (\boldsymbol{\varepsilon} + \boldsymbol{\tau}_{\mathrm{d}}) + \mathbf{v}$$

$$\mathbf{D}\dot{\mathbf{r}} + (\mathbf{K}_{\mathrm{v}} + \mathbf{C})\mathbf{r} = \varsigma_1$$



$$\mathbf{v} = -(\varepsilon_{\mathrm{N}} + b_{\mathrm{d}})\,\mathrm{sgn}(\mathbf{r})$$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

2．稳定性及收敛性分析

定义Lyapunov函数：

$$L=\frac{1}{2}\mathbf{r}^{\mathrm{T}}\mathbf{D}\mathbf{r}+\frac{1}{2}\mathrm{tr}\left(\tilde{\mathbf{W}}^{\mathrm{T}}\mathbf{F}^{-1}\tilde{\mathbf{W}}\right)$$

$$\dot{L}=\mathbf{r}^{\mathrm{T}}\mathbf{D}\dot{\mathbf{r}}+\frac{1}{2}\mathbf{r}^{\mathrm{T}}\dot{\mathbf{D}}\mathbf{r}+\mathrm{tr}\left(\tilde{\mathbf{W}}^{\mathrm{T}}\mathbf{F}^{-1}\dot{\tilde{\mathbf{W}}}\right)$$

则将（9.71）式代入上式，得

$$\dot{L}=-\mathbf{r}^{\mathrm{T}}\mathbf{K}_{\mathrm{v}}\mathbf{r}+\frac{1}{2}\mathbf{r}^{\mathrm{T}}\left(\dot{\mathbf{D}}-2\mathbf{C}\right)\mathbf{r}+\mathrm{tr}\tilde{\mathbf{W}}^{\mathrm{T}}\left(\mathbf{F}^{-1}\dot{\tilde{\mathbf{W}}}+\boldsymbol{\varphi}\mathbf{r}^{\mathrm{T}}\right)+\mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon}+\boldsymbol{\tau}_{\mathrm{d}}+\mathbf{v}\right)$$

根据机器人物理特性有 $\mathbf{r}^{\mathrm{T}}\left(\dot{\mathbf{D}}-2\mathbf{C}\right)\mathbf{r}=0$

得 $\dot{L}=-\mathbf{r}^{\mathrm{T}}\mathbf{K}_{\mathrm{v}}\mathbf{r}+\mathrm{tr}\tilde{\mathbf{W}}^{\mathrm{T}}\left(\mathbf{F}^{-1}\dot{\tilde{\mathbf{W}}}+\boldsymbol{\varphi}\mathbf{r}^{\mathrm{T}}\right)+\mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon}+\boldsymbol{\tau}_{\mathrm{d}}+\mathbf{v}\right)$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

取 $\dot{\tilde{\mathbf{W}}} = -\mathbf{F}\boldsymbol{\varphi}\mathbf{r}^{\mathrm{T}}$，即神经网络自适应律为：

$$\tilde{\mathbf{W}} = \mathbf{W} - \hat{\mathbf{W}} \Rightarrow \dot{\tilde{\mathbf{W}}} = -\dot{\hat{\mathbf{W}}} \qquad \dot{\hat{\mathbf{W}}} = \mathbf{F}\boldsymbol{\varphi}\mathbf{r}^{\mathrm{T}} \tag{9.73}$$

则

$$\dot{L} = -\mathbf{r}^{\mathrm{T}}\mathbf{K}_{\mathrm{v}}\mathbf{r} + \mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon} + \boldsymbol{\tau}_{\mathrm{d}} + \mathbf{v}\right)$$

由于

$$\mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon} + \boldsymbol{\tau}_{\mathrm{d}} + \mathbf{v}\right) =$$

则

$$\mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon} + \boldsymbol{\tau}_{\mathrm{d}}\right) + \mathbf{r}^{\mathrm{T}}\mathbf{v} = \mathbf{r}^{\mathrm{T}}\left(\boldsymbol{\varepsilon} + \boldsymbol{\tau}_{\mathrm{d}}\right) - \|\mathbf{r}\|\left(\boldsymbol{\varepsilon}_{\mathrm{N}} + \mathbf{b}_{\mathrm{d}}\right) \le 0$$

$$\dot{L} \le 0$$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

（三）仿真实例

选两关节机器人力臂系统，其动力学模型为：

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\mathbf{G}(\mathbf{q})+\mathbf{F}(\dot{\mathbf{q}})+\boldsymbol{\tau}_{\mathrm{d}}=\boldsymbol{\tau}$$

其中

$$\mathbf{D}(\mathbf{q})=\begin{bmatrix} p_1+p_2+2p_3\cos q_2 & p_2+p_3\cos q_2 \\ p_2+p_3\cos q_2 & p_2 \end{bmatrix} \qquad \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})=\begin{bmatrix} -p_3\dot{q}_2\sin q_2 & -p_3(\dot{q}_1+\dot{q}_2)\sin q_2 \\ p_3\dot{q}_1\sin q_2 & 0 \end{bmatrix}$$

$$\mathbf{G}(\mathbf{q})=\begin{bmatrix} p_4 g\cos q_1+p_5 g\cos(q_1+q_2) \\ p_5 g\cos(q_1+q_2) \end{bmatrix} \qquad \mathbf{F}(\dot{\mathbf{q}})=0.02\,\mathrm{sgn}(\dot{\mathbf{q}}) \qquad \boldsymbol{\tau}_{\mathrm{d}}=\begin{bmatrix} 0.2\sin(t) & 0.2\sin(t) \end{bmatrix}^{\mathrm{T}}$$

取 $\mathbf{p}=[p_1,p_2,p_3,p_4,p_5]=[2.9,0.76,0.87,3.04,0.87]$

9.9 基于模型整体逼近的机器人RBF网络自适应控制

RBF网络高斯基函数参数的取值对神经网络控制的作用很重要，如果参数取值不合适，将使高斯基函数无法得到有效的映射，从而导致RBF网络无效。故c按网络输入值的范围取值，取b=10，网络的初始权值取零，网络输入取$\mathbf{z}=[\mathbf{e} \quad \dot{\mathbf{e}} \quad \mathbf{q}_d \quad \dot{\mathbf{q}}_d \quad \ddot{\mathbf{q}}_d]$

系统的初始状态为[0.09, 0, -0.09, 0]，两个关节的位置指令分别为

$q_{1d}=0.1\sin t$，$q_{2d}=0.1\sin t$，控制参数取 $\mathbf{K}_v=\mathrm{diag}\{20,20\}$，$\mathbf{F}=\mathrm{diag}\{15,15\}$，

$\Lambda=\mathrm{diag}\{5,5\}$，在鲁棒项中，取 $\varepsilon_N=0.20$，$b_d=0.10$。

```
z1=[e(1);de(1);qd(1);dqd(1);ddqd(1)];
z2=[e(2);de(2);qd(2);dqd(2);ddqd(2)];
for j=1:1:node
    h1(j)=exp(-norm(z1-c(:,j))^2/(b*b));
    h2(j)=exp(-norm(z2-c(:,j))^2/(b*b));
end
```

9.9 基于模型整体逼近的机器人RBF网络自适应控制

采用Simulink和S函数进行控制系统的仿真。首先采用针对$f(x)$进行逼近的控制器子程序chap9_6ctrl.m，控制律取式（9.72），自适应律取式（9.75），仿真结果如图9-31至图9-33所示。

