

基于伴随方程的时滞系统参数辨识框架及其应用^{*}

陈靖天 张丽[†]

(南京航空航天大学 航空航天结构力学及控制全国重点实验室, 南京 210016)

摘要 时滞微分方程广泛应用于描述系统当前状态与过去状态之间的动态关联, 涵盖生物力学、工程学、物理学等多个领域, 尤其适用于含时滞的复杂系统建模. 由于时滞效应能够显著影响系统的动态行为、控制效果和稳定性, 准确识别时滞参数成为动力学系统研究中的核心挑战之一. 为提高辨识精度和效率, 本文提出了一种基于伴随方程和梯度下降算法的时滞微分方程参数辨识算法. 该算法利用伴随方程的解析特性, 通过逆向求解, 精确计算系统响应对参数的梯度, 从而实现高效的参数更新. 本文给出了算法的数学推导, 并基于 Python 平台开发对应计算框架, 使用自动更新, 动态插值的解类并封装了一个支持求解时变参数的时滞微分方程求解器. 利用简化算法和并行计算, 优化求解流程并降低计算复杂度, 增强实际应用中的可操作性. 为了验证算法的有效性, 本文以一个二自由度非线性振子系统为例, 进行了数值仿真实验. 仿真结果表明, 该方法能够准确识别系统的时滞参数.

关键词 时滞微分方程, 参数辨识, 伴随方程, 动力学反问题

中图分类号: O313

文献标志码: A

Adjoint Equation Based Framework and Application for Delayed System Parameter Identification^{*}

Chen Jingtian Zhang Li[†]

(State Key Laboratory of Mechanics and Control for Aerospace Structures, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract Delay differential equations are widely used to describe dynamic connections between system's current states and its past states. They are particularly suited for modelling complex systems with delays, such as dynamic systems arisen from biology, engineering and physics. Since delay effects can significantly impact system dynamic behaviour, control performance and stability, accurate identification of delay parameters has become a core challenge. To improve both accuracy and efficiency, this study proposes a delay parameter identification algorithm based on adjoint equations and the gradient descent method. By leveraging the analytical properties of adjoint equations, this method solves the adjoint system backward in time, allowing for the precise calculation of the gradient of the system response with respect to the parameters, thus facilitating efficient parameter updates. This study details the mathematical derivation of the algorithm and develops a computational framework on the Python platform, incorporating automatic updates, dynamic interpolation, and a solver for delay differential equations with time-varying parameters. By simplifying the algorithm and utilizing parallel computing, the solution

2024-09-30 收到第 1 稿, 2024-10-22 收到修改稿.

^{*} 国家自然科学基金资助项目 (12272167), National Natural Science Foundation of China (12272167).

[†] 通信作者 E-mail: zhangli@nuaa.edu.cn

process is optimized to reduce computational complexity, enhancing its practical applicability. To validate the effectiveness of the algorithm, a numerical identification experiment is conducted for a two-degree-of-freedom nonlinear spring-mass system. The results demonstrate that the method accurately identifies the system delay parameters while maintaining a low error margin.

Key words delay differential equations, parameter identification, adjoint equation, inverse dynamic problem

引言

时滞微分方程(delay differential equations)是一类特殊的泛函微分方程,具有与传统常微分方程(ordinary differential equations)不同的无穷维度、滞后性和继承性特点^[1].它描述了动态系统当前状态与过去状态之间的关系,能够比较完整、真实地反映实际系统的状态.根据系统当前状态与过去状态、未来状态的关联性,时滞微分方程可被归纳为滞后型、中立型、超前型以及混合型方程.

随着时滞微分方程理论的发展,使用该类方程对系统进行建模的方法在动力学研究中被广泛采用,时滞逐渐成为动力学系统建模中不可缺少的重要变量.在物理学、信息科学、化学、工程学、经济学和生物数学等领域,时滞已成为研究重点之一,旨在揭示时滞变化对系统特征所产生的影响^[2-5].对于动力学控制系统而言,时滞所带来的滞后效应将不可避免地影响系统的响应、控制效果和稳定性.忽略时滞常常会导致系统出现较差的力学特性.因此,在面对未知力学系统时,首先需要对目标力学系统进行学习和研究.系统参数辨识作为工程中研究未知系统的第一步,其具有识别系统结构,标定参数大小,分析系统特性,找到关键变量等重要作用,诸多研究均关注了未知系统的参数识别问题^[6-8].然而与一般系统有所不同的是,时滞在实际力学系统中难以直接测量和估计,这对系统的精确建模和仿真构成了巨大的挑战.因此,在动力学系统中开展针对时滞的辨识工作有较高价值.

