第 5 章

CHAPTER 5

机械手迭代学习控制

5.1 迭代学习控制的数学基础

迭代学习控制是通过迭代修正达到某种控制目标的改善，它的算法较为简单，且能在给定的时间范围内实现未知对象实际运行轨迹以高精度跟踪给定期望轨迹，且不依赖系统的精确数学模型，因而一经推出，就在控制领域得到了广泛的运用。下面介绍学习控制的稳定性和收敛性分析时用到的基本数学知识[1,2]。

5.1.1 矩阵的迹及初等性质

定义 设 $\boldsymbol{A}$ 是 n 阶方阵，则称 $\boldsymbol{A}$ 的主对角元素的和为 $\boldsymbol{A}$ 的迹，记作 $\mathrm{tr}(\boldsymbol{A})$。即若

$$\boldsymbol{A}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & a_{ii} & a_{in} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \tag{5.1}$$

则

$$\mathrm{tr}(\boldsymbol{A})=\sum_{i=1}^{n} a_{ii} \tag{5.2}$$

设 $\boldsymbol{A}$、$\boldsymbol{B}$ 都是 n 阶方阵，λ、μ 为任意复数，则矩阵的迹具有如下性质：

(1) $\mathrm{tr}(\lambda\boldsymbol{A}+\mu\boldsymbol{B})=\lambda\mathrm{tr}(\boldsymbol{A})+\mu\mathrm{tr}(\boldsymbol{B})$。

(2) $\mathrm{tr}(\boldsymbol{A})=\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}})$。

(3) 若 $\boldsymbol{A}\in\boldsymbol{C}^{m\times n}$，$\boldsymbol{B}\in\boldsymbol{C}^{n\times m}$，则 $\mathrm{tr}(\boldsymbol{AB})=\mathrm{tr}(\boldsymbol{BA})$。

5.1.2 向量范数和矩阵范数

1. 向量范数

任取 $\boldsymbol{x}\in\boldsymbol{C}^{n}$，且 $\boldsymbol{x}=(\xi_1 \quad \xi_2 \quad \cdots \quad \xi_n)^{\mathrm{T}}$，可定义

$$\|\boldsymbol{x}\|_1=\sum_{i=1}^{n}|\xi_i| \tag{5.3}$$

$$\| \boldsymbol{x} \|_2 = \left(\sum_{i=1}^{n} | \xi_i |^2 \right)^{1/2} \tag{5.4}$$

$$\| \boldsymbol{x} \|_\infty = \max_{1 \leqslant i \leqslant n} | \xi_i | \tag{5.5}$$

上述三个范数都是 $\boldsymbol{C}^n$ 中的向量范数，分别称为 1-范数，2-范数和∞-范数，这三个范数实际上都是 p -范数的特殊情形。

p -范数定义如下：

$$\| \boldsymbol{x} \|_p = \left(\sum_{i=1}^{n} | \xi_i |^p \right)^{1/p}, \quad 1 \leqslant p < +\infty \tag{5.6}$$

2. 矩阵范数

定义：若对任意矩阵 $\boldsymbol{A} \in \boldsymbol{C}^{m \times n}$，都有实数 $\| \boldsymbol{A} \|$ 与之对应，且满足下面的范数公理：

(1) 正定性：$\| \boldsymbol{A} \| \geqslant 0$，当且仅当 $\boldsymbol{A} = 0$ 时 $\| \boldsymbol{A} \| = 0$。

(2) 齐次性：对任何 $\lambda \in \boldsymbol{C}$，$\| \lambda \boldsymbol{A} \| = | \lambda | \, \| \boldsymbol{A} \|$。

(3) 三角不等式：对任何 $\boldsymbol{A}, \boldsymbol{B} \in \boldsymbol{C}^{m \times n}$，有

$$\| \boldsymbol{A} + \boldsymbol{B} \| \leqslant \| \boldsymbol{A} \| + \| \boldsymbol{B} \|$$

则称这个实数 $\| \boldsymbol{A} \|$ 为矩阵 $\boldsymbol{A}$ 的范数。

$$\| \boldsymbol{A} \|_{V_1} \overset{\text{def}}{=} \sum_{j=1}^{m} \sum_{i=1}^{n} | a_{ij} | \tag{5.7}$$

$$\| \boldsymbol{A} \|_{V_\infty} \overset{\text{def}}{=} \max_{i,j} | a_{ij} | \quad \text{（契比雪夫范数）} \tag{5.8}$$

$$\| \boldsymbol{A} \|_{V_p} \overset{\text{def}}{=} \left(\sum_{j=1}^{m} \sum_{i=1}^{n} | a_{ij} |^p \right)^{1/p}, \quad 1 \leqslant p \leqslant +\infty \tag{5.9}$$

当 $p = 2$ 时，称 $\| \boldsymbol{A} \|_{V_2} = \| \boldsymbol{A} \|_F \overset{\text{def}}{=} \left(\sum_{j=1}^{m} \sum_{i=1}^{n} | a_{ij} |^2 \right)^{1/2}$ 为 $\boldsymbol{A}$ 的 Frobenius 范数，简称 F-范数，是最常用的范数之一。它就是酉矩阵 $\boldsymbol{C}^{m \times n}$ 中的内积 $\boldsymbol{A} | \boldsymbol{B} = \mathrm{tr}(\boldsymbol{B}^{\mathrm{H}} \boldsymbol{A})$ 所诱导的范数：

$$\| \boldsymbol{A} \|_F^2 = (\boldsymbol{A} \mid \boldsymbol{A}) = \mathrm{tr}(\boldsymbol{B}^{\mathrm{H}} \boldsymbol{A}) = \sum_{j=1}^{m} \sum_{i=1}^{n} | a_{ij} |^2 \tag{5.10}$$

F-范数具有下列良好的性质：

性质 1 设 $\boldsymbol{A} \in \boldsymbol{C}^{m \times n}$，对酉矩阵 $\boldsymbol{U} \in \boldsymbol{C}^{m \times m}$，$\boldsymbol{V} \in \boldsymbol{C}^{n \times n}$，恒有

$$\| \boldsymbol{A} \|_F = \| \boldsymbol{U}\boldsymbol{A} \|_F = \| \boldsymbol{A}\boldsymbol{V} \|_F = \| \boldsymbol{U}\boldsymbol{A}\boldsymbol{V} \|_F \tag{5.11}$$

性质 2 设 $\boldsymbol{A} \in \boldsymbol{C}^{m \times n}$，$\boldsymbol{B} \in \boldsymbol{C}^{n \times l}$，则有

$$\| \boldsymbol{A} \|_F \leqslant \| \boldsymbol{A} \|_F \| \boldsymbol{B} \|_F \tag{5.12}$$

性质 3 在矩阵空间 $\boldsymbol{C}^{n \times n}$ 上的任意实函数，记为 $\| \cdot \|$，如果对所有的 $\boldsymbol{A}, \boldsymbol{B} \in \boldsymbol{C}^{n \times n}$，$\lambda \in \boldsymbol{C}$，都满足以下 4 个条件：

(1) $\| \boldsymbol{A} \| \geqslant 0$，当且仅当 $\boldsymbol{A} = \boldsymbol{0}$ 时，有 $\| \boldsymbol{A} \| = 0$。

(2) $\| \lambda \boldsymbol{A} \| = | \lambda | \, \| \boldsymbol{A} \|$。

(3) $\| \boldsymbol{A} + \boldsymbol{B} \| \leqslant \| \boldsymbol{A} \| + \| \boldsymbol{B} \|$。

(4) $\| \boldsymbol{A}\boldsymbol{B} \| \leqslant \| \boldsymbol{A} \| \, \| \boldsymbol{B} \|$。

则称 $\| \cdot \|$ 为相容的矩阵范数，或简称矩阵范数。显然，矩阵的 F-范数是一种相容的矩阵范数。

5.2　迭代学习控制方法介绍

迭代学习控制(iterative learning control,ILC)是智能控制中具有严格数学描述的一个分支。1984 年,Arimoto[1]等提出了迭代学习控制的概念,该控制方法适合于具有重复运动性质的被控对象,它不依赖于系统的精确数学模型,能以非常简单的方式处理不确定度相当高的非线性强耦合动态系统。目前,迭代学习控制在学习算法、收敛性、鲁棒性、学习速度及工程应用研究上取得了巨大的进展。

近年来,迭代学习控制理论和应用在国外得到快速发展,取得了许多成果。在国内,迭代学习控制理论也得到广泛的重视,有许多重要著作出版[2~5],发表了许多综述性论文[6~9]。

5.2.1　迭代学习控制基本原理

设被控对象的动态过程为

$$\dot{\boldsymbol{x}}(t)=f(\boldsymbol{x}(t),\boldsymbol{u}(t),t),\quad \boldsymbol{y}(t)=g(\boldsymbol{x}(t),\boldsymbol{u}(t),t) \tag{5.13}$$

其中,$\boldsymbol{x}\in\mathbf{R}^n$、$\boldsymbol{y}\in\mathbf{R}^m$、$u\in\mathbf{R}^r$ 分别为系统的状态、输出和输入变量,$f(\cdot)$、$g(\cdot)$ 为适当维数的向量函数,其结构与参数均未知。若期望控制 $\boldsymbol{u}_{\mathrm{d}}(t)$存在,则迭代学习控制的目标为:给定期望输出 $\boldsymbol{y}_{\mathrm{d}}(t)$ 和每次运行的初始状态 $\boldsymbol{x}_k(0)$,要求在给定的时间 $t\in[0,T]$内,按照一定的学习控制算法通过多次重复的运行,使控制输入 $\boldsymbol{u}_k(t)\to\boldsymbol{u}_{\mathrm{d}}(t)$,而系统输出 $\boldsymbol{y}_k(t)\to\boldsymbol{y}_{\mathrm{d}}(t)$。第 k 次运行时,式(5.13) 表示为

$$\dot{\boldsymbol{x}}_k(t)=f(\boldsymbol{x}_k(t),\boldsymbol{u}_k(t),t),\quad \boldsymbol{y}_k(t)=g(\boldsymbol{x}_k(t),\boldsymbol{u}_k(t),t) \tag{5.14}$$

跟踪误差为

$$\boldsymbol{e}_k(t)=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{y}_k(t) \tag{5.15}$$

迭代学习控制可分为以下开环学习和闭环学习两种方法:

(1) 开环学习控制的方法是:第 $k+1$ 次的控制等于第 k 次控制再加上第 k 次输出误差的校正项,即

$$\boldsymbol{u}_{k+1}(t)=\mathrm{L}(\boldsymbol{u}_k(t),\boldsymbol{e}_k(t)) \tag{5.16}$$

(2) 闭环学习控制的方法是:取第 $k+1$ 次运行的误差作为学习的修正项,即

$$\boldsymbol{u}_{k+1}(t)=\mathrm{L}(\boldsymbol{u}_k(t),\boldsymbol{e}_{k+1}(t)) \tag{5.17}$$

