文章编号: 1000-6893(2005) 03-0328-06

高阶 Riccati 方程加权阵选择方法及其在飞控中的应用

王　欣[1,2], 史忠科[1]
(1. 西北工业大学 自动化学院, 陕西 西安　710072)
(2. 长安大学 工程机械学院, 陕西 西安　710064)

Approach to Selecting the Weight Matrics of High Order Riccati Equation and Its Application to Airplane Control

WANG Xin[1,2], SHI Zhong-ke[1]
(1. Institute of Automation, Northwestern Polytechnical University, Xi′an　710072, China)
(2. School of Engineering Machinery, Chang′an University, Xi′an　710064, China)

摘　要: 阐述了飞机着陆控制中存在的主要矛盾和问题,说明了采用鲁棒控制方法的必要性和可行性,设计了鲁棒稳定的 H_∞ 状态反馈控制器。针对高阶 Riccati 方程加权矩阵选择困难的问题,给出了一种快速有效的参数选择算法:通过在一定区间范围内,随机改变权矩阵的取值,来搜索出满足条件的 Riccati 方程的解阵。给出了仿真算例并从理论及应用两个角度讨论了方法的优点与不足。

关键词: H_∞ 控制; Riccati 方程; 加权矩阵; 鲁棒控制; 飞机着陆控制
中图分类号: V249.1　　文献标识码: A

Abstract: The main problems of airplane landing control and solving methods are discussed in this paper. H-infinity robust control for airplane landing is studied. A parameter selecting approach is proposed to determine the weight matrics of high order Riccati equation. By using this approach the weight matrics are created randomly in a specified area, and the positive solutions of Riccati equation can be searched after repeated calculation. A robust H-infinity state feedback controller is developed. Simulations show that stability and speed of the landing period are guaranteed by using the controller presented. Both the advantages and disadvantages of the approach are addressed.

Key words: H-infinity control; Riccati equation; weight matrix; robust control; airplane landing control

着陆是飞机飞行过程中极为关键的阶段,也是飞行事故多发的一个阶段。在这一阶段,突出的矛盾来自于:控制器本身在设计上存在的固有局限性以及外界干扰的影响。

在着陆阶段,飞机的各状态变量——尤其是高度与速度变化剧烈,常规方法在进行控制器设计时,通常对模型进行小扰动线性化,将不同状态下的飞机模型用某一特定状态的线性模型近似。对于实际着陆过程而言,飞行状态是不断变化的,从而真实模型也是不断变化的,因此采用这种设计方法获得的控制器就必然存在应用上的局限性。

解决这一问题有两种思路,一是针对不同的飞行状态设计不同的控制器,在实际飞行时进行控制器切换。这种方法理论上可以提供精度较高的控制,但实现起来非常复杂,代价也比较大。另一种解决方法就是采用一个近似的模型,在设计的同时考虑模型不确定性的问题,保证控制器能够适应一定范围的模型参数变化——这就要求采用鲁棒控制方法。另一方面,从外界干扰的角度,阵风是影响飞机安全着陆的重要因素,由于阵风的风力、风向等均无法事先准确知道,因此只能将其看作是不确定的外部干扰。H_∞ 鲁棒控制方法的突出优点在于,在进行控制器设计时,只要求给出模型不确定或干扰的上界,因此,很适合于解决各种模型不确定及干扰抑制问题,本文即选择了这种方法进行阵风干扰下鲁棒控制器的设计。

H_∞ 状态反馈控制器可通过求解 Riccati 方程得到,飞机是一个复杂高维的系统,在进行 H_∞ 控制器设计时,求解高阶 Riccati 方程是不可避免的。

在求解 Riccati 方程之前,需要先确定方程中

收稿日期: 2004-04-19; 修订日期: 2005-03-01
基金项目: 国家杰出青年基金(69925306)资助项目

加权矩阵的取值。在加权阵选取方面, 虽然发展了一些研究方法, 但由于这些方法在理论上大都十分复杂, 实现起来比较困难, 所以传统上仍然大量采用手工调节或经验值选取等办法来解决一些实际问题。手工选择加权矩阵的办法通常可以用于处理低维问题, 但用于复杂高维问题则显得困难和不足, 难于取得令人满意的结果。这一因素也是 H_∞控制在理论上比较成熟, 而在工程应用方面显得滞后的原因之一。