目前,主流时滞识别方法主要有如下几种:一是采用传统含梯度的优化搜索方法,例如改进最小二乘法^[9]、样条曲线法^[10]等.此类方法将时滞微分方程近似为常微分方程,并将时滞独立表示和计算.二是采用启发式算法,避开对梯度的直接计算,例如遗传算法^[11]、粒子群算法^[12]、贝叶斯优化^[13]

等对全局最优较敏感的方法.此类方法使用代理模型,避免对系统参数直接求梯度,能够在先验知识较少的情况下对未知系统展开直接研究,并且有一定的工程部署潜力.三是利用神经网络的非线性拟合能力和时间序列预测功能进行时滞系统辨识.在建模、预测和控制方面,此类方法也被称作数据驱动的神经网络动力学系统辨识.该方法革新了复杂系统研究的思路.数据驱动的方法对于系统的物理模型和先验知识并不依赖,相反,其通过系统的输入输出信号来识别未知模型.基于稀疏识别方法的时滞微分方程重构^[14],深度神经网络时序预测^[15,16],以及基于神经网络的神经时滞微分方程^[17,18]等都展现了数据驱动方法对未知系统辨识问题的处理能力.

为了更精确地从先验物理模型出发,本文围绕梯度优化搜索方法,采用了一种基于伴随方程^[19]以及梯度下降法的算法对含时滞的动力学系统进行辨识和标定.该方法能够利用参数未知的物理模型,对系统初始条件、各种设定参数以及控制量进行分析,给出当参数数值变化时,系统响应与参数的梯度关系.首先,将描述算法的数学原理,以及搭建在 Python 平台的时滞微分方程识别算法的计算逻辑和框架.辨识算法包含了一个时变时滞微分方程求解器,逆向伴随微分方程求解器,以及一个并行梯度求解器.随后,一个基于二自由度弹簧振子的数值仿真实验将用来检验算法的有效性.

1 伴随梯度法建模

含时滞的动力学系统识别问题可以用一个受微分方程约束的优化问题表示.这种表示方法将引入一个损失函数来描述辨识结果和实际物理模型的差异.通过对差异的一系列数学处理,便能够计算相应参数的梯度.伴随梯度法是通过伴随方程的构建,以避免对微分方程约束的非线性泛函优化问

题直接进行求解,转而通过微分方程在伴随空间的计算,间接给出系统参数的梯度.该方法不限制微分方程的非线性程度,非线性时滞微分方程可以直接进行计算.以单时滞系统为例,以下给出通过伴随方程求解系统参数梯度的过程.

具有单时滞的时滞微分方程可以表示为:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{f}[\mathbf{x}(t), \mathbf{x}(t-\tau), \mathbf{p}], \quad 0 < t \leq T \\ \mathbf{x}(t) &= \mathbf{x}_0(t), \quad -\tau \leq t \leq 0\end{aligned}\quad (1)$$

其中 $\mathbf{x} \in \mathbb{R}^n$ 是系统的状态变量, $\mathbf{f}(\cdot)$ 是一个非线性函数, τ 是系统的常时滞, T 表示系统的终值时刻, \mathbf{x}_0 表示系统的初始条件函数, \mathbf{p} 表示系统除时滞以外的参数, $\mathbf{p} = [p_1, p_2, \dots, p_m]^T \in \mathbb{R}^m$. 假设用一组识别参数生成的响应记为 $\hat{\mathbf{x}}$, 通过其与系统真实响应 \mathbf{x} 之差的内积可以构造出一个损失函数 J . 使用该损失函数构建的如下优化问题能够用于描述系统参数与时滞的辨识任务:

$$\begin{aligned}\operatorname{argmin}_{\mathbf{p}, \tau} \quad & J(\hat{\mathbf{x}}, \tau, \mathbf{p}) \\ &= \int_0^T \langle \hat{\mathbf{x}}(t) - \mathbf{x}(t), \dot{\hat{\mathbf{x}}}(t) - \dot{\mathbf{x}}(t) \rangle dt \\ \text{sub.} \quad & \dot{\hat{\mathbf{x}}}(t) = \mathbf{f}[\hat{\mathbf{x}}(t), \hat{\mathbf{x}}(t-\tau), \mathbf{p}] \\ & \hat{\mathbf{x}}(t) = \hat{\mathbf{x}}_0(t, \mathbf{p})\end{aligned}\quad (2)$$

与一般优化问题不同的是,公式(2)需要在优化的同时处理时滞微分方程约束.对此泛函优化问题,引入伴随方程和伴随向量,通过伴随空间可直接求解关于系统参数的梯度.

