

doi: 10.12052/gdutxb.170082

一种基于自适应动态规划的协同优化算法

刘毅, 章云

(广东工业大学 自动化学院, 广东 广州 510006)

摘要: 采用值迭代的自适应动态规划的收敛条件是迭代性能指标函数初始化为任意半正定函数. 根据此收敛条件, 本文研究了迭代性能指标函数的初始化和更新方法, 提出了一种基于自适应动态规划的协同优化算法. 仿真结果表明, 该协同优化算法令迭代的残差快速减小, 大幅提高了自适应动态规划的收敛速度.

关键词: 自适应动态规划; 值迭代; 协同优化

中图分类号: TP273

文献标志码: A

文章编号: 1007-7162(2017)06-0015-05

A Cooperative Optimization Algorithm Based on Adaptive Dynamic Programming

Liu Yi, Zhang Yun

(School of Automation, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: If the initial iterative performance index function is a positive semi-definite function, then the value iteration of adaptive dynamic programming will converge to the optimal. This is the convergence condition of value-iteration based adaptive dynamic programming. Based on the condition, the initializing and updating methods for iterative performance index function is studied and a cooperative optimization algorithm based on adaptive dynamic programming is proposed. The simulation results show that the proposed algorithm can rapidly reduce the iteration residuals and greatly improve the convergence rate of adaptive dynamic programming.

Key words: adaptive dynamic programming; value iteration; cooperative optimization

动态规划是一种逆向计算方法, 必须在时间上逆向求解最优控制序列. 虽然动态规划能精确求解哈密顿-雅可比-贝尔曼方程, 但会导致计算量和存储量随着控制序列的长度也即解的时间维度的增加而急剧增长, 也即“维数灾”问题^[1]. 由Werbos提出的自适应动态规划通过迭代计算求解哈密顿-雅可比-贝尔曼方程^[2-5], 迭代控制策略和迭代性能指标函数逐渐逼近最优控制策略和最优性能指标函数, 避免了“维数灾”问题. 自适应动态规划有4种基本的实现结构: 启发式动态规划(HDP)、二次启发式规划(DHP)、执行依赖启发式动态规划(ADHDP)和执行依赖二次启发式规划(ADDHP)^[2-3]. 文献[6]提出了全新的自适应动态规划实现结构: 全局二次启发式规划(GDHP)和执行依赖全局二次启发式规划(ADGDHP).

值迭代是自适应动态规划的一种迭代方式, 需要初始化迭代性能指标函数^[7-8]. Al-Tamimi等^[9]给出了一种应用于离散时间仿射非线性系统的值迭代, 首次指出迭代性能指标函数初始化为零可保证迭代收敛到最优, 并给出了数学证明. Zhang等^[10]研究了离散仿射非线性系统的最优跟踪控制问题, 并给出了值迭代的收敛性和最优性证明. Wang等^[11]给出了应用于离散时间非仿射非线性系统的值迭代的收敛性和最优性证明. 文献[9-11]要求迭代性能指标函数初始化为零. Wei等^[12]给出一种保证迭代控制策略是稳定控制策略的迭代性能指标函数的初始化方法, 并给出了收敛性证明. 文献[13-14]证明了迭代性能指标函数初始化为半正定函数可保证值迭代收敛到最优, 极大地放松了值迭代对迭代性能指标函数的初始化要求.

收稿日期: 2017-04-11

基金项目: 国家自然科学基金资助项目(U1501251, 51307025); 高等学校博士学科点专项科研基金资助项目(20124420130001)

作者简介: 刘毅(1979-), 男, 博士研究生, 主要研究方向为智能控制、优化控制.