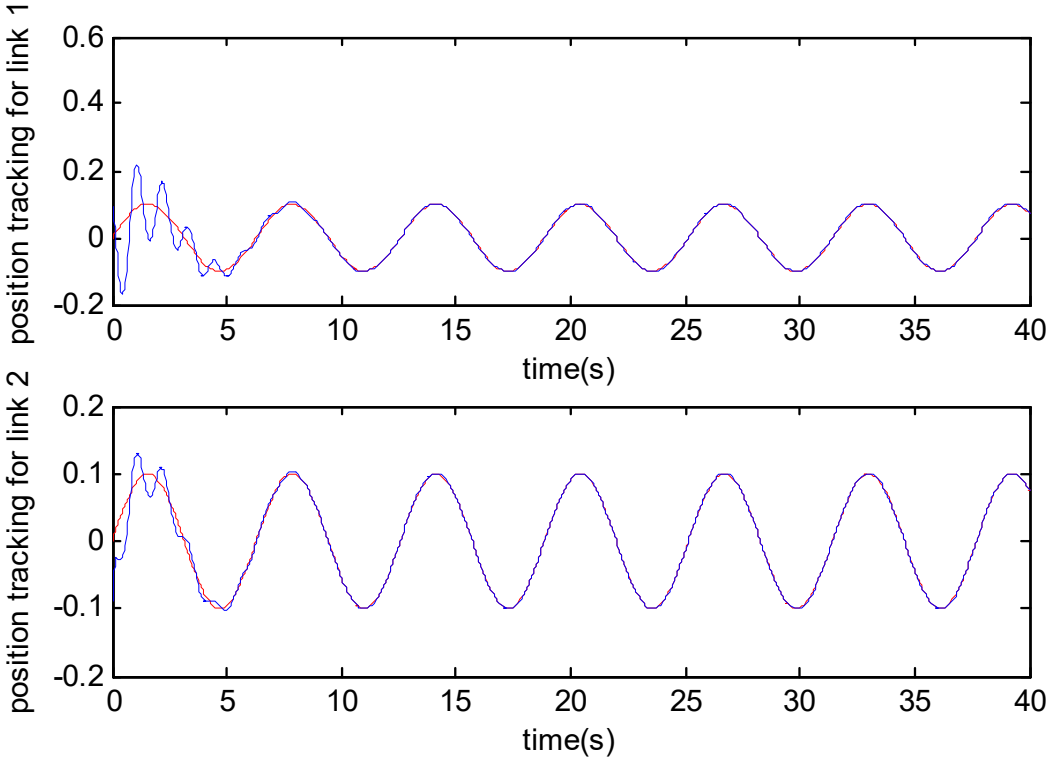


图9-31 关节1及关节2的位置跟踪

9.9 基于模型整体逼近的机器人RBF网络自适应控制

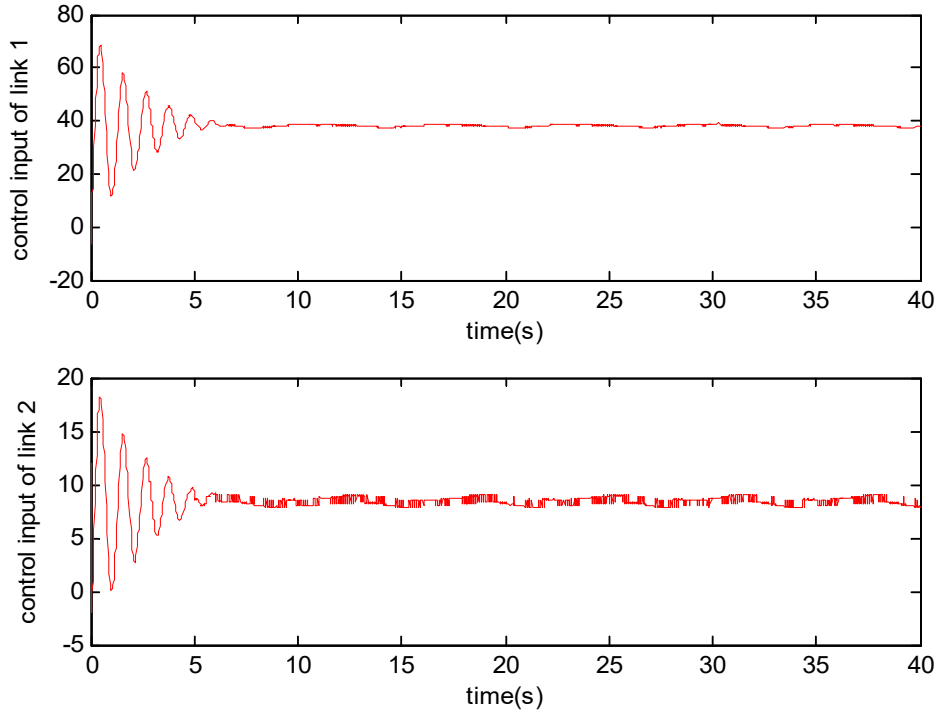


图9-32 关节1及关节2的控制输入

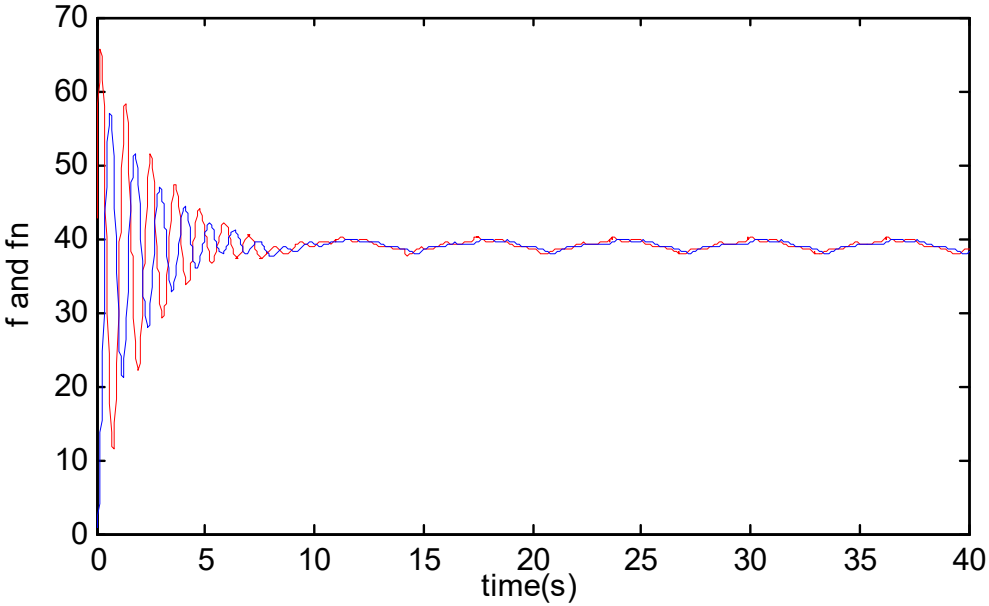


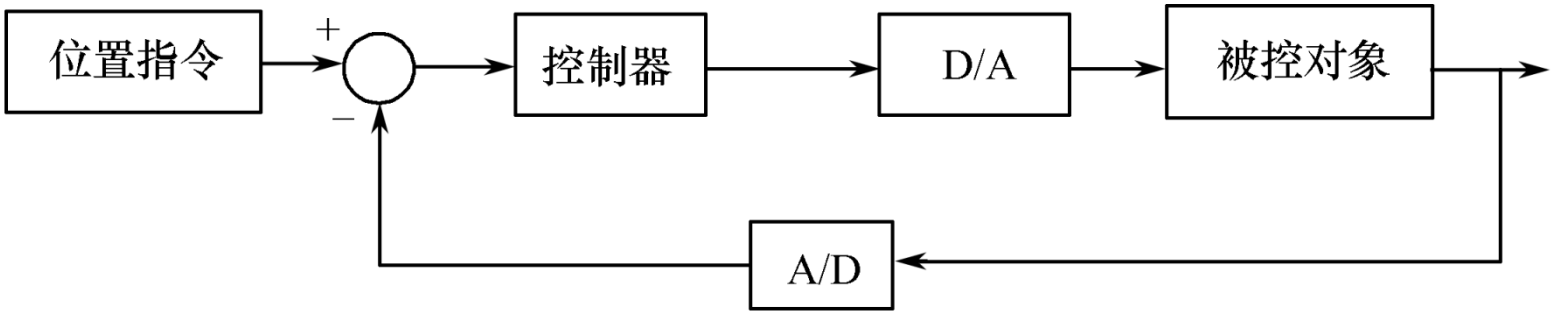
图9-33 关节1及关节2的$\|\mathbf{f}(\mathbf{x})\|$及其逼近$\|\hat{\mathbf{f}}(\mathbf{x})\|$