1.1 伴随方程

对于公式(1)中的时滞微分方程,利用泛函优化中的常用工具拉格朗日恒等式(Lagrange equality),可以得到一个新的微分方程,该微分方程以 $\mathbf{w}(t) \in \mathbb{R}^n$ 为状态变量,以 $-\tau$ 为时滞.该方程的表达式如下所示:

$$\begin{aligned}\frac{d\mathbf{w}(t)}{dt} &= -\frac{\partial \mathbf{f}^T(t)}{\partial \dot{\mathbf{x}}} \mathbf{w}(t) - \\ & \frac{\partial \mathbf{f}^T(t+\tau)}{\partial \dot{\mathbf{x}}} \mathbf{w}(t+\tau), \quad t \leq T\end{aligned}\quad (3)$$

该微分方程与公式(1),在形式和时滞表达上完全匹配.公式(3)以 $-\tau$ 为时滞,是一个反向的时滞微分方程,描述了一个终值问题,并沿着时间轴逆向进行演化;公式(1)是一个正向的时滞微分方程,描述了一个初值问题,沿着时间轴正向进行演化.因此,公式(3)又被称作原时滞微分方程的伴随

方程. $\mathbf{w}(t)$ 也被称为伴随变量,可参与并简化系统参数梯度的计算.由于时滞的存在以及方程(3)对系统未知参数的依赖,很难提前给出 $\mathbf{w}(t)$ 的稳定性.因此,直接通过方程(3)的形式与数值积分方法在给定的边界条件以及有限的时间区间 $[0, T]$ 上求解 $\mathbf{w}(t)$,得到的数值解向量即可用于计算下面一节所描述的梯度表达式.

1.2 梯度表达式

为了求解系统响应对系统参数与时滞的梯度,需要对时滞微分方程进行变分:

$$\delta \dot{\mathbf{x}}'(t) = \frac{\partial \mathbf{f}}{\partial \dot{\mathbf{x}}} \delta \dot{\mathbf{x}}(t) + \frac{\partial \mathbf{f}}{\partial \dot{\mathbf{x}}_\tau} \delta \dot{\mathbf{x}}_\tau + \frac{\partial \mathbf{f}}{\partial \mathbf{p}} \delta \mathbf{p}(t) + \frac{\partial \mathbf{f}}{\partial \dot{\mathbf{x}}_\tau} \frac{\partial \dot{\mathbf{x}}_\tau}{\partial \tau} \delta \tau \quad (4)$$

其中 $\dot{\mathbf{x}}_\tau$ 为 $\dot{\mathbf{x}}(t-\tau)$ 的简写.将该方程两边同时左乘 $\mathbf{w}_i^T(t)$, 其中 $i=1, 2, \dots, n$, 并对时间 t 在区间 $[0, T]$ 上积分,利用分步积分法分离方程的左端项.此外,令 $\mathbf{w}_i(T) = [0, \dots, 0, 1_{i-th}, 0, \dots, 0]^T$, 代入方程中可得 $\mathbf{w}_i^T(T) \delta \mathbf{x}(T) = \delta x_i(T)$. 公式(4)在上述变换下可以得到如下结果:

$$\begin{aligned}\delta x_i(T) &= \mathbf{w}_i^T(0) \delta \mathbf{x}(0) + \int_0^T [\mathbf{w}_i'(t) + \frac{\partial \mathbf{f}^T}{\partial \dot{\mathbf{x}}} \mathbf{w}_i(t) + \\ & \frac{\partial \mathbf{f}^T(t+\tau)}{\partial \dot{\mathbf{x}}} \mathbf{w}_i(t+\tau)]^T \delta \dot{\mathbf{x}}(t) dt + \int_0^T \mathbf{w}_i^T(t) \cdot \\ & \frac{\partial \mathbf{f}}{\partial \mathbf{p}} \delta \mathbf{p} dt - \int_{-\tau}^{T-\tau} \mathbf{w}_i^T(t+\tau) \frac{\partial \mathbf{f}(t+\tau)}{\partial \dot{\mathbf{x}}} \frac{d\dot{\mathbf{x}}(t)}{dt} \delta \tau dt\end{aligned}\quad (5)$$

可以清楚地看到公式(5)中出现了伴随方程的表达式,将其消去后,令 $\delta \mathbf{p}$ 和 $\delta \tau$ 趋向于 0, 则可以获得系统参数与时滞在时滞微分方程约束下的灵敏度,即系统状态分别对系统参数及对应时滞的偏导数:

$$\begin{aligned}\frac{\partial \dot{x}_i(T)}{\partial \mathbf{p}} &= \int_0^T \mathbf{w}_i^T(t) \frac{\partial \mathbf{f}}{\partial \mathbf{p}} dt, \\ \frac{\partial \dot{x}_i(T)}{\partial \tau} &= - \int_{-\tau}^{T-\tau} \mathbf{w}_i^T(t+\tau) \frac{\partial \mathbf{f}(t+\tau)}{\partial \dot{\mathbf{x}}} \frac{d\dot{\mathbf{x}}(t)}{dt} dt\end{aligned}\quad (6)$$