本文所给出的快速搜索算法, 能够在给定指标要求下, 自动调整加权矩阵的参数, 快速找到满足条件的加权阵和 Riccati 方程的解阵。将这一算法应用于飞机着陆 H_∞鲁棒控制器设计, 获得了良好的效果。

1　状态反馈 H_∞控制的理论描述

为简便起见, 这里不加证明地给出本文所涉及的 H_∞控制理论中的有关定理和结论。

问题 1　设广义被控对象的状态空间实现为

$$\left.\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}_1\boldsymbol{w} + \boldsymbol{B}_2\boldsymbol{u} \\ \boldsymbol{z} &= \boldsymbol{C}_1\boldsymbol{x} + \boldsymbol{D}_{11}\boldsymbol{w} + \boldsymbol{D}_{12}\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{x} \end{aligned}\right\} \tag{1}$$

设广义被控对象满足以下假设:

①$(\boldsymbol{A}, \boldsymbol{B}_2)$ 能稳定, $(\boldsymbol{C}_1, \boldsymbol{A})$ 能观测;

②$\boldsymbol{D}_{11} = \boldsymbol{0}$, $\boldsymbol{D}_{12}$列满秩;

③$\boldsymbol{D}_{12}^{\mathrm{T}}[\boldsymbol{C}_1\quad \boldsymbol{D}_{12}] = [\boldsymbol{0}\quad \boldsymbol{R}]$。

引理 1[1]　对于满足假设 ①~ ③的线性定常系统(1) 和给定的正数 $\gamma > 0$, 存在状态反馈阵 $\boldsymbol{K}$, 使闭环系统内部稳定且

$$\|\boldsymbol{T}_{zw}(\mathrm{j}\omega)\|_\infty < \gamma \tag{2}$$

的充分必要条件是, 存在矩阵 $\boldsymbol{P} = \boldsymbol{P}^{\mathrm{T}} > \boldsymbol{0}$, 满足如下矩阵 Riccati 方程

$$\begin{aligned} &\boldsymbol{P}\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}(\gamma^{-2}\boldsymbol{B}_1\boldsymbol{B}_1^{\mathrm{T}} - \\ &\boldsymbol{B}_2\boldsymbol{R}^{-1}\boldsymbol{B}_2^{\mathrm{T}})\boldsymbol{P} + \boldsymbol{C}_1^{\mathrm{T}}\boldsymbol{C}_1 = \boldsymbol{0} \end{aligned} \tag{3}$$

其中: $\boldsymbol{T}_{zw}(j\omega)$为 $\boldsymbol{w}$ 到 $\boldsymbol{z}$ 的传递函数。状态反馈阵

$$\boldsymbol{K} = -\boldsymbol{R}^{-1}\boldsymbol{B}_2^{\mathrm{T}}\boldsymbol{P} \tag{4}$$

相应的系统状态反馈控制律为

$$\boldsymbol{u} = \boldsymbol{K}\boldsymbol{x} = -\boldsymbol{R}^{-1}\boldsymbol{B}_2^{\mathrm{T}}\boldsymbol{P}\boldsymbol{x}$$

在进行控制器设计时, 选择 $\gamma < 1$, 以保证系统的鲁棒稳定性, 设计指标式(2)变为

$$\|\boldsymbol{T}_{zw}(\mathrm{j}\omega)\|_\infty < 1 \tag{5}$$

实际中常取 $\boldsymbol{C}_1 = \begin{bmatrix} \boldsymbol{Q}^{1/2} \\ \boldsymbol{0} \end{bmatrix}$, $\boldsymbol{D}_{12} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{R}^{1/2} \end{bmatrix}$。易知 $\boldsymbol{Q} = \boldsymbol{C}_1^{\mathrm{T}}\boldsymbol{C}_1$, $\boldsymbol{R} = \boldsymbol{D}_{12}^{\mathrm{T}}\boldsymbol{D}_{12}$。

