基于多目标人工鱼群算法的符号回归

刘 庆, 任海鹏†, 姚俊良, 刘 龙

(西安理工大学 陕西省复杂系统控制与智能信息处理重点实验室, 陕西 西安 710048)

摘要: 针对现有符号回归方法仅关注拟合误差而忽略模型简化的问题, 提出了一种基于多目标的人工鱼群算法, 将拟合误差与模型复杂度同时作为目标函数进行优化. 以二叉堆对语法树编码, 优良分支得以稳定地遗传和继承, 也更易解码. 在引入蒙版、邻域、小生境、拥挤度等概念的基础上, 设计和定义了适用于二叉堆编码的随机游动、觅食、追尾、逃脱等人工鱼行为算子. 详尽的实验表明, 提出算法在符号回归过程中能获取高质量的Pareto解. 此外, 对从Pareto前沿上选取折衷解及降低算法内存开销的方法也进行了讨论.

关键词: 符号回归; 多目标优化; 语法树; 二叉堆

引用格式: 刘庆, 任海鹏, 姚俊良, 等. 基于多目标人工鱼群算法的符号回归. 控制理论与应用, 2020, 37(2): 340 – 354

DOI: 10.7641/CTA.2019.80896

Symbolic regression via a multi-objective artificial fish school algorithm

LIU Qing, REN Hai-peng†, YAO Jun-liang, LIU Long

(Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing,
Xi’an University of Technology, Xi’an Shaanxi 710048, China)

Abstract: Aiming at the issue that the existing methods for symbolic regression focus on minimizing the fitting error merely while ignore the model simplification, a multi-objective artificial fish school algorithm is proposed for minimizing the fitting error and the model complexity simultaneously during symbolic regression. The parse tree is encoded as the form of binary heap, such that fine branches of the parse tree could be stably inherited, and suchlike binary heap-based representation is easier to decode. By introducing the conceptions such as mask, neighborhood, niche, and crowding degree, several behavior operators performed by the artificial fish, including randomly moving, foraging, following, and escaping, are defined. Exhaustive simulation results show that the proposed algorithm is capable of obtaining high-quality Pareto solutions during symbolic regression. Besides, the method of determining a trade-off solution from the obtained Pareto front and that of reducing the memory overhead are also discussed.

Key words: symbolic regression; multi-objective optimization; parse tree; binary-heap

Citation: LIU Qing, REN Haipeng, YAO Junliang, et al. Symbolic regression via a multi-objective artificial fish school algorithm. *Control Theory & Applications*, 2020, 37(2): 340 – 354

1 引言

数据是信息的载体, 从数据中发掘相关事物的内在联系、掌握未知系统的运行规律, 一直是人们获取知识的重要手段. 根据观测数据辨识和构建多个变量间耦合关系的过程被称为数据驱动建模. 目前, 数据驱动建模大致有3类实现方法: 回归分析[1–4]、神经网络[5–6], 以及符号回归[7–8].

回归分析要求预先给出模型假设(model assumption), 然后根据观测数据确定各回归系数. 给出模型假设的前提是具备与变量耦合关系相关的先验知识. 对于一个完全未知的“黑盒子(black box)”来说, 难以给出恰当的模型假设, 而不合理的模型假设往往导致过拟合(over-fitting), 从而限制了回归分析的应用场景.

神经网络以调整神经节点连接权值的方式拟合观测数据, 因而不要求模型假设, 但其所建模型也是以连接权值的形式存在, 因而无法以明确的代数解析表达式对变量之间的耦合关系进行实质刻画. 所以, 神

收稿日期: 2018−11−19; 录用日期: 2019−05−16.
†通信作者. E-mail: renhaipeng@xaut.edu.cn; Tel.: +86 13379268118.
本文责任编委: 王凌.
国家自然科学基金项目(61502385, 61673318), 陕西省特支计划科技创新领军人才项目支持资助.
Supported by the National Natural Science Foundation of China (61502385, 61673318) and the Shaanxi Provincial Special Support Program for Science and Technology Innovation Leader.

经网络本质上是为一个“黑盒子”找到另一个等效的“黑盒子”, 这对模型的理解和数据的解释造成了困难. 例如, 文献[9]就指出在煤矿双电机驱动带式输送机的能耗建模中前馈(back propagation, BP)神经网络得到的模型不具备外推性, 而基于解析表达式的输送机能耗模型才更为合理. 此外, 当变量耦合关系比较复杂时, 神经网络也存在过拟合的现象.

符号回归以语法树(parse tree)对模型进行表达, 例如著名的质能方程$E = M \cdot C^2$可表示成图1所示语法树, 质量M与光速C以图中所示语法树的连接关系耦合, 从而得出能量E. 不同的语法树对应不同的模型, 以优化语法树结构的方式最小化其模型在观测数据上的拟合误差, 就是符号回归的过程. 换言之, 符号回归是一个优化问题. 由于符号回归得到的模型是对语法树优化的结果, 所以不要求预先给出模型假设, 而语法树对变量耦合关系的清晰刻画也使“黑盒子”得以“白化”. 因此, 以符号回归的方式构建的模型在合理性与可解释性两个方面均显著优于回归分析和神经网络.

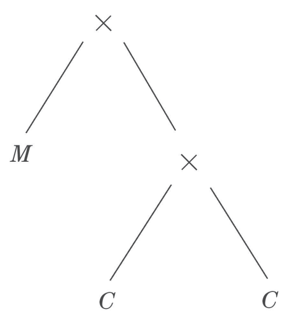


图 1 语法树示例

Fig. 1 Diagram of a parse tree

1992年, John Koza[10]开创性地提出了遗传规划(genetic programming, GP), 通过重新定义遗传算法(genetic algorithm, GA) 中的交叉和变异实现了语法树的优化, 奠定了符号回归的基础. 针对GP易出现代码膨胀(code bloat)及产生无效语法树的问题[11–12], Ferrira 提出了基因表达式规划 (gene expression programming, GEP)[12]. GEP同样以语法树表示模型, 但是对语法树以开放式阅读框架(open reading frame, ORF)进行编码, 使GEP对语法树的遗传操作灵活而安全. 得益于此, GEP在解决符号回归问题时的效率比GP高4个数量级[12]. 在此之后, 以GEP为蓝本的各种变体算法也被相继提出[13–15]. Li X等人[13]采用前缀记号法对GEP的ORF进行存储, 显著提高了GEP的收敛速度. Zhong J 等人[14]提出了一种自学习 GEP (self-learning GEP, SL–GEP), 通过在ORF串中实行子函数嵌套机制提高了对复杂模型的构造能力. Peng Y 等人[15]则通过引入堆栈技术进行解码进一步提高了GEP的解码效率. 近年来, 群智能算法在符号回归中的应用同样引人关注[16–18]. Qi F 等人[18]利用收敛速度更快的粒子群算法(particle swarm optimization, PSO)实现了对语法树的演化, 在收敛时间和平均收敛代数方面均优于传统GP. Karaboga D等人则将蜂群算法(artificial bee colony, ABC)[17]应用于符号回归, 具有比传统GP更好的灵活性与鲁棒性. 不过, 由于Qi F和Karaboga D等人[17–18]的研究均直接沿用了GP中的语法树作为模型表达, 因而仍然存在代码膨胀及产生无效语法树的问题. 鉴于此, Liu Q等人[16]将GEP中的ORF串和鱼群算法(artificial fish school algorithm, AFSA)的优化框架进行了结合, 取得了在拟合误差与求解速度方面均优于GEP的结果.

目前, 已报道的各类符号回归算法不少, 但都忽略了一个基本问题. 符号回归的核心任务是建模, 其目的在于解释数据, 从而揭示规律, 进而预测趋势. 已有算法仅聚焦于模型拟合误差的最小化, 得到的模型往往十分复杂. 奥卡姆剃刀(Occam’s razor)准则是一个被物理学、天文学等自然科学领域广为沿用的基础性原则, 它主张保留与经验观察一致的最简单假设. 尽管在机器学习领域中对什么是“最简单”尚存争议, 但这一思想在机器学习领域已被广为接受和运用. 实际上, 形式简洁的模型往往更有效. 一方面, 简洁的模型不会过度解释噪声, 有利于避免过拟合; 另一方面, 简洁的模型也更易理解. 因此, 本文将符号回归考虑成一个同时对模型拟合误差和模型复杂度最小化的多目标优化问题, 并提出了一种基于二叉堆的多目标鱼群算法(multi-objective artificial fish school algorithm, MO–AFSA)对其求解. 详尽的算法描述及仿真在随后小节给出.

2 拟合误差与模型复杂度的量化

符号回归本质上是一个组合优化的过程, 决策变量是$+$, $-$, $\times$, $\div$, sin$(\cdot)$, exp$(\cdot)$等运算符以及x, y, π等变量或常量, 要得到的模型$y = \phi(x)$就是定义在由这些运算符、变量或常量构成的符号集合S上的一个最优映射$\phi(\cdot)$.