其中,L 为线性或非线性算子。

5.2.2　基本的迭代学习控制算法

Arimoto 等首先给出了线性时变连续系统的 D 型迭代学习控制律[1]

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{\Gamma}\dot{\boldsymbol{e}}_k(t) \tag{5.18}$$

其中,$\boldsymbol{\Gamma}$ 为常数增益矩阵。在 D 型算法的基础上,相继出现了 P 型、PI 型、PD 型迭代学习控制律。从一般意义来看它们都是 PID 型迭代学习控制律的特殊形式,PID 迭代学习控制律表示为

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{\Gamma}\dot{\boldsymbol{e}}_k(t)+\boldsymbol{\Phi}\boldsymbol{e}_k(t)+\boldsymbol{\Psi}\int_0^t\boldsymbol{e}_k(\tau)\mathrm{d}\tau \tag{5.19}$$

其中，$\boldsymbol{\Gamma}$、$\boldsymbol{\Phi}$、$\boldsymbol{\Psi}$为学习增益矩阵。算法中的误差信息使用$\boldsymbol{e}_k(t)$称为开环迭代学习控制，如果使用$\boldsymbol{e}_{k+1}(t)$则称为闭环迭代学习控制，如果同时使用$\boldsymbol{e}_k(t)$和$\boldsymbol{e}_{k+1}(t)$则称为开闭环迭代学习控制。

此外，还有高阶迭代学习控制算法、最优迭代学习控制算法、遗忘因子迭代学习控制算法和反馈-前馈迭代学习控制算法等。

5.2.3 迭代学习控制主要分析方法

学习算法的收敛性分析是迭代学习控制的核心问题，这方面的研究成果很丰富。

1. 基本的收敛性分析方法

对于如下线性离散系统：

$$\begin{cases}\boldsymbol{x}(t+1)=\boldsymbol{A}\boldsymbol{x}(t)+\boldsymbol{B}\boldsymbol{u}(t)\\ \boldsymbol{y}(t)=\boldsymbol{C}\boldsymbol{x}(t)\end{cases} \tag{5.20}$$

迭代学习控制算法为

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{\Gamma}\boldsymbol{e}_k(t+1) \tag{5.21}$$

针对学习算法式(5.21)的收敛性，有以下两种分析方法：

(1) 压缩映射方法：即系统要求满足全局 Lipschitz 条件和相同的初始条件，如果$\|\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma}\|<1$，则有

$$\|\boldsymbol{e}_{k+1}\|=\|(\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma})\boldsymbol{e}_k\|<\|\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma}\|\,\|\boldsymbol{e}_k\|<\|\boldsymbol{e}_k\| \tag{5.22}$$

此时算法是单调收敛的。该方法依赖于范数的选择，常用的有l_1-范数、l_2-范数、l_∞-范数及λ-范数。在收敛性证明过程中常用到 Bellman-Gronwall 引理。

(2) 谱半径条件法：如果谱半径ρ满足$\rho(\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma})\leqslant\rho<1$，则有

$$\lim_{k\to\infty}\|\boldsymbol{e}_k\|=\lim_{k\to\infty}\|(\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma})\boldsymbol{e}_{k-1}\|=\lim_{k\to\infty}\rho(\boldsymbol{I}-\boldsymbol{C}\boldsymbol{B}\boldsymbol{\Gamma})^k\|\boldsymbol{e}_0\| \tag{5.23}$$

即$\lim_{k\to\infty}\|\boldsymbol{e}_k\|=0$。

2. 基于 2-D 理论的分析方法

迭代学习控制系统的学习是按两个相互独立的方向进行：时间轴方向和迭代次数轴方向，因此迭代学习过程本质上是二维系统，可利用成熟的 2-D 系统理论系统地研究和分析时间域的稳定性和迭代次数域的收敛性问题。2-D 系统的稳定性理论为迭代学习控制的收敛性证明提供了一种非常有效的方法，2-D 系统理论中的 Roesser 模型成为迭代学习控制中最基本的分析模型。

3. 基于 Lyapunov 直接法的设计方法

Lyapunov 直接法已广泛用于非线性动态系统的控制器设计和分析中，在研究非线性不确定系统时，该方法是最重要的应用工具之一。受 Lyapunov 直接法的启发，在时间域和迭代域能量函数的概念得到研究，它为学习控制在迭代域设计和收敛性分析方面提供了一种新的研究方法。

在迭代域能量函数的迭代学习控制方法基础上，发展了鲁棒和自适应迭代学习控制，可解决具有参数或非参数不确定性非线性系统控制器的设计。近年来反映时间域和迭代域系统能量的组合能量函数方法也应用于迭代学习控制，它可保证在迭代域跟踪误差的渐进收敛以及在时间域具有有界和逐点跟踪的动态特性，并且控制输入在整个迭代区间内是范数

收敛的，适用于一类不具有全局 Lipschitz 条件的非线性系统。通过能量函数的方法，许多新的控制方法，如反演设计和非线性优化方法都作为系统设计工具应用到迭代学习控制中。此外，还有最优化分析方法、频域分析法等分析方法。

5.2.4　迭代学习控制的关键技术

1. 学习算法的稳定性和收敛性

稳定性与收敛性是研究当学习律与被控系统满足什么条件时，迭代学习控制过程才是稳定收敛的。算法的稳定性保证了随着学习次数的增加，控制系统不发散，但是对于学习控制系统而言，仅仅稳定是没有实际意义的，只有使学习过程收敛到真值，才能保证得到的控制为某种意义下最优的控制。收敛是对学习控制的最基本要求，多数学者在提出新的学习律的同时，基于被控对象的一些假设，给出了收敛的条件。例如，Arimoto 在最初提出 PID 型学习控制律时，仅针对线性系统在 D 型学习律下的稳定性和收敛条件作了证明。

2. 初始值问题

运用迭代学习控制技术设计控制器时，只需要通过重复操作获得的受控对象的误差或误差导数信号。在这种控制技术中，迭代学习总要从某初始点开始，初始点指初始状态或初始输出。几乎所有的收敛性证明都要求初始条件是相同的，解决迭代学习控制理论中的初始条件问题一直是人们追求的目标之一。目前已提出的迭代学习控制算法大多数要求被控系统每次运行时的初始状态在期望轨迹对应的初始状态上，即满足初始条件：

$$x_k(0)=x_d(0),\quad k=0,1,2,\cdots \tag{5.24}$$

当系统的初始状态不在期望轨迹上，而在期望轨迹的某一很小的邻域内时，通常把这类问题归结为学习控制的鲁棒性问题研究。

3. 学习速度问题

在迭代学习算法研究中，其收敛条件基本上都是在学习次数 $k\to\infty$ 下给出的。而在实际应用场合，学习次数 $k\to\infty$ 显然是没有任何实际意义的。因此，如何使迭代学习过程更快地收敛于期望值是迭代学习控制研究中的另一个重要问题。

ILC 本质上是一种前馈控制技术，大部分学习律尽管证明了学习收敛的充分条件，但收敛速度还是很慢。可利用多次学习过程中得到的知识来改进后续学习过程的速度，例如，采用高阶迭代控制算法、带遗忘因子的学习律、利用当前项或反馈配置等方法构造学习律，可使收敛速度大大加快。

4. 鲁棒性问题

迭代学习控制理论的提出有浓厚的工程背景，因此仅仅在无干扰条件下讨论收敛性问题是不够的，还应讨论存在各种干扰的情形下系统的跟踪性能。一个实际运行的迭代学习控制系统除了存在初始偏移外，还或多或少存在状态扰动、测量噪声、输入扰动等各种干扰。鲁棒性问题讨论存在各种干扰时迭代学习控制系统的跟踪性能。具体地说，一个迭代学习控制系统是鲁棒的，指系统在各种有界干扰的影响下，其迭代轨迹能收敛到期望轨迹的邻域内，而当这些干扰消除时，迭代轨迹会收敛到期望轨迹。

5.3 机械手轨迹跟踪迭代学习控制仿真实例

5.3.1 控制器的设计

考虑一个 n 关节的机械手，其动态性能可以由以下二阶非线性微分方程描述：

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}+\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}+\boldsymbol{G}(\boldsymbol{q})=\boldsymbol{u}-\boldsymbol{\tau}_{\mathrm{d}} \tag{5.25}$$

其中，$\boldsymbol{q}\in\mathbf{R}^n$ 为关节角位移量，$\boldsymbol{M}(\boldsymbol{q})\in\mathbf{R}^{n\times n}$ 为机械手的惯性矩阵，$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\in\mathbf{R}^n$ 表示离心力和哥氏力，$\boldsymbol{G}(\boldsymbol{q})\in\mathbf{R}^n$ 为重力项，$\boldsymbol{\tau}\in\mathbf{R}^n$ 为控制力矩，$\boldsymbol{\tau}_{\mathrm{d}}\in\mathbf{R}^n$ 为各种误差和扰动。

设系统所要跟踪的期望轨迹为 $\boldsymbol{y}_{\mathrm{d}}(t)$，$t\in[0,T]$。系统第 i 次输出为 $\boldsymbol{y}_i(t)$，令 $\boldsymbol{e}_i(t)=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{y}_i(t)$。

在学习开始时，系统的初始状态为 $\boldsymbol{x}_0(0)$。学习控制的任务为通过学习控制律设计 $\boldsymbol{u}_{i+1}(t)$，使第 $i+1$ 次运动误差 $\boldsymbol{e}_{i+1}(t)$减少。采用以下三种基于反馈的迭代学习控制律：

(1) 闭环 D 型：

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{K}_{\mathrm{d}}(\dot{\boldsymbol{q}}_{\mathrm{d}}(t)-\dot{\boldsymbol{q}}_{k+1}(t)) \tag{5.26}$$

(2) 闭环 PD 型：

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{K}_{\mathrm{p}}(\boldsymbol{q}_{\mathrm{d}}(t)-\boldsymbol{q}_{k+1}(t))+\boldsymbol{K}_{\mathrm{d}}(\dot{\boldsymbol{q}}_{\mathrm{d}}(t)-\dot{\boldsymbol{q}}_{k+1}(t)) \tag{5.27}$$

(3) 指数变增益 D 型：

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\boldsymbol{K}_{\mathrm{p}}(\boldsymbol{q}_{\mathrm{d}}(t)-\boldsymbol{q}_{k+1}(t))+\boldsymbol{K}_{\mathrm{d}}(\dot{\boldsymbol{q}}_{\mathrm{d}}(t)-\dot{\boldsymbol{q}}_{k+1}(t)) \tag{5.28}$$