通信作者: 章云(1963-), 男, 教授, 博士生导师, 主要研究方向为优化控制、非线性控制和多智能体技术等. E-mail: yun@gdut.edu.cn

文献[9-14]指出,如果值迭代的迭代次数趋向无穷,那么迭代性能指标函数和迭代控制策略分别等于最优性能指标函数和最优控制策略.无穷次迭代是无法实现的,有限次迭代所得的迭代性能指标函数和迭代控制策略是含有残差的近似最优性能指标函数和近似最优控制策略.如果为迭代计算给定一个收敛精度,则可通过多次迭代让残差达到收敛精度的要求.由于放松了对迭代次数的要求,自适应动态规划的实时性将无法得到保证.如果给定迭代次数,则不能保证残差达到收敛精度的要求.

值迭代的残差大小与初始的迭代性能指标函数有关.初始的迭代性能指标函数影响着迭代次数和收敛精度.本文根据文献[13-14]的证明结果,研究了一种在有限迭代次数下保证收敛精度的快速自适应动态规划算法.

1 问题描述

本文所描述的离散时间非仿射非线性系统如下:

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k, \mathbf{u}_k), \quad k = 0, 1, 2, \dots \quad (1)$$

其中 $\mathbf{x}_k \in \mathbb{R}^n$ 是状态向量, $\mathbf{u}_k \in \mathbb{R}^m$ 是控制向量. $\forall \mathbf{x}_k, \mathbf{u}_k$, $F(\mathbf{x}_k, \mathbf{u}_k)$ 连续, $F(0, 0) = 0$. $\mathbf{x} = 0$ 是系统(1)的稳定状态.另外,系统(1)是可控的.

优化目标是寻找控制序列 $\underline{\mathbf{u}}_k = (\mathbf{u}_k, \mathbf{u}_{k+1}, \dots)$,使得性能指标函数(2)最小:

$$J(\mathbf{x}_k, \underline{\mathbf{u}}_k) = \sum_{t=k}^{\infty} U(\mathbf{x}_t, \mathbf{u}_t). \quad (2)$$

其中 $U(\mathbf{x}_t, \mathbf{u}_t)$ 称为效用函数,是连续的正定函数.假设采用状态反馈对系统(1)进行控制,即 $\mathbf{u}_k = \mathbf{u}(\mathbf{x}_k)$.另外假设反馈控制满足 $\mathbf{u}(0) = 0$.

根据Bellman最优定理,最优性能指标函数满足如下哈密顿-雅可比-贝尔曼方程:

$$J^*(\mathbf{x}_k) = \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J^*(\mathbf{x}_{k+1})\}. \quad (3)$$

式(3)也可表示为

$$J^*(\mathbf{x}_k) = U(\mathbf{x}_k, \mathbf{u}^*(\mathbf{x}_k)) + J^*(\mathbf{x}_{k+1}). \quad (4)$$

其中 $\mathbf{u}^*(\mathbf{x}_k)$ 是最优控制策略:

$$\mathbf{u}^*(\mathbf{x}_k) = \arg \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J^*(\mathbf{x}_{k+1})\}. \quad (5)$$

即

$$\mathbf{u}^*(\mathbf{x}_k) = \arg \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J^*(\mathbf{x}_{k+1})\}. \quad (6)$$

2 快速自适应动态规划算法

2.1 自适应动态规划的值迭代

设 $\mathbf{u}_i(\mathbf{x}_k)$ 是迭代控制策略, $J_i(\mathbf{x}_k)$ 是迭代性能指标函数, $i = 0, 1, 2, \dots$ 是迭代次数.初始化 $J_0(\mathbf{x})$,值迭代的迭代过程为

$$\mathbf{u}_i(\mathbf{x}_k) = \arg \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J_i(\mathbf{x}_{k+1})\}. \quad (7)$$

$$J_{i+1}(\mathbf{x}_k) = \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J_i(\mathbf{x}_{k+1})\}. \quad (8)$$

文献[13-14]证明了若 $J_0(\mathbf{x})$ 是半正定函数,则 $\lim_{i \rightarrow \infty} J_i(\mathbf{x}_k) = J^*(\mathbf{x}_k)$, $\lim_{i \rightarrow \infty} \mathbf{u}_i(\mathbf{x}_k) = \mathbf{u}^*(\mathbf{x}_k)$.