在本文考虑的优化框架下, 评价模型$y = \phi(x)$需从拟合误差f_1与模型复杂度f_2两个方面进行评价. 本节给出量化这两个指标的方法.

2.1 拟合误差

本文以平均绝对误差对模型$y = \phi(x)$的拟合误差进行量化. 也就是说, 拟合误差f_1是关于映射$\phi(\cdot)$的函数, 具体根据式(1)进行计算:

$$f_1(\phi) = \frac{1}{n} \cdot \sum_{i=1}^{n} |\phi(x_i) - y_i|, \tag{1}$$

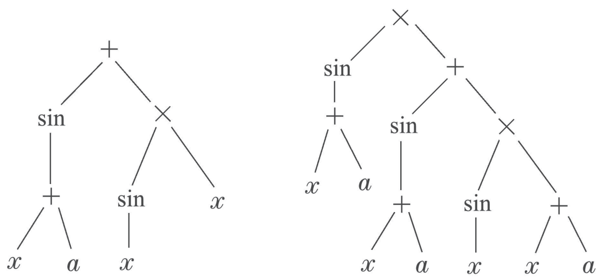
其中: n表示训练样本的个数; (x_i, y_i)表示第i个训练样本.

2.2 模型复杂度

在回归分析和神经网络中, 正则化是实现模型约简的普遍做法, 有助于提高模型的泛化能力. 然而, 符号回归以语法树对模型进行描述, 正则化并不适用. 实际上, 语法树上语义节点的多寡, 与模型表达式的复杂程度直接相关. 因此, 本文以语法树节点个数对其进行量化, 即$f_2 = N_{\text{node}}$. 例如, 图2(a)–(b)分别给出了2个语法树, 其语义节点数分别为9和16, 它们分别具有形如式(2)–(3)的模型表达式. 显然, 式(2)具有比式(3)更为简洁的表达式结构.

$$f(x) = \sin(x + a) + x \sin x, \tag{2}$$

$$f(x) = \sin(x + a)(\sin(x + a) + \sin x(x + a)). \tag{3}$$



(a) 9个节点的语法树　(b) 16个节点的语法树

图 2 两个具有不同节点数的语法树示例

Fig. 2 Two parse trees with different number of nodes

3 用于符号回归的多目标人工鱼群算法

3.1 人工鱼编码

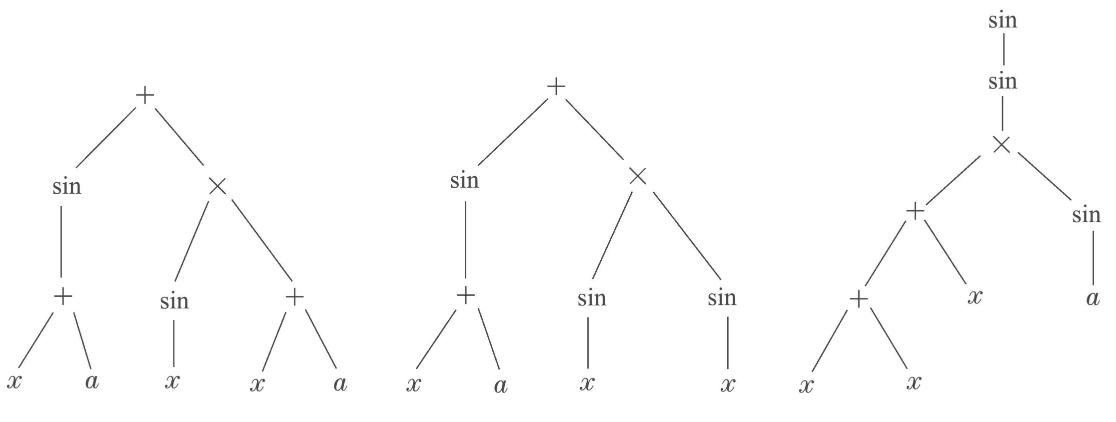
利用AFSA优化语法树, 关键是要将语法树映射为一个便于算法操作的线性串. GEP中的ORF串结构是符号回归常采用的映射形式. 本节在分析ORF的基础上提出一种基于二叉堆的全新编码.

3.1.1 ORF串的缺点

ORF串具有一段长度可变的非编码区, 该区域不产生任何语法表达. 非编码区是一种有益的编码冗余, 其意义在于确保操作ORF串的安全性, 弱化了对ORF串的操作限制. 这一点在文献[16]报道的工作中得到了确认. 但笔者也发现, ORF串对语法树的映射表达并不稳定. 现以图3所示的3个语法树为例进行说明.

根据ORF编码规则, 将语法树上的语义节点按自上而下、从左至右的次序逐个排列, 构成ORF的编码区, 其后补足一定数量的非运算节点作为非编码区. 表1列出了图3各语法树对应的ORF串, 其中: s表示$\sin(\cdot)$算子, 各ORF串的非编码区以下划线标出. ORF–2仅1个符号位与ORF–1不同(以框线标出), 这解释了二者语法树结构的近似, 见图3(a)–(b); 但情况并非总是如此, ORF–3 也仅有 1 个符号位与 ORF–1 不同(以框线标出), 但二者语法树的结构却差异巨大. 这说明, 相近的ORF串不一定表达相近的语法树. 从进化计算的角度看, 该现象违背了模式块定理的前提假设[19].

分析易得, 即便只对 ORF 串做轻微操作, 如修改ORF串的某1个符号位, 作为表现型的语法树也可能发生巨大变化. ORF串对语法树映射表达时的这种不稳定性, 不利于优良分支结构的继承, 从而影响符号回归的效果. 此外, ORF串具有可变长的非编码区, 将其解码为语法树时须首先确定其有效编码长度, 这需要执行专门的算法[16,20]. 符号回归是一个迭代最优化过程, 期间会对ORF频繁解码, 确定ORF有效长度的运算会显著劣化符号回归的效率.



(a) ORF−1语法树　(b) ORF−2语法树　(c) ORF−3语法树

图 3 语法树示例

Fig. 3 Diagram of parse tree

表 1 ORF串(下划线部分为非编码区)

Table 1 ORF-string (the underlined part is the non-coding region)

ORF串															
ORF–1	+	s	×	+	s	+	x	a	x	x	a	$\underline{x}$	$\underline{x}$		
ORF–2	+	s	×	+	s	$\boxed{s}$	x	a	x	x	$\underline{a}$	$\underline{x}$	$\underline{x}$		
ORF–3	$\boxed{s}$	s	×	+	s	+	x	a	x	x	$\underline{a}$	$\underline{x}$	$\underline{x}$		

3.1.2 二叉堆编码

针对 ORF 的前述缺点, 本文设计了一种基于二叉堆(binary heap, BH)的全新编码. 具体说来, 二叉堆BH中键值为i的节点$\text{BH}_{[i]}$总有$\text{BH}_{[2i+1]}$与$\text{BH}_{[2i+2]}$两个后继节点. 利用这一性质, 本文设计了灵活的映射规则.

现以图4举例说明, 语法树的根节点‘+’在二叉堆中的键值为0, 即$\text{BH}_{[0]} =$‘+’, 其后继节点则分别对应二叉堆节点$\text{BH}_{[1]} =$‘s’与$\text{BH}_{[2]} =$‘×’; 虽然$\text{BH}_{[1]} =$‘s’具有$\text{BH}_{[3]} =$‘+’和$\text{BH}_{[4]} =$‘×’两个后继, 但$\text{BH}_{[1]} =$‘s’表示的$\sin(\cdot)$是一个单目运算符, 因而在语法树中并不对其右子树进行语法表达, 故在图4中以虚线表示, 同理的还有$\text{BH}_{[5]} =$‘s’的右子树; 更进一步, 若将$\text{BH}_{[1]} =$‘s’修改为诸如‘x’或‘a’这样的非运算节点, 可将其视作0目运算符, 则$\text{BH}_{[1]}$在语法树中的左、右子树均不进行语法表达. 另外, 本文规定由$(2k+1)$个语义节点构成的二叉堆编码中, 从$\text{BH}_{[k]}$至$\text{BH}_{[2k]}$的$(k+1)$个语义节点只接受非运算节点的赋值以确保映射的语法树叶节点上不出现单目或双目运算.