上述表达式利用伴随变量 $\mathbf{w}_i(t)$ 给出了系统参数和时滞与系统响应间的关系.利用公式(2)中的损失函数 J 和公式(5),并在时间上进行离散,便可将损失函数关于系统参数与时滞的梯度表示为如下:

$$\frac{\partial J}{\partial \mathbf{p}} = 2\Delta t \sum_{j=0}^{N-1} \sum_{i=1}^n [\dot{x}_i(t_j) - x_i(t_j)] \frac{\partial \dot{x}_i(t_j)}{\partial \mathbf{p}},$$

$$\frac{\partial J}{\partial \tau} = 2\Delta t \sum_{j=0}^{N-1} \sum_{i=1}^n [\dot{x}_i(t_j) - x_i(t_j)] \frac{\partial \dot{x}_i(t_j)}{\partial \tau} \quad (7)$$

其中, Δt 为离散的时间步长, N 为总采样数据个数. 只要获得公式(3)和(6)中所有矩阵的数据, 在给定系统响应情况下, 优化一个未知的时滞微分方程便成为可能. 需要说明的是, 上述推导可以推广应用于多参数多时滞模型, 多参数多时滞优化功能已经集成到下文描述的优化框架内.

2 Python 计算框架构建

在 Python 平台上, 搭建一个针对上述优化问题的计算框架. 该计算框架的目的是快速完成关于时滞微分方程优化任务的数值计算. 该框架分三个子模块: 一个是正反向时滞微分方程的数值积分模块, 并集成了含时变参数的方程求解功能; 一个是并行的梯度解算器; 一个是梯度优化器. 接下来将着重说明三个子模块的构建方法和计算逻辑.

2.1 微分方程求解器

时滞微分方程求解器是整个优化算法功能的支柱. 在算法中, 两个时滞微分方程需要串行求解, 其中在求解伴随时滞微分方程时, 求解器需要接受时变参数, 并且沿时间轴逆向求解. 此外, 时滞微分方程的求解需要回溯系统的历史响应, 且每一时间步计算均需要回溯. 需要回溯的历史数据所对应的时间节点并不一定是已经记录在内存中的节点, 为了保证回溯正常进行, 需要实现对任意时刻系统解数据的索取功能.

使用类 V 来表示时滞微分方程的解. 该类通过时滞微分方程的初始条件进行实例化, 有两个便于时滞微分方程求解的功能: 记录并更新时滞微分方程求解器返回的某时间步系统状态变量数据; 对给定时间步的数据进行插值索取. 这两个功能均通过三次 Hermite 插值实现. 以 n 点插值为例, 给出获得任意时间步插值数据的数学表达. 设 $\hat{x}_i \in C^1[a, b]$ 且 $t_0^i, t_1^i, t_2^i, \dots, t_n^i \in [a, b]$ 为 \hat{x}_i 已经求解的时间步, 那么对 \hat{x}_i 进行三次 Hermite 插值可以得到:

$$\begin{aligned} \hat{x}_i(t) &= \sum_{j=0}^n \hat{x}_i(t_j^i) H_{n,j}(t) + \sum_{j=0}^n \dot{\hat{x}}_i(t_j^i) \hat{H}_{n,j}(t), \\ H_{n,j}(t) &= [1 - 2(t - t_j^i) L'_{n,j}(t_j^i)] L_{n,j}^2(t), \\ \hat{H}_{n,j}(t) &= (t - t_j^i) L_{n,j}^2(t), \end{aligned} \quad (8)$$

其中, $L_{n,j}(t)$ 函数为第 j 个 n 次拉格朗日多项式基函数, $(\cdot)'$ 表示时间导数. 此插值能够较准确给出任意时刻时滞微分方程的解, 因为其兼顾了给定点的数据及其导数大小, 匹配了时滞微分方程对其解 \hat{x} 和 \hat{x}' 的约束.

如图 1 所示, 该类在计算时接受求解器返回的数据, 并用一个 $V.update()$ 函数将其保存, 在接受时间并给出历史数据时, 先以 t 为中心, 在其左右共取 n 个点作为插值采样点, 返回插值结果 $\hat{x}_i(t)$. 在计算时, 为了保证计算速度, 数据记录步长 Δt 取 0.01 s, 而 n 则取为 50. 这样动态保留导数信息的插值无需在每个时间步积分时遍历所有已经求解并保存的数据, 并且在精度上不弱于全局插值算法. 如此能够减少每次前向积分时回溯历史值所需要的插值时间, 因为每次仅需对固定大小的数据长度进行插值即可.

