混沌时间序列的遗传演化建模

张 伟, 吴智铭, 杨根科
(上海交通大学自动化系, 上海 200030)

摘 要: 从实测时间序列中构造混沌系统的模型是非线性时间序列分析中的一个重要议题, 本文利用遗传编程方法(GP), 在尽可能大的函数空间内搜索合适的模型结构, 并引入粒子群算法(PSO)在动态模型结构情况下进行非线性参数估计, 取得了较好效果. 此外, 演化建模的实现结合了非线性时间序列分析(NTSA)的结果, 以 NTSA 的结果指导演化建模参数的选取并作为模型优劣的评判标准, 改进了经典 GP 算法对混沌系统建模的应用效果.

关键词: 混沌时间序列分析; 遗传演化建模; 非线性参数估计; 粒子群算法; 非线性系统辨识
中图分类号: O231.3 文献标识码: A 文章编号: 0372-2112 (2005) 04-0748-04

Genetic Programming Modeling On Chaotic Time Series

ZHANG Wei, WU Zhi-ming, YANG Gen-ke
(*Dept. of Automation, Shanghai Jiaotong University, Shanghai* 200030, *China*)

Abstract: This paper proposes Genetic Programming Modeling (GPM) algorithm on chaotic time series. GP is used here to search for appropriate model structures in function space, and Particle Swarm Optimization (PSO) algorithm is introduced for Nonlinear Parameter Estimation (NPE) of dynamic model structures. In addition, GPM integrates the results from Nonlinear Time Series Analysis (NTSA) to adjust the parameters and as the criterion of founded models. The simulation shows the effectiveness of such improvements on modeling chaotic time series.

Key words: chaotic time series analysis; genetic programming modeling; nonlinear parameter estimation; particle swarm optimization; nonlinear system identification

1 引言

许多系统的外在复杂行为都可由内在的简单混沌规律得到解释[1], 但如何寻求这些简单的规律却一直是非线性系统辨识与建模的难点, 究其根源在于混沌系统的耗散性和初值敏感性特点[2]. 神经网络[2]与多项式[3]是两种常用的全局建模工具, 但是, 这些全局建模方法大都无法给出简明直观的模型表达形式, 特别是在处理未知混沌系统数据的情况下带有很大盲目性, 难以结合已有的非线性分析结果和相关经验. 本文引入遗传编程(GP)方法对混沌时间序列建模, 通过 GP 在由指定函数算子复合形成的函数空间中搜索能够尽可能反映系统行为的模型. 与一般遗传演化建模[4]不同的是, 本文将遗传演化建模(GPM)与非线性时间序列分析(NTSA)结合起来, 以NTSA 的结果指导 GPM 的运行, 并且引入混沌系统的不变特征量而不是单纯的拟合精度作为衡量模型质量的标准, 从而改善了 GPM 的建模效果. 此外, 基于粒子群算法(PSO)的动态非线性模型参数估计也降低了 GPM 的运算量; 改进的模型输出计算方法亦使 GPM 可以应用在较大数据量场合, 并更加适应混沌系统建模的要求.

本文如下组织, 第 2 节介绍遗传演化建模 GPM 的框架, 第 3 节介绍 GPM 的基本原理和实现, 第 4 节以 Logistic 映射和 Chebyshev 映射为例, 考察了 GPM 建模的效果, 并给出了一些应用 GPM 的经验, 最后为总结.

2 遗传演化建模(GPM)的框架

GPM 系统采用 C++ 和 Matlab 混合编程实现. 其流程框架如图 1 所示.

首先应对实测时间序列数据进行适当地预处理, 可用小波方法适当降噪并滤除其中可能的非平稳趋势, 过多的噪音易导致 GPM 建模结果发散. NTSA 模块负责对实测数据进行分析, 获取系统的嵌入维数 m、重构延迟 τ、最大 *Lyapunov* 指数、最大可预测区间 L 等信息以指导后继的建模过程. GP 负责搜索函数空间以获取比较好的模型结构, 非线性参数估计 NPE 负责确定模型参数. 在 GPM 中, 通常我们可以得到系统模型的一组备选描述, 因此有必要人工地对这些模型进行评估和筛选以得到最终结果.