2　对加权矩阵 $\boldsymbol{Q}$、$\boldsymbol{R}$ 的讨论[2]

加权矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{R}$ 来源于二次型性能指标

$$J = \int_0^\infty \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z}\,\mathrm{d}t = \int_0^\infty (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u})\,\mathrm{d}t \tag{6}$$

其中: 矩阵 $\boldsymbol{Q} = \boldsymbol{Q}^{\mathrm{T}} \geqslant 0$, $\boldsymbol{R} = \boldsymbol{R}^{\mathrm{T}} > \boldsymbol{0}$。指标式(6) 与式(2) 有如下关系

$$\|\boldsymbol{T}_{zw}(\mathrm{j}\omega)\|_\infty < 1 \Leftrightarrow \sup_{\omega \in L_2[0, +\infty)} \frac{\|\boldsymbol{z}\|_2^2}{\|\boldsymbol{w}\|_2^2} < 1$$

设加权矩阵为对称阵主要是为了计算方便, 且不失一般性。此外, 由于任何对称矩阵必与某个对角矩阵相似, 所以实际应用中通常取加权阵为对角阵。

一般来说, 加强控制 $\boldsymbol{u}$ 可以加快输出从非零状态转移到零状态的速度, 但实际上, $\boldsymbol{u}$ 受到物理因素的限制不可能是任意大, 因此一般要求控制 $\boldsymbol{u}$ 有界, 而且最好小一些以减少能量的消耗, 为此要对 $\boldsymbol{u}$ 加一些限制指标。考虑到 $\boldsymbol{u}$ 的各个分量往往并非同等重要, 所以对各个分量加以相应的权, 常取 $\int_0^\infty \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u}\,\mathrm{d}t$ 作为衡量控制函数的指标。

矩阵 $\boldsymbol{R}$ 要求必须正定, 否则, 其对角线上会出现等于零或小于零的元素, 导致“控制任意变大以至越大越省能量”的矛盾。另外, 计算中需要用到矩阵 $\boldsymbol{R}$ 的逆, $\boldsymbol{R}$ 正定, 才能保证其逆的存在。

同理, 对状态可加以 $\int_0^\infty \boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}\,\mathrm{d}t$ 形式的指标限制, 且其值越小, 系统的状态从非零状态到达零状态的速度越快, 由于状态的某些分量可能无关紧要, 不必加以限制, 所以加权阵 $\boldsymbol{Q}$ 一般只要求半正定即可。

3　加权矩阵选择算法的设计与实现

对 Riccati 方程加权矩阵选取方法的理论研究起于 20 世纪 70 年代, 当时主要用于求解最优控制问题, 早期发展的一些方法, 如灵敏度方法、按主导极点分布选择加权矩阵等, 这些方法理论上比较繁琐, 实现起来也十分困难[3]。此后, 由于加权矩阵与某些具实际意义指标间的关系一直未能系统地建立起来, 所以工程中也始终缺乏简便实用的方法。近年来, 致力于这一研究的学者也相继提出了一些理论或实用的方法[4~6], 如采用专家系统, 矩阵试验等, 但这些方法大都缺乏普遍性, 当被控对象改变时, 需重新进行设计或试验, 且大都不能在时域中直接使用。而目前在实际中通常采用的手工试探选取的方法几乎无法直接用