2.2 基于自适应动态规划的协同优化算法

粒子群算法会初始化多个携带不同可行解的粒子,各粒子在迭代过程中通过其他粒子的解来更新自己的解.参考粒子群算法,本节提出了基于自适应动态规划的协同优化算法(Cooperative Optimization Based on Adaptive Dynamic Programming, COADP).COADP会创建多个值迭代“粒子”.在每一次迭代里,COADP根据所有“粒子”的迭代性能指标函数来重新构建各“粒子”的迭代性能指标函数.

设一个进程(或智能体、粒子等)运行一个值迭代.设 A 是进程的数量, $J_i^{(a)}(\mathbf{x}_k)$ 是进程 a 的迭代性能指标函数, $\mathbf{u}_i^{(a)}(\mathbf{x}_k)$ 是进程 a 的迭代控制策略, $a = 1, 2, \dots, A$.

(1) COADP的初始化.

由于迭代性能指标函数可初始化为任意半正定函数,COADP将首先初始化 A 个进程及 A 个互不相同的半正定函数 $J_0^{(a)}(\mathbf{x}_k)$.

(2) COADP的迭代.

命名 $V_{i+1}^{(a)}(\mathbf{x}_k)$ 为进程 a 的备选迭代性能指标函数.参考式(7)和(8),进程 a 以如下方式迭代:

$$\mathbf{u}_i^{(a)}(\mathbf{x}_k) = \arg \min_{\mathbf{u}(\mathbf{x}_k)} \{U(\mathbf{x}_k, \mathbf{u}(\mathbf{x}_k)) + J_i^{(a)}(\mathbf{x}_{k+1})\}. \quad (9)$$

$$V_{i+1}^{(a)}(\mathbf{x}_k) = U(\mathbf{x}_k, \mathbf{u}_i^{(a)}(\mathbf{x}_k)) + J_i^{(a)}(\mathbf{x}_{k+1}). \quad (10)$$

与式(8)不同的是,式(10)没有立刻更新 $J_{i+1}^{(a)}(\mathbf{x}_k)$,而是把 $U(\mathbf{x}_k, \mathbf{u}_i^{(a)}(\mathbf{x}_k)) + J_i^{(a)}(\mathbf{x}_{k+1})$ 作为一个备选函数保存于 $V_{i+1}^{(a)}(\mathbf{x}_k)$.

(3) COADP的更新.

设第 B_{i+1} 个进程的 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$ 与 $J^*(\mathbf{x}_k)$ 最相似(本节称 $V_{i+1}^{(B_0)}(\mathbf{x}_k)$ 为最优备选迭代性能指标函数,初始化为 $V_0^{(B_0)}(\mathbf{x}_k) = 0$).COADP的更新方法是:找出 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$,然后在 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$ 的邻域内构造 $J_{i+1}^{(a)}(\mathbf{x}_k)$.

若 $V_{i+1}^{(a)}(\mathbf{x}_k) = J_i^{(a)}(\mathbf{x}_k)$, 那么式(10)与式(4)是相同的, $V_{i+1}^{(a)}(\mathbf{x}_k) = J^*(\mathbf{x}_k)$. 由此可推断, 若 $\|V_{i+1}^{(a)}(\mathbf{x}_k) - J_i^{(a)}(\mathbf{x}_k)\| \approx 0$, 则 $V_{i+1}^{(a)}(\mathbf{x}_k) \approx J^*(\mathbf{x}_k)$. 据此, COADP通过式(11)寻找 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$:

$$B_{i+1} = \arg \min_a \left\{ \left\| \frac{V_{i+1}^{(a)}(\mathbf{x}_k) - J_i^{(a)}(\mathbf{x}_k)}{V_{i+1}^{(a)}(\mathbf{x}_k)} \right\| \right\}. \quad (11)$$