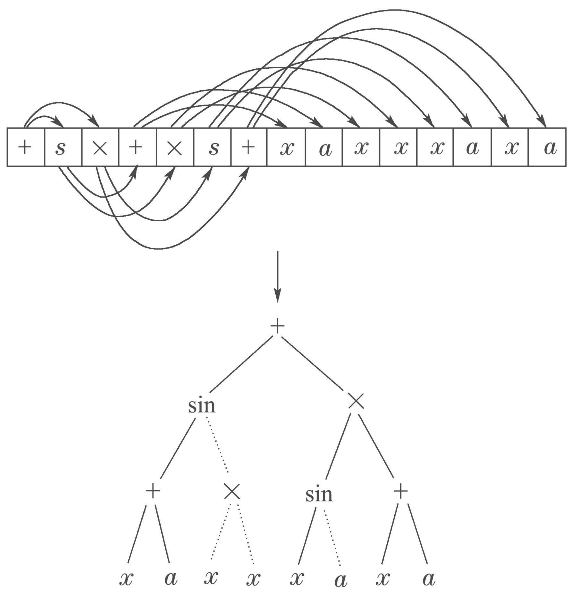


图 4 二叉堆与语法树的对应关系

Fig. 4 Correspondence relationship between binary heap and parse tree

在语法表达的稳定性方面, 二叉堆编码明显优于ORF串. 实际上, 图4语法树去掉虚线部分就是图3(a)所示的ORF–1语法树. 将表1中的ORF–1根节点从‘+’替换为‘s’即得到ORF–3, 前面笔者已经知道这一操作使得ORF–1的语法树发生了巨大的拓扑变化. 作为对比, 在图4所示的二叉堆编码上进行同样操作, 将根节点对应的$\text{BH}_{[0]} =$‘+’替换为‘s’. 根据二叉堆编码映射规则, 可得图5语法树(实线部分). 显然, 新得到的语法树的确继承了图4语法树的拓扑, 这与表1中ORF–1被修改后发生的语法树拓扑的巨大变化形成了明显对比. 二叉堆编码在表达语法树时优良的稳定性符合模式定理的前提假设, 有利于合理语法分支的继承. 此外, 由于可从二叉堆编码的任意键值i处根据$(2i+1)$和$(2i+2)$直接索引后继节点, 因而解码可以从根节点出发并以递归方式完成, 避免了采用ORF串时必须首先确定其有效编码长度的计算负担.

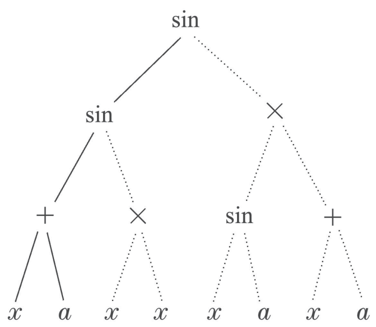


图 5 修改图4语法树的根节点后得到的语法树

Fig. 5 Parse tree obtained by altering the root of the original parse tree shown in Fig. 4

3.1.3 蒙版

本文在二叉堆编码的基础上进一步提出蒙版(mask)的概念, 以下给出其具体定义及相关性质.

定义 1 蒙版(mask)是二叉堆编码BH中各语义节点是否进行语法表达的状态刻画, 可视作对BH的一种假想覆盖, 用一段与BH等长的0–1码串表示, 且满足:

$\text{mask}_{[i]} = 0$时, 语义节点$\text{BH}_{[i]}$被蒙版覆盖不表达语法;

$\text{mask}_{[i]} = 1$时, 语义节点$\text{BH}_{[i]}$未被覆盖进而表达语法.

例如, 图4中二叉堆编码与其蒙版的关系如图6所示.

关于蒙版状态的确定, 并不需要专门的算法. 根据第3.1.2节中所述的二叉堆编码规则, 在以递归方式将1个二叉堆解码为一个语法树时, 易将蒙版状态同步带回, 此处不再赘述.

BH	+	s	×	+	×	s	+	x	a	x	x	x	a	x	a
蒙版	1	1	1	1	0	1	1	1	1	0	0	1	0	1	1

被蒙版覆盖部分不作语法表达　未被蒙版覆盖部分

图 6　二叉堆编码与其对应蒙版关系示例

Fig. 6 Diagram of binary heap and its corresponding mask

根据蒙版的定义, 可得到以下性质及推论:

性质 1　二叉堆编码表达的语法树节点个数等于其蒙版状态之和.

该性质可用式(4)表示:

$$f_2 = N_{\text{node}} = \sum_{i=0}^{2k+1} \text{mask}_{[i]}. \tag{4}$$

性质 2　对于二叉堆编码上的任意语义节点 $\text{BH}_{[i]}$:

a) $\text{mask}_{[i]} = 0$时, 对$\text{BH}_{[i]}$作任意修改, 语法树保持不变;

b) $\text{mask}_{[i]} = 1$时, 以同目运算符替换$\text{BH}_{[i]}$, 语法树改变但结构保持不变; 以异目运算符替换$\text{BH}_{[i]}$, 语法树结构也改变.

性质2的推论　表达相同结构语法树的任意二叉堆编码具有相同的蒙版状态.

3.2　人工鱼行为算子与外部档案维护

AFSA是由国内学者在模拟自然水域中鱼群游弋过程的基础上提出的一种群体智能优化算法, 人工鱼(artificial fish, AF)通过执行随机游动、觅食、追尾等行为算子实现对问题解空间的搜索[21]. 利用AFSA求解本文问题的关键是如何在二叉堆编码上定义搜索邻域、拥挤度并实现相关行为算子, 本小节对此展开讨论.

定义 2　任意人工鱼AF–x的搜索邻域可定义为与其二叉堆编码具有相同蒙版状态的所有AF的集合.

定义 3　规模为Pn的人工鱼种群若以图7所示队列表示, 则以任意人工鱼AF_i为中心、从AF_{i-r}至AF_{i+r}的$(2r+1)$条人工鱼构成一个小生境(niche), $(2r+1)$为其容量. 特别地, 对于下标为$s \in [i-r, i+r]$ $(0 < i < Pn-1)$的任意人工鱼, 若$s < 0$, s取$s + Pn$的和; 若$s > 0$, s取s除以Pn的余数. 该操作使AF_0与AF_{Pn-1}在逻辑上相连, 构成一个逻辑上的环状拓扑. 这样一来, 人工鱼种群中就有Pn个容量为$(2r+1)$的小生境.

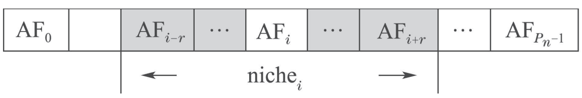


图 7　人工鱼小生境示例

Fig. 7 Diagram of AF niche

定义 4　人工鱼AF–x所在的容量为$(2r+1)$的小生境中, 若有n条人工鱼进入了AF–x的搜索邻域, 则$\delta(x) = \dfrac{n}{2r+1}$称为人工鱼AF–$x$搜索邻域的拥挤度.

一条任意人工鱼AF–x通过执行各行为算子搜索问题的解空间, 其本质是对语法树的搜索, 语法树以二叉堆编码的形式存在, 因而各行为算子的实质是对二叉堆编码的操作. 一条任意人工鱼AF–x的各个行为算子可描述如下:

1) 随机游动 (randomly moving), 人工鱼 AF–x 在其搜索邻域内随机移动rs个海明距离的行为.

根据定义2, AF–x在随机游动后, 其二叉堆编码的蒙版状态不能发生改变. 否则, 意味着AF–x进入了新的搜索邻域. 因此, 该行为的实现方式是从AF–x的二叉堆编码BH中任选rs个满足$\text{mask}_{[i]} = 1$的$\text{BH}_{[i]}$以同目运算符进行替换, 取步长$rs = \min\{rnd, \text{AF–}x.f_2\}$, 其中$rnd$为区间[minStep, maxStep]上的随机整数.

2) 觅食(foraging), 人工鱼AF–x对其搜索邻域进行抽样和感知, 从而向更优位置游动的行为.

人工鱼AF–x对其所在搜索邻域进行随机抽样, 将得到的抽样状态y与AF–x进行比较, 若AF–$x \prec y$ (AF–x受y支配), 则以y更新AF–x并结束觅食; 否则, AF–x继续对邻域进行抽样, 直到连续Tn次抽样后均未获得可支配AF–x的抽样状态y, 则觅食行为结束.

需要解释的是, AF–x对其搜索邻域进行抽样得到状态y的过程, 可通过复制AF–x的副本并以调用随机游动的方式模拟实现.

3) 追尾(following), 人工鱼AF–x向其所在小生境内的非支配解y移动rs个海明距离的行为.

计算人工鱼AF–x所在小生境内的所有非支配解在目标空间中的拥挤距离Cd(crowding distance)[22] (与定义4中拥挤度无关). 设拥挤距离最大的非支配解为AF–y, 若Cd(AF–x)<Cd(AF–y), 则从AF–y的二叉堆编码中随机选rs个语义节点$\text{BH}_{[i]}$对AF–x的对应位置进行替换, 要求步长$rs = \min\{rnd, \|\text{mask} - x \& \text{mask} - y\|_1\}$且选取的$rs$个位置须满足$\text{mask}{-}x_{[i]} = 1$, $\text{mask}{-}y_{[i]} = 1$, 其中$rnd$取区间[minStep, maxStep]上的随机整数; 若Cd(AF–x) ⩾ Cd(AF–y), 则AF–x转

而执行随机游动行为. 追尾行为执行后蒙版状态可能发生改变, 因此需对蒙版状态进行及时更新.