于高阶系统[7]。

本文设计的“加权矩阵随机选择算法”旨在发挥计算机强大的运算能力,通过在给定区域进行给定密度的搜索,来获得到满足条件的解。方法的设计基于蒙特卡罗(Monte Carlo)原理,或称计算机随机模拟原理,是一种基于“随机数”的计算方法。Monte Carlo 方法的计算复杂性不依赖于维数。因此适合于解决那些由于维数过高而无法计算的问题。随机搜索类算法实现起来相对比较容易,一般是在理论指导不足的情况下采用的一类方法。本算法的收敛性及获得解的可靠性虽然缺乏理论的严格保证,但可通过验证结果的有效性来弥补。

算法的核心是用一组随机产生的数作为加权阵 $\boldsymbol{Q}$、$\boldsymbol{R}$ 的对角线元素值,探测解阵的正定性,不断重复这一过程,改变 $\boldsymbol{Q}$、$\boldsymbol{R}$ 取值,在大量运算中搜索到满足要求的解阵。

主要实现过程如下(为简便和突出重点,略去了判断系统可控、可观性等步骤,只对算法实现的关键环节作出描述):

(1) 输入矩阵 $\boldsymbol{A}$, $\boldsymbol{B}$, Γ 及常数 γ, C_Q, C_R 和 N。可调常数 $C_Q>0$、$C_R>0$ 用以控制不同对角线元素值的变化幅度,也即限制了搜索范围;N 为设定的运算次数;

(2) 随机产生加权矩阵 $\boldsymbol{Q}$、$\boldsymbol{R}$ 的对角线元素,初始值由函数“rand”产生,为范围在(0,1)之间的随机数,与正常数 C_Q, C_R 相乘后可改变这一范围。随机产生的矩阵必须满足 $\boldsymbol{Q}\geqslant\boldsymbol{0}$ 及 $\boldsymbol{R}>\boldsymbol{0}$;

(3) 调用求解 Riccati 方程的函数;

(4) 将得到的解阵 $\boldsymbol{P}$ 回代入原方程,检验解的准确性;

(5) 求出状态反馈阵 $\boldsymbol{K}$ 以及闭环系统的全部特征值;

(6) 判断解阵的对称性。这一环节很必要,因为在求解高阶 Riccati 方程时,阶次升高很容易导致解阵失去对称性,所以增加这一步骤来保证所得解是对称的;

(7) 判断解的正定性;

(8) 判断闭环系统特征值是否都位于左半平面;

(9) 以上几个判断条件不满足时,重新计算。

算法采用 C++ 语言实现,其中涉及到矩阵运算的部分使用 Matlab C++ 数学库(Matlab C++ Math Library)来完成,这样做的目的主要有二:一方面,搜索算法本身需做大量计算,耗费机时较多,须选用效率高的编程语言。作为一种以解释方式运行的高级计算机语言,Matlab 程序的执行效率较低。笔者做过实验,如果单纯用 Matlab 语言实现全部程序,速度较 C/C++ 方式慢得多,无法满足实际需要,不宜采用。另一方面,一般高级语言(如 C/C++)在处理矩阵运算时,编程过程复杂,需定义大量变量及数组,程序的可读性与可扩展性都不好。使用这种方式既可以提高算法的速度,又可以利用 Mablab 中强大的矩阵运算功能及其提供的工具箱函数。

Matlab 的鲁棒控制工具箱(robust control box)中提供了 Aresolv 这一标准 Riccati 方程求解函数,Aresolv 内部使用了两种方法,即“特征结构法(eigenstructure approach)”和“舒尔向量法(Schur vector approach)”,由于 Matlab C++ 数学库中没有提供这一函数,笔者用 C++ 语言实现了该算法,虽然稍嫌麻烦,但总的来说,所花费的工作还是值得的。

求解标准 Riccati 方程的成熟算法较多,如“矩阵符号函数法”等,也可以选用一种完全用 C/C++(或其它高级语言)实现。由于在 C++ 中调用 Matlab C++ 数学库编程有前述诸多优点,所以笔者推荐使用这种方式。