为了实现可比性, 式(11)中采用的是相对变化率. 在重新构建 $J_{i+1}^{(a)}(\mathbf{x}_k)$ 的时候, 令第 B_{i+1} 个进程的 $J_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$ 直接等于 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$, 而其他进程的 $J_{i+1}^{(a)}(\mathbf{x}_k)$ 则在 $V_{i+1}^{(B_{i+1})}(\mathbf{x}_k)$ 的邻域内进行构建:

$$J_{i+1}^{(a)}(\mathbf{x}_k) = \begin{cases} V_{i+1}^{(B_{i+1})}(\mathbf{x}_k), & a = B_{i+1}; \\ \max \{ V_{i+1}^{(B_{i+1})}(\mathbf{x}_k) + Ra_{i+1}^{(a)} \cdot D \cdot V_{i+1}^{(B_{i+1})}(\mathbf{x}_k), 0 \}, & a \neq B_{i+1}. \end{cases} \quad (12)$$

其中 $Ra \in [-1, 1]$ 是随机数, $D > 0$ 是调整范围. 式(12)保证了 $J_{i+1}^{(a)}(\mathbf{x}_k)$ 是半正定函数, 收敛性不会受到影响.

图1是COADP的流程图. 当 $\|V_{i+1}^{(B_{i+1})}(\mathbf{x}_k) - V_i^{(B_i)}(\mathbf{x}_k)\|$ 小于预先设置的收敛精度 ε , 则近似认为迭代已收敛, 迭代结束.

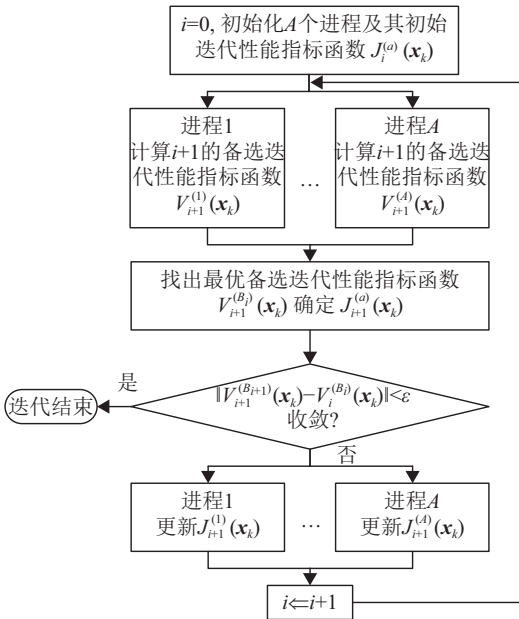


图1 COADP算法的流程图

Fig.1 Flow chart of COADP

3 仿真测试

图2是COADP的结构图, 其中每个进程均采用HDP结构. 采用三层BP神经网络近似 $J_i^{(a)}(\mathbf{x}_k)$ 和 $u_i^{(a)}(\mathbf{x}_k)$. 下面给出3个例子来验证COADP的有效性.

第一和第二个例子来自文献[11], 第三个例子与文献[12]、[15]和[16]相同.

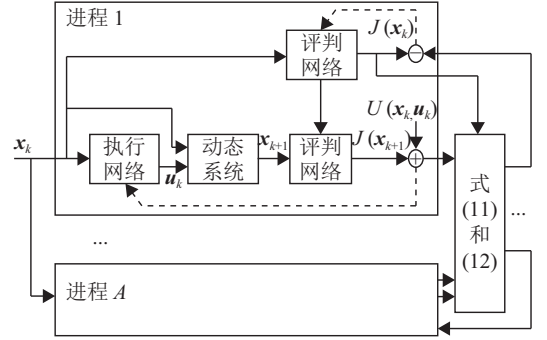


图2 COADP算法的结构图

Fig.2 Scheme of COADP

例1: 非仿射非线性系统为

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \sin(\mathbf{x}_k + \mathbf{u}_k). \quad (13)$$