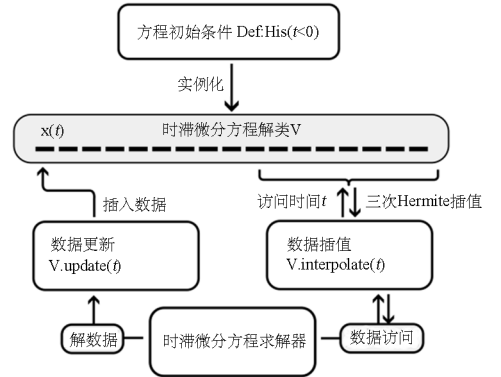


图 1 时滞微分方程解类的构建

Fig. 1 Construction of class for time-delay differential equations

图 2 给出的是含时变参数的时滞微分方程求解器架构. 该求解器使用了一个完整的可变参数常微分方程积分器 (VODE), 并对其进行了二次封装. 为了将时滞微分方程转化为常微分方程, 时滞项被视作为一个动态变量, 其随着求解时间 t 的变化返回 $\hat{x}(t - \tau)$ 的值. 当求解伴随时滞微分方程时, 其中的一些时变参数矩阵如 $\partial f^T(t)/\partial \hat{x}$ 则被封装成函数的形式, 等待调用.

两个类 V 的对象 \hat{x} 和 \hat{x}' 被初始化并赋予初始条件. \hat{x}' 的导数信息被送入 \hat{x} 完成 Hermite 插值. 待求解的方程被写入求解器中并使用动态步长进行求解, 按 Δt 记录求解的数据并回送 \hat{x} 和 \hat{x}' , \hat{x} 记录系统状态变量的解, \hat{x}' 记录解的导数信息. 求解器在积分时回溯 \hat{x} 在 $t - \tau$ 时的值, 回溯时完成三次 Hermite 插值操作并继续送入求解器进行求解. 求解伴随时滞微分方程时, 进行的计算步骤与上述

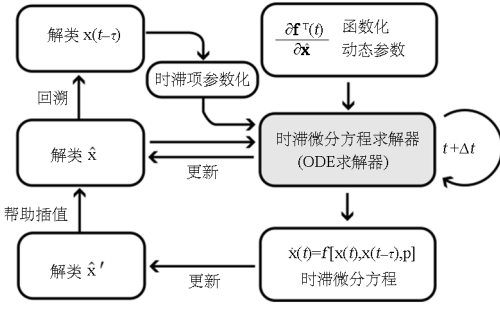


图2 时滞微分方程求解器流程

Fig. 2 Time delay differential equations solver flow

相同,只需将时间步长取为 $-\Delta t$,此时 \hat{x} 和 \hat{x}' 以及求解器都会反向进行计算。

2.2 并行梯度解算

对时滞微分方程和伴随时滞微分方程的求解结束后,紧接着需要对数据进行组合并按照公式(6)和(7)来计算系统参数和时滞的梯度.设公式(6)中的 $T=t_j$,并让 t_j 遍历区间 $[0, T]$.为了计算 $\partial \hat{x}_i(t_j)/\partial p$ 和 $\partial \hat{x}_i(t_j)/\partial \tau$,需要预先计算 $w_i(t)$, $t \in [0, t_j]$,而不同的 t_j 代表伴随方程的终值时刻需要反复调整.因此,上述过程需要进行 $2 \times N \times n$ 次数值积分,这显著地增加了计算的代价.事实上,对于相同维度的伴随方程,在 $t_j \in [0, T]$ 时,其终值条件均相同: $w_i(t_j) = [0, \dots, 0, 1_{i-th}, 0, \dots, 0]^T$.相同的终值条件使得不同 t_j 计算得到的解实则是 $w_i(T)$ 的一部分,对于 $t_j \in [0, T]$,可以得到 $w_i(t) = w_i(T - t_j + t)$ 的等式关系.利用此关系进行计算实际上减少了 $N - 1$ 次独立对伴随微分方程的数值积分.最终,梯度表达式在对公式(6)的调整下被简化为:

$$\begin{aligned} \frac{\partial \hat{x}_i(t_j)}{\partial p} &= \int_0^{t_j} w_i^T(T - t_j + t) \frac{\partial f}{\partial p} dt, \\ \frac{\partial \hat{x}_i(t_j)}{\partial \tau} &= - \int_{-\tau}^{T-\tau} w_i^T(T - t_j + t + \tau) \times \\ &\quad \frac{\partial f(t + \tau)}{\partial \hat{x}} \frac{d\hat{x}(t)}{dt} dt \end{aligned} \quad (9)$$

然而,对于公式(9),仍需要对其进行积分操作.在数值计算中 w_i 作为一个时滞微分方程解类的一个实例,其可以被视为一个函数,可以在给定的时间区间内返回一个连续的函数值.如果利用多项式插值进行数值积分,时间计算成本较大,因为其需要对 w_i 内记录的离散数组反复回溯.因此,算法采用了并行的离散数值积分。