4 飞机着陆 H_∞ 控制器设计

对于本文所研究的飞机着陆控制问题,系统的广义被控对象定义为

$$\left.\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}_1\boldsymbol{w}+\boldsymbol{B}_2\boldsymbol{u} \\ \boldsymbol{z} &= \begin{bmatrix}\boldsymbol{Q}^{1/2}\\ \boldsymbol{0}\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{R}^{1/2}\end{bmatrix}\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{x} \end{aligned}\right\} \tag{7}$$

其中:$\boldsymbol{A}(10\times10)$ 为系统矩阵;$\boldsymbol{B}_2(10\times4)$ 为输入矩阵;$\boldsymbol{B}_1(10\times4)$ 为干扰矩阵;状态向量 $\boldsymbol{x}=[u\quad w\quad q\quad \vartheta\quad h\quad \phi\quad \varphi\quad p\quad r\quad v]^{\mathrm{T}}$,其中 u, v, w 分别为飞机速度沿 x, y, z 轴的投影分量;ϑ, ϕ, φ 为俯仰角、偏航角及滚转角;h 为飞行高度;p, q, r 分别为飞机绕质心转动的角速度沿 x, y, z 轴的投影分量;输入向量 $\boldsymbol{u}=[\delta_e\quad \delta_c\quad \delta_a\quad \delta_r]^{\mathrm{T}}$,其中 δ_e, δ_c, δ_a, δ_r 分别为升降舵偏角、鸭翼偏角、副翼偏转角和方向舵偏角;$\boldsymbol{w}=[w_1\quad w_2\quad w_3\quad w_4]^{\mathrm{T}}$ 为干扰向量。$\boldsymbol{A}$, $\boldsymbol{B}_1$, $\boldsymbol{B}_2$ 的具体取值为

$$
A=\begin{bmatrix}
0 & 0.0044 & -0.0039 & 7.4449 & 0 & 0 & 0 & 0 & 12.6221 & 0\\
0 & -0.0192 & 8.1045 & -6.3728 & 0 & 0 & 0 & -12.6221 & 0 & 0\\
0 & -0.0985 & -0.2415 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0.6503 & 0.7597 & 0 & -6.1569 & 0 & 0 & 9.5888 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.3163 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & -0.8560 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.2415 & -0.7982 & -0.01985\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 18.7258 & 0.9178 & 0.0001\\
0 & 0 & 0 & 0 & 0 & 0 & -7.4449 & 0 & -8.1045 & 0.0015
\end{bmatrix}
$$

$$
B_1=\begin{bmatrix}
0 & 0.0044 & -0.0039 & 7.4449\\
0 & -0.0192 & 8.1045 & -6.3728\\
0 & -0.0192 & -0.2415 & 0\\
0 & 0 & 1.0000 & 0\\
0.6503 & 0.7597 & 0 & -6.1569\\
0 & 0.0014 & 0.0001 & 0.0005\\
0 & -0.0098 & 0.0181 & 0.0079\\
0 & -0.0019 & -0.0028 & -1.0761\\
0 & 0 & 0.0003 & -0.0082\\
0.0166 & 0.0872 & 1.0000 & 0.0187
\end{bmatrix},\
B_2=\begin{bmatrix}
0.0127 & -0.0159 & 0 & 0\\
-0.0357 & 0.0051 & 0 & 0\\
-4.6881 & 4.1303 & 0 & 0\\
0 & 0 & 0 & 0\\
0 & 0 & 0 & 0\\
0 & 0 & 0 & 0\\
0 & 0 & 0 & 0\\
0 & 0 & 0.5952 & 0.1364\\
0 & 0 & -0.9812 & 0.0485\\
0 & 0 & -0.0044 & 1.0812
\end{bmatrix}
$$

本文目标是设计状态反馈控制器, 使系统达到如下性能指标要求:

(1) 闭环系统是内部稳定的, 即 $\boldsymbol{A}+\boldsymbol{B}_2\boldsymbol{K}$ 是稳定的;