其中 $\mathbf{x}_k \in \mathbb{R}$, $\mathbf{u}_k \in \mathbb{R}$. $\mathbf{x}_k = 0$ 是该系统的稳定状态. 效用函数为 $U(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{x}_k^T \mathbf{x}_k + \mathbf{u}_k^T \mathbf{u}_k$. 初始状态为 $\mathbf{x}_0 = 1.5$. 所有进程的执行网络均是1-8-1结构, 评判网络均是1-8-1结构. 执行网络和评判网络的学习率均为0.02. 执行网络和评判网络的训练次数均设定为2 000次. 各进程的执行网络均初始化为相同的神经网络.

首先令进程数量 $A=3$, 初始化各进程的评判网络为: $J_0^{(1)}(\mathbf{x}_k) = 0$, $J_0^{(2)}(\mathbf{x}_k) = 6\mathbf{x}_k^T \mathbf{x}_k$, $J_0^{(3)}(\mathbf{x}_k) = 12\mathbf{x}_k^T \mathbf{x}_k$. 图3是 $A=3$ 时的 $V_i^{(B_i)}(\mathbf{x}_k)$ 收敛曲线. 为了进行比较, 图3~图8中加入了 $J_0(\mathbf{x}_k) = 0$ 的且采用式(7)和式(8)进行迭代的 $J_i(\mathbf{x}_k)$ 收敛曲线(图中简称zero-ADP). 若 $D = 0.1$, 则调整范围过小. 与zero-ADP相比, COADP的残差减小得快, 但不明显. 若 $D = 1$, 则调整范围合适, 残差快速减小. 若 $D = 2$, 则调整范围过大. 线程数量 A 不足以支持如此大范围的搜索, 残差减小的速度没有明显加快.

修改进程数量 $A=7$, 新增的4个进程的评判网络初始化为 $J_0^{(4)}(\mathbf{x}_k) = 2\mathbf{x}_k^T \mathbf{x}_k$, $J_0^{(5)}(\mathbf{x}_k) = 4\mathbf{x}_k^T \mathbf{x}_k$, $J_0^{(6)}(\mathbf{x}_k) = 8\mathbf{x}_k^T \mathbf{x}_k$, $J_0^{(7)}(\mathbf{x}_k) = 10\mathbf{x}_k^T \mathbf{x}_k$. 各进程的执行网络仍旧初始化为相同的神经网络. 图4是 $A=7$ 时的 $V_i^{(B_i)}(\mathbf{x}_k)$ 收敛曲线. 若 $D = 0.1$, 则调整范围过小, 即使 A 增大了也无法显著地加快残差减小的速度. 若 $D = 1$, 则调整范围合适, 残差快速减小. 若 $D = 2$, 则调整范围足够大. 此时线程数量 A 足以实现在如此大范围内的有效搜索, 残差减小的速度进一步加快, 协同优化效果明显.

若 D 足够大, 则通过式(12)更新所得的某一 $J_{i+1}^{(a)}(\mathbf{x}_k)$ (设为 $J_{i+1}^{(\bar{a})}(\mathbf{x}_k)$) 可能会远大于 $J^*(\mathbf{x}_k)$. 若 A 不足,

则可供选择的 $V_{i+1}^{(a)}(\mathbf{x}_k)$ 数量太少, $V_{i+2}^{(B_{i+2})}(\mathbf{x}_k)$ 选择 $V_{i+2}^{(\bar{a})}(\mathbf{x}_k)$ 的可能性就会增大. 因此在图3中, $D=2$ 的 $V_3^{(B_3)}(\mathbf{x}_k)$ 出现远大于 $J^*(\mathbf{x}_k)$ 的现象.