9.10 神经网络数字控制

（一）基本原理

在工程实际中，控制算法一般在计算机或DSP中实现，这就需要将控制算法离散化。本节讨论神经网络自适应控制律的数字化实现方法。

数字控制系统结构如下图所示，其中控制器为数字控制算法，被控对象的输入、输出为模拟信号，通过D/A和A/D与数字信号相连接。



9.10 神经网络数字控制

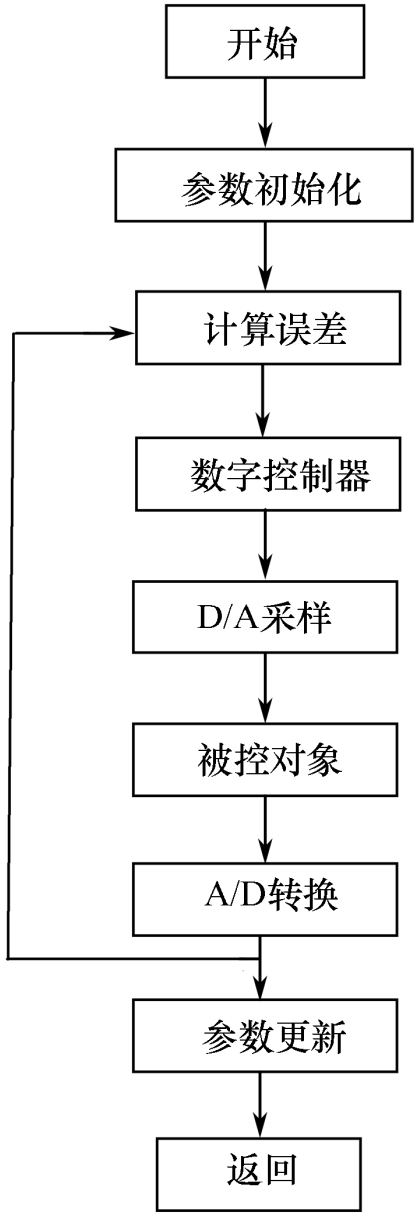


图9-35　数字控制算法程序框图

9.10 神经网络数字控制

针对9.8节中的控制算法，选择被控对象为单电机模型，其动力学模型为

$$\mathbf{D}\ddot{\mathbf{q}}+\mathbf{C}\dot{\mathbf{q}}+\mathbf{G}=\boldsymbol{\tau}+d \tag{9.74}$$

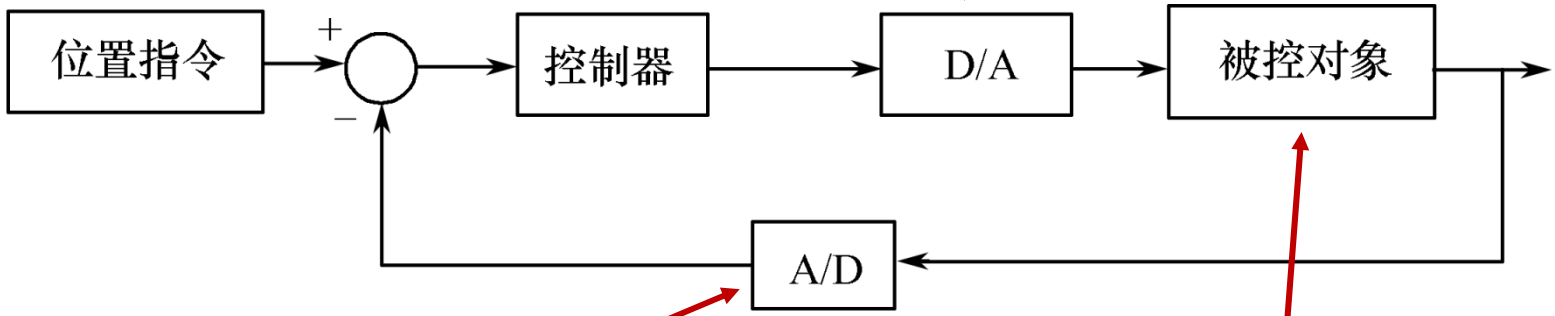
式中，$D_0=\frac{3}{4}ml^2$　　$G_0=m\lg\cos q$

取$x_1=q, x_2=\dot{q}$，则方程式(9.76)可转化为动力学方程

$$\begin{aligned}\dot{x}_1&=x_2\\ \dot{x}_2&=D^{-1}(\tau+d-C\dot{q}-G)\end{aligned} \tag{9.75}$$

在Matlab仿真中，在每个采样时间 T 内，采用Runge-Kutta迭代算法求解式(9.75)，从而实现连续被控对象的离散求解，仿真中采用了Matlab函数“ode45”进行离散积分求解。

9.10 神经网络数字控制




```
tol_1=tol(k);
para=tol_1;    %D/A
```

```
xk=xx(length(xx),:);  %A/D
```

```
[t,xx]=ode45('chap9_8plant',tSpa
n,xk,[],para);  %Plant
```

```
dx(1)=x(2);
dx(2)=inv(D)*(tol+d-C*x(2)-G);
```

9.10 神经网络数字控制

针对自适应律的离散化问题，一种方法是利用采样时间进行差分的离散化，另一种方法是利用数值迭代方法进行离散化。下面介绍一种高精度数值迭代方法——RKM(Runge-Kutta-Merson)方法。

以离散化 $\dot{x} = f(t, x)$ 为例，采样时间为T，

- 如果采用差分方法离散化，n+1时刻的x值为

$$x_{n+1} = x_n + Tf(t_n, x_n)$$

- 而采用RKM方法，则n+1时刻的x值为

$$x_{\mathrm{n+1}} = x_{\mathrm{n}} + \frac{1}{6}\left(k_1 + 4k_4 + k_5\right) \qquad (9.76)$$

9.10 神经网络数字控制

其中

$$k_1 = Tf\left(t_n, x_n\right)$$

$$k_2 = Tf\left(t_n + \frac{1}{3}T, x_n + \frac{1}{3}k_1\right)$$

$$k_3 = Tf\left(t_n + \frac{1}{3}T, x_n + \frac{1}{6}k_1 + \frac{1}{6}k_2\right)$$

$$k_4 = Tf\left(t_n + \frac{1}{2}T, x_n + \frac{1}{8}k_1 + \frac{3}{8}k_3\right)$$

$$k_5 = Tf\left(t_n + T, x_n + \frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4\right)$$

针对每个采样时间，采用RKM迭代算法式(9.76)进行求解，考虑到自适应律表达式中权值$\hat{\omega}$相当于式(9.76)中的x_n，即

$$w_{n+1} = w_n + \frac{1}{6}\left(k_1 + 4k_4 + k_5\right)$$

且没有出现时间变量t，采样时间为ts，故离散求解式(9.76)的Matlab程序如下：

9.10 神经网络数字控制

```
if S==1
        w(i,1)=w_1(i,1)+ts*(gama*h(i)*xi'*P*B-
k1*gama*norm(xi)*w_1(i,1)); % Adaptive law
elseif S==2
        k1=ts*(gama*h(i)*xi'*P*B-k1*gama*norm(xi)
*w_1(i,1));
        k2=ts*(gama*h(i)*xi'*P*B-
k1*gama*norm(xi)*(w_1(i,1)+1/3*k1));
        k3=ts*(gama*h(i)*xi'*P*B-
k1*gama*norm(xi)*(w_1(i,1)+1/6*k1+1/6*k2));
        k4=ts*(gama*h(i)*xi'*P*B-
k1*gama*norm(xi)*(w_1(i,1)+1/8*k1+3/8*k3));
        k5=ts*(gama*h(i)*xi'*P*B-
k1*gama*norm(xi)*(w_1(i,1)+1/2*k1-3/2*k3+2*k4));
        w(i,1)=w_1(i,1)+1/6*(k1+4*k4+k5);
end
```

9.10 神经网络数字控制

（二）仿真实例

仿真中，采用控制律式(9.53)和自适应律式(9.65)。在离散化自适应律式(9.65)微分方程时，取两种离散化方法：当S=1时，采用简单的差分方法进行离散化，当S=2时，采用RKM方法进行离散化。

取m=1，l=1，g=9.8，d=0.5sin(t)，

$C_0 = 2 \qquad \Delta D = 0.8D_0 \qquad \Delta C = 0.8C_0 \qquad \Delta G = 0.8G_0$

系统的初始状态为x=[0,0]。

9.10 神经网络数字控制

位置指令为q_d=0.5sin(k·ts)，采样时间取ts=0.001，取控制器参数为kp=40，kv=20，$Q=\begin{bmatrix}2000 & 0\\ 0 & 2000\end{bmatrix}$，取自适应律参数为γ=5，$k_1$=0.01，RBF网络的隐含层节点数取10，网络权值的初始值取0，高斯基参数初始值的选取见控制器主程序chap9_8.m。取M=1，未采用神经网络补偿，仿真结果如图9-36所示。取M=2，采用神经网络补偿，仿真结果如图9-37和图9-38所示。

```
if M==1        %No compensation
    tol(k)=D0*(ddqd(k)-kv*de(k)-
kp*e(k))+C0*dq(k)+G0(k);
elseif M==2   %Neural network
compensation
    tol(k)=D0*(ddqd(k)-kv*de(k)-
kp*e(k))+C0*dq(k)+G0(k)-D0*fp(k);
end
```