5.3.2 仿真实例

本节针对二关节机械手，介绍一种机械手 PD 型反馈迭代学习控制的仿真设计方法。针对二关节机械手控制系统式(5.25)，各项表示为：

$$\boldsymbol{M}=[m_{ij}]_{2\times2}$$

其中，$m_{11}=m_1l_{c1}^2+m_2(l_1^2+l_{c2}^2+2l_1l_{c2}\cos q_2)+I_1+I_2$，$m_{12}=m_{21}=m_2(l_{c2}^2+l_1l_{c2}\cos q_2)+l_2$，$m_{22}=m_2l_{c2}^2+I_2$，

$$\boldsymbol{C}=[c_{ij}]_{2\times2}$$

其中，$c_{11}=h\dot{q}_2$，$c_{12}=h\dot{q}_1+h\dot{q}_2$，$c_{21}=-h\dot{q}_1$，$c_{22}=0$，$h=-m_2l_1l_{c2}\sin q_2$

$$\boldsymbol{G}=[G_1\quad G_2]^{\mathrm{T}}$$

其中，$G_1=(m_1l_{c1}+m_2l_1)g\cos q_1+m_2l_{c2}g\cos(q_1+q_2)$，$G_2=m_2l_{c2}g\cos(q_1+q_2)$，干扰项为 $\tau_{\mathrm{d}}=[0.3\sin t\quad 0.1(1-\mathrm{e}^{-t})]^{\mathrm{T}}$。

机械手系统参数为 $m_1=10$，$m_2=5$，$l_1=1$，$l_2=0.5$，$l_{c1}=0.5$，$l_{c2}=0.25$，$I_1=0.83$，$I_2=0.3$，$g=9.8$。

根据式(5.26)～式(5.28)，采用三种闭环迭代学习控制律，其中，$M=1$ 为 D 型迭代学习控制，$M=2$ 为 PD 型迭代学习控制，$M=3$ 为指数变增益 D 型迭代学习控制。

两个关节的角度指令分别为 $\sin(3t)$和 $\cos(3t)$，为了保证被控对象初始输出与指令初值一致，取被控对象的初始状态为 $\boldsymbol{x}(0)=[0\quad 3\quad 1\quad 0]^{\mathrm{T}}$。取 PD 型迭代学习控制，即 $M=2$，仿真结果如图 5-1～图 5-3 所示。

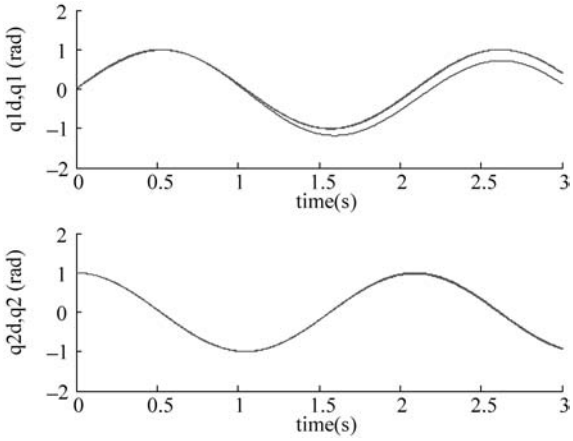


图 5-1　20 次迭代学习的角度跟踪过程

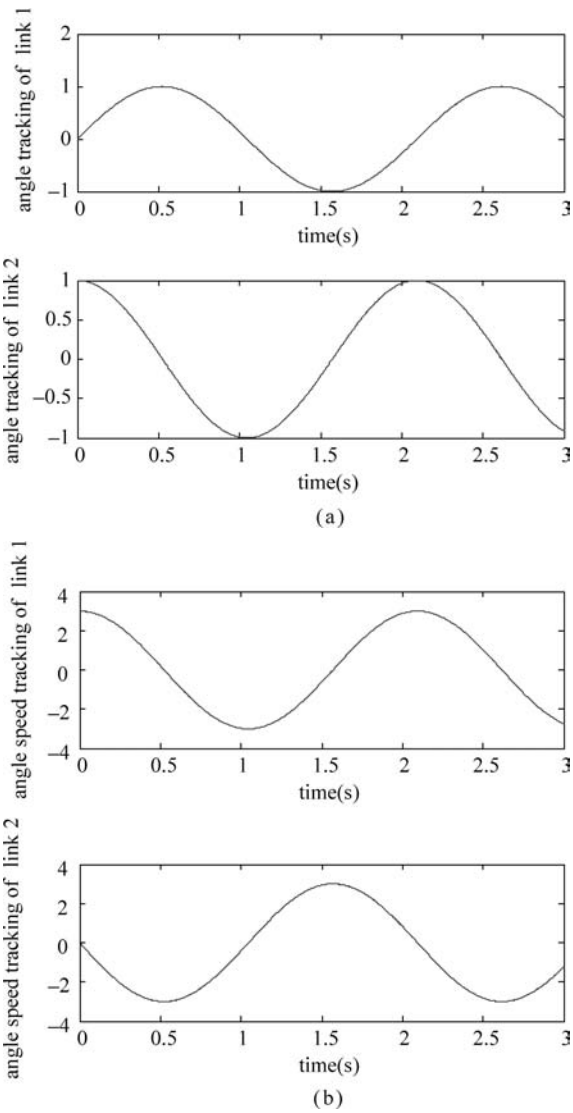


图 5-2　第 20 次迭代学习的角度跟踪

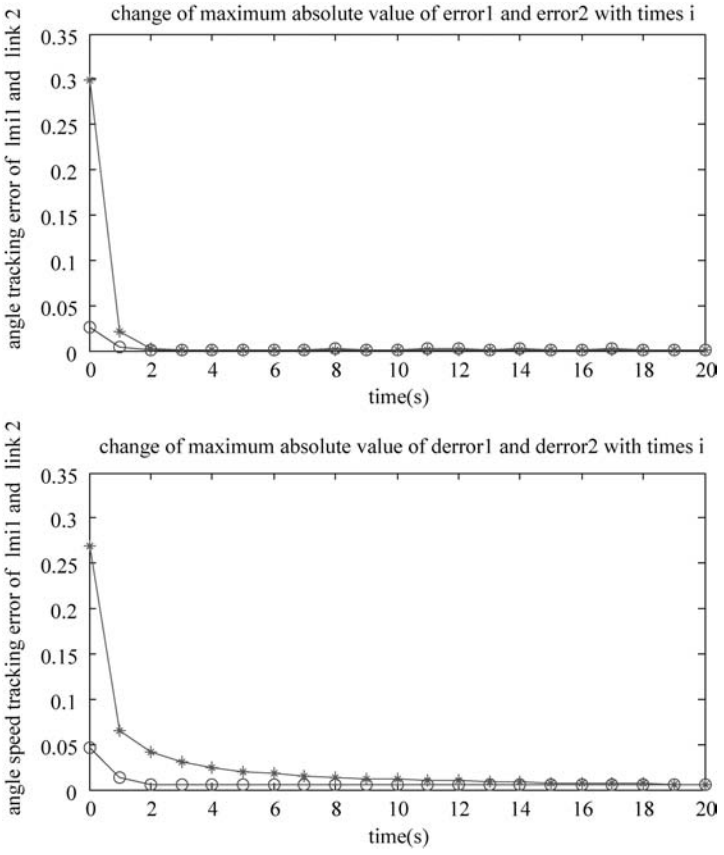


图 5-3　第 20 次迭代学习角度和角速度跟踪误差收敛过程

仿真程序如下：

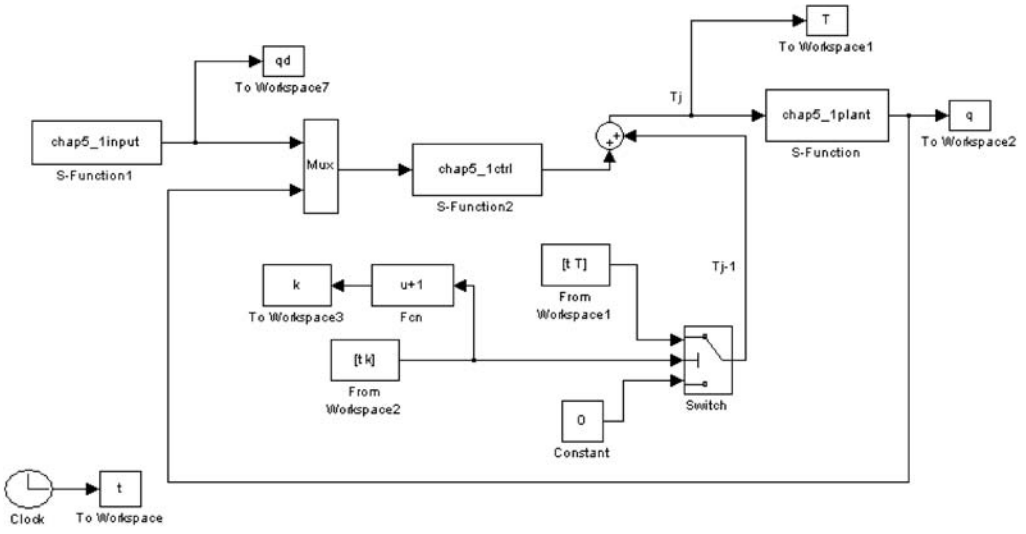
(1) 主程序：chap5_1main.m。

```
clear all;
close all;

t = [0:0.01:3]';
k(1:301) = 0;                % Total initial points
k = k';
T1(1:301) = 0;
T1 = T1';
T2 = T1;
T = [T1 T2];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
M = 20;
for i = 0:1:M                % Start Learning Control
i
pause(0.01);

sim('chap5_1sim',[0,3]);
```

```
q1 = q(:,1);
dq1 = q(:,2);
q2 = q(:,3);
dq2 = q(:,4);

q1d = qd(:,1);
dq1d = qd(:,2);
q2d = qd(:,3);
dq2d = qd(:,4);

e1 = q1d - q1;
e2 = q2d - q2;
de1 = dq1d - dq1;
de2 = dq2d - dq2;

figure(1);
subplot(211);
hold on;
plot(t,q1,'b',t,q1d,'r');
xlabel('time(s)');ylabel('q1d,q1 (rad)');

subplot(212);
hold on;
plot(t,q2,'b',t,q2d,'r');
xlabel('time(s)');ylabel('q2d,q2 (rad)');

j = i + 1;
times(j) = i;
e1i(j) = max(abs(e1));
e2i(j) = max(abs(e2));
de1i(j) = max(abs(de1));
de2i(j) = max(abs(de2));
end                          % End of i
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2);
subplot(211);
plot(t,q1d,'r',t,q1,'b');
xlabel('time(s)');ylabel('angle tracking of link 1');
subplot(212);
plot(t,q2d,'r',t,q2,'b');
xlabel('time(s)');ylabel('angle tracking of link 2');

figure(3);
subplot(211);
plot(t,dq1d,'r',t,dq1,'b');
xlabel('time(s)');ylabel('angle speed tracking of link 1');
subplot(212);
plot(t,dq2d,'r',t,dq2,'b');
xlabel('time(s)');ylabel('angle speed tracking of link 2');

figure(4);
```

```
subplot(211);
plot(times,e1i,'* - r',times,e2i,'o - b');
title('change of maximum absolute value of error1 and error2 with times i');
xlabel('time(s)');ylabel('angle tracking error of lmi1 and link 2');
subplot(212);
plot(times,de1i,'* - r',times,de2i,'o - b');
title('change of maximum absolute value of derror1 and derror2 with times i');
xlabel('time(s)');ylabel('angle speed tracking error of lmi1 and link 2');
```