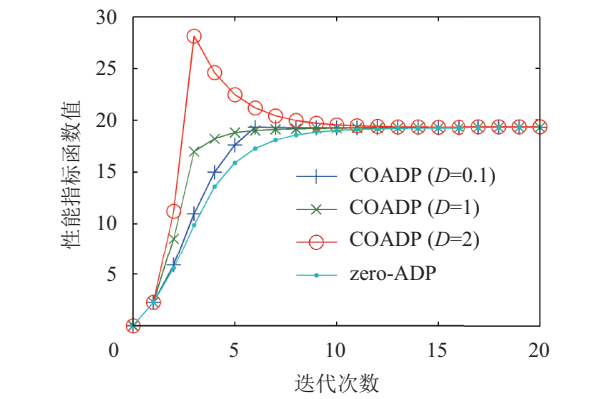


图 3 例1的最优备选迭代性能指标函数收敛过程($A=3$)

Fig.3 Convergence of the best optional iterative performance index function for Example 1 ($A=3$)

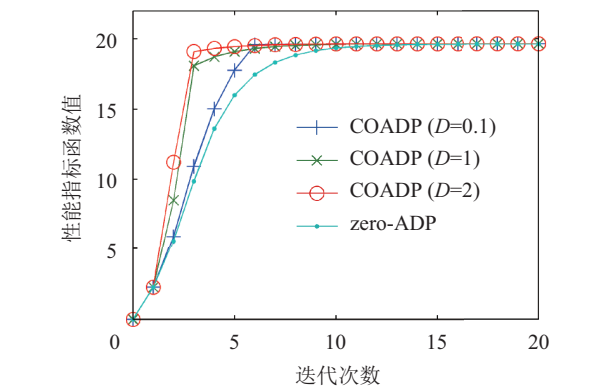


图 4 例1的最优备选迭代性能指标函数收敛过程($A=7$)

Fig.4 Convergence of the best optional iterative performance index function for Example 1 ($A=7$)

例2: 非仿射非线性系统为

$$\mathbf{x}_{k+1} = \begin{bmatrix} -\mathbf{x}_{1k}\mathbf{x}_{2k} \\ 1.5\mathbf{x}_{2k} + \sin(\mathbf{x}_{2k}^2 + \mathbf{u}_k) \end{bmatrix}. \quad (14)$$

其中 $\mathbf{x}_k = [\mathbf{x}_{1k}, \mathbf{x}_{2k}]^T \in \mathbb{R}^2, \mathbf{u}_k \in \mathbb{R}$. $\mathbf{x}_k = 0$ 是该系统的稳定状态. 效用函数 $U(\mathbf{x}_k, \mathbf{u}_k)$ 与例1相同. 初始状态为 $\mathbf{u}_0 = [0.5, -1]^T$. 所有进程的执行网络均是2-8-1结构, 评判网络均是2-8-1结构. 其他设置与例1相同. 图5、6分别是 $A=3$ 、 $A=7$ 时的 $V_i^{(B_i)}(\mathbf{x}_k)$ 收敛曲线. 对于系统(14), 若 $D=0.1$, 则调整范围过小, COADP算法的效果不明显; 若 $D=0.5$, 则调整范围合适, 残差快速减少, 但 A 对残差减小的速度影响不大; 若 $D=1$, 则调整范围足够大. 若增大 A 可令残差减小的速度进一步加快, 协同优化效果明显. D 和 A 对残差的影响与例1相同.

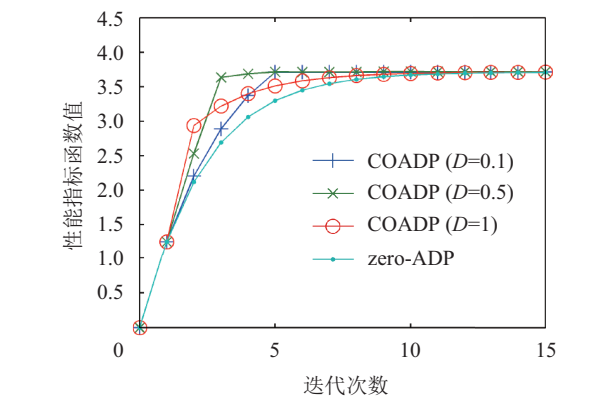


图 5 例2的最优备选迭代性能指标函数收敛过程($A=3$)