9.10 神经网络数字控制

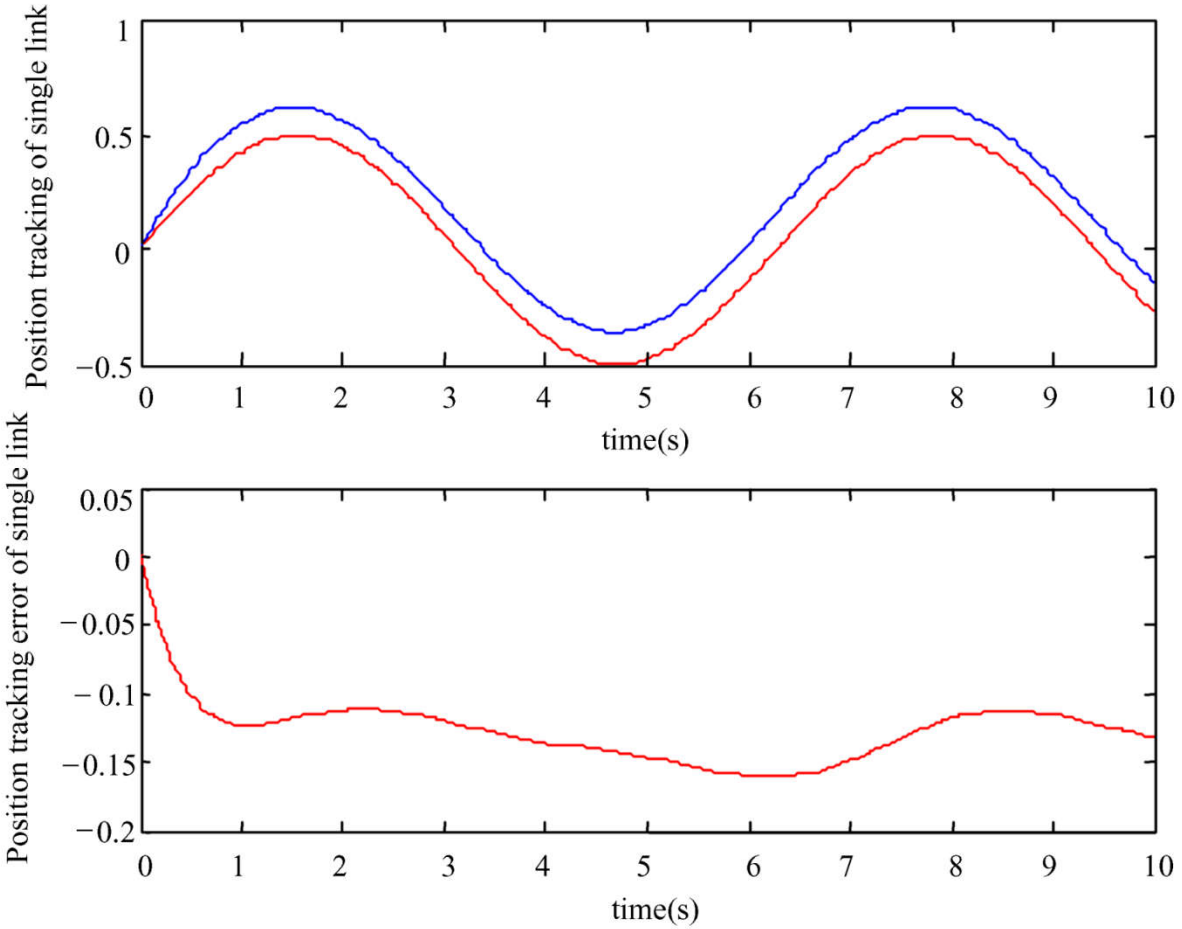


图9-36 未用神经网络补偿的位置跟踪及其误差（M=1）

9.10 神经网络数字控制

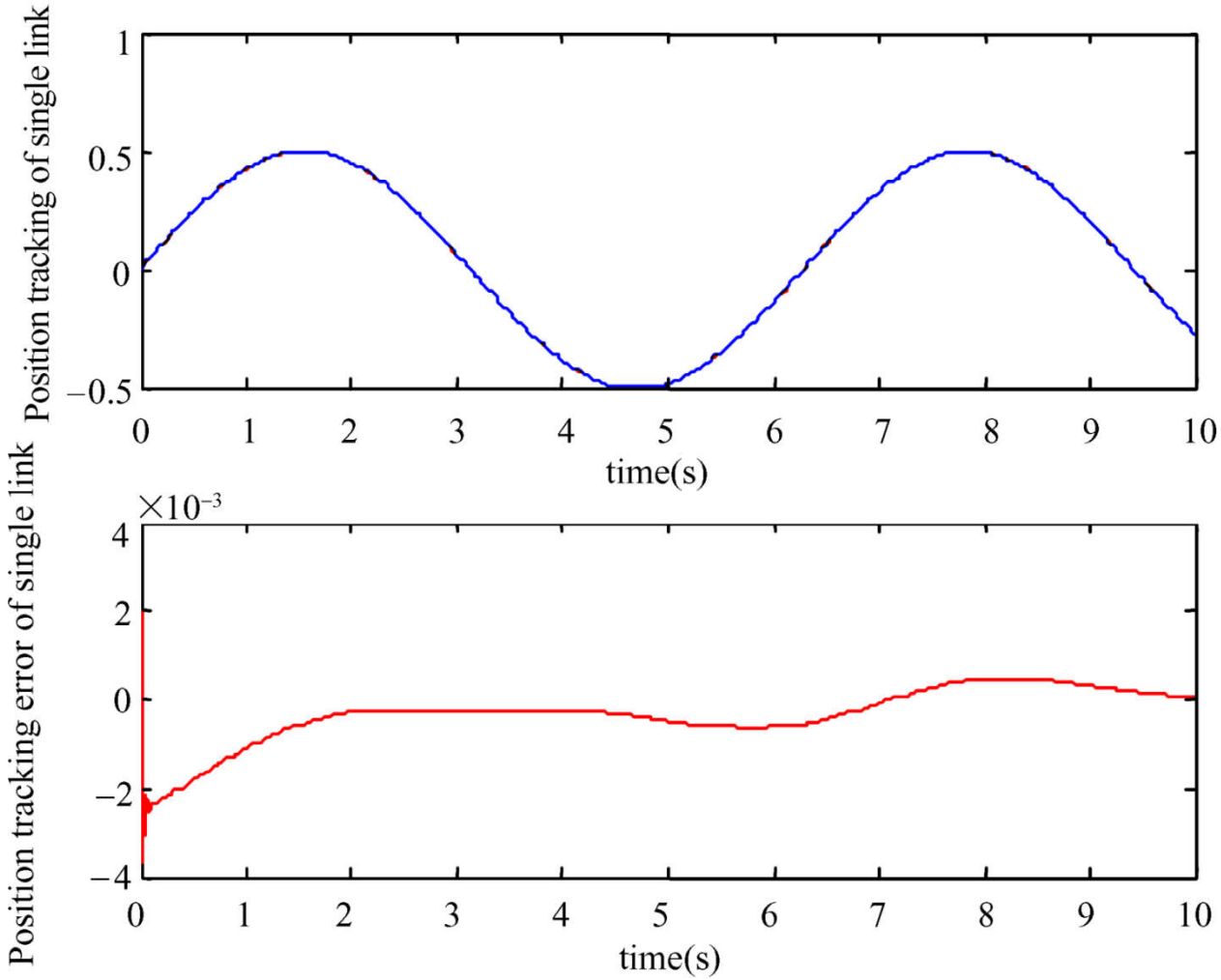


图9-37 采用神经网络补偿的位置跟踪及其误差（M=2）

9.10 神经网络数字控制

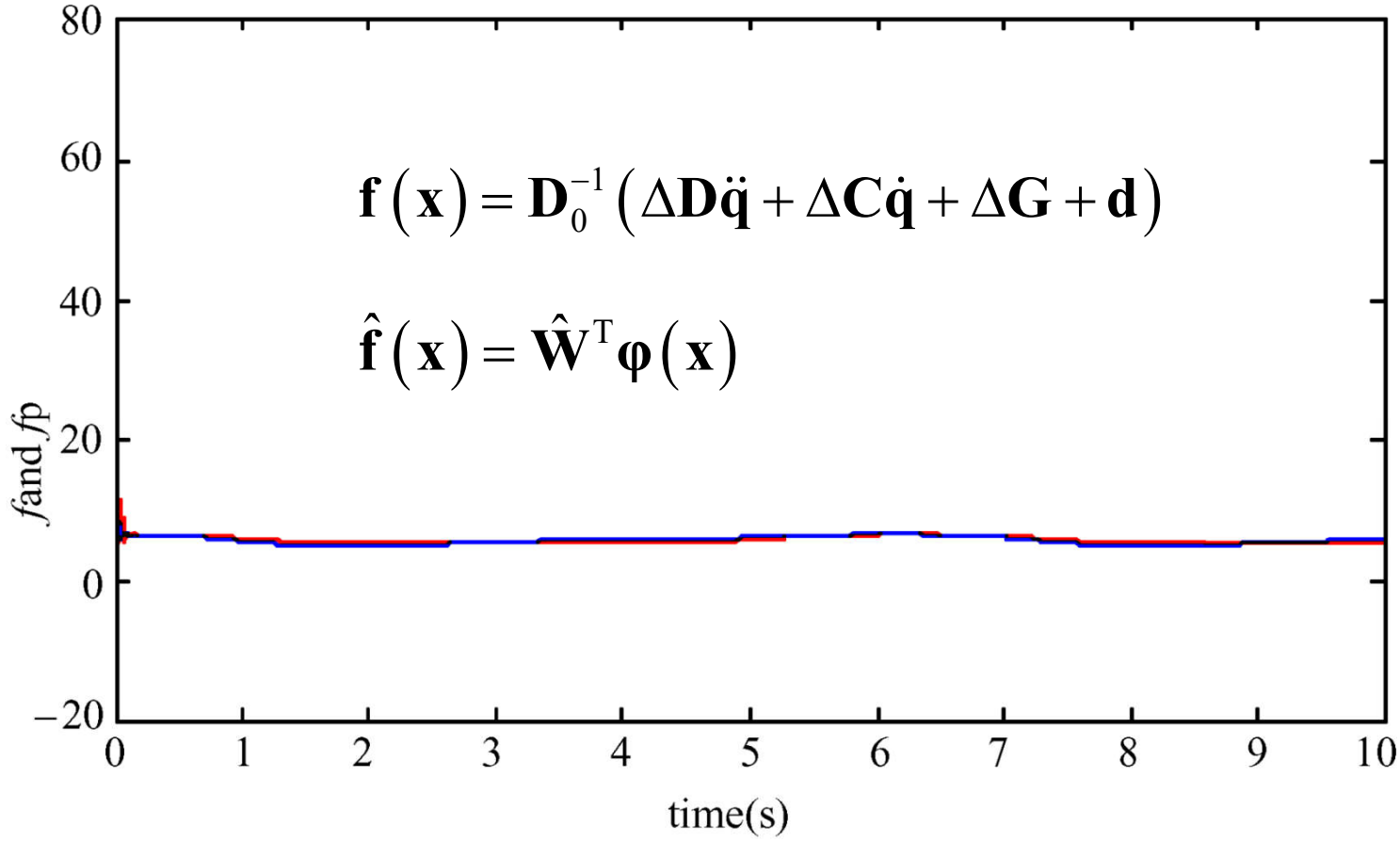


图9-38 不确定项及其神经网络逼近结果（M=2）

9.11 离散系统的RBF网络控制

离散时间控制系统设计在实际工程中具有重要意义。设计离散数字控制器有两种方法：

- 一种方法首先基于连续系统设计连续控制器，然后再将其离散化；
- 另一种方法是基于离散系统直接设计离散控制器。

本节讨论第二种离散控制方法，即非线性离散系统的神经网络控制方法。在离散系统的Lyapunov稳定性分析中，容易含有系统状态和神经网络权值的耦合平方项，使离散控制设计比连续系统控制律设计复杂。

9.11 离散系统的RBF网络控制

（一）系统描述

考虑如下非线性离散系统:

$$y(k+1)=f\left(\boldsymbol{x}(k)\right)+u(k) \tag{9.77}$$

式中 $\boldsymbol{x}(k)=\left[y(k)\quad y(k-1)\quad\cdots\quad y(k-n+1)\right]^{\mathrm{T}}$ 为状态向量，u(k)为控制输入，y(k)为系统输出，假设非线性光滑函数 $f(x(k))$为未知。控制任务为 y(k) 跟踪$y_d(k)$。

9.11 离散系统的RBF网络控制

（二）经典控制器设计

定义跟踪误差为 $e(k)=y(k)-y_d(k)$。如果$f(x(k))$已知，则可设计反馈线性化控制律为

$$u(k)=y_d(k+1)-f(\boldsymbol{x}(k))-c_1e(k) \qquad (9.78)$$

将式 (9.78)代入式 (9.77)中，得

$$y(k+1)=f(\boldsymbol{x}(k))+y_d(k+1)-f(\boldsymbol{x}(k))-c_1e(k)$$

可得到渐进收敛误差动态方程为

$$e(k+1)+c_1e(k)=0 \qquad (9.79)$$

如果取 $|c_1|<1$ ，则可保证$k\to\infty$时，$e(k)\to 0$。

$$-1<c_1<1 \Rightarrow -1<-c_1<1 \Rightarrow e(k+1)=-c_1e(k)<e(k)$$

9.11 离散系统的RBF网络控制

（三）自适应神经网络控制器设计

如果 $f(x(k))$ 是未知的，可用RBF神经网络逼近 $f(x(k))$。神经网络的输出为

$$\hat{f}\left(\boldsymbol{x}(k)\right)=\hat{\boldsymbol{w}}(k)^{\mathrm{T}}\,\boldsymbol{h}\left(\boldsymbol{x}(k)\right) \tag{9.80}$$