3 遗传演化建模的基本原理和实现描述

系统建模的两个基本问题是模型结构的选择和模型参数的估计. 通常的建模过程总是事先假定某种模型结构然后进

收稿日期: 2003-08-25; 修回日期: 2004-12-06
基金项目: 国家自然科学基金(No. 60174009, No. 70071017)

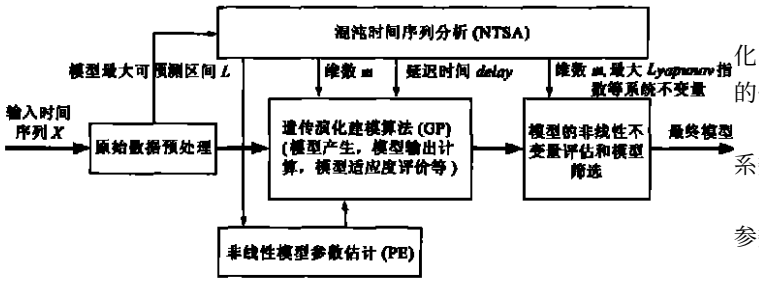

图 1 遗传演化建模 (GPM) 的框架

行细化和参数估计, 如 ARMA 建模和多项式建模[3]. 这种方法对混沌系统通常是失效的, 因为混沌系统的复杂表现往往受简单的低维确定性运动规律所支配, 并且常常包含正反馈环节、切换环节、延迟环节或相互作用环节等. 这导致上述通用模型结构在表述混沌系统时效果反而不好. 而且, 在数据量增大情况下经常出现模型结构发散和复杂化的趋势. 因此, 为了更好的对混沌系统建模, 有必要在更大的函数空间中对模型结构进行搜索, 这在 GPM 中是通过 GP 实现的.

模型结构初步确定之后, 进一步的问题就是如何估计模型参数. 由于 GPM 中模型结构是动态产生的并且常常具有非线性结构, 通常的参数估计算法难以应用, 所以本文引入粒子群算法(PSO)[5,6]进行参数估计. 相比其他算法如 GA, PSO 具有运算量小, 可调参数较少的优点.

3.1　遗传编程算法(GP)

GP 负责搜索合适的模型结构, 它以树状结构表示个体[7], 每一棵树的中序遍历形式即对应着一种模型结构, 其中, 树叶结点选自终结符集合 *TS*, 树中结点取自函数算子集合 *FS*. 在 GPM 的实现中, $TS_{\max}=\{\text{CONSTANT}, \boldsymbol{X}\}$, $FS_{\max}=\{add, multiply, sub, divide, cos, sin, acos, exp\}$. 确定模型结构的过程等效于在函数空间 *F*(模型族)中搜索的过程, *F* 可由 *FS* 中的基本函数算子经过有限次运算和复合得到, 所以 GP 所能搜索的最大模型空间即由 *FS* 的闭包 *span*(*FS*) 确定. 根据对模型的先验知识, 可以仅选取最能表达模型规律的算子参与建模, 即实际的 *TS* 和 *FS* 是上述最大集合的子集. 有人引入 *z* 算子(延迟算子)[8,9], 借助 *z* 算子提升模型的维数以增强模型的表述力, 但事实上, 这个维数是可以借助 NTSA 分析出来的.

遗传算子设计如下[4]:

(1)初始化算子: 即模型树的随机生成过程, 初始群体的一半按广度优先原则生成, 另外一半则按深度优先原则生成, 后者生成的个体包含所有 *FS* 中的函数算子.

(2)交叉算子: 随机地在两个父体中选择杂交点, 然后以杂交点为根结点, 交换相应子树得到新的个体.

(3)变异算子: 随机地在父体中选择某个结点为变异点, 若变异点是非叶结点, 则等概率执行下述操作: (a) 删除以该变异点为根结点的子树并以终结符替代; (b) 删除以该变异点为根结点的子树, 并且在变异点插入一棵随机生成的新子树; 若变异点是叶结点, 则随机的从 *TS* 中选择其他终结符替代当前符号.

3.2　模型的非线性参数估计(NPE)算法

为避免好的模型结构因参数不合理而导致在演化中消亡, GPM 采取模型结构搜索与参数估计相分离的做法. 模型参数按照如下规则生成:

(1) 为每一个函数算子和终结符算子附加一个乘系数作为模型参数;

(2) 若终结符为 CONSTANT, 则直接作为一个模型参数进入模型表达式, 不需附加系数;

(3) 若函数算子为线性算子, 也不需附加系数, 因为线性算子乘系数的功能完全可由其自变量前的系数完成.