Fig.5 Convergence of the best optional iterative performance index function for Example 2 ($A=3$)

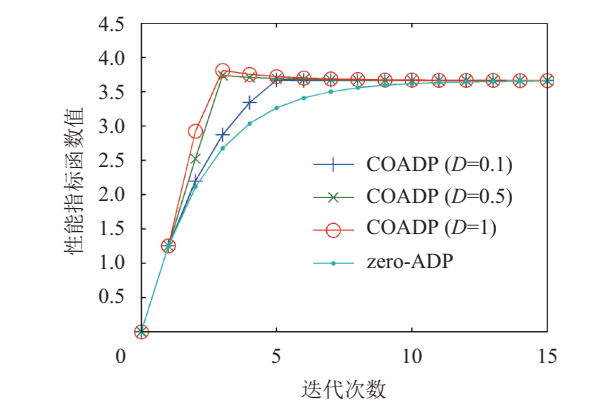


图 6 例2的最优备选迭代性能指标函数收敛过程($A=7$)

Fig.6 Convergence of the best optional iterative performance index function for Example 2 ($A=7$)

例3: 仿射非线性系统为

$$\mathbf{x}_{k+1} = \begin{bmatrix} (\mathbf{x}_{1k}^2 + \mathbf{x}_{2k}^2 + \mathbf{u}_k) \cos(\mathbf{x}_{2k}) \\ (\mathbf{x}_{1k}^2 + \mathbf{x}_{2k}^2 + \mathbf{u}_k) \sin(\mathbf{x}_{2k}) \end{bmatrix}. \quad (15)$$

其中 $\mathbf{x}_k = [\mathbf{x}_{1k}, \mathbf{x}_{2k}]^T \in \mathbb{R}^2, \mathbf{u}_k \in \mathbb{R}$. $\mathbf{x}_k = 0$ 是该系统的稳定状态. 效用函数与例1相同. 初始状态为 $\mathbf{x}_0 = [1, -1]^T$. 所有进程的执行网络均是2-8-1结构, 评判网络均是2-8-1结构. 其他设置与例1相同. 图7、8分别是 $A=3$ 、 $A=7$ 时的 $V_i^{(B_i)}(\mathbf{x}_k)$ 收敛曲线. 对于系统(15), 若 $D=0.1$, 则调整范围过小, COADP算法的效果不明显; 若 $D=1$, 则调整范围合适, 残差快速减少, 但 A 对残差减小的速度影响不大; 若 $D=3$, 则调整范围足够大. 若增大 A 可令残差减小的速度进一步加快, 协同优化效果明显. D 和 A 对残差的影响与例1相同.

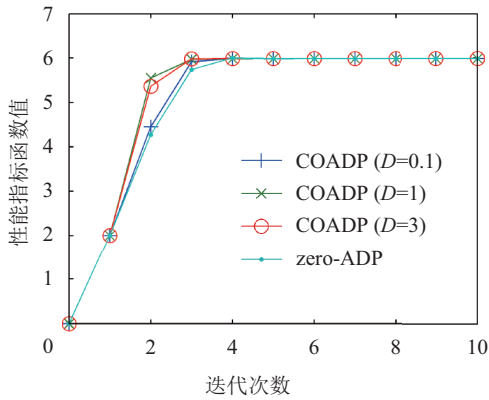


图7 例3的最优迭代性能指标函数收敛过程($A=3$)

Fig.7 Convergence of the best optional iterative performance index function for Example 3 ($A=3$)