其中 $\hat{\boldsymbol{w}}(k)$ 为神经网络输出权值向量，$\boldsymbol{h}(x(k))$ 为高斯基函数。

根据RBF网络逼近定理，对于任意小的非零逼近误差 ε_f，则存在某个最优权值向量 $\boldsymbol{w}^*$，使

$$f(\boldsymbol{x})=\hat{f}\left(\boldsymbol{x},\boldsymbol{w}^*\right)-\Delta_f(\boldsymbol{x}) \tag{9.81}$$

其中 $\Delta_f(\boldsymbol{x})$ 为最优神经网络逼近误差，$\left|\Delta_f(\boldsymbol{x})\right|<\varepsilon_f$ 。

9.11 离散系统的RBF网络控制

则神经网络逼近误差为

$$\begin{aligned}\tilde{f}\left(\boldsymbol{x}(k)\right) &= f\left(\boldsymbol{x}(k)\right) - \hat{f}\left(\boldsymbol{x}(k)\right) \\ &= \hat{f}\left(\boldsymbol{x}, \boldsymbol{w}^{*}\right) - \Delta_f\left(\boldsymbol{x}(k)\right) - \hat{\boldsymbol{w}}(k)^{\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k)\right) \\ &= \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k)\right) - \hat{\boldsymbol{w}}(k)^{\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k)\right) - \Delta_f\left(\boldsymbol{x}(k)\right) \\ &= -\tilde{\boldsymbol{w}}(k)^{\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k)\right) - \Delta_f\left(\boldsymbol{x}(k)\right)\end{aligned} \tag{9.82}$$

其中 $\tilde{\boldsymbol{w}}(k) = \hat{\boldsymbol{w}}(k) - \boldsymbol{w}^{*}$

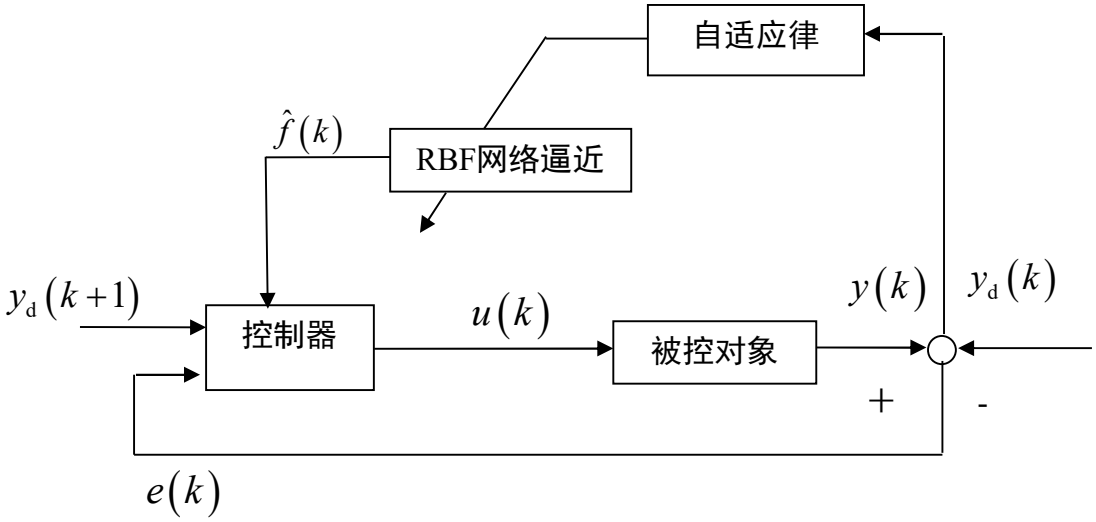
$$\tilde{f}\left(\boldsymbol{x}(k)\right) = -\tilde{\boldsymbol{w}}(k)^{\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k)\right) - \Delta_f\left(\boldsymbol{x}(k)\right)$$