(2) 闭环系统满足 $\|\boldsymbol{T}_{zw}(j\omega)\|_\infty<1$。

易验证广义对象式(7)满足假设条件①~③, 根据引理 1, 要得到满足指标要求的状态反馈控制器, 需要求解以下矩阵 Riccati 方程

$$
\boldsymbol{PA}+\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}(\gamma^{-2}\boldsymbol{B}_1\boldsymbol{B}_1^{\mathrm{T}}-\boldsymbol{B}_2\boldsymbol{R}^{-1}\boldsymbol{B}_2^{\mathrm{T}})\boldsymbol{P}+\boldsymbol{Q}=\boldsymbol{0} \tag{8}
$$

取得对称正定解 $\boldsymbol{P}$。其中 $\boldsymbol{Q}$, $\boldsymbol{R}$ 即为所要调整的加权矩阵。

给定 $\gamma=0.8$, 计算中干扰信号取满足 $|w_i|\leqslant 5$ 的随机值, 用本文给出的算法, 每次批量运算可得不同组解, 进行一定比较和选择后, 取其中一组解(加权阵和解阵的具体值见附录), 得反馈阵如下

闭环系统特征值为

$$
\lambda_{1,\cdots,10}=\\
[-2.7109\pm 5.3099i\quad -4.7055\pm 2.3573i\\
-4.4023\quad -2.266\quad -0.91925\\
-0.07874\quad -0.023977\quad -0.035415]
$$

解阵 $\boldsymbol{P}$ 与相应的加权阵 $\boldsymbol{Q}$、$\boldsymbol{R}$ 的计算结果见附录。

各状态(增量)随时间的响应曲线如图 1~图 5。

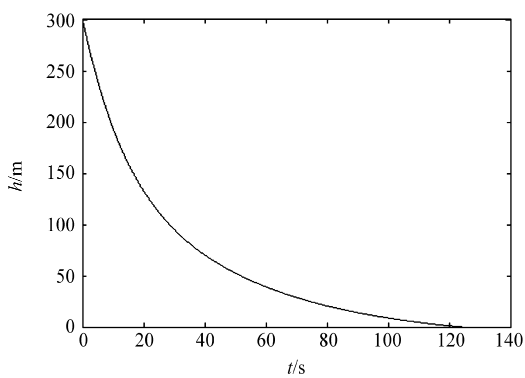

图 1　高度 h 随时间的响应曲线

Fig. 1　The response plot of h

$$
K=\begin{bmatrix}
4.0511 & 3.628 & 1.1504 & -24.492 & 0.077235 & -0.72809 & 57.898 & -1.6841 & -0.7521 & 0.36412\\
-1.901. & -0.98424 & -1.4482 & 1.9501 & 0.01186 & 1.0215 & -26.062 & 1.8986 & 0.83945 & -0.42168\\
1.7567 & 1.4959 & 0.12192 & -10.763 & 0.011767 & -0.52517 & 28.807 & -25.463 & -5.346 & 1.9927\\
-0.4257 & -0.232 & -0.051744 & 1.3883 & 0.018602 & 1.7125 & 2.7298 & -4.3443 & -1.2638 & -0.90241
\end{bmatrix}
$$