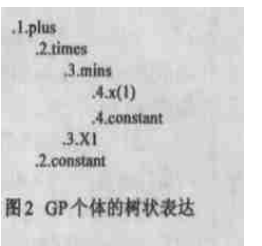

图2 GP个体的树状表达

以图 2 所示个体为例, 包含待评估参数的模型表达式如下所示, 其中 $\boldsymbol{X}(k)$ 表示将时间序列输入 $\boldsymbol{X}$ 延迟 *k* 个时间单位:

$$plus(times(minus(\boldsymbol{A}(1)*\boldsymbol{X}(1), \boldsymbol{A}(2)), \boldsymbol{A}(3)*\boldsymbol{X}(1)), \boldsymbol{A}(4))$$

与通常结构已知的 NPE 问题不同, GPM 中模型结构是动态产生的, 因此很可能会产生结构不合理的模型; 此外, 即使是结构合理的模型, 在不同参数取值下也会得到不合理的输出, 这导致通常的参数估计方法无法在 GPM 应用. 为了解决在动态模型结构、模型光滑与可导性未知情况下的参数估计, 这里采用粒子群算法(PSO)[5,6] 在参数空间中搜索尽可能合理的取值. PSO 是一种新型的基于群体智能的优化算法, 其运行机理包含着深刻的社会认知规律[10], 与经典遗传算法相比, 它通常只需较少的粒子数(通常 4-35)、较少的演化代数(一般小于 200) 即可达到较好的计算效果, 且算法结构简单, 可调参数较少, 实现方便, 有助于提高 GPM 运行效率.

3.3　模型输出的计算

在确定模型结构、模型参数和模型输入的情况下, 可以对模型表达式进行解析并计算模型输出. 出于实现方便的考虑, 本文在 GP 阶段仍采用均方误差 MSE 作为衡量模型序列与实测序列是否接近的标准. 传统的 GP 建模方法[4] 通常只能处理小样本数据, 这是因为数据量增大后, 数据中包含的噪音成分会导致模型结构发散, 特别是混沌系统的初值敏感特点更加剧了这种发散趋势. 为了适合大数据量输入和混沌系统要求, GPM 采用多级迭代预测计算模型输出, 如图 3 所示:

图 3 多级迭代预测计算模型输出值

由于系统的混沌本质限制了模型的预测能力[11~13], 所以在预测模型输出时只能迭代若干步, 不宜太多. 考虑到最大 *Lyapunov* 指数 λ 刻画了混沌系统中相邻轨道的指数发散速率, 所以最长预测时间 *L* 可以取做 λ 的倒数, 即 $L\approx 1/\lambda$ 作为系统确定性预测的时间上界[12]. 在数据精度比较高时, *L* 可以适当大一些.

3.4 模型个体的评估

模型个体的评估策略决定着GP的演化方向. 在GPM中, 个体 i 的适应值由下式决定:

$$fitness(i)=\frac{1}{1+precision(i)}\cdot\alpha(h) \quad (1)$$

其中, $precision(i)=\sum_{k=1}^{n}[x(k)-\hat{x}(k)]$, 表示该个体所代表的模型的精度, $x(k)$ 为实测序列, $\hat{x}(k)$ 为模型输出序列, 变换 $1/(1+precision(i))$ 的倒相作用使得模型误差较小的个体具有较高的适应度值, 同时又可以平滑剧烈的模型误差波动, 使GP运行效果更佳. $\alpha(h)$ 为调整项因子, 由树深度 h 决定, 控制着模型复杂度, 使得GPM倾向于寻找那些结构简单的模型:

$$\alpha(h)=\begin{cases}abs(h-h')/2^{h-2} & h\geqslant 4\\ abs(h-h')\times 4 & h\leqslant 2\\ abs(h-h') & else\end{cases} \quad (2)$$

h' 表示期待树深度, 这里取为2.5.