9.11 离散系统的RBF网络控制

采用神经网络逼近未知函数，根据式（9.78），控制律可设计为

$$u(k)=y_{\mathrm{d}}(k+1)-\hat{f}(\boldsymbol{x}(k))-c_1 e(k) \tag{9.83}$$

下图为基于神经网络逼近的自适应控制系统框图。



9.11 离散系统的RBF网络控制

将式(9.83)代入式(9.77)，可得

$$e(k+1)=\tilde{f}\left(\boldsymbol{x}(k)\right)-c_1 e(k)$$

则

$$e(k)+c_1 e(k-1)=\tilde{f}\left(\boldsymbol{x}(k-1)\right) \qquad (9.84)$$

式 (9.84)的另一种表达方式为

$$e(k)=\Gamma^{-1}\left(z^{-1}\right)\tilde{f}\left(\boldsymbol{x}(k-1)\right) \qquad (9.85)$$

其中 $\Gamma\left(z^{-1}\right)=1+c_1 z^{-1}$，$z^{-1}$为离散时间延时因子。

9.11 离散系统的RBF网络控制

定义一个新的误差函数为

辅助控制信号

$$e_1(k)=\beta\left(e(k)-\Gamma^{-1}\left(z^{-1}\right)v(k)\right) \qquad (9.86)$$

其中$\beta>0$。

$$e(k)=\Gamma^{-1}\left(z^{-1}\right)\tilde{f}\left(\boldsymbol{x}(k-1)\right)$$

将式(9.85)代入式(9.86)，可得

$$e_1(k)=\beta\Gamma^{-1}\left(z^{-1}\right)\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)=\beta\frac{1}{1+c_1z^{-1}}\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)$$

9.11 离散系统的RBF网络控制

$$e_1(k)\left(1+c_1z^{-1}\right)=\beta\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)$$
$$=e_1(k)+c_1e_1(k)z^{-1}$$
$$=e_1(k)+c_1e_1(k-1)$$

整理可得

$$e_1(k-1)=\frac{\beta\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)-e_1(k)}{c_1} \tag{9.87}$$

设计自适应律为

$$\Delta\hat{\boldsymbol{w}}(k)=\begin{cases}\dfrac{\beta}{\gamma c_1^2}\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)e_1(k) & \text{if } |e_1(k)|>\varepsilon_f/\mathrm{G}\\ 0 & \text{if } |e_1(k)|\le\varepsilon_f/\mathrm{G}\end{cases} \tag{9.88}$$

其中$\Delta\hat{\boldsymbol{w}}(k)=\hat{\boldsymbol{w}}(k)-\hat{\boldsymbol{w}}(k-1)$，γ和G是严格的正常数。

9.11 离散系统的RBF网络控制

（四）稳定性分析

根据闭环系统的性能要求，定义离散时间Lyapunov函数为

$$V(k)=e_1^2(k)+\gamma\tilde{\boldsymbol{w}}^{\mathrm{T}}(k)\tilde{\boldsymbol{w}}(k) \qquad (9.89)$$

$V(k)$的一阶差分为

$$\Delta V(k)=V(k)-V(k-1)$$

$$=e_1^2(k)-e_1^2(k-1)+\gamma\left(\tilde{\boldsymbol{w}}^{\mathrm{T}}(k)+\tilde{\boldsymbol{w}}^{\mathrm{T}}(k-1)\right)\left(\tilde{\boldsymbol{w}}(k)-\tilde{\boldsymbol{w}}(k-1)\right)$$

稳定性分析分为以下三步进行。

9.11 离散系统的RBF网络控制

首先，将式（9.87）的 $e_1(k-1)$ 代入上式，可得

$$\Delta V(k)=e_1^2(k)-\frac{e_1^2(k)+\beta^2\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)^2-2\beta\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)e_1(k)}{c_1^2}$$

$$+\gamma\left(\left(\hat{\boldsymbol{w}}(k)-\boldsymbol{w}^*\right)^{\mathrm{T}}+\left(\hat{\boldsymbol{w}}(k-1)-\boldsymbol{w}^*\right)^{\mathrm{T}}\right)\left(\left(\hat{\boldsymbol{w}}(k)-\boldsymbol{w}^*\right)-\left(\hat{\boldsymbol{w}}(k-1)-\boldsymbol{w}^*\right)\right)$$

$$=\boxed{-V_1}+\frac{2\beta\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)e_1(k)}{c_1^2}+\gamma\left(\Delta\hat{\boldsymbol{w}}^{\mathrm{T}}(k)+2\tilde{\boldsymbol{w}}^{\mathrm{T}}(k-1)\right)\Delta\hat{\boldsymbol{w}}(k)$$

≤ 0

其中 $V_1=\dfrac{e_1^2(k)\left(1-c_1^2\right)}{c_1^2}+\dfrac{\beta^2\left(\tilde{f}\left(\boldsymbol{x}(k-1)\right)-v(k)\right)^2}{c_1^2}\geq 0$

9.11 离散系统的RBF网络控制

其次，将式(9.82)代入上式的 $\tilde{f}\left(\boldsymbol{x}(k-1)\right)$ 中，整理可得

$$\Delta V(k)=-V_1+\frac{2\beta\left(-\tilde{\boldsymbol{w}}(k-1)^{\mathrm{T}}\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)-\Delta_f\left(\boldsymbol{x}(k-1)\right)-v(k)\right)e_1(k)}{c_1^2}$$

$$+\gamma\Delta\hat{\boldsymbol{w}}^{\mathrm{T}}(k)\Delta\hat{\boldsymbol{w}}(k)+2\gamma\tilde{\boldsymbol{w}}^{\mathrm{T}}(k-1)\Delta\hat{\boldsymbol{w}}(k)$$

$$=-V_1+2\tilde{\boldsymbol{w}}^{\mathrm{T}}(k-1)\left(\gamma\Delta\hat{\boldsymbol{w}}(k)-\frac{\beta}{c_1^2}\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)e_1(k)\right)$$

$$-\frac{2\beta}{c_1^2}\left(\Delta_f\left(\boldsymbol{x}(k-1)\right)+v(k)\right)e_1(k)+\gamma\Delta\hat{\boldsymbol{w}}^{\mathrm{T}}(k)\Delta\hat{\boldsymbol{w}}(k)$$

9.11 离散系统的RBF网络控制

最后，将自适应律式（9.88）代入上式，可得

$$\Delta V(k)=\begin{cases}-V_1-\dfrac{2\beta}{c_1^2}\left(\Delta_f\left(\boldsymbol{x}(k-1)\right)+v(k)\right)e_1(k)+ \\ \left(\dfrac{\beta}{\sqrt{\gamma}c_1^2}\right)^2\boldsymbol{h}^{\mathrm{T}}\left(\boldsymbol{x}(k-1)\right)\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)e_1^2(k),\ \text{if}\ \left|e_1(k)\right|>\varepsilon_f/G \\ -V_1-\dfrac{2\beta}{c_1^2}\left[\left(\left(\tilde{\boldsymbol{w}}^{\mathrm{T}}(k-1)\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)\right)+\right.\right. \\ \left.\left.v(k)+\Delta_f\left(\boldsymbol{x}(k-1)\right)e_1(k)\right],\ \text{if}\ \left|e_1(k)\right|\le\varepsilon_f/G\end{cases} \tag{9.90}$$

9.11 离散系统的RBF网络控制

辅助控制信号$v(k)$的设计应保证$e_1(\mathrm{k})\to 0$，从而$e(\mathrm{k})\to 0$。

设计辅助控制信号为

$$v(k)=v_1(k)+v_2(k) \tag{9.91}$$

其中 $$v_1(k)=\frac{\beta}{2\gamma c_1^2}\boldsymbol{h}^{\mathrm{T}}\left(\boldsymbol{x}(k-1)\right)\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)e_1(k)$$

$$v_2(k)=Ge_1(k)$$

$$\boxed{e_1(k)=\beta\left(e(k)-\Gamma^{-1}\left(z^{-1}\right)v(k)\right)}$$

9.11 离散系统的RBF网络控制

- 如果$\left|e_1(k)\right| > \varepsilon_f / G$，将式(9.91)代入式 (9.90)，整理可得

$$\Delta V(k) = \underbrace{-V_1}_{\leq 0} - \frac{2\beta}{c_1^2}\left(\Delta_f\left(\boldsymbol{x}(k-1)\right) + Ge_1(k)\right)e_1(k)$$

$$\leq -\frac{2\beta}{c_1^2}\left(\Delta_f\left(\boldsymbol{x}(k-1)\right) + Ge_1(k)\right)e_1(k)$$