4) 逃脱(escaping), 人工鱼AF–x所在搜索邻域过于拥挤时, 移动rs个海明距离以逃脱所在邻域的行为.

判断人工鱼AF–x所在搜索邻域是否过于拥挤的依据是定义4给出的拥挤度$\delta(x)$是否大于一个可容忍的拥挤程度上限Uc. 若$\delta(x) > Uc$, 表明AF–x的搜索邻域过于拥挤, 则AF–x 执行逃脱行为. 根据定义 2, AF–x逃脱所在邻域意味其二叉堆编码的蒙版状态在行为执行后必须发生改变, 否则AF–x未逃出所在邻域. 因此, 该行为的实现方式是从AF–x的二叉堆编码 BH中任选rs个满足$\mathrm{mask}_{[i]} = 1$的语义节点$\mathrm{BH}_{[i]}$, 对其中$\mathrm{BH}_{[i]}(i \in [0, k-1])$的语义节点以任意异目运算符替换, 对$\mathrm{BH}_{[i]}(i \in [k, 2k])$的语义节点以任意0目运算符替换, 取步长$rs = \min\{rnd, \mathrm{AF}\text{–}x.f_2\}$, 其中$rnd$为区间[minStep, maxStep]上的随机整数.

该行为执行后蒙版状态必然发生改变, 因此需对蒙版状态进行及时更新.

5) 维护外部档案(external archive maintainance), 更新与维护外部档案的目的是确保人工鱼种群在状态更新时能得到多样性好且分布均匀的Pareto前沿. 本文采用基于格密度的维护策略[23], 对目标空间以8×8的方式分块, 则格密度取值等于各块内的人工鱼个数. 每次迭代所得Pareto非劣解集与外部档案集进行合并, 并剔除其中的被支配解.

3.3 算法步骤

前面小节详细描述了各行为算子的实现方法及外部档案的维护策略, 本节以伪代码的形式给出多目标AFSA的执行步骤.

MO–AFSA算法伪代码:

0) 输入符号回归数据集;

1) 初始化AF种群并评价每条人工鱼AF;

2) 初始化每条人工鱼AF的蒙版mask;

3) 设置人工鱼外部档案集external archive;

4) for iter ← 1 to MaxIteration

```
    for x ← 1 to Pn
        if δ(AF–x)> Uc
            AF–x.escaping(·);
            AF–x.update(mask–x);
        end
        AF–i.foraging(·);
        if ∃ AF–y ∈ Niche–x ≻ AF–x
            AF–x.following(AF–y);
            AF–x.update(mask–x);
        else
            AF–x.randomly_moving(·);
        end
            AF–x.update(external archive)
    end
end
```

5) 输出外部档案集external archive.

4 仿真实验

笔者以C++对所提算法MO–AFSA进行编程, 在配置为Intel Core i7–6700 CPU 3.4 GHz, 8 GB RAM的计算平台上对其进行仿真. 算例取自文献[24], 要求被测算法将从函数$y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$上随机选取的20个点(见表2)作为训练样本进行函数建模. 由于该函数在$x = 0$处存在“尖点” (cusp), 因而不易建模, 现已成为考察数据驱动建模方法的benchmark算例并被广泛使用[15–16, 24–25]. 本文算法MO–AFSA的参数按表3设定. 作为对比, 本文对回归分析、神经网络、GEP[24]、SL–GEP[14]、AFSA[16]等已有数据驱动建模方法也进行了仿真. 此外,为了考察本文算法MO–AFSA得到的Pareto前沿的质量, 本文还将符号回归中已广泛运用的GP[10]与经典的NSGA[26]的多目标优化框架相结合, 编程实现了多目标优化版本的GP, 记作NSGP, 将其作为评价参考.

表 2　训练样本集: $y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$上随机选取的20个点

Table 2　Training set: 20 points randomly chosen from the function $y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$

序号	x	y
1	–0.2639725157548009	3.194980662652764
2	0.0578905532656938	1.990520017259985
3	0.3340252901096346	8.396637039972868
4	–0.2363345775644623	3.070889769728257
5	–0.8557443825668047	5.879467636957033
6	–0.0194437136332785	–0.7753263223284588
7	–0.1921343881833043	2.834702257744086
8	0.5293079101246271	12.21547266421373
9	–0.007889741187284598	–2.498039834186359
10	0.4389698049506311	10.40717348588088
11	–0.1075592926980396	2.09413635645908
12	–0.2745569943771633	3.239272780108398
13	–0.05953332196045281	1.197012847673475
14	0.3844929939583523	9.355807691898551
15	–0.8749230207363339	6.006424530013026
16	–0.236546636250546	3.071897290438372
17	–0.1678759417045577	2.674400531309863
18	0.9506821818220914	22.48196398441491
19	0.9469791595773622	22.37501611873555
20	0.6393399100595915	14.5701285332337

表 3 本文算法MO–AFSA参数设置

Table 3 Parameter setting of the proposed MO–AFSA

Pn	40
r	3
T_{n}	5000
U_{c}	0.6
minStep	3
maxStep	6
k	$2^{12}-1$
MaxIteration	10^4
运算符&终止符	+, –, *, /, ln, exp, sin, cos, sqrt, sq, cub, pow10, abs, x, 0.811, 0.618, π, 1.112, 2, 0.5

图8(a)–(b)分别展示了多项式回归与高斯回归两种回归分析方法在仿真算例上的建模结果. 作为对比, 函数$y=4.251x^2+\ln x^2+7.243\mathrm{e}^x$的真实曲线也在图8(a)–(b)中进行了显示. 可以看出, 不论如何调整多项式的阶次或高斯函数项数, 均无法获得满意的拟合效果, 所得模型与仿真算例函数的真实曲线明显悖离. 其原因在于预先给出的模型假设(model assumption)并非算例函数中变量耦合关系的真实反映, 尽管通过优化回归系数可以最小化拟合误差, 但模型假设的先天不足导致回归模型始终处于欠拟合(under-fitting)或过拟合(over-fitting) 的状态.

图9展示了神经网络对测试算例的拟合情况. 本文使用的4个神经网络均为3层网络, 包括1个输入层、1个输出层、1个隐层. 4个神经网络的隐层分别具有10, 20, 30, 40个神经元, 激活函数采用Sigmoid函数. 为避免过拟合, 本文还使用了Bayes正则化技术. 从图9可以看出, 较之回归分析, 神经网络的拟合效果并未见有显著改善, 拟合曲线仍明显有悖于仿真算例函数的真实曲线, 尤其在训练样本分布稀疏的部分可靠性较差.