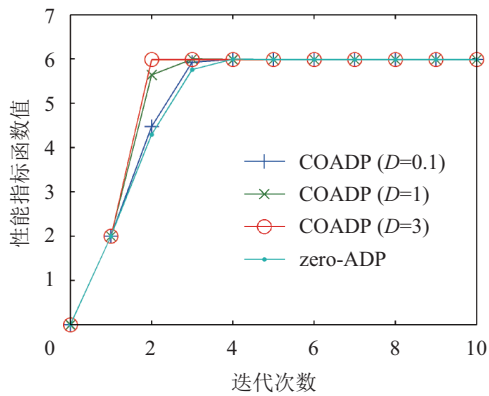


图8 例3的最优迭代性能指标函数收敛过程($A=7$)

Fig.8 Convergence of the best optional iterative performance index function for Example 3 ($A=7$)

以上3个例子的仿真结果表明,如果 D 过小,或者如果 D 过大而 A 过小,则COADP算法的效果与zero-ADP的效果相若,COADP算法不能显著加快残差减小的速度.在实际应用中,对于不同的系统, A 和 D 的恰当取值是难以确定的.为了充分发挥COADP算法的优势,应把 A 和 D 设置为较大的数值.

4 总结

本文研究了一种在有限迭代次数下保证收敛精度的快速自适应动态规划算法.本文提出的协同优化算法能令残差快速减小,令自适应动态规划更为实用.

参考文献:

[1] BELLMAN R E. Dynamic Programming[M]. Princeton: Princeton University Press, 1957.
[2] WERBOS P J. Advanced forecasting methods for global crisis warning and models of intelligence [J]. General Systems Yearbook, 1977, 22(6): 25-38.

[3] WERBOS P J. Approximate Dynamic Programming for Real-time Control and Neural Modeling, in Handbook of Intelligent Control[M]. New York: Van Nostrand Reinhold, 1992.
[4] MILLER W T, SUTTON R S, WERBOS P J. A Menu of Designs for Reinforcement Learning Over Time, in Neural Networks for Control[M]. Cambridge: MIT Press, 1991.
[5] MURRAY J J, COX C J, LENDARIS G G, SAEKS R. Adaptive dynamic programming [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 2002, 32(2): 140-153.
[6] PROKHOROV D V, WUNSCH D C. Adaptive critic designs [J]. IEEE Transactions on Neural Networks, 1997, 8(5): 997-1007.
[7] LEWIS F L, LIU D. Reinforcement learning and adaptive dynamic programming for feedback control [J]. IEEE Circuits & Systems Magazine, 2009, 9(3): 32-50.
[8] LEWIS F L, VRABIE D, VAMVOUDAKIS K G. Reinforcement learning and feedback control: using natural decision methods to design optimal adaptive controllers [J]. IEEE Control Systems, 2012, 32(6): 76-105.
[9] AL-TAMIMI A, LEWIS F L, ABU-KHALAF M. Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2008, 38(4): 943-949.
[10] ZHANG H, WEI Q, LUO Y. A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2008, 38(4): 937-942.
[11] WANG D, LIU D, WEI Q, ZHAO D, JIN N. Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming [J]. Automatica, 2012, 48(8): 1825-1832.
[12] WEI Q, LIU D. A novel iterative θ -adaptive dynamic programming for discrete-time nonlinear systems [J]. IEEE Transactions on Automation Science and Engineering, 2014, 11(4): 1176-1190.
[13] WEI Q, LIU D, LIN H. Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems [J]. IEEE Transactions on Cybernetics, 2016, 46(3): 840-853.
[14] 刘毅, 章云. 基于值迭代的自适应动态规划的收敛条件 [J]. 广东工业大学学报, 2017, 34(5): 10-14.
LIU Y, ZHANG Y. A convergence condition of value-iteration based adaptive dynamic programming [J]. Journal of Guangdong University of Technology, 2017, 34(5): 10-14.
[15] NAVARRO-LOPEZ E M. Local feedback passivation of nonlinear discrete-time systems through the speed-gradient algorithm [J]. Automatica, 2007, 43(7): 1302-1306.
[16] SIRA-RAMIREZ H. Non-linear discrete variable structure systems in quasi-sliding mode [J]. International Journal of Control, 1991, 54(5): 1171-1187.