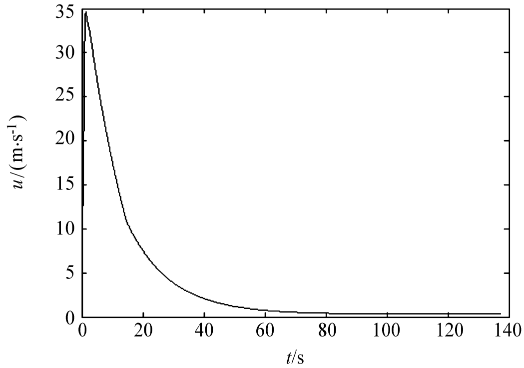

图 2 速度分量 u 随时间的响应曲线

Fig. 2 The response plot of u

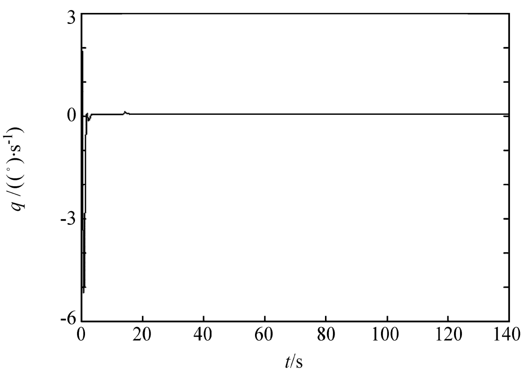

图 3 角速度分量 q 随时间的响应曲线

Fig. 3 The response plot of q

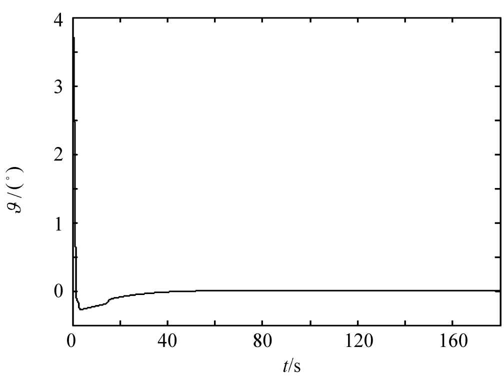

图 4 俯仰角 ϑ 随时间的响应曲线

Fig. 4 The response plot of ϑ

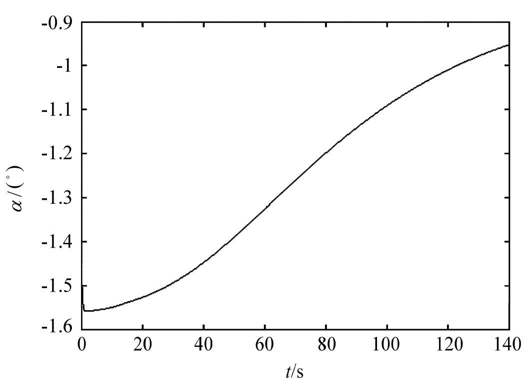

图 5 迎角 α 随时间的响应曲线

Fig. 5 The response plot of α

5 结 论

高阶 Riccati 方程加权阵选择方法有以下优点:

(1) 思想较为简单, 实现起来容易, 易于工程技术人员掌握;

(2) 解决实际问题时, 不需要有加权阵选择的先验经验, 能快速获得满足条件的解阵, 简便实用;

(3) 被控对象发生改变时, 算法基本不需要改变;

(4) 除了作为参数选择的工具以外, 还可以用于探测解存在的区域, 并获得加权阵 $\boldsymbol{Q}$、$\boldsymbol{R}$ 的经验取值, 便于在有实际需要时进一步在敏感区域中寻找更多解。

方法的不足在于:

(1) 在实现上主要依靠大批量数据运算, 发挥了工具——计算机的长处, 但还没有能更充分地糅合控制理论本身的思想和成熟规律, 智能化显得不足, 这也是本方法需进一步改进和深化的地方;

(2) 由于理论指导上的不足, 不能保证在任一搜索区间都能找到解, 当所给区间不存在解时, 需要调整搜索范围。

该方法适用于阶数较高的、在加权阵选取方面没有先验经验的系统。除可以用于 H_∞ 控制器设计, 还可用于最优控制等需要选择权矩阵的设计问题。

对以上所给的应用实例而言, 因为被控对象维数较高, 若采用手工调整参数的方法, 几乎不可能得到满足设计要求的 Riccati 方程正定解; 而使用本文给出的算法, 这一问题得到了较为满意的解决, 仿真结果表明此方法是可行和有效的。

参 考 文 献

[1] Kwakernaak H. Robust control and H_∞ optimization[J]. Automatica, 1993, 29(2): 255- 273.