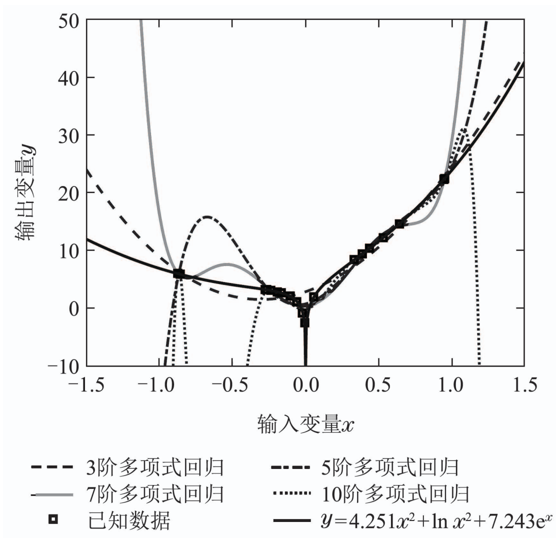


图 8(a) 基于多项式回归模型的回归分析

Fig. 8(a) Regression analysis based on polynomial model

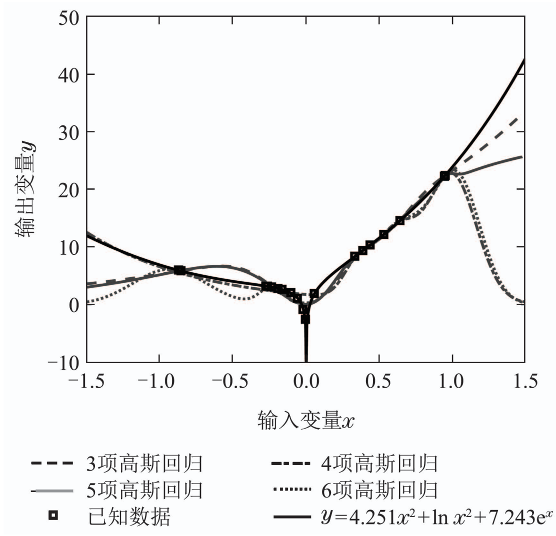


图 8(b) 基于高斯回归模型的回归分析

Fig. 8(b) Regression analysis based on Gaussian model

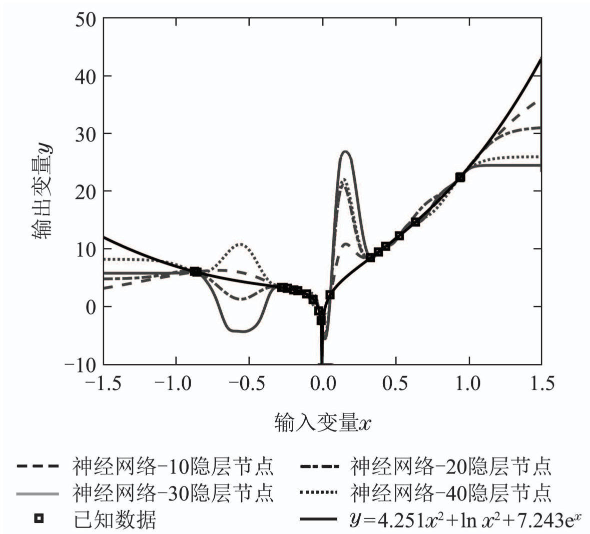


图 9 神经网络建模

Fig. 9 Neural network modeling

表4列出了本文算法MO–AFSA以多目标优化的方式对仿真算例$y=4.251x^2+\ln x^2+7.243\mathrm{e}^x$进行符号回归所得到的10个非支配解, 这10个函数模型对训练样本的拟合精度从上到下依次提高, 而函数模型的复杂度则自下而上依次降低. 将这10个非支配解模型以函数曲线的形式绘于图10, 通过与算例函数$y=4.251x^2+\ln x^2+7.243\mathrm{e}^x$的真实曲线进行比较, 可以看出它们对算例样本的拟合效果均显著优于图8–9中回归分析和神经网络得到的结果. 图10所示结果不仅表明了本文算法的有效性, 同时印证了符号回归相对于回归分析和神经网络的优越性. 此外, 为了将本文算法MO–AFSA与其他符号回归算法进行对比, 本文还在表4中列出了GEP[24], SL–GEP[14], AFSA[16], NSGP等算法得到的结果, 其中: GEP和AFSA所得结果分别在文献[24]和文献[16]中有报道, SL–GEP和NSGP所得结果则是对算法进行编程实现后的实测结果.

表 4 各符号回归算法根据表2中训练样本集得到的函数模型

Table 4 Functional models obtained by each compared symbolic regression algorithm based on the training set listed in Table 2

对比算法	f_1	f_2	模型表达式
MO–AFSA	0.2273	15	$y = 10^{\cos 1.112} \cdot (10^{\sin x} + 0.5) + \pi + \ln x^2;$
	0.1290	16	$y = \pi + ((\mathrm{e}^{\sin x})^2 \cdot 2^2 + \ln \lvert x^2 \rvert - x);$
	0.0670	18	$y = (\mathrm{e}^x - (\sin(\sin x)^3 - \lvert 0.618 \rvert))^3 + \mathrm{e}^{1.112} + \ln x^2;$
	0.0506	20	$y = \ln(x \cdot x \cdot \cos 0.5) + \dfrac{x^2}{\lvert 0.5^2 \rvert} + \mathrm{e}^x \cdot \mathrm{e}^2;$ $\bullet$
	0.0492	22	$y = (\mathrm{e}^x + (\cos(0.811)^2 \cdot x)^2) \cdot (\lvert 0.811 \rvert + 1.112)^3 + \ln(1.112 + \lvert x^2 \rvert);$ $\blacksquare$
	0.0446	23	$y = (\ln 10^x)^2 + \mathrm{e}^{2+x} - \dfrac{\sin \lvert \ln 1.112 + x^3 \rvert}{0.618} + \ln(x^2);$
	0.0332	25	$y = \dfrac{\mathrm{e}^x \cdot (0.5 + \frac{0.5}{\pi}) \cdot 10^{\cos 0.5} + x^2 \cdot \pi}{\cos 0.811} + \ln(\sin x^2);$
	0.0111	26	$y = \mathrm{e}^{x+2} + \ln(\sin x^2) + \dfrac{\frac{x^3}{(\sin 0.5)^2}}{x} - \mathrm{e}^{\ln 1.112 - 2 + x};$
	0.0084	28	$y = \ln(\mathrm{e}^{\frac{0.811^2 \cdot x^2}{0.618}} \cdot x)^2 + \dfrac{(\lvert \pi \rvert + 0.618 - \ln(\ln \pi)) \cdot \mathrm{e}^x}{0.5};$
	0.0048	36	$y = \ln(x^2) + (\dfrac{\pi \cdot \lvert \frac{x \cdot x}{\mathrm{e}^x} \rvert}{\cos((\frac{\cos 2}{0.618})^2)} + \cos(\ln 0.5) + 10^{0.811}) \cdot \mathrm{e}^x + \sin(\dfrac{x}{1.112})^2;$
SL–GEP[14]	0.0065	33	$y = \sin(x \cdot \sin\ 0.618 + \mathrm{e}^{5-\pi} - 2) + \ln(x \cdot x) + \lvert 1 \cdot (\lvert x + x \rvert^2 + \mathrm{e}^{2+x}) \rvert + 0.811;$
AFSA[16]	0.0140	34	$y = \ln(x^2) - ((\mathrm{e}^{\frac{0.618}{\mathrm{e}^{\ln(0.618)^3}}}) \cdot (\sin \dfrac{1.112}{\frac{\pi}{\mathrm{e}^x}} - \dfrac{x^2}{10^{0.618-0.811-0.811}}) - \mathrm{e}^{2+x});$
GEP[24]	0.0441	25	$y = (\ln(10^{(x+0.613)} \cdot \dfrac{x^2}{0.203})) + \mathrm{e}^{(x^2+0.811)} + 10^{\sin x} + 10^{\sin(\sin\ x)};$
NSGP	0.0839	21	$y = \ln(\sqrt{\dfrac{\frac{\mathrm{e}^{10^{1.112}}}{(10^{0.811-10^x})^2} \cdot \pi}{2}} - x \cdot x^2);$
	0.0618	23	$y = \dfrac{\mathrm{e}^{\mathrm{e}^{\sin x}+2}}{2 + \frac{\sqrt{(\frac{0.618}{0.811})^2}}{\sqrt{\sin \sqrt{\pi}}}} + \ln x^2;$
	0.0577	29	$y = (\ln(x^2 \cdot (10^{\lvert \mathrm{e}^x \rvert})^2) + \dfrac{\pi}{\sqrt{0.618}}) \cdot \sqrt{\lvert \sqrt{\cos 0.811} \rvert \cdot (x + \mathrm{e}^{\lvert x \rvert})} + 0.5^2;$
	0.0354	35	$y = \ln(\dfrac{x \cdot (\cos(\frac{0.618}{1.112}))^3}{0.5} \cdot 10^{\frac{\sqrt{0.811}}{\lvert 0.811 \rvert}})^2 + (\ln 10^{0.811 \cdot (\sin \frac{x}{1.112} - \lvert \ln 0.618 \rvert) + \sin 2})^3;$
	0.0343	39	$y = \cos(\pi - \dfrac{2 - 0.618}{\pi^3} - (10^\pi \cdot \sqrt{\pi})^3) + (1.112 \cdot x + \sin 2)^3 \cdot 2^2 \cdot 0.618 +$ $\ln x^2 + \mathrm{e}^{1.112^2} + 1.112;$
	0.0112	43	$y = \ln \dfrac{\frac{(\sin x)^2}{\cos \sqrt{0.618}}}{\frac{\frac{x + \mathrm{e}^{0.5} \cdot 0.5}{\frac{\lvert 0.5^3 \rvert}{\cos x}} + \cos 0.618}{10^{x^2}}} + \mathrm{e}^{x - \ln \lvert (\lvert 0.618 - 0.5 \rvert) \rvert} + (0.811^2)^2;$
	0.0088	48	$y = \ln \lvert x^2 \cdot (10^{\frac{\mathrm{e}^{0.618}}{0.811}} - \pi) \rvert - \dfrac{\ln \left\lvert \lvert \cos\ x^3 \rvert^3 \right\rvert}{\sin 2} + (\sin(\cos(\cos x)) + x) \cdot \lvert \mathrm{e}^{1.112} \rvert \cdot$ $(x + \lvert 1.112^2 + x \rvert + (0.811 + \cos \pi)^2);$
	0.0078	57	$y = (10^x - \cos(1.112 \cdot \sin(\cos(10^\pi + 0.811^2)) - 10^{(\sqrt{0.5})^3} \cdot x)) \cdot (\mathrm{e}^{\frac{(\ln 0.5)^2}{0.811}} \cdot$ $(\cos \dfrac{x}{10^{0.811+0.618-x^2}})^3) + \ln(\lvert x \rvert \cdot 1.112 \cdot (\sqrt{2} - (0.811 - 2)) \cdot \pi \cdot \pi)^2.$