最优神经网络逼近误差

由 $\left|\Delta_f(\boldsymbol{x})\right| < \varepsilon_f$，$\left|e_1(k)\right| > \varepsilon_f / G$，则

$$\left|e_1(k)\right| > \frac{\left|\Delta_f\left(\boldsymbol{x}(k-1)\right)\right|}{G} \Rightarrow e_1^2(k) > -\frac{\Delta_f\left(\boldsymbol{x}(k-1)\right)e_1(k)}{G}$$

因此 $\left(\Delta_f\left(\boldsymbol{x}(k-1)\right) + Ge_1(k)\right)e_1(k) > 0$

从而可得 $\Delta V(k) < 0$

- 如果$\left|e_1(k)\right| \leq \varepsilon_f / G$，则可保证跟踪性能，且$\Delta V(k)$可以任意小。

9.11 离散系统的RBF网络控制

仿真说明如下：

（1）由式(9.86)可得 $e_1(k)=\beta\left(e(k)-\dfrac{1}{1+c_1z^{-1}}v(k)\right)$，则

$$e_1(k)\left(1+c_1z^{-1}\right)=\beta\left(e(k)\left(1+c_1z^{-1}\right)-v(k)\right)$$

因此

$$e_1(k)=-c_1e_1(k-1)+\beta\left(e(k)+c_1e(k-1)-v(k)\right) \qquad (9.92)$$

9.11 离散系统的RBF网络控制

（2） 通过Lyapunov稳定性分析，如果$k\to\infty$，$e_1(k)\to 0$，由式(9.91)：

$$v(k)=v_1(k)+v_2(k)$$

$$v_1(k)=\frac{\beta}{2\gamma c_1^2}\boldsymbol{h}^{\mathrm{T}}(\boldsymbol{x}(k-1))\boldsymbol{h}(\boldsymbol{x}(k-1))e_1(k)$$

$$v_2(k)=Ge_1(k)$$

可得$v(k)\to 0$，

再由式 (9.92)，$e_1(k)=-c_1e_1(k-1)+\beta(e(k)+c_1e(k-1)-v(k))$

很显然 $e(k)+c_1e(k\text{-}1)\to 0$，考虑到 $|c_1|<1$，可得$e(k)\to 0$。

9.11 离散系统的RBF网络控制

（3）$v(k)$ 是一个虚拟变量，由式(9.91)，令

$$v_1'(k)=\frac{\beta}{2\gamma c_1^2}\boldsymbol{h}^{\mathrm{T}}\left(\boldsymbol{x}(k-1)\right)\boldsymbol{h}\left(\boldsymbol{x}(k-1)\right)$$

```
v1_bar(k)=beta/(2*gama*c1^2)*h*h';
```

可得 $$v(k)=\left(v_1'(k)+G\right)e_1(k)$$

将$v(k)$入式(9.92)中，可得，

$$e_1(k)=-c_1e_1(k-1)+\beta\left(e(k)+c_1e(k-1)-\left(v_1'(k)+G\right)e_1(k)\right)$$

进一步整理得

```
e1(k)=(-c1*e1_1+beta*(e(k)+c1*e_1))/(1+beta*(v1_bar(k)+G));
```

$$e_1(k)=\frac{-c_1e_1(k-1)+\beta\left(e(k)+c_1e(k-1)\right)}{1+\beta\left(v_1'(k)+G\right)} \qquad (9.93)$$

9.11 离散系统的RBF网络控制

（五）仿真实例

考虑离散非线性系统为

$$y(k)=f\left(x(k-1)\right)+u(k-1)$$

其中 $f\left(x(k-1)\right)=\dfrac{0.5y(k-1)\left(1-y(k-1)\right)}{1+\exp\left(-0.25y(k-1)\right)}$ 。

假设 $f(x(k\text{-}1))$ 为已知，控制律采用式(9.78)，取c_1=-0.01，仿真结果如图9.40和图9.41所示。

$$u(k)=y_{\mathrm{d}}(k+1)-f\left(\boldsymbol{x}(k)\right)-c_1e(k)$$

9.11 离散系统的RBF网络控制

图 9.40 $f\left(x\left(k-1\right)\right)$ 已知时的位置跟踪

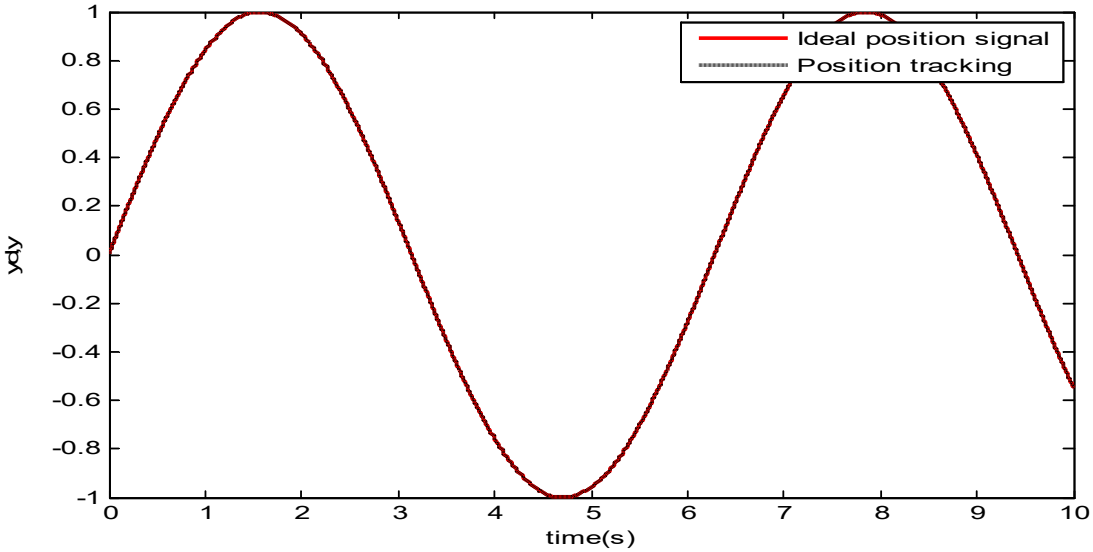
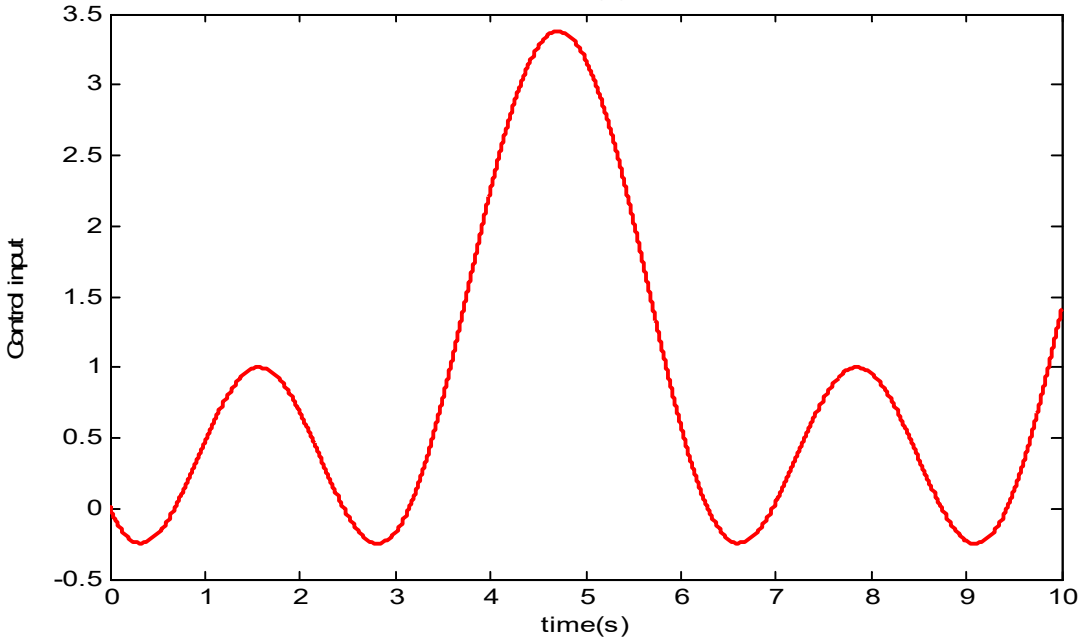