(2) Simulink 子程序：chap5_1sim. mdl。



(3) 被控对象子程序：chap5_1plant. m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
switch flag,
case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
case 1,
    sys = mdlDerivatives(t,x,u);
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 4;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 4;
sizes.NumInputs      = 2;
sizes.DirFeedthrough = 1;
```

```
sizes.NumSampleTimes  = 1;
sys = simsizes(sizes);
x0  = [0;3;1;0];          % Must be equal to x(0) of ideal input
str = [];
ts  = [0 0];
function sys = mdlDerivatives(t,x,u)
Tol = [u(1) u(2)]';

g = 9.8;
m1 = 10;m2 = 5;
l1 = 1;l2 = 0.5;
lc1 = 0.5;lc2 = 0.25;
I1 = 0.83;I2 = 0.3;

d11 = m1 * lc1^2 + m2 * (l1^2 + lc2^2 + 2 * l1 * lc2 * cos(x(3))) + I1 + I2;
d12 = m2 * (lc2^2 + l1 * lc2 * cos(x(3))) + I2;
d21 = d12;
d22 = m2 * lc2^2 + I2;
D = [d11 d12;d21 d22];
h = - m2 * l1 * lc2 * sin(x(3));
c11 = h * x(4);
c12 = h * x(4) + h * x(2);
c21 = - h * x(2);
c22 = 0;
C = [c11 c12;c21 c22];
g1 = (m1 * lc1 + m2 * l1) * g * cos(x(1)) + m2 * lc2 * g * cos(x(1) + x(3));
g2 = m2 * lc2 * g * cos(x(1) + x(3));
G = [g1;g2];

a = 1.0;
d1 = a * 0.3 * sin(t);
d2 = a * 0.1 * (1 - exp( - t));
Td = [d1;d2];

S = - inv(D) * C * [x(2);x(4)] - inv(D) * G + inv(D) * (Tol - Td);

sys(1) = x(2);
sys(2) = S(1);
sys(3) = x(4);
sys(4) = S(2);
function sys = mdlOutputs(t,x,u)
sys(1) = x(1);              % Angle1:q1
sys(2) = x(2);              % Angle1 speed:dq1
sys(3) = x(3);              % Angle2:q2
sys(4) = x(4);              % Angle2 speed:dq2
```

(4) 控制器子程序：chap5_1ctrl.m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
switch flag,
case 0,
```

```
    [sys,x0,str,ts] = mdlInitializeSizes;
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 2;
sizes.NumInputs      = 8;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];
function sys = mdlOutputs(t,x,u)
q1d = u(1);dq1d = u(2);
q2d = u(3);dq2d = u(4);

q1 = u(5);dq1 = u(6);
q2 = u(7);dq2 = u(8);

e1 = q1d - q1;
e2 = q2d - q2;
e = [e1 e2]';
de1 = dq1d - dq1;
de2 = dq2d - dq2;
de = [de1 de2]';

Kp = [100 0;0 100];
Kd = [500 0;0 500];

M = 2;
if M == 1
    Tol = Kd * de;                      % D Type
elseif M == 2
    Tol = Kp * e + Kd * de;             % PD Type
elseif M == 3
    Tol = Kd * exp(0.8 * t) * de;       % Exponential Gain D Type
end
sys(1) = Tol(1);
sys(2) = Tol(2);
```

(5) 指令程序：chap5_1input.m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
```

```
switch flag,
case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 4;
sizes.NumInputs      = 0;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];
function sys = mdlOutputs(t,x,u)
q1d = sin(3 * t);
dq1d = 3 * cos(3 * t);
q2d = cos(3 * t);
dq2d = -3 * sin(3 * t);