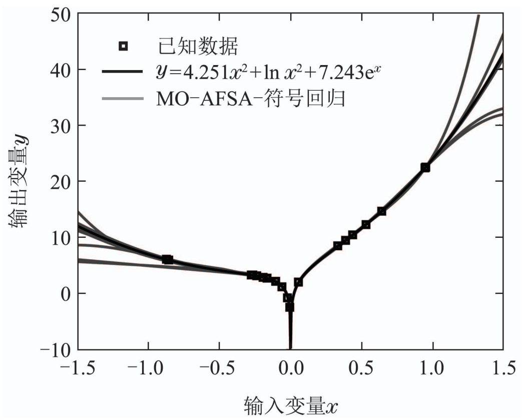


图 10 基于本文算法MO–AFSA的符号回归建模

Fig. 10 Symbolic regression modeling based on the proposed MO–AFSA

由于GEP, SL–GEP, AFSA仅考虑了最小化拟合误差而未对模型复杂度加以考虑, 因而得到的模型结构较为复杂, 尤其是SL–GEP和AFSA得到的函数模型, 虽然拟合误差很小, 但是复杂的模型结构难于解释和理解. 本文算法MO–AFSA以多目标优化的方式进行符号回归, 兼顾了拟合误差和模型复杂度, 有助于发现反映变量真实耦合关系的模型. 例如表4中由本文算法MO–AFSA获得的两个分别标注了“●”和“■”的函数模型, 不仅具有较小的拟合误差, 同时具有较小的模型复杂度. 若将这两个函数模型中出现的所有常数化为数值形式, 则分别得到

- $\bullet \to y = 4x^2 + \ln 0.8776x^2 + 7.3891\mathrm{e}^x$,
- $\blacksquare \to y = 4.4537x^2 + \ln 1.112x^2 + 7.1111\mathrm{e}^x$.

不难发现, 以上两个函数已经十分接近仿真算例的真实函数关系 $y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$. 尽管NSGP同样以多目标优化的方式进行符号回归, 但其并未得到复杂度与本文算法MO–AFSA所得模型相当的结果. NSGP得到的8个非支配解中有5个在模型复杂度的指标上甚至劣于SL–GEP和AFSA. 究其原因, 主要在于NSGP对语法树的交叉和变异操作容易造成代码膨胀(code bloat), 进而导致模型复杂度的大幅增加[11]. 这是GP系算法的固有问题, 仅通过多目标优化无法从根本上解决, 也反证了本文算法MO–AFSA在简化模型结构方面的优越性能. 在对算例$y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$的仿真测试中, GEP, SL–GEP, AFSA, NSGP等算法得到的函数模型均具有较满意的拟合误差, 但多数模型的结构较为复杂, 虽无明确证据表明这些复杂的模型出现了过拟合, 但与仿真算例的真实函数关系确有明显悖离.

图11展示了MO–AFSA得到的Pareto前沿与GEP, SL–GEP, AFSA, NSGP等对比算法得到的函数模型在由目标函数f_1和f_2张成的目标空间中的分布情况. 由图可见, 在MO–AFSA得到的Pareto前沿上总能找出一个或多个函数模型分别支配GEP, AFSA, 或NSGP得到的结果; 尽管SL–GEP能够得到一个不劣于MO–AFSA的结果, 但仅得到一个模型显然无法比拟MO–AFSA提供的模型多样性及可选择性. 本质上, MO–AFSA得到的Pareto前沿是对潜在函数模型的批量输出, 通过在Pareto前沿上进行合理选取, 能够得到更易解释的高质量函数模型.

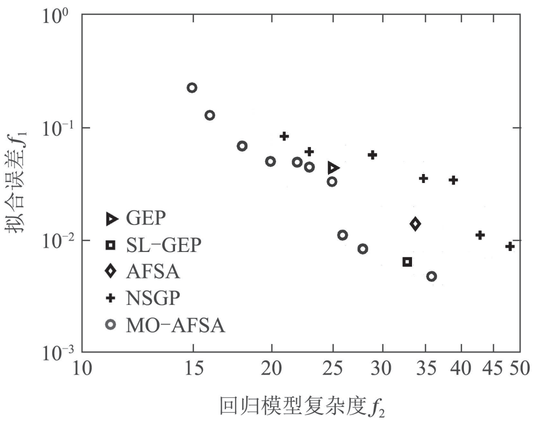


图 11 符号回归算法对算例$y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$的所得模型在目标空间的比较

Fig. 11 Comparison of the models obtained by the symbol regression algorithms for the example $y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$ in the objective-space

前文仿真算例$y = 4.251x^2 + \ln x^2 + 7.243\mathrm{e}^x$的建模难点主要在于$x = 0$处存在“尖点”(cusp), 而尖点两侧函数瓣的非线性程度并不高. 为考察本文算法MO–AFSA对强非线性模型的逼近性能, 从函数$y = \sin(x \cdot \cos(5x))$的曲线上随机选取15个点(见表5)作为训练样本构造了新的仿真算例, 并且同样与回归分析、神经网络、GEP, SL–GEP, AFSA, NSGP等已有的数据驱动建模方法进行了对比.

表 5 训练样本集: $y = \sin(x \cdot \cos(5x))$ 上随机选取的15个点

Table 5 Training set: 15 points randomly chosen from the function $y = \sin(x \cdot \cos(5x))$

序号	x	y	序号	x	y
1	0.005	0.005	9	1.391	0.8860
2	0.498	−0.3857	10	1.578	−0.0568
3	0.611	−0.5718	11	1.597	−0.2071
4	0.698	−0.6100	12	1.610	−0.3085
5	0.897	−0.2008	13	1.682	−0.7246
6	0.942	−0.0023	14	1.810	−0.9936
7	1.331	0.9458	15	1.952	−0.9631
8	1.385	0.8954			

图12和图13分别展示了两种回归分析方法和神经网络对$y = \sin(x \cdot \cos(5x))$的逼近结果. 相较而言, 高斯回归具有更好的逼近效果. 令人意外的是, 尽管采用了Bayes正则化技术, 拥有40个隐层节点的神经网络依然出现了严重的过拟合, 原因主要在于40个隐层节点的神经网络要求更多的训练样本, 而测试算例中的15个训练样本数量显然太少.

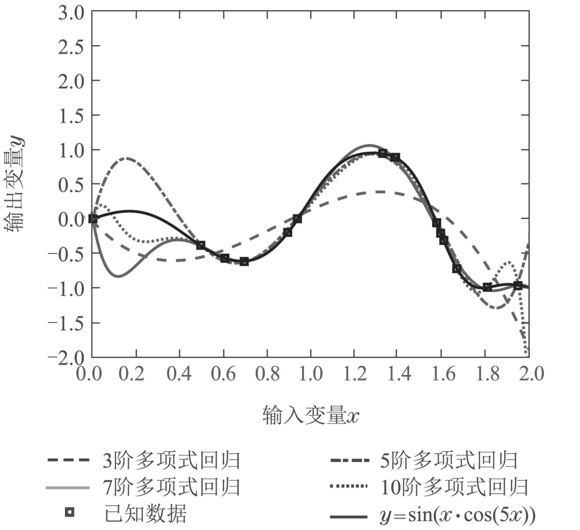


图 12(a) 基于多项式回归模型的回归分析

Fig. 12(a) Regression analysis based on polynomial model

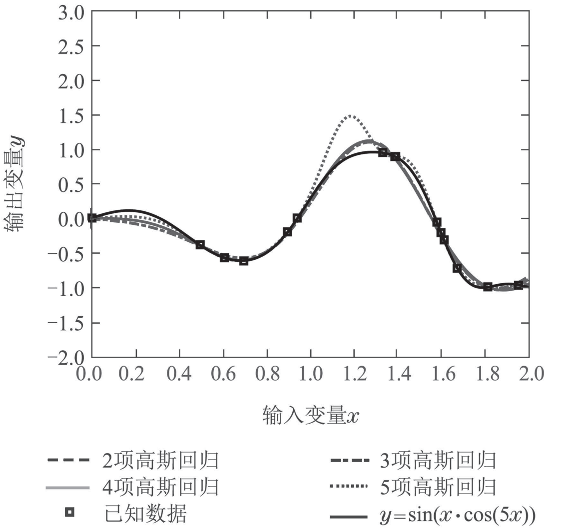


图 12(b) 基于高斯回归模型的回归分析

Fig. 12(b) Regression analysis based on Gaussian model

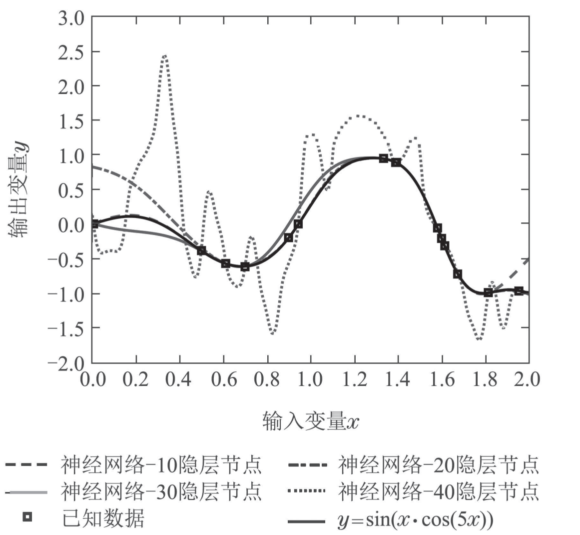


图 13 神经网络建模

Fig. 13 Neural network modeling

表6列出了各算法对算例$y=\sin(x\cdot\cos(5x))$的建模结果. 将表6中由本文算法MO–AFSA得到的6个非劣解模型以函数曲线的形式绘制于图14, 通过与算例函数$y=\sin(x\cdot\cos(5x))$真实曲线的比较, 可以看出本文算法MO–AFSA的拟合效果显著优于两种回归分析方法和神经网络, 同时也印证了MO–AFSA对强非线性模型良好的逼近性能. 例如表6中由本文算法MO–AFSA获得的两个分别标注了"●"和"■"的函数模型, 不仅具有较小的拟合误差, 同时具有较小的模型复杂度. 若将这两个函数模型中出现的所有常数化为数值形式, 则分别得到:

$$\bullet\rightarrow y=\sin(1.05x\cdot\cos(5.06x)),$$

$$\blacksquare\rightarrow y=\cos(1.612-x\cdot\cos(5x)),$$

其中: 模型$\bullet\rightarrow y=\sin(1.05x\cdot\cos(5.06x))$与算例的真实函数模型$y=\sin(x\cdot\cos(5x))$已经十分接近; 而$\blacksquare\rightarrow y=\cos(1.612-x\cdot\cos(5x))$中$1.612\approx\pi/2$, 根据诱导公式$\cos(\pi/2-\alpha)=\sin\alpha$, 该模型甚至在数学本质上已经与算例函数模型具有完全一致的形式.

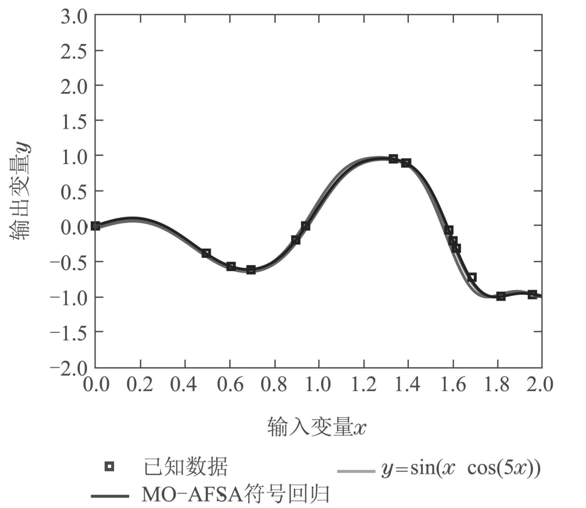


图 14 基于本文算法MO–AFSA的符号回归建模

Fig. 14 Symbolic regression modeling based on the proposed MO–AFSA

图15对GEP, SL–GEP, AFSA, NSGP等符号回归方法进行了比较, 将各算法得到的模型在由目标函数f_1和f_2张成的目标空间中进行了展示. 相较于GEP, SL–GEP, AFSA, NSGP等符号回归算法, MO–AFSA得到的函数模型普遍具有更为简洁的模型结构, 有利于模型的理解和解释. 对算例函数

$$y=\sin(x\cdot\cos(5x))$$

进行的仿真表明, 本文算法MO–AFSA不仅具备对强非线性模型的表达能力, 而且在简化模型结构方面具有显著优势.

表 6 各符号回归算法根据表5中训练样本集得到的函数模型

Table 6 Functional models obtained by each compared symbolic regression algorithm based on the training set listed in Table 5

对比算法	f_1	f_2	模型表达式
MO–AFSA	0.0587	14	$y=\sin(\sqrt{1.112}\cdot x\cdot\cos(1.112+\pi+0.811)\cdot x);$ ●
	0.0296	17	$y=\cos(1.112+0.5-x\cdot\cos(x\cdot 2+x+x\cdot 2));$ ■
	0.0107	21	$y=\ln \mathrm{e}^{\frac{\sin(\frac{\cos(\frac{x\cdot\sqrt{2}}{\cos\sqrt{\mathrm{e}^{0.5}}})}{1.112})}{\frac{x}{\ln \mathrm{e}^{1.112}}}};$
	0.00618744	22	$y=\ln \mathrm{e}^{\sin(\cos((x\cdot(2^3-2))-x)\cdot(x+0.5\cdot\sqrt{\sqrt{(\sin\pi)^3}}))};$
	0.00618501	24	$y=\sin(\lvert x+\ln \mathrm{e}^{\mathrm{e}^2\cdot\lvert\sin\pi\cdot 2^2\rvert}\rvert\cdot\cos(\frac{x}{\frac{0.5}{2}}+x));$
	0.0050	40	$y=\sin(\ln \mathrm{e}^{(\cos(2^2\cdot\lVert x\rVert+x)+\sqrt{\sin \mathrm{e}^{\sin(\sin 0.5)}}-(\frac{\sqrt{\lVert 0.618\rVert^2}}{0.618})^2)\cdot(\ln\cos\frac{((0.618)^2)^3}{\frac{\pi}{x}}+x)});$
SL–GEP[14]	0.0061693	40	$y=\sin(x\cdot\cos(0.618\times 1.112+\ln\lvert\cos 0.811\rvert+\frac{x}{0.618-0.811}+\frac{\ln\lvert 0.811\cdot(0.5-x)\rvert}{\frac{\pi}{\frac{x}{x\cdot x}+\pi\cdot 1.112}}));$
AFSA[16]	0.0062	48	$y=\sin((\frac{\lvert 0.811\rvert+\sqrt{\cos(0.5+0.618)}}{1.112}\cdot\lvert\frac{(\sqrt{((1.112)^3)^3}-10^{(0.811-0.811)^2})\cdot x}{0.811}\rvert)\cdot \cos\frac{x}{\frac{(2\cdot(\sqrt{0.5})^2+\sin\pi)^3}{10\cdot((\cos\pi)^2)^3\cdot 0.5}});$
GEP[24]	0.0581	40	$y=\sin(\sin(\frac{\cos(x\cdot 0.5\cdot\sqrt{10^2})-(10^{1.112}\cdot \mathrm{e}^{(\sqrt{\sqrt{\pi}}-\frac{\pi^3}{2})})}{\lvert\frac{\sqrt{0.811}}{(\sqrt{x})^2}\rvert\cdot \mathrm{e}^{\sqrt{(\cos(\cos 0.618))^3}}\cdot 0.618}));$
NSGP	0.0649	27	$y=\cos(x\cdot 1.112\cdot\ln(\sin(\mathrm{e}^2+\cos\pi))^2)\cdot x\cdot\cos(\sin(\mathrm{e}^{1.112}\cdot\cos\ln(x^3+1.112)));$
	0.0538	43	$y=\sin(x\cdot(\cos(0.618+(x\cdot(\sqrt{\ln 2+\sin 0.618+(1.112+1.112)\cdot(10^{\cos 0.811})^2}\rVert+ \pi+\ln 0.811)-\cos(\cos(\sin(\ln 0.811)))))-\mathrm{e}^{(\ln x-2)^3}));$
	0.0529164	50	$y=\frac{\sin(\ln\frac{\mathrm{e}^{\cos x\cdot ln2}\cdot 2}{(\sqrt{(\mathrm{e}^x)^3})^3})}{1.112}\cdot\frac{\lvert\sin(x\cdot(\frac{(\frac{\frac{\ln 10^{\lvert 1.112\rvert}}{0.618}}{10^{\pi}\cdot\ln\lvert 0.811\rvert})^3}{\frac{\cos x}{0.618}}-\cos(0.5\cdot\sin 1.112)))\rvert}{0.811};$
	0.0141512	59	$y=\sin(\cos\frac{\frac{x^3}{\lvert\sqrt{10^{\ln(\frac{1.112^2}{10^{\pi}}(x+1.112))}}\rvert}}{\lvert((\frac{0.811}{\cos\ln 1.112})^3+0.618+\cos\lvert x\rvert-\frac{\pi}{0.618})^2\rvert}\cdot x\cdot\cos((\frac{1.112}{\lvert x\rvert}\cdot \mathrm{e}^{\pi^2}-\lvert(\lvert\sqrt{1.112+\pi}\rvert)^3\rvert)\cdot x)).$

为进一步考察本文算法MO–AFSA的收敛情况，本文以表7列出的4个常用算例对算法进行测试，并以反向世代距离(inverted generational distance, IGD)作为量化算法收敛性能的指标对算法收敛性能进行考察. 由于表7中算例问题的真实Pareto前沿无从知晓，因而本文算法MO–AFSA对各算例分别独立运行30次，并以所得的30个Pareto前沿中的所有非支配解作为参考前沿(reference front)，据此计算各次运行的IGD.