[2] 谢绪恺. 现代控制理论基础[M]. 沈阳: 辽宁人民出版社, 1981. 376- 399.

Xie X K. Fundamentals of modern control theory[M]. Shenyang: Liaoning People's Publishing House, 1981. 376 - 399. (in Chinese)

[3] 解学书. 最优控制——理论与应用[M]. 北京: 清华大学出版社, 1986. 329- 376.

Xie X S. Optimal control theory and application[M]. Beijing: Tsinghua University Press, 1986. 329- 376. (in Chinese)

[4] 吴旭东, 解学书. H_∞ 鲁棒控制中的加权阵选择[J]. 清华大学学报, 1997, 37(1): 27- 30.

Wu X D, Xie X S. Weighting function matrix selection in H_∞ robust control[J]. Journal of Tsinghua University, 1997, 37(1): 27- 30. (in Chinese)

[5] Yang C D, Ju H S, Liu S W. Experimental design of H_∞ weighting functions for flight control systems[A]. Proceedings of the American Control Conference[C]. 1994. 2516- 2520.

[6] Postlethwaite I, Tsai M C, Gu D W. Weighting function selection in H_∞ design[A]. Proceedings of the IFAC Conference[C]. 1992. 104- 109.

[7] Wang X, Shi Z K. A hierarchical approach for H_∞ control and its application to airplane landing under wind disturbance[J]. Control Theory and Applications, 1999, 16(6): 873- 876.

作者简介:

王　欣(1974-)　女, 河北藁城人, 长安大学工程机械学院讲师, 博士, 2004 年毕业于西北工业大学控制理论与控制工程专业, 主要从事智能控制及鲁棒控制等研究。E-mail: small.2003@tom.com。

附录

仿真计算得 Riccati 方程(8)的一组解阵与相应的加权阵如下

$$
\boldsymbol{P}=\begin{bmatrix}
47.373 & 52.145 & 0.34474 & -418.26 & 1.5673 & 0.88497 & 692.28 & -4.542 & -2.1977 & 0.81687 \\
52.145 & 58.328 & 0.24649 & -469.26 & 1.7706 & 1.6001 & 763.12 & -3.7611 & -1.8065 & 0.63474 \\
0.34474 & 0.24649 & 0.1733 & -1.2682 & 0.0024575 & -0.12007 & 4.832 & -0.23874 & -0.10611 & 0.05253 \\
-418.26 & -469.26 & -1.2682 & 3782.4 & -14.295 & -14.142 & -6123.3 & 27.82 & 13.458 & -4.5875 \\
1.5673 & 1.7706 & 0.0024575 & -14.295 & 0.09046 & 0.44481 & 22.5 & -0.032302 & -0.015829 & -0.0015577 \\
0.88479 & 1.6001 & -0.12007 & -14.142 & 0.44481 & 23.366 & -12.775 & 3.0059 & 1.6603 & -1.0378 \\
692.28 & 763.12 & 4.832 & -6123.3 & 22.5 & -12.775 & 10186 & -77.289 & -37.728 & 10.514 \\
-4.542 & -3.7611 & -0.23874 & 27.82 & -0.032302 & 3.0059 & -77.289 & 24.984 & 7.0304 & 1.9862 \\
-2.1977 & -1.8065 & -0.10611 & 13.458 & -0.015829 & 1.6603 & -37.728 & 7.0304 & 2.5595 & -0.57091 \\
0.81687 & 0.63474 & 0.05253 & -4.5875 & -0.0015577 & -1.0378 & 10.514 & 1.9862 & -0.57091 & 0.58397
\end{bmatrix}
$$

$\boldsymbol{Q}$= diag(0.059476　0.018128　0.56716　0.95389　0.0006759　0.48974　0.43087　0.11445　0.2329　0.92253)

$\boldsymbol{R}$= diag(0.71008　0.49123　0.31343　0.36872)

(责任编辑: 李泓洁)