每个维度所对应的 w_i 被 i 个进程并行求解,求得的数据 w_i 将被送入公式(9)等待积分计算.按时间间隔 Δt 离散 $[0, t_j]$ 时间区间,并从 w_i 中取得待积分数据.需要注意的是,按照待 Δt 离散 $[0, t_j]$ 时间区间对应的 $w_i(t)$ 刚好是存储在实例中的离散数组,免去了调用时的插值过程.积分数据则被存储为一个三维矩阵,其中一维对应状态向量的维度,一维对应数据自身长度,一维对应积分的时间上限.这样存储的目的是将每一离散时刻需要积分的数据按时间和维度排列,以内存占用换取较短的计算时间,避免在积分时串行循环调用数值积分方法.最终,数据被送入 $n \times N$ 个并行进程中同时进行离散 Simpson 积分,最终得到系统状态分别对系统参数及对时滞的偏导数并代入求解公式(7)。

2.3 基于梯度的优化

当梯度求解完毕后,算法使用最常见的自适应矩估计算法(adaptive moment estimation, Adam)梯度优化器来对系统参数和时滞进行优化^[21].Adam算法通过计算一阶和二阶矩以获得动态梯度序列,从而实现快速稳定的优化过程.该方法首先计算梯度的一阶矩估计 m_t ,考虑过去的梯度,得到梯度的移动平均值.随后计算梯度的二阶矩估计 v_t ,考虑过去梯度的平方,得到平方梯度的移动平均值.利用上述的一阶矩和二阶矩进行偏差修正.其参数更新策略如下:

$$\begin{aligned} g_t &= \frac{\partial J}{\partial p_i} \\ m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \end{aligned} \quad (10)$$

其中 g_t 表示参数和时滞的梯度, η 表示优化的学习率, t 表示某个离散的特定时刻.为了增强优化器的收敛速度和泛化能力,需要对优化器中的参数根据不同辨识任务进行人为指定.在本文动力学系统辨识仿真算例中,公式(10)中的三个参数分别指定为 $\beta_1 = 0.75, \beta_2 = 0.999, \epsilon = 10^{-8}$ 。

3 数值计算案例

考虑一个二自由度质量—弹簧振子系统^[9],如图 3 所示,两个质量块 m_1 和 m_2 由一个三次非线性弹簧和线性阻尼 c_1 连接,其中三次非线性弹簧的线性刚度系数为 k_1 ,非线性刚度系数为 μ_1 、 m_2 . 通过线性弹簧 k_2 和线性阻尼 c_2 与地面连接. 在质量块 m_1 和 m_2 上分别有一个时滞微分控制器 g_1 和 g_2 控制弹簧振子系统的振动,用 p_1 、 p_2 、 p_3 、 p_4 表示控制器增益,控制器 g_1 和 g_2 的表达式为:

$$\begin{aligned} g_1 &= p_1 \dot{x}_1(t-\tau) + p_2 \dot{x}_2(t-\tau) \\ g_2 &= p_3 \dot{x}_1(t-\tau) + p_4 \dot{x}_2(t-\tau) \end{aligned} \quad (11)$$

取系统参数为 $m_1=0.25$ kg, $m_2=0.37$ kg, $k_1=1.2\pi^2$ N/m, $k_2=0.8\pi^2$ N/m, $\mu_1=2$ N/m³, $c_1=0.02\pi$ N·s/m, $c_2=0.08\pi$ N·s/m. 系统状态方程如下所示:

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ A_1 \\ A_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ p_1 \dot{x}_1(t-\tau) + p_2 \dot{x}_2(t-\tau) \\ p_3 \dot{x}_1(t-\tau) + p_4 \dot{x}_2(t-\tau) \end{bmatrix}$$

$$A_1 = 7.3741(-x_1 + x_2) + 8(-x_1 + x_2)^2 + 0.251327(-\dot{x}_1 + \dot{x}_2)$$

$$A_2 = 32.0095(x_1 - x_2) + 5.4054(x_1 - x_2)^3 - 21.3397x_2 + 0.17(\dot{x}_1 - \dot{x}_2) - 0.679\dot{x}_2 \quad (12)$$

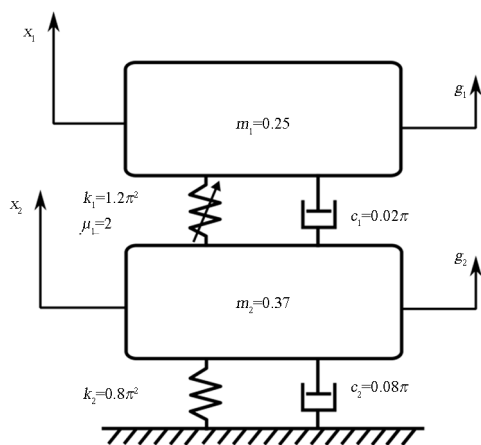


图 3 质量—弹簧振子系统示意图

Fig. 3 Schematic diagram of a mass-spring oscillator

利用上文描述的辨识算法对该系统展开参数和时滞辨识. 在辨识前,假设方程(12)的初始条件为定常值,在 $t \leq 0$ 时, x_1 和 x_2 的值均为 0.1 m,而 \dot{x}_1 和 \dot{x}_2 的值均为 0 m/s,且均不随着时间变化而改变. 辨识该系统的主要目的是从响应中得到系统