sys(1) = q1d;
sys(2) = dq1d;
sys(3) = q2d;
sys(4) = dq2d;
```

5.4　线性时变连续系统迭代学习控制

5.4.1　系统描述

Arimoto[1]等给出了线性时变连续系统为

$$\begin{cases}\dot{\boldsymbol{x}}(t)=\boldsymbol{A}(t)\boldsymbol{x}(t)+\boldsymbol{B}(t)\boldsymbol{u}(t)\\ \boldsymbol{y}(t)=\boldsymbol{C}(t)\boldsymbol{x}(t)\end{cases} \tag{5.29}$$

其开环 PID 型迭代学习控制律为

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\left(\boldsymbol{\Gamma}\frac{\mathrm{d}}{\mathrm{d}t}+\boldsymbol{L}+\boldsymbol{\Psi}\int\mathrm{d}t\right)\boldsymbol{e}_k(t) \tag{5.30}$$

其中，$\boldsymbol{\Gamma}$，$\boldsymbol{L}$，$\boldsymbol{\Psi}$ 为学习增益矩阵。

5.4.2　控制器设计及收敛性分析

定理 5.1[4]　若由式(5.29)和式(5.30)描述的系统满足如下条件：

(1) $\| \boldsymbol{I}-\boldsymbol{C}(t)\boldsymbol{B}(t)\boldsymbol{\Gamma}(t) \| \leqslant \bar{\rho}<1$。

(2) 每次迭代初始条件一致，即 $\boldsymbol{x}_k(0)=\boldsymbol{x}_0(k=1,2,3,\cdots)$，$\boldsymbol{y}_0(0)=\boldsymbol{y}_d(0)$，则当 $k\to\infty$ 时，有 $\boldsymbol{y}_k(t)\to\boldsymbol{y}_d(t)$，$\forall t\in[0,T]$。

下面给出该定理的简单分析，可参考文献[4]的证明过程。

由式(5.29)及条件式(5.30)得

$$\boldsymbol{y}_{k+1}(0)=\boldsymbol{C}\boldsymbol{x}_{k+1}(0)=\boldsymbol{C}\boldsymbol{x}_k(0)=\boldsymbol{y}_k(0)$$

则 $\boldsymbol{e}_k(0)=0(k=0,1,2,\cdots)$，即系统满足初始条件。

非齐次一阶线性微分方程 $\dot{\boldsymbol{x}}(t)=\boldsymbol{A}(t)\boldsymbol{x}(t)+\boldsymbol{B}(t)\boldsymbol{u}(t)$ 的解为

$$\begin{aligned}\boldsymbol{x}(t)&=\boldsymbol{C}\exp\left(\int_0^t\boldsymbol{A}\,\mathrm{d}\tau\right)+\exp\left(\int_0^t\boldsymbol{A}\,\mathrm{d}\tau\right)\int_0^t\boldsymbol{B}(\tau)\boldsymbol{u}(\tau)\exp\left(\int_0^\tau-\boldsymbol{A}\,\mathrm{d}\delta\right)\mathrm{d}\tau\\&=\boldsymbol{C}\exp(\boldsymbol{A}t)+\exp(\boldsymbol{A}t)\int_0^t\boldsymbol{B}(\tau)\boldsymbol{u}(\tau)\exp(-\boldsymbol{A}\tau)\mathrm{d}\tau\\&=\boldsymbol{C}\exp(\boldsymbol{A}t)+\int_0^t\exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}(\tau)\boldsymbol{u}(\tau)\mathrm{d}\tau\end{aligned}$$

取 $\boldsymbol{\Phi}(t,\tau)=\exp(\boldsymbol{A}(t-\tau))$，则

$$\boldsymbol{x}_k(t)-\boldsymbol{x}_{k+1}(t)=\int_0^t\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)(\boldsymbol{u}_k(\tau)-\boldsymbol{u}_{k+1}(\tau))\mathrm{d}\tau$$

由于 $\boldsymbol{e}_k(t)=\boldsymbol{y}_d(t)-\boldsymbol{y}_k(t)$，$\boldsymbol{e}_{k+1}(t)=\boldsymbol{y}_d(t)-\boldsymbol{y}_{k+1}(t)$，则

$$\begin{aligned}\boldsymbol{e}_{k+1}(t)-\boldsymbol{e}_k(t)&=\boldsymbol{y}_k(t)-\boldsymbol{y}_{k+1}(t)=\boldsymbol{C}(t)(\boldsymbol{x}_k(t)-\boldsymbol{x}_{k+1}(t))\\&=\int_0^t\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)(\boldsymbol{u}_k(\tau)-\boldsymbol{u}_{k+1}(\tau))\mathrm{d}\tau\end{aligned}$$

即

$$\boldsymbol{e}_{k+1}(t)=\boldsymbol{e}_k(t)-\int_0^t\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)(\boldsymbol{u}_{k+1}(\tau)-\boldsymbol{u}_k(\tau))\mathrm{d}\tau$$

将 PID 型控制律式(5.30)代入上式，则第 $k+1$ 次输出的误差为：

$$\boldsymbol{e}_{k+1}(t)=\boldsymbol{e}_k(t)-\int_0^t\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)[\boldsymbol{\Gamma}(\tau)\dot{\boldsymbol{e}}_k(\tau)+\boldsymbol{L}(\tau)\boldsymbol{e}_k(\tau)+\boldsymbol{\Psi}(\tau)\int_0^\tau\boldsymbol{e}_k(\delta)\mathrm{d}\delta]\mathrm{d}\tau \tag{5.31}$$

利用分部积分公式，令 $\boldsymbol{G}(t,\tau)=\boldsymbol{C}(t)\boldsymbol{B}(\tau)\boldsymbol{\Gamma}(\tau)$，有

$$\begin{aligned}\int_0^t\boldsymbol{C}(t)\boldsymbol{B}(\tau)\boldsymbol{\Gamma}(\tau)\dot{\boldsymbol{e}}_k(\tau)\mathrm{d}\tau&=\boldsymbol{G}(t,\tau)\boldsymbol{e}_k(\tau)\Big|_0^t-\int_0^t\frac{\partial}{\partial\tau}G(t,\tau)\boldsymbol{e}_k(\tau)\mathrm{d}\tau\\&=\boldsymbol{C}(t)\boldsymbol{B}(\tau)\boldsymbol{\Gamma}(\tau)\boldsymbol{e}_k(\tau)-\int_0^t\frac{\partial}{\partial\tau}\boldsymbol{G}(t,\tau)\boldsymbol{e}_k(\tau)\mathrm{d}\tau\end{aligned} \tag{5.32}$$

将式(5.32)代入式(5.31)，得

$$\begin{aligned}\boldsymbol{e}_{k+1}(t)=&[\boldsymbol{I}-\boldsymbol{C}(t)\boldsymbol{B}(t)\boldsymbol{\Gamma}(t)]\boldsymbol{e}_k(t)+\int_0^t\frac{\partial}{\partial\tau}\boldsymbol{G}(t,\tau)\boldsymbol{e}_k(\tau)\mathrm{d}\tau-\\&\int_0^t\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{L}(\tau)\boldsymbol{e}_k(\tau)\mathrm{d}\tau-\\&\int_0^t\int_0^\tau\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{\psi}(\tau)\boldsymbol{e}_k(\sigma)\mathrm{d}\sigma\mathrm{d}\tau\end{aligned} \tag{5.33}$$

将式(5.33)两端取范数，有

$$\begin{aligned}\|\boldsymbol{e}_{k+1}(t)\| &\leqslant \|\boldsymbol{I}-\boldsymbol{C}(t)\boldsymbol{B}(t)\boldsymbol{\Gamma}(t)\|\,\|\boldsymbol{e}_k(t)\|+\int_0^t\left\|\frac{\partial}{\partial\tau}\boldsymbol{G}(t,\tau)\right\|\|\boldsymbol{e}_k(\tau)\|\mathrm{d}\tau+\\ &\int_0^t\|\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{L}(\tau)\|\,\|\boldsymbol{e}_k(\tau)\|\mathrm{d}\tau+\\ &\int_0^t\int_0^\tau\|\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{\psi}(\tau)\|\,\|\boldsymbol{e}_k(\sigma)\|\mathrm{d}\sigma\mathrm{d}\tau\\ &\leqslant \|\boldsymbol{I}-\boldsymbol{C}(t)\boldsymbol{B}(t)\boldsymbol{\Gamma}(t)\|\,\|\boldsymbol{e}_k(t)\|+\int_0^t b_1\|\boldsymbol{e}_k(\tau)\|\mathrm{d}\tau+\\ &\int_0^t\int_0^\tau b_2\|\boldsymbol{e}_k(\sigma)\|\mathrm{d}\sigma\mathrm{d}\tau\end{aligned}\tag{5.34}$$

其中，

$$b_1=\max\left\{\sup_{t,\tau\in[0,T]}\left\|\frac{\partial}{\partial\tau}\boldsymbol{G}(t,\tau)\right\|,\sup_{t,\tau\in[0,T]}\|\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{L}(\tau)\|\right\}$$

$$b_2=\sup_{t,\tau\in[0,T]}\|\boldsymbol{C}(t)\boldsymbol{\Phi}(t,\tau)\boldsymbol{B}(\tau)\boldsymbol{\psi}(\tau)\|$$

将式(5.34)两端同乘以 $\exp(-\lambda t)$，$\lambda>0$，并考虑 $\int_0^t\exp(\lambda\tau)\mathrm{d}\tau=\frac{\exp(\lambda t)-1}{\lambda}$，有

$$\begin{aligned}\exp(-\lambda t)\int_0^t b_1\|\boldsymbol{e}_k(\tau)\|\mathrm{d}\tau&=\exp(-\lambda t)\int_0^t b_1\|\boldsymbol{e}_k(\tau)\|\exp(-\lambda\tau)\exp(\lambda\tau)\mathrm{d}\tau\\ &\leqslant b_1\exp(-\lambda t)\|\boldsymbol{e}_k(\tau)\|_\lambda\int_0^t\exp(\lambda\tau)\mathrm{d}\tau\\ &=b_1\exp(-\lambda t)\|\boldsymbol{e}_k(\tau)\|_\lambda\frac{\exp(\lambda t)-1}{\lambda}\\ &=\frac{b_1}{\lambda}\|\boldsymbol{e}_k(\tau)\|_\lambda\exp(-\lambda t)(\exp(\lambda t)-1)\\ &=b_1\frac{(1-\exp(-\lambda t))}{\lambda}\|\boldsymbol{e}_k(\tau)\|_\lambda\\ &\leqslant b_1\frac{(1-\exp(-\lambda T))}{\lambda}\|\boldsymbol{e}_k(\tau)\|_\lambda\end{aligned}$$

根据范数的性质[4]，有

$$\exp(-\lambda t)\int_0^t\int_0^\tau b_2\|\boldsymbol{e}_k(\sigma)\|\mathrm{d}\sigma\mathrm{d}\tau\leqslant b_2\left(\frac{1-\exp(-\lambda T)}{\lambda}\right)^2\|\boldsymbol{e}_k(\tau)\|_\lambda$$

则

$$\|\boldsymbol{e}_{k+1}\|_\lambda\leqslant\bar{\rho}\|\boldsymbol{e}_k\|_\lambda\tag{5.35}$$

其中，$\bar{\rho}=\bar{\rho}+b_1\frac{1-\exp(-\lambda T)}{\lambda}+b_2\left(\frac{1-\exp(-\lambda T)}{\lambda}\right)^2$。由于 $\bar{\rho}<1$，则当 λ 取足够大时，可使 $\bar{\rho}<1$。因此 $\lim\limits_{k\to\infty}\|\boldsymbol{e}_k\|_\lambda=0$。

如果将式(5.30)中的 $\boldsymbol{e}(k)$ 改为 $\boldsymbol{e}(k+1)$，则为闭环 PID 型迭代学习控制律。同定理 5.1 的分析过程，可证明闭环 PID 迭代学习控制律。

5.4.3　仿真实例

考虑二输入、二输出线性系统：

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix}=\begin{bmatrix}-2 & 3\\ 1 & 1\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}+\begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}u_1(t)\\ u_2(t)\end{bmatrix}$$

$$\begin{bmatrix}y_1(t)\\ y_2(t)\end{bmatrix}=\begin{bmatrix}2 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}$$

期望跟踪轨迹为

$$\begin{bmatrix}y_{1d}(t)\\ y_{2d}(t)\end{bmatrix}=\begin{bmatrix}\sin(3t)\\ \cos(3t)\end{bmatrix},\quad t\in[0,1]$$

由于 $\boldsymbol{CB}=\begin{bmatrix}2 & 2\\ 0 & 1\end{bmatrix}$，取 $\boldsymbol{\Gamma}=\begin{bmatrix}0.95 & 0\\ 0 & 0.95\end{bmatrix}$，可满足定理 5.1 中的条件(1)，在控制律式(5.30)中取 $\boldsymbol{L}=\begin{bmatrix}2.0 & 0\\ 0 & 2.0\end{bmatrix}$，系统的初始状态为 $\begin{bmatrix}x_{1(0)}(0)\\ x_{2(0)}(0)\end{bmatrix}=\begin{bmatrix}0\\ 1\end{bmatrix}$。

在 chap5_2sim.mdl 程序中，选择 Simulink 的 Manual Switch 开关，将开关向下，取 PD 型开环迭代学习控制律，仿真结果如图 5-4～图 5-6 所示。将开关向上，采用 PD 型闭环迭代学习控制律，仿真结果如图 5-7～图 5-9 所示。可见，闭环收敛速度好于开环收敛速度。

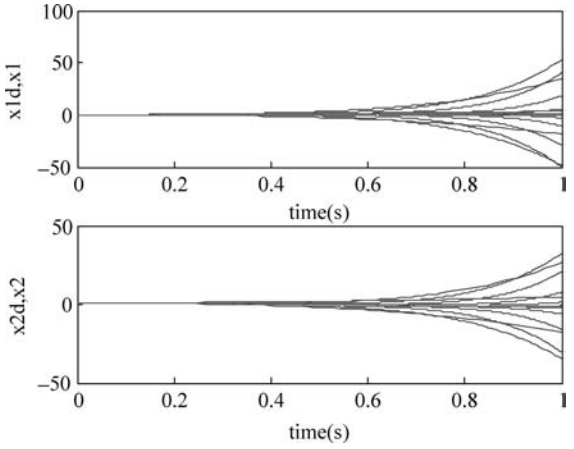


图 5-4　30 次迭代学习的跟踪过程(开环 PD 控制)

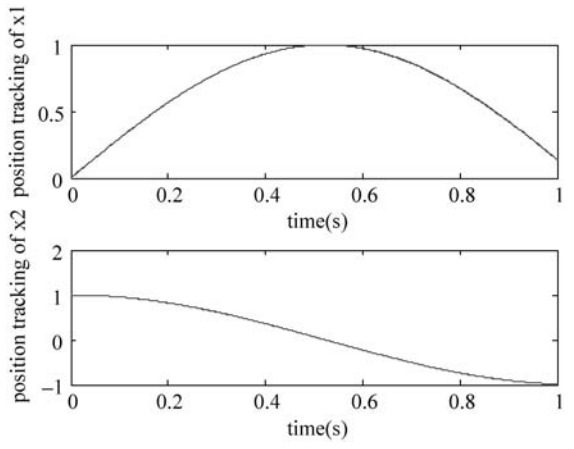


图 5-5　第 30 次迭代学习的位置跟踪(开环 PD 控制)

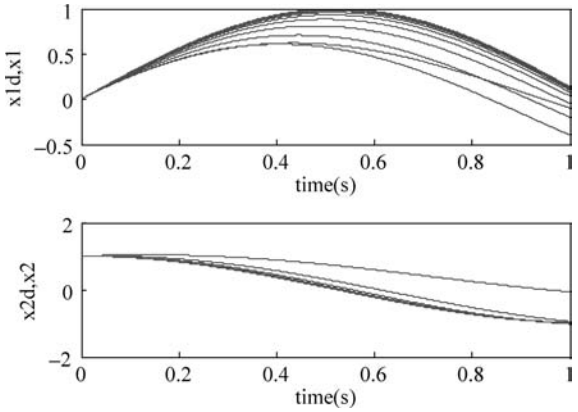


图 5-6　30 次迭代学习的跟踪过程(闭环 PD 控制)

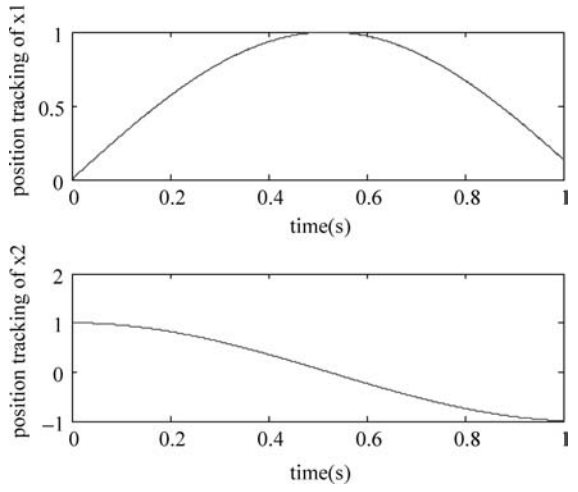


图 5-7　第 30 次迭代学习的位置跟踪(闭环 PD 控制)

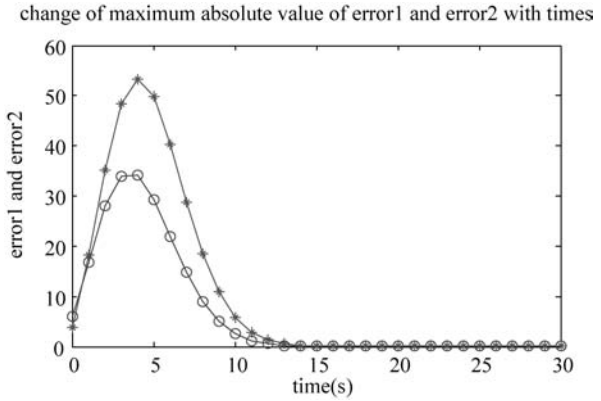


图 5-8　30 次迭代过程中误差最大绝对值的收敛过程(开环 PD 控制)

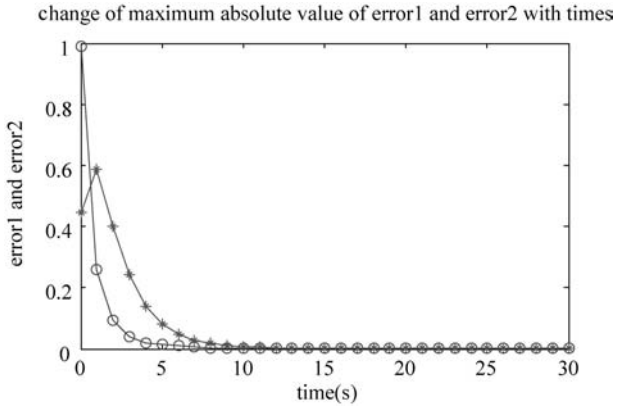


图 5-9　30 次迭代过程中误差最大绝对值的收敛过程(闭环 PD 控制)

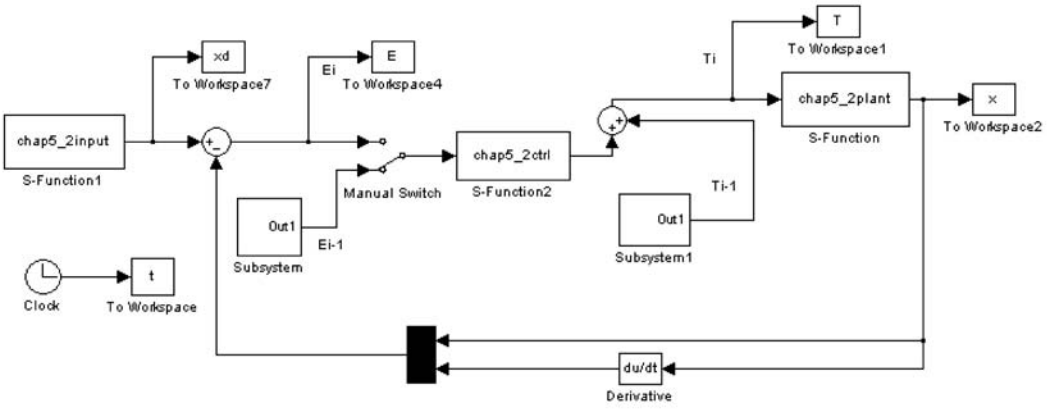
仿真程序如下：

(1) 主程序：chap5_2main.m。