3.5 模型比较

GPM所得模型是否真实地反映了实际系统的运行规律? 一般的建模方法都是通过比较实测序列与模型序列的误差来说明模型与系统的等价性, 但是, 对Chaos系统, 所得模型除了应能较好的拟合特定初值下的实测序列, 更应能够正确反映系统内在的混沌规律, 因此有必要引入一些混沌系统的不变特征量作为衡量模型质量的标准, 因为这些特征量与初值无关且能反映系统的混沌本质. 本文采用最大Lyapunov指数 λ 和关联维 D 作为比较标准, 它们的近似值可直接从观测序列中估计得到[13].

3.6 模型的筛选

由于数据噪音、算法本身的有偏性、系统本身的复杂性等各方面原因, GP输出的最优解未必是最优的模型. 事实上, GP的末代群体和每代最优都可以作为备选模型, 模型筛选就是要从这些备选模型中人工地选出能尽可能反映系统实际动力学行为的最终模型. 与3.3模型输出计算不同, 这里的模型序列采用任意合法初值经多次迭代产生, 模型比较采用3.5中基于混沌不变量的比较方式. 如果模型序列和实测序列具有比较接近的不变量, 则可认为这个模型比较好地反映了实际系统的混沌特征.

4 实验及讨论

4.1 Logistic混沌序列的建模

Logistic 映射:

$$x(n)=rx(n-1)(1-x(n-1))\quad x\in(0,1),\ n\in N \quad (3)$$

是一个典型的混沌映射, 在 $r>3.5699$ 时会产生复杂的动态行为. 取 $r=4$, $x(0)=0.2$, 迭代生成长度10000的混沌序列, 从该序列中可直接计算有关的系统特征量如下: 重构延迟 $\tau=8$, 嵌入维数 $m=4$, 关联维 $D\approx 3.13$, 最大 *Lyapunov* 指数 $\lambda\approx 0.51$. 相应的最大可预测长度 L 约为2步, 信噪比较高时, L 的值可以适当增大; 而这样估计出的维数 m 由于数据噪音等原因一般会偏大, 实际建模时可以先从较小的值开始尝试, 这里取 $m=1$.

表1 Logistic序列备选模型, A 为参数向量, X 为自变量, *plus*、*times* 和 *minus* 分别是+, *, – 算子

序	模型(参数取值略)
1	*plus(times(minus(A(1)*X(1), A(2)), A(3)*X(1)), A(4))*
2	*plus(times(minus(A(1)*X(1), A(2)), A(3)), A(4)*X(1))*
3	*plus(times(minus(A(1)*X(1), A(2)*X(1)), A(3)*X(1)), A(4)*X(1))*
4	*times(plus(A(1), A(2)*X(1)), A(3)*X(1))*
5	*plus(times(minus(A(1)*X(1), A(2)*X(1)), plus(A(3), A(4))), A(5)*X(1))*
6	*times(plus(A(1), A(2)*X(1)), times(plus(A(3), A(4)*X(1)), A(5)*X(1)))*
7	*times(plus(A(1), plus(A(2), A(3)*X(1))), times(plus(A(4)*X(1), A(5)*X(1)), minus(A(6)*X(1), A(7)*X(1))))*
8	*times(plus(A(1), A(2)*X(1)), plus(A(3)*X(1), A(4)))*
9	*plus(minus(A(1)*X(1), A(2)), minus(A(3)*X(1), A(4)*X(1)))*

从上述Logistic序列中选取长度100的数据片断作为GPM输入, 取群体规模 $M=5$, 演化代数10, 交叉概率0.8, 变异概率0.6, 期待树深度 $h'=2.5$, $FS=\{plus, minus, times\}$, $TS=\{CONSTANT, X(1)\}$; GPM输出的一些备选模型如表1所示. 选择这些个体为备选模型的原因是因为它们具有较高的适应度, 并且具有可能产生混沌行为的潜在结构. 为提高运算效率, GPM内嵌的PSO-NPE算法只用较少的代数对模型进行初步参数估计, 但在获取上述备选模型后, 可增加PSO的粒子数和演化代数进行更精确的参数估计. 其中, 模型4在20个粒子演化80代时最小均方误差 *MSE* 就降到0.0739, 对应参数 $A=\{10.0000, -10.0000, 0.3993\}$, 经整理后模型为 $x(n)=3.993x(n-1)\cdot(1-x(n-1))$, 与实际模型十分接近, 如图4所示. 模型3在20个粒子演化80代后得到 $A=\{-8.2138, -4.7130, 1.1438, 4.0018\}$, $MSE=0.0704$, 并且具有与实际系统相接近的最大 *Lyapunov* 指数0.54. 事实上, 模型3约简后为 $x(n)=4.0018x(n-1)-4.0042x^2(n-1)$, 等同于Logistic模型. 可见, GPM可以成功地从Logistic序列中重建系统模型.