时滞以及反馈参数的数值. 设系统时滞的真值 τ 为 0.5 s,而方程中表示控制器的四个参数被设为: $p_1=0.14$, $p_2=0.09$, $p_3=0.07$, $p_4=-0.11$,单位均为 1/s. 优化的初始时滞 τ 为 0.75 s,反馈参数的初始条件与真实值存在较大偏差,如下所示: $p_1=0.20$, $p_2=0.15$, $p_3=0.15$, $p_4=-0.05$,单位均为 1/s. 利用上述初始条件和上文的参数辨识算法迭代 200 次来寻找系统的真实参数. 从图 4 可以看出,该识别算法能够给出很好的识别效果,识别系统的响应和真实系统响应的均方误差仅为 2.43×10^{-5} . 图中虚线代表参数的真实大小,实线表示的是随着迭代进行,参数辨识的结果. 通过观察辨识迭代过程,可以发现算法能够准确识别时滞和系统参数的大小. 此外,该实例还表明算法能够处理非线性问题. 考虑到公式(12)中同时存在二阶以及三阶的非线性项,识别过程中参数和时滞的数值尽管有所波动,但结果仍非常理想. 该数值仿真实验验证了其对于含非线性以及时滞反馈的系统的识别能力.

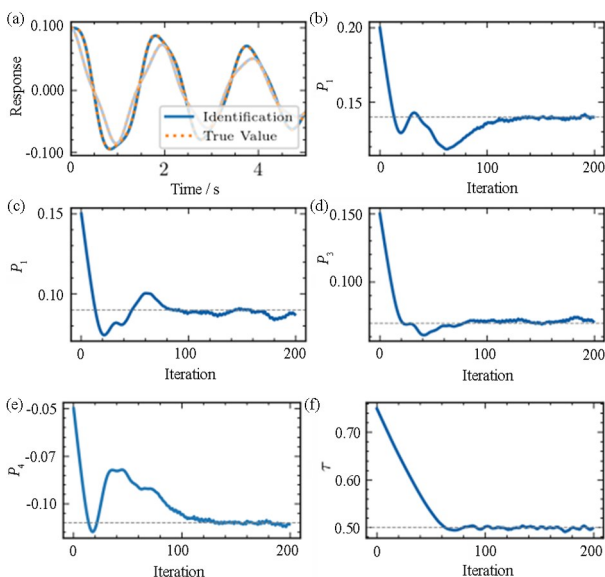


图 4 质量—弹簧振子系统识别结果

Fig. 4 Identification results of the mass-spring oscillator

此外,在上述实验的基础上,考虑了噪声对该计算框架辨识效果的影响. 在上述数值仿真得到的响应中,加入信噪比(SNR)分别为 20 dB、30 dB、40 dB 的白噪声. 表 1 中给出了在四种不同信噪比下的识别结果. 在无噪声的情况下辨识结果与参考值几乎一致,随着噪声信号的增加,除了时滞以外的参数均有偏离参考值的趋势,但误差仍然可控. 时滞参数随噪声强度的波动变化不大. 随着噪声强

度增大,识别准确度降低,这是由于本文算法是严格的时域辨识算法,对时域信号较为敏感,时域的信号噪声会一定程度影响梯度计算中所需数值积分的准确度.

表 1 不同信噪比下的识别结果
Table 1 Identification results with different signal-to-noise ratios

SNR	$p_1/(1/s)$	$p_2/(1/s)$	$p_3/(1/s)$	$p_4/(1/s)$	τ/s
No noise	0.140	0.090	0.070	-0.109	0.500
40	0.140	0.089	0.069	-0.109	0.500
30	0.139	0.098	0.059	-0.099	0.500
20	0.178	0.127	0.047	-0.157	0.489

4 结论

本文基于伴随方程与梯度下降算法对时滞微分方程展开了参数辨识.通过几个关键的计算步骤展示了该算法的数学逻辑.此外还详细介绍了一个基于 Python 平台的辨识框架.其中,详细说明了数据存储方法、求解器搭建方法和计算流程,并利用简化手段降低了参数识别的计算复杂度.该算法平台计算逻辑简单,调用额外接口少,便于使用,可以对任意时滞微分方程的未知参数展开相应的辨识工作.为了验证算法的准确程度,本文辨识了一个二自由度振子系统的参数和其控制器的时滞,验证了算法的准确性,并分析了含有噪声情况下该算法的准确度.该算法给予了研究者对含时滞未知系统进行初步辨识的能力,具有实际应用潜力,未来将在实际工程力学模型中进一步验证其有效性.