```
% Iterative D-Type Learning Control
clear all;
close all;

t=[0:0.01:1]';
k(1:101)=0;              % Total initial points
k=k';
T1(1:101)=0;
T1=T1';
T2=T1;
T=[T1 T2];

k1(1:101)=0;             % Total initial points
k1=k1';
E1(1:101)=0;
E1=E1';
E2=E1;
E3=E1;
E4=E1;
E=[E1 E2 E3 E4];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
M=30;
for i=0:1:M              % Start Learning Control
i
pause(0.01);

sim('chap5_2sim',[0,1]);

x1=x(:,1);
```

```
x2 = x(:,2);

x1d = xd(:,1);
x2d = xd(:,2);
dx1d = xd(:,3);
dx2d = xd(:,4);

e1 = E(:,1);
e2 = E(:,2);
de1 = E(:,3);
de2 = E(:,4);
e = [e1 e2]';
de = [de1 de2]';

figure(1);
subplot(211);
hold on;
plot(t,x1,'b',t,x1d,'r');
xlabel('time(s)');ylabel('x1d,x1');

subplot(212);
hold on;
plot(t,x2,'b',t,x2d,'r');
xlabel('time(s)');ylabel('x2d,x2');

j = i + 1;
times(j) = i;
e1i(j) = max(abs(e1));
e2i(j) = max(abs(e2));
de1i(j) = max(abs(de1));
de2i(j) = max(abs(de2));
end                        % End of i
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2);
subplot(211);
plot(t,x1d,'r',t,x1,'b');
xlabel('time(s)');ylabel('position tracking of x1');
subplot(212);
plot(t,x2d,'r',t,x2,'b');
xlabel('time(s)');ylabel('position tracking of x2');

figure(3);
plot(times,e1i,'* - r',times,e2i,'o - b');
title('Change of maximum absolute value of error1 and error2 with times');
xlabel('time(s)');ylabel('error1 and error2');
```

（2）Simulink 程序：chap5_2sim. mdl。



（3）被控对象子程序：chap5_2plant. m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
switch flag,
case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
case 1,
    sys = mdlDerivatives(t,x,u);
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 2;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 2;
sizes.NumInputs      = 2;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [0;1];
str = [];
ts  = [0 0];
function sys = mdlDerivatives(t,x,u)
A = [ - 2 3;1 1];
C = [1 0;0 1];
B = [1 1;0 1];
Gama = 0.95;
norm(eye(2) - C * B * Gama);            % Must be smaller than 1.0

U = [u(1);u(2)];
```

```
dx = A * x + B * U;
sys(1) = dx(1);
sys(2) = dx(2);
function sys = mdlOutputs(t,x,u)
sys(1) = x(1);
sys(2) = x(2);
```

(4) 控制器子程序：chap5_2ctrl.m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
switch flag,
case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 2;
sizes.NumInputs      = 4;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];
function sys = mdlOutputs(t,x,u)
e1 = u(1);e2 = u(2);
de1 = u(3);de2 = u(4);

e = [e1 e2]';
de = [de1 de2]';

Kp = 2.0;
Gama = 0.95;
Kd = Gama * eye(2);

Tol = Kp * e + Kd * de;                    % PD Type

sys(1) = Tol(1);
sys(2) = Tol(2);
```

(5) 指令程序：chap5_2input.m。

```
function [sys,x0,str,ts] = spacemodel(t,x,u,flag)
switch flag,
```

```
case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
case 3,
    sys = mdlOutputs(t,x,u);
case {2,4,9}
    sys = [];
otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates     = 0;
sizes.NumDiscStates     = 0;
sizes.NumOutputs        = 4;
sizes.NumInputs         = 0;
sizes.DirFeedthrough    = 1;
sizes.NumSampleTimes    = 1;
sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];
function sys = mdlOutputs(t,x,u)
x1d = sin(3 * t);
dx1d = 3 * cos(3 * t);
x2d = cos(3 * t);
dx2d = -3 * sin(3 * t);