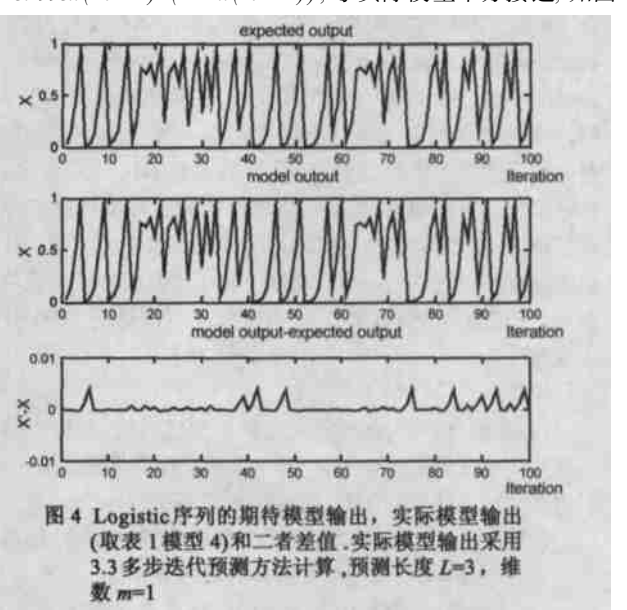

图4 Logistic序列的期待模型输出, 实际模型输出(取表1模型4)和二者差值. 实际模型输出采用3.3多步迭代预测方法计算, 预测长度 $L=3$, 维数 $m=1$

4.2 Chebyshev 混沌序列的建模

Chebyshev 映射具有如下形式:

$$x(n)=\cos(1.8\cos^{-1}(x(n-1))) \tag{4}$$

为考察 GPM 对噪音的敏感程度,在上述序列中加入适量白噪音成分,SNR= 35db. LS-GP[8]演化 51 代得到的最佳模型为:

$$x(n)=1.9832\cos(0.6545x(n)+0.0183\sin^{-1}(x(n-3))) \tag{5}$$

H. Leung 等提出的改进算法 ILS-GP 得到的结果为[8,9]:

$$x(n)=1.0025\cos(1.8003\cos^{-1}(1.0000x(n))) \tag{6}$$

GPM 则能很快地稳定在最优结构上:

$$x(n)=\boldsymbol{A}(1)\cos(\boldsymbol{A}(2)\cos^{-1}(\boldsymbol{A}(3)x(n-1))) \tag{7}$$

从而将这个问题蜕化为一个 NPE 问题. PSO-NPE 输出为 $\boldsymbol{A}=\{0.9954, 1.8024, 1.0000\}$. 可见, GPM 同样具有较好的抗噪能力,这是因为 GPM 中结合了 NTSA 的分析结果, 我们可以预知系统维数 m、可预测长度 L 等系统信息, 而不必完全依赖建模算法在建模过程中去决定维数, 从而提高了效率和准确性.

4.3 有关 GPM 的参数设置和一些经验

与神经网络等全局建模方法相比较, GPM 能够比较容易地结合系统先验知识, 从而获得较好的建模结果. 准确的先验知识如系统维数、已知模型结构模式、可预测长度等有助于提高建模的质量. 函数算子集合 *FS*、终结符集合 *TS* 和初始群体的设置也都需要先验知识的指导.

算法运行初期, GPM 可能会产生大量的不可行解和一些无意义的平凡解, 但伴随系统的演化, 它们所占的比例会降低,可以引入编辑算子以减弱这些非法个体的影响, 对提高效率很有帮助. 此外, 避免复杂的模型结构也会促使 GPM 减少非法解的产生, 因为复杂结构非法的概率相对较高. 而且, 结构简单的模型在揭示内部规律上往往优于复杂的模型.