参考文献

[1] HU H Y, WANG Z H. Dynamics of controlled mechanical systems with delayed feedback [M]. Berlin; Springer Berlin, 2002.

[2] KOLOMEITSEV E E, VOSKRESENSKY D N. Time delays and advances in classical and quantum systems [J]. Journal of Physics G: Nuclear and Particle Physics, 2013, 40(11): 113101.

[3] STEPAN G. Delay effects in the human sensory system during balancing [J]. Philosophical Transactions Series A, Mathematical, Physical, and Engineering Sciences, 2009, 367(1891): 1195-1212.

[4] LI Y L, JOHANSSON R, YIN Y X. Acceleration

feedback control for nonlinear teleoperation systems with time delays [J]. International Journal of Control, 2015, 88(3): 507-516.

[5] FANELLI V, MADDALENA L. A time delay model for the diffusion of a new technology [J]. Nonlinear Analysis: Real World Applications, 2012, 13(2): 643-649.

[6] 常迎香, 王淑英, 李险峰, 等. 一个新混沌系统的参数辨识[J]. 动力学与控制学报, 2009, 7(3): 235-238.

CHANG Y X, WANG S Y, LI X F, et al. Parameter identification of a new chaotic system [J]. Journal of Dynamics and Control, 2009, 7(3): 235-238. (in Chinese)

[7] 刘昊鹏, 朱云鹏, 罗忠, 等. 多自由度非线性系统动态参数化模型建模方法研究[J]. 动力学与控制学报, 2017, 15(1): 15-24.

LIU H P, ZHU Y P, LUO Z, et al. Modeling method on dynamic parametrical model of nonlinear multi-degree of freedom systems [J]. Journal of Dynamics and Control, 2017, 15(1): 15-24. (in Chinese)

[8] 郑罡, 曹和生, 杜宗松, 等. Winkler 地基梁动力学系统的无量纲化与参数识别[J]. 动力学与控制学报, 2024, 22(1): 60-68.

ZHENG G, CAO H S, DU Z S, et al. Dimensionless and parameter identification of dynamic system of beam on Winkler foundation [J]. Journal of Dynamics and Control, 2024, 22(1): 60-68. (in Chinese)

[9] ZHANG X X, XU J. Identification of time delay in nonlinear systems with delayed feedback control [J]. Journal of the Franklin Institute, 2015, 352(8): 2987-2998.

[10] BANKS H T, DANIEL LAMM P K. Estimation of delays and other parameters in nonlinear functional differential equations [J]. SIAM Journal on Control and Optimization, 1983, 21(6): 895-915.

[11] JAMILLA C U, MENDOZA R G, MENDOZA V M P. Parameter estimation in neutral delay differential equations using genetic algorithm with multi-parent crossover [J]. IEEE Access, 2021, 9: 131348-131364.

[12] MODARES H, ALFI A, FATEH M M. Parameter identification of chaotic dynamic systems through an improved particle swarm optimization [J]. Expert Systems with Applications, 2010, 37(5): 3714-

- 3720.
- [13] SHEIBANI M, OU G. A Bayesian optimized framework for successful application of unscented Kalman filter in parameter identification of MDOF structures [J]. *Journal of Low Frequency Noise, Vibration and Active Control*, 2021, 40(4): 1712—1730.
- [14] LEYLAZ G, WANG S, SUN J Q. Identification of nonlinear dynamical systems with time delay [J]. *International Journal of Dynamics and Control*, 2022, 10(1): 13—24.
- [15] OPREA M, WALTH M, STEPHANY R, et al. Learning the delay using neural delay differential equations [EB/OL]. (2023-04-03)[2024-09-30]. <https://arxiv.org/abs/2304.01329v2>.
- [16] VINODBHAI C D, DUBEY S. Numerical solution of neutral delay differential equations using orthogonal neural network [J]. *Scientific Reports*, 2023, 13: 3164.
- [17] ZHU Q X, GUO Y, LIN W, et al. Neural delay differential equations: system reconstruction and image classification [EB/OL]. (2023-04-11)[2024-09-30]. <https://arxiv.org/abs/2304.05310v1>.
- [18] ZHU Q X, SHEN Y F, LI D S, et al. Neural piecewise-constant delay differential equations [J]. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, 36(8): 9242—9250.
- [19] MARCHUK G I, SHUTYAEV V, BOCHAROV G. Adjoint equations and analysis of complex systems: Application to virus infection modelling [J]. *Journal of Computational and Applied Mathematics*, 2005, 184(1): 177—204.
- [20] MARCHUK G I. Adjoint equations and analysis of complex systems [M]. Dordrecht, Nertherlands: Kluwer Academic Publishers, 1995.
- [21] KINGMA D P. Adam: A method for stochastic optimization [EB/OL]. (2014-12-22)[2024-09-30]. <https://arxiv.org/abs/1412.6980>.