sys(1) = x1d;
sys(2) = x2d;
sys(3) = dx1d;
sys(4) = dx2d;
```

5.5 任意初始状态下的迭代学习控制

下面介绍一种任意初始状态下的学习控制方法[10]及其仿真设计方法。

5.5.1 问题的提出

假设一种系统为

$$\begin{cases}\dot{\boldsymbol{x}}(t)=\boldsymbol{A}\boldsymbol{x}(t)+\boldsymbol{B}\boldsymbol{u}(t)\\ \boldsymbol{y}(t)=\boldsymbol{C}\boldsymbol{x}(t)\\ \boldsymbol{x}(t_0)=\boldsymbol{x}(0)\end{cases} \tag{5.36}$$

其中，$\boldsymbol{x}(t)\in\mathbf{R}^n$，$\boldsymbol{u}(t)$，$\boldsymbol{y}(t)\in\mathbf{R}^m$，$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$ 为相应维数的常阵且满足假设

$$\operatorname{rank}(\boldsymbol{C}\boldsymbol{B})=m \tag{5.37}$$

设系统所要跟踪的期望轨迹为 $\boldsymbol{y}_{\mathrm{d}}(t)$，$t\in[0,T]$。系统第 i 次输出为 $\boldsymbol{y}_i(t)$，令 $\boldsymbol{e}_i(t)=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{y}_i(t)$。

在学习开始时，系统的初始状态为 $\boldsymbol{x}_0(0)$，初始控制输入为 $\boldsymbol{u}_0(t)$。学习控制的任务为

已知第 i 次运动的 $\boldsymbol{u}_i(t)$、$\boldsymbol{x}_i(0)$ 和 $\boldsymbol{e}_i(t)$，通过学习控制律设计 $\boldsymbol{u}_{i+1}(t)$ 和 $\boldsymbol{x}_{i+1}(0)$，第 $i+1$ 次运动误差 $\boldsymbol{e}_{i+1}(t)$ 将减少。

5.5.2　控制器的设计

首先介绍范数如下：

$$\| \boldsymbol{e}(t) \|_{\infty} = \max_{1 \leqslant i \leqslant m} | e^{(i)}(t) | \tag{5.38}$$

$$\| \boldsymbol{G} \|_{\infty} = \max_{1 \leqslant i \leqslant m} \left\{ \sum_{j=1}^{m} | g^{(i,j)} | \right\} \tag{5.39}$$

$$\| \boldsymbol{e}(t) \|_{\lambda} = \sup_{1 \leqslant t \leqslant T} \{ \exp(-\lambda t) \| \boldsymbol{e}(t) \|_{\infty} \} \tag{5.40}$$

其中，$e^{(i)}(t)$ 为 $e(t) \in \mathbf{R}^m$ 中的第 i 个元素，$g^{(i,j)}$ 是 $G \in \mathbf{R}^{m \times m}$ 中的第 i，j 个元素，$\lambda > 0$。

学习控制律及初始状态学习律分别为

$$\boldsymbol{u}_{i+1}(t) = \boldsymbol{u}_i(t) + \boldsymbol{L}\dot{\boldsymbol{e}}_i(t) \tag{5.41}$$

$$\boldsymbol{x}_{i+1}(0) = \boldsymbol{x}_i(0) + \boldsymbol{B}\boldsymbol{L}\boldsymbol{e}_i(0) \tag{5.42}$$

其中，$\boldsymbol{L} \in \mathbf{R}^{m \times m}$ 为常阵。

定理 5.2[10]　若学习控制律(5.41)及初始状态学习律(5.42)满足以下条件：

(1) $u_0(t)$ 在 $[0,T]$ 上连续，$\boldsymbol{y}_d(t)$ 在 $[0,T]$ 上连续可微。

(2) $\| \boldsymbol{I}_{\infty} - \boldsymbol{C}\boldsymbol{B}\boldsymbol{L} \| < 1$。

则当 $i \to \infty$ 时，有

$$\boldsymbol{y}_i(t) \to \boldsymbol{y}_d(t), \quad t \in [0,T]$$

下面给出该定理的详细分析过程，可参考文献[10]的证明过程。

由方程式(5.36)的解为

$$\boldsymbol{x}(t) = \exp(\boldsymbol{A}t)\boldsymbol{x}(0) + \int_0^t \exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{u}(t)\mathrm{d}\tau$$

则

$$\boldsymbol{x}_{i+1}(t) = \exp(\boldsymbol{A}t)\boldsymbol{x}_{i+1}(0) + \int_0^t \exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{u}_{i+1}(\tau)\mathrm{d}\tau$$

将式(5.41)和(5.42)代入上式，得

$$\boldsymbol{x}_{i+1}(t) = \exp(\boldsymbol{A}t)[\boldsymbol{x}_i(0) + \boldsymbol{B}\boldsymbol{L}\boldsymbol{e}_i(0)] + \int_0^t \exp(A(t-\tau))[\boldsymbol{B}\boldsymbol{u}_i(t) + \boldsymbol{L}\dot{\boldsymbol{e}}_i(t)]\mathrm{d}\tau$$

则

$$\begin{aligned}
\boldsymbol{e}_{i+1}(t) &= \boldsymbol{y}_d(t) - \boldsymbol{y}_{i+1}(t) = \boldsymbol{y}_d(t) - \boldsymbol{C}\boldsymbol{x}_{i+1}(t) \\
&= \boldsymbol{y}_d(t) - \boldsymbol{C}\left\{ \exp(\boldsymbol{A}t)[\boldsymbol{x}_i(0) + \boldsymbol{B}\boldsymbol{L}\boldsymbol{e}_i(0)] + \int_0^t \exp(\boldsymbol{A}(t-\tau))[\boldsymbol{B}\boldsymbol{u}_i(t) + \boldsymbol{L}\dot{\boldsymbol{e}}_i(t)]\mathrm{d}\tau \right\} \\
&= \boldsymbol{y}_d(t) - \Big[\boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{x}_i(0) + \boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{B}\boldsymbol{L}\boldsymbol{e}_i(0) + \\
&\quad \int_0^t \boldsymbol{C}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{u}_i(t)\mathrm{d}\tau + \int_0^t \boldsymbol{C}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{L}\dot{\boldsymbol{e}}_i(t)\mathrm{d}\tau \Big]
\end{aligned}$$

采用分部积分方法：

$$\int_0^t x\dot{y}\,\mathrm{d}t = xy\,|_0^t - \int_0^t \dot{x}y\,\mathrm{d}t$$

则

$$\int_0^t \boldsymbol{C}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\dot{\boldsymbol{e}}_i(t)\mathrm{d}\tau$$

$$=[\boldsymbol{C}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(t)]\Big|_0^t-\int_0^t(-1)\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(t)\mathrm{d}\tau$$

$$=\boldsymbol{CBL}\boldsymbol{e}_i(t)-\boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{BL}\boldsymbol{e}_i(0)+\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau$$

$$\begin{aligned}\boldsymbol{e}_{i+1}(t)&=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{x}_i(0)-\boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{BL}\boldsymbol{e}_i(0)-\int_0^t\boldsymbol{C}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{u}_i(\tau)\mathrm{d}\tau-\\&\quad\boldsymbol{CBL}\boldsymbol{e}_i(t)+\boldsymbol{C}\exp(\boldsymbol{A}t)\boldsymbol{BL}\boldsymbol{e}_i(0)-\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\\&=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{C}\left[\exp(\boldsymbol{A}t)\boldsymbol{x}_i(0)+\int_0^t\exp(\boldsymbol{A}(t-\tau))\boldsymbol{B}\boldsymbol{u}_i(\tau)\mathrm{d}\tau\right]-\\&\quad\boldsymbol{CBL}\boldsymbol{e}_i(t)-\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\\&=\boldsymbol{y}_{\mathrm{d}}(t)-\boldsymbol{y}_i(t)-\boldsymbol{CBL}\boldsymbol{e}_i(t)-\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\end{aligned}$$

即

$$\begin{aligned}\boldsymbol{e}_{i+1}(t)&=\boldsymbol{e}_i(t)-\boldsymbol{CBL}\boldsymbol{e}_i(t)-\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\\&=(\boldsymbol{I}_{\mathrm{m}}-\boldsymbol{CBL})\boldsymbol{e}_i(t)-\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\end{aligned}$$

上式两边同乘以 $\mathrm{e}^{-\lambda t}$，取范数，并考虑

$$\|\boldsymbol{XY}\|\leqslant\|\boldsymbol{X}\|\ \|\boldsymbol{Y}\|,\quad \|\boldsymbol{X}+\boldsymbol{Y}\|\leqslant\|\boldsymbol{X}\|+\|\boldsymbol{Y}\|$$

则

$$\begin{aligned}&\exp(-\lambda t)\|\boldsymbol{e}_{i+1}(t)\|_\infty\\&\leqslant\|(\boldsymbol{I}_m-\boldsymbol{CBL})\boldsymbol{e}_i(t)\|_\infty\exp(-\lambda t)+\left\|\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\right\|_\infty\exp(-\lambda t)\\&\leqslant\|(\boldsymbol{I}_m-\boldsymbol{CBL})\|_\infty\|\boldsymbol{e}_i(t)\|_\infty\exp(-\lambda t)+\left\|\int_0^t\boldsymbol{CA}\exp(\boldsymbol{A}(t-\tau))\boldsymbol{BL}\boldsymbol{e}_i(\tau)\mathrm{d}\tau\right\|_\infty\exp(-\lambda t)\\&=\|(\boldsymbol{I}_{\mathrm{m}}-\boldsymbol{CBL})\|_\infty\|\boldsymbol{e}_i(t)\|_\lambda+\|\boldsymbol{CABL}\|_\infty\left\|\int_0^t\exp(\boldsymbol{A}(t-\tau))\boldsymbol{e}_i(\tau)\mathrm{d}\tau\right\|_\infty\exp(-\lambda t)\\&=\|(\boldsymbol{I}_m-\boldsymbol{CBL})\|_\infty\|\boldsymbol{e}_i(t)\|_\lambda+\|\boldsymbol{CABL}\|_\infty\|\boldsymbol{h}(t)\|_\lambda\end{aligned}$$

其中，$\boldsymbol{h}(t)=\int_0^t\exp(\boldsymbol{A}(t-\tau))\boldsymbol{e}_i(\tau)\mathrm{d}\tau$。

根据 λ 范数的性质[4]，当 $\lambda>\boldsymbol{A}$ 时，有

$$\|\boldsymbol{h}(t)\|_\lambda\leqslant\frac{1-\exp((\boldsymbol{A}-\lambda)T)}{\lambda-\boldsymbol{A}}\|\boldsymbol{e}_i(t)\|_\lambda$$

则

$$\|\boldsymbol{e}_{i+1}(t)\|_\lambda\leqslant\left(\|\boldsymbol{I}_{\mathrm{m}}-\boldsymbol{CBL}\|_\infty+\|\boldsymbol{CABL}\|_\infty\frac{1-\exp((\boldsymbol{A}-\lambda)T)}{\lambda-\boldsymbol{A}}\right)\|\boldsymbol{e}_i(t)\|_\lambda=\rho\|\boldsymbol{e}_i(t)\|_\lambda$$

定义

$$\rho = \| \boldsymbol{I}_m - \boldsymbol{CBL} \|_\infty + \| \boldsymbol{CABL} \|_\infty \frac{1-\exp((\boldsymbol{A}-\lambda)T)}{\lambda - \boldsymbol{A}} \tag{5.43}$$

当 λ 足够大时，$\frac{1-\exp((\boldsymbol{A}-\lambda)T)}{\lambda-\boldsymbol{A}} \to 0$，考虑条件(2)，则

$$\rho < 1$$

当 $i \to \infty$ 时，有

$$\| \boldsymbol{e}_{i+1}(t) \|_\lambda \to 0, \quad t \in [0, T]$$

由 ρ 的定义可知，当取 $\boldsymbol{L} = (\boldsymbol{CB})^{-1}$ 时，$\| \boldsymbol{I}_m - \boldsymbol{CBL} \|_\infty = 0$，$\rho$ 的值最小，收敛速度最快。

5.5.3　仿真实例

考虑多输入多输出非线性系统：

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -2 & 3 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

期望跟踪轨迹为

$$\begin{bmatrix} y_{1d}(t) \\ y_{2d}(t) \end{bmatrix} = \begin{bmatrix} 1.5t \\ 1.5t \end{bmatrix}, \quad t \in [0, 1]$$

由于 $\boldsymbol{CB} = \begin{bmatrix} 2 & 2 \\ 0 & 1 \end{bmatrix}$，故取 $\boldsymbol{L} = (\boldsymbol{CB})^{-1} = \begin{bmatrix} 0.5 & -1 \\ 0 & 1 \end{bmatrix}$，可满足定理 5.2 中的条件(2)。

根据式(5.41)和式(5.42)，学习控制律及初始状态学习律分别为

$$\begin{bmatrix} u_{1(i+1)}(t) \\ u_{2(i+1)}(t) \end{bmatrix} = \begin{bmatrix} u_{1(i)}(t) \\ u_{2(i)}(t) \end{bmatrix} + \begin{bmatrix} 0.4 & -0.5 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} \dot{e}_{1(i)}(t) \\ \dot{e}_{2(i)}(t) \end{bmatrix}$$

$$\begin{bmatrix} x_{1(i+1)}(0) \\ x_{2(i+1)}(0) \end{bmatrix} = \begin{bmatrix} x_{1(i)}(0) \\ x_{2(i)}(0) \end{bmatrix} + \begin{bmatrix} 0.4 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} e_{1(i)}(0) \\ e_{2(i)}(0) \end{bmatrix}$$

系统的初始控制输入为 $\begin{bmatrix} u_{1(0)}(t) \\ u_{2(0)}(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$，系统的初始状态为 $\begin{bmatrix} x_{1(0)}(0) \\ x_{2(0)}(0) \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$。仿真结果如图 5-10～图 5-15 所示。

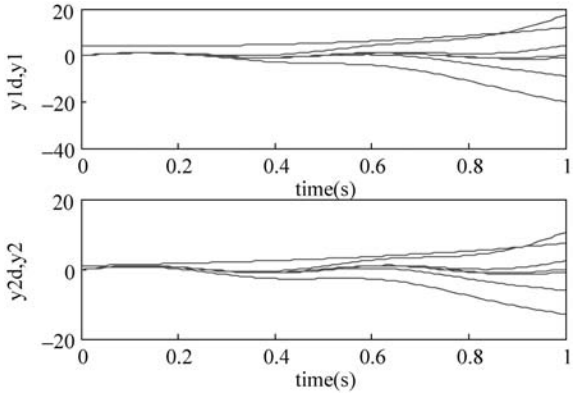


图 5-10　5 次迭代对象输出的跟踪过程

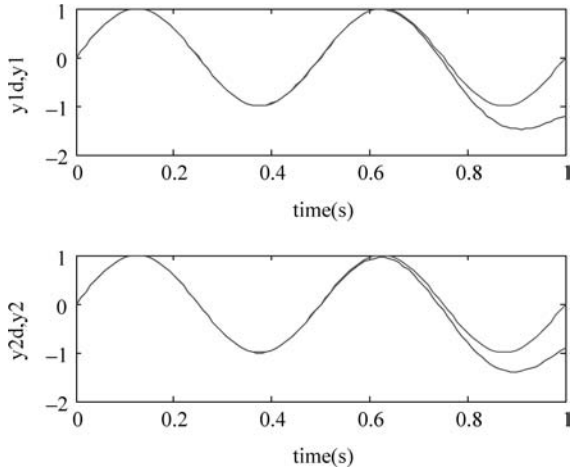


图 5-11　5 次迭代后正弦位置跟踪

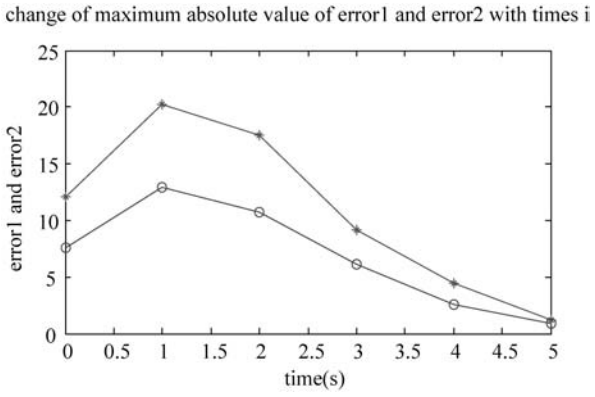


图 5-12　5 次迭代过程中误差范数的收敛过程

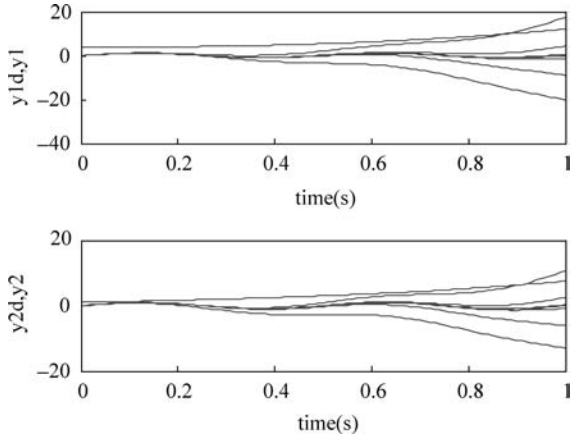


图 5-13　10 次迭代对象输出的跟踪过程

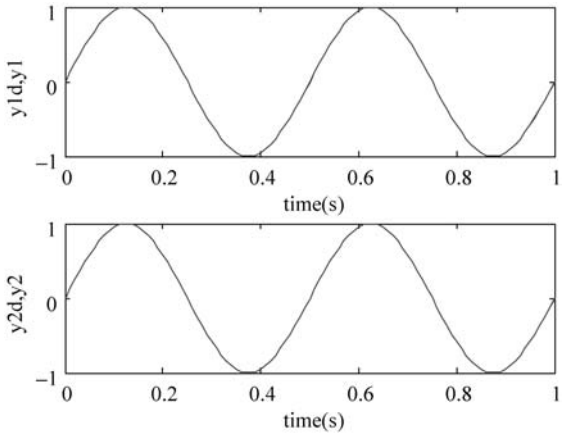


图 5-14　10 次迭代后正弦位置跟踪

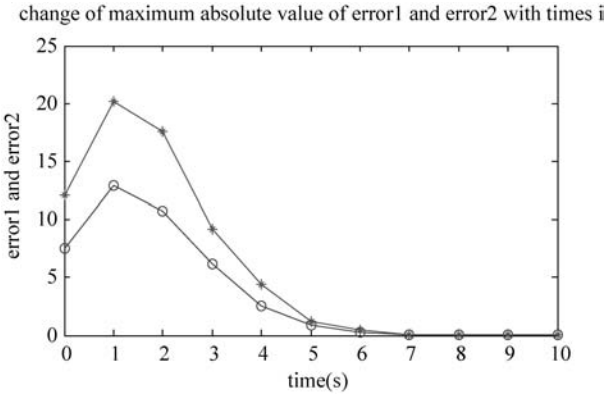


图 5-15　10 次迭代过程中误差范数的收敛过程

仿真程序如下：

(1) 主程序：chap5_3.m。