动态结构下的非线性参数估计是一个难点, 因为其中经常会遇到非法输出、无穷大等不合理情况; 此外, 参数搜索空间设置的是否合理亦会影响 NPE 的结果. 寻求大参数空间内高效的 NPE 算法有助于提高 GPM 的效果.

GPM 的群体规模和演化代数一般不必太大, 个体取 10–30, 演化代数取 10–50 即可. 这是因为 GPM 只要求尽可能大地搜索模型空间, 并不要求全部收敛到所谓的最优解上, 如 3.5 所述, 在 MSE 衡量标准下, 最后所谓的算法最优也未必就是最合适的模型.

5 结论

许多系统的外在复杂行为都是由内在的简单混沌规律决定的, GPM 提供了一种从有限的实测时间序列中恢复出系统模型的方法. 与多项式、神经网络等全局建模方法不同的是, GPM 可以搜索广大函数空间并得到较好的模型结构, 这有助于发现系统内在的规律并得到简明的模型表述. 仿真实验表明, 用 GPM 得到的模型比较接近实测时间序列的行为, 二者具有近似相等的不变特征量, 能够正确反应系统的混沌特征.

参考文献:

[1] R Hegger, H Kantz. Practical implementation of nonlinear time series methods: The TISEAN software package online documentation[R]. http://www.mpiipks-dresden.mpg.de/~tisean, 2000.

[2] 魏荣, 卢俊国, 李军, 王执铨. 离散混沌系统的小波模型和定量分析[J]. 电子学报, 2002, 30(1): 73–75.
WEI Rong, LU Junguo, LI Jun, WANG Zhiquan. A new wavelet model for identification of discrete chaotic systems and qualitative analysis of model[J]. Acta Electronica Sinica, 2002, 30(1): 73–75(in Chinese).

[3] 简相超, 郑君里. 一种正交多项式混沌全局建模方法[J]. 电子学报, 2002, 30(1): 76–78.
JIAN Xiangchao, ZHEN Junli. A chaotic global modeling method based on orthogonal polynomials[J]. Acta Electronica Sinica, 2002, 30(1): 76–78(in Chinese).

[4] 潘正君, 康立山, 陈毓屏. 演化计算[M]. 北京, 清华大学出版社/广西科学技术出版社, 1998.

[5] J Kennedy, R Eberhart. Particle swarm optimization[A]. Proc IEEE Int Conf on Neural Networks[C]. USA: IEEE Press, 1995, 4: 1942–1948.

[6] Shi Yuhui, R Eberhart. A modified particle swarm optimizer[A]. Proc IEEE Int Conf on Evolutionary Computation[C]. Anchorage, Alaska: IEEE Press, May 1998: 69–73.

[7] J R Koza, Genetic Programming. A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems[M]. USA: Starr ford University, http://www.genetic programming.com/jkpubs72to93.html# anchor484765, 1990.

[8] H Leung, V Varadan. System modelling and design using genetic programming[A]. The 1st IEEE International Conference on Cognitive Informatics[C]. Banff, Canada: IEEE, Aug 2002.

[9] V Varadan, H Leung. Reconstruction of polynomial systems from noisy time series measurements using genetic programming[J]. IEEE Trans. Industrial Electronics, 2001, 48(4): 742–748.

[10] 谢晓锋, 张文俊, 杨之廉. 微粒群算法综述[J]. 控制与决策, 2003, 18(2): 129–134.

[11] H Kantz, T Schreiber. Nonlinear time series analysis[M]. UK: Cambridge University Press, 1997.

[12] 吕金虎, 路君安, 陈士华. 非线性时间序列分析及其应用[M]. 武汉大学出版社, 2002.

[13] M T Rosenstein, J J Collins, C J de Luca. A practical method for calculating largest lyapunov exponents from small data sets[J]. Physica D 1993, 65(1–2): 117–134.

作者简介:

张　伟　男, 1975 年出生于河北沧州, 1997 年毕业于华东理工大学计算机系, 1999 年进入上海交通大学自动化系直接攻读博士学位, 主要兴趣为复杂网络、智能数据分析. E-mail: zhang_wi@sjtu.edu.cn

吴智铭　男, 1936 年出生于上海, 教授, 博士生导师, 研究方向为制造系统中的生产计划与调度、智能计算方法、混合系统.

杨根科　男, 1963 年出生于山西, 教授, 研究方向为混合系统.