图 9.41 $f\left(x\left(k-1\right)\right)$ 已知时的控制输入



9.11 离散系统的RBF网络控制

假设 $f(x(k\text{-}1))$ 为未知，并用RBF神经网络对其进行逼近，RBF神经网络的结构为1-9-1，由$f(x(k\text{-}1))$ 的表达式可知，网络输入可取$y(k\text{-}1)$，高斯函数的参数c_i和b_i分别选为

[-2 -1.5 -1.0 -0.5 0 0.5 1.0 1.5 2] 和15(i=1，j=1,2,…,9)，神经网络的初始权值取（0，1）之间的随机数。控制对象的初始值为0，理想跟踪信号为$y_d(k)$=sint。$e_1(k)$ 由式(9.93)计算可得，控制律采用式(9.83)，自适应律采用式(9.88)，控制参数取c_1=-0.01，β=0.001，γ=0.001，ε_f=0.003，G=50000。仿真结果如图9-42至图9-44所示。

9.11 离散系统的RBF网络控制

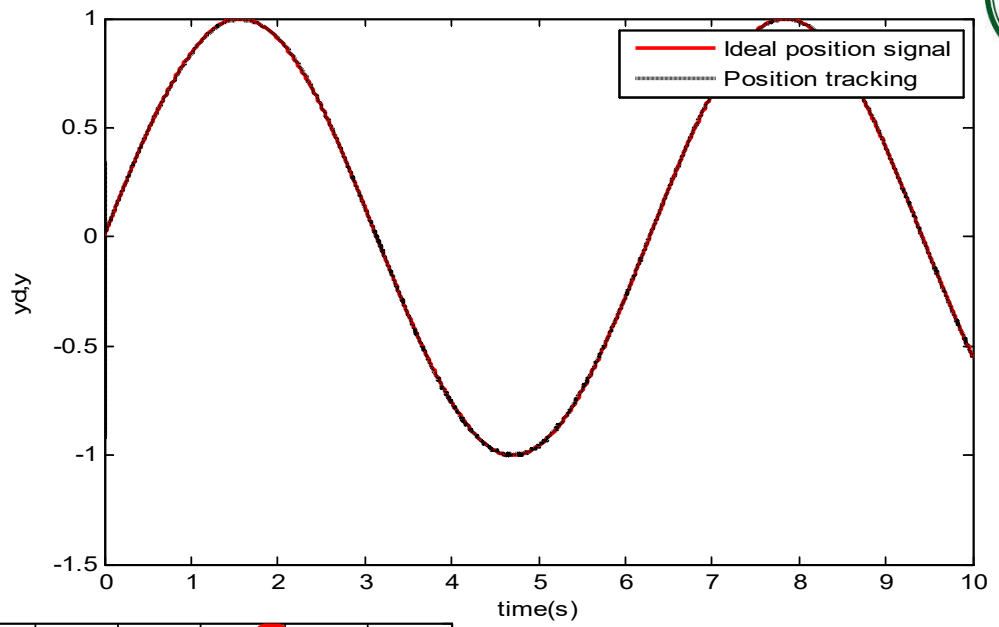


图 9.42 $f\left(x\left(k-1\right)\right)$ 未知时的位置跟踪

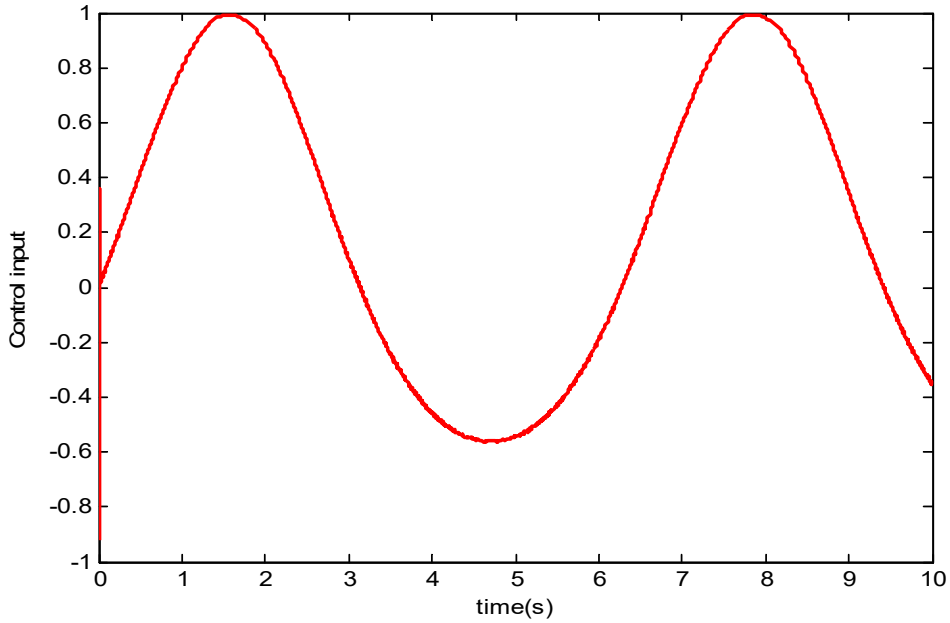


图 9.43 $f\left(x\left(k-1\right)\right)$ 未知时的控制输入

9.11 离散系统的RBF网络控制

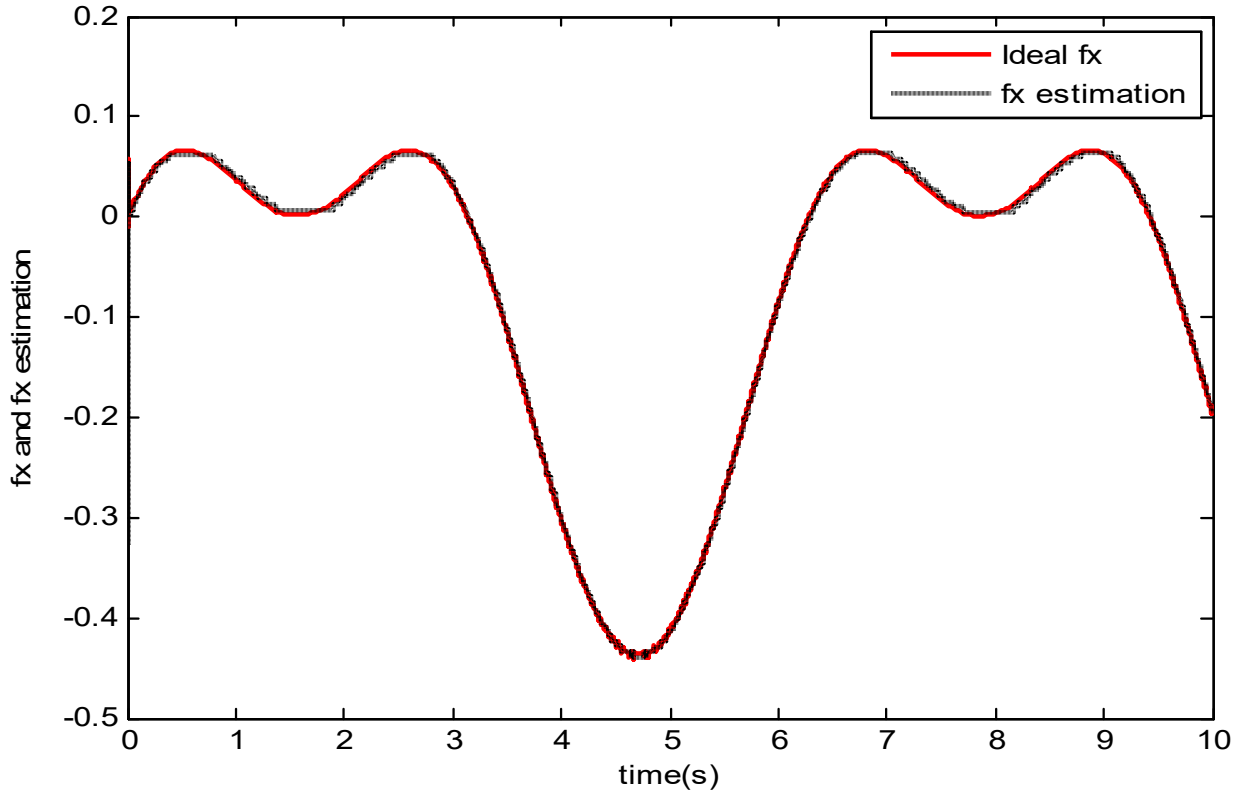


图 9.44　$f\left(x\left(k-1\right)\right)$及其逼近

思考与练习

第九章的9-1、9-2、9-3

7月20日前交！
命名规则：姓名_学号_第九章作业
ic_sysu@163.com