```
% Learning Control with an arbitrary initial state
clear all;
close all;
global A B

A = [ - 2 3;1 1];
B = [1 1;0 1];
C = [2 0;0 1];
L = inv(C * B);

ts = 0.01;
for k = 1:1:101
    u1(k) = 0;u2(k) = 0;

end
xk0 = [2;1];
```

```
% xk0 = [ - 2; - 1];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
M = 10;
for i = 0:1:M              % Start Learning Control
i
pause(0.005);
if i == 0
   xki = xk0;
else
   yd0 = 0;
   yi0 = [2 * xk0(1);xk0(2)];
   e0 = yd0 - yi0;
   xki = xk0 + B * L * e0;
end
xk0 = xki;                 % 用 xk0 存储上次运行的初始状态

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k = 1:1:101
time(k) = (k - 1) * ts;

S = 2;
if S == 1
    y1d_1(k) = 1.5 * (k - 1) * ts;
    y2d_1(k) = 1.5 * (k - 1) * ts;
    y1d(k) = 1.5 * k * ts;
    y2d(k) = 1.5 * k * ts;
elseif S == 2
    y1d_1(k) = sin(4 * pi * (k - 1) * ts);
    y2d_1(k) = sin(4 * pi * (k - 1) * ts);
    y1d(k) = sin(4 * pi * k * ts);
    y2d(k) = sin(4 * pi * k * ts);
end

TimeSet = [(k - 1) * ts k * ts];
para = [u1(k);u2(k)];
if k == 1                  % Initial state at times M
   xk = xk0;
end

y1_1(k) = 2 * xk(1);
y2_1(k) = xk(2);
% xk
[tt,xx] = ode45('chap5_3plant',TimeSet,xk,[],para);
% xx
xk = xx(length(xx),:);
y1(k) = 2 * xk(1);
y2(k) = xk(2);

e1_1(k) = y1d_1(k) - y1_1(k);
e2_1(k) = y2d_1(k) - y2_1(k);
```

```
e1(k) = y1d(k) - y1(k);
e2(k) = y2d(k) - y2(k);

de1(k) = (e1(k) - e1_1(k))/ts;
de2(k) = (e2(k) - e2_1(k))/ts;
dek = [de1(k);de2(k)];
Uk = [u1(k);u2(k)];
U = Uk + L * dek;           % Control law: Uk is U(i - 1), dek is near to de(i - 1)

u1(k) = U(1);
u2(k) = U(2);
end                         % End of k

figure(1);
subplot(211);
hold on;
plot(time,y1d_1,'r',time,y1_1,'b');
xlabel('time(s)');ylabel('y1d,y1');

subplot(212);
hold on;
plot(time,y2d_1,'r',time,y2_1,'b');
xlabel('time(s)');ylabel('y2d,y2');

i = i + 1;
times(i) = i - 1;
e1i(i) = max(abs(e1_1));
e2i(i) = max(abs(e2_1));
end                         % End of i
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2);
subplot(211);
plot(time,y1d_1,'r',time,y1_1,'b');
xlabel('time(s)');ylabel('y1d,y1');
subplot(212);
plot(time,y2d_1,'r',time,y2_1,'b');
xlabel('time(s)');ylabel('y2d,y2');

figure(3);
plot(times,e1i,'* - r',times,e2i,'o - b');
title('change of maximum absolute value of error1 and error2 with times i');
xlabel('times');ylabel('error1 and error2');
```

(2) 子程序：chap5_3plant.m。

```
function dx = PlantModel(t,x,flag,para)
global A B
dx = zeros(2,1);

u = para;
dx = A * x + B * u;
```

实际工程中，机械手常在有限时间内执行重复性的控制任务。针对这种有限区间执行重复性的控制任务，迭代学习控制是一种有效的控制方法。因此，机械臂轨迹跟踪的迭代学习控制得到广泛研究[11~15]。

常规的机械手迭代学习控制方法需要假设机械臂初始状态与期望轨迹初始状态相等，而这在实际过程中常常无法满足，因而针对带有初始角度偏移的机械手迭代学习控制问题的研究具有一定工程意义[14,15]。

参考文献

[1] Arimoto S, Kawamura S, Miyazaki F. Bettering operation of robotics by leaning[J]. Journal of Robotic System, 1984, 1(2): 123-140.

[2] 林辉，王林. 迭代学习控制理论[M]. 西安：西北工业大学出版社，1998.

[3] 孙明轩，黄宝健. 迭代学习控制[M]. 北京：国防工业出版社，1999.

[4] 谢胜利. 迭代学习控制的理论与应用[M]. 北京：科学出版社，2005.

[5] 于少娟，齐向东，吴聚华. 迭代学习控制理论及应用[M]. 北京：机械工业出版社，2005.

[6] 方忠，韩正之，陈彭年. 迭代学习控制新进展[J]. 控制理论与应用，2002，19(2)：161-165.

[7] 石成英，林辉. 迭代学习控制技术的原理、算法及应用[J]. 机床与液压，2004，19：80-83.

[8] 许建新，侯忠生. 学习控制的现状与展望[J]. 自动化学报，2005，31(6)：943-955.

[9] 李仁俊，韩正之. 迭代学习控制综述[J]. 控制与决策，2005，20(9)：961-966.

[10] 任雪梅，高为炳. 任意初始状态下的迭代学习控制[J]. 自动化学报，1994，20(1)：74-79.

[11] Ouyang P R, Zhang W J, Gupta M M. An adaptive switching learning control method for trajectory tracking of robot manipulators[J]. Mechatronics, 2006, 16: 51-61.

[12] Tayebi A. Adaptive iterative learning control for robot manipulators[J]. Automatica, 2004, 40: 1195-1203.

[13] Kang M K, Lee J S, Han K L. Kinematic path-tracking of mobile robot using iterative learning control[J]. Journal of Robotic Systems, 2005, 22(2): 111-121.

[14] Chen Y Q, Wen C, Gong Z, et al. An iterative learning controller with initial state learning[J]. IEEE Transaction on Automatic Control, 1999, 44(2): 371-376.

[15] Park K H, Bien Z. A generalized iterative learning controller against initial state error[J]. International Journal of Control, 2000, 73(10): 871-881.