图16以盒图的形式展示了本文算法MO–AFSA在4个算例问题上的IGD分布情况. 从图16可以看出，本文算法MO–AFSA在各算例上的IGD分布仅产生较少的离群点(outlier)，特别是在对F1, F2, F4三个算例盒图出现短“须”(whisker)，说明IGD序列的上四分位

数至上止点的分布也比较集中, 进一步表明本文算法MO–AFSA具有较好的收敛一致性.

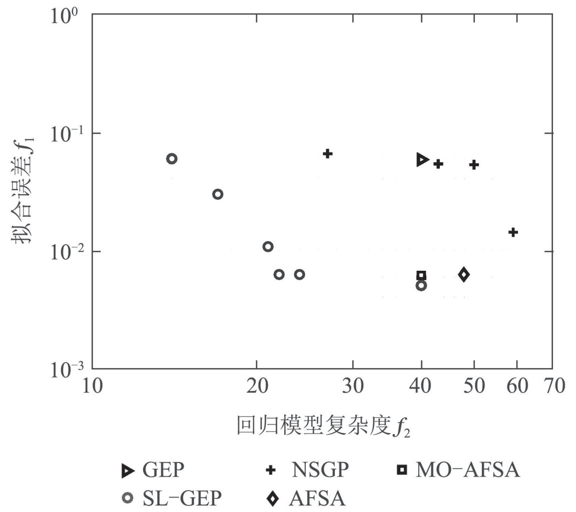


图 15 符号回归算法对算例$y = \sin(x \cdot \cos(5x))$的所得模型在目标空间的比较

Fig. 15 Comparison of the models obtained by the symbol regression algorithms for the example $y = \sin(x \cdot \cos(5x))$ in the objective-space

表 7 用于收敛性测试的符号回归标准算例

Table 7 Symbolic regression benchmarks for convergence test

P	真实函数关系	训练样本
$F1$	$y = 3 \cdot (x+1)^3 + 2 \cdot (x+1)^2 + (x+1)$	$U[-1, 1, 200]$
$F2$	$y = \ln(x+1) + \ln(x^2+1)$	$U[-1, 1, 200]$
$F3$	$y = \sin x + \sin(x + x^2)$	$U[-1, 1, 200]$
$F4$	$y = \sqrt{x}$	$U[0, 4, 200]$

备注 $U[a, b, c]$表示从区间$[a, b]$上以均匀分布从函数上随机选取c个样本

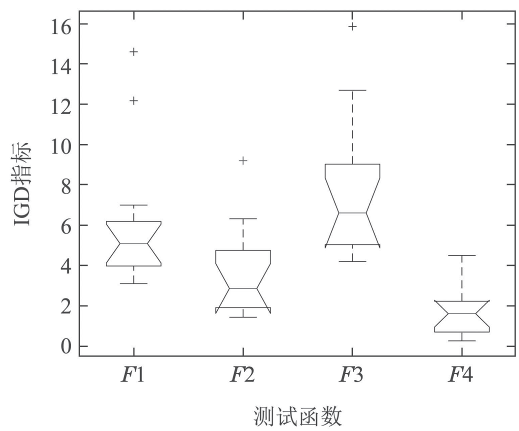


图 16 MO–AFSA对表7中4个benchmark问题的IGD值

Fig. 16 Corresponding IGD-value of MO-AFSA when solving the benchmarks listed in Table 7

5 讨论

5.1 模型的选择

本文算法MO–AFSA是以多目标优化的方式对能够拟合训练样本的潜在函数模型进行批量输出, 与GEP, SL–GEP, AFSA等已有符号回归算法相比, 其显著优势在于MO–AFSA能够批量输出高质量的潜在函数模型以供选择. 然而, 如何从获得的Pareto前沿上选取一个折衷解仍是一个开放性的问题. 本节提出一个可能的折衷解的选取方法并就此进行讨论.

在Pareto前沿上的非支配解中, 一些潜在函数模型通过容忍较大的拟合误差来简化模型结构, 另一些则通过提高模型复杂度来抑制拟合误差. 本文倾向于选择2个目标函数间最划算的折衷. 根据边际效用递减规律, 任何一个目标函数都不应被过分偏好. 由于整个目标空间被其原点(0, 0)支配, 可将目标空间原点(0, 0)视作一个基准点. 当一个潜在函数模型仅倾向于最小化一个优化目标而不是对两个优化目标进行平衡, 则该潜在函数模型一定远离基准点. 基于这样的认知, 本文建议在Pareto前沿上选取距离目标空间原点(0, 0)欧式距离最短的非支配解作为最终模型输出. 考虑到拟合误差与模型复杂度在量级上的显著差异可能会影响对两个目标的偏好, 笔者建议在使用前述方法对非支配解进行选取之前应首先根据式(5)对目标空间进行归一化, 其中: T_{right}表示Pareto前沿最右侧的解, T_{left}表示Pareto前沿最左侧的解, T_i表示要归一化的解.

需要强调的是, 如何从Pareto前沿上选取一个合适的折衷解在很大程度上取决于具体的应用场景. 上述方法仅作为一个通用的可替换方案给出.

$$\begin{cases} F_1(T_i) = \dfrac{f_1(T_i) - f_1(T_{\text{left}})}{f_1(T_{\text{right}}) - f_1(T_{\text{left}})}, \\ F_2(T_i) = \dfrac{f_2(T_i) - f_2(T_{\text{right}})}{f_2(T_{\text{left}}) - f_2(T_{\text{right}})}. \end{cases} \tag{5}$$

5.2 内存开销的降低

本文算法MO–AFSA在对语法树进行编码表达时采用了二叉堆结构. 该结构在表达语法树时具有优良的遗传稳定性, 有利于合理语法分支的遗传和继承. 此外, 由于从二叉堆编码的任意键值i处可直接根据$(2i+1)$和$(2i+2)$访问其后继节点, 因而解码可从根节点出发并以递归方式完成, 避免了采用ORF串时必须首先确定有效串长的计算负担. 以上优点是以较大的内存开销为代价. 具体说来, 要得到一个深度为h的语法树, 需要一个容量为$V = (2h-1)$的二叉堆. 同样深度的语法树, 采用二叉堆编码的内存开销高于ORF串编码. 针对这一问题, 本文的观点是: 一方面, 目前各种计算平台的内存资源已十分富余, 不会造成

资源瓶颈; 另一方面, 即使对于内存资源稀缺的应用场景, 亦可采取多树表达策略(multi-tree-based expression)减缓二叉堆编码对内存的消耗, 即: 利用多个语法树以级联耦合的方式实现深度语法树的表达. 这一策略已在GEP[12]中广泛应用, 因而可在本文算法MO–AFSA中直接使用, 从而显著降低算法的内存开销.

5.3 参数的选择

本文提出的MO–AFSA涉及多个参数, 本小节就这些参数的设置进行简要讨论.

1) 种群规模Pn.

MO–AFSA对解空间的搜索依赖于人工鱼个体间的协作, 因而要求人工鱼种群具备一定的规模. 以表7中F1为算例, 分别测试本文算法MO–AFSA在种群规模Pn分别取8, 14, 20, 40, 60, 80, 100时的收敛性能. 具体做法是保持其余参数不变, 对种群规模Pn的不同取值, 将本文算法分别独立运行30次并计算本文算法在不同Pn下的IGD值. 图17以盒图的形式对测的结果进行了展示. 可以看出, 种群规模Pn小于20的情况下, 收敛性能不佳, 这是因为过小的种群规模难以有效形成群体智能; 而当种群规模Pn取值达到20以上时, 算法能够保证较好的收敛性能. 不过, 考虑到过大的种群规模同时会增加算法耗时, Pn的合理取值范围一般在20至60之间.

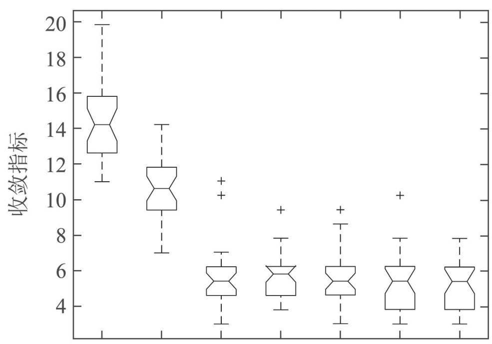


图 17 MO–AFSA的参数Pn对算法收敛性能的影响

Fig. 17 Influence of parameter Pn of MO–AFSA on convergence performance

2) 抽样次数$T_{\rm n}$.

$T_{\rm n}$是人工鱼对其所在邻域进行随机抽样搜索的次数上限. 为讨论$T_{\rm n}$对本文算法MO–AFSA收敛性能的影响, 同样以表7中$F1$为算例, 测试算法在$T_{\rm n}$各取值下的IGD值, 如图18所示. 不难看出, 当$T_{\rm n}$较小时, 人工鱼对其所在邻域的勘探并不十分充分, 因而算法未能达到较好的收敛性能; 而当$T_{\rm n} > 3000$时, 算法收敛性能较好, 这主要得益于人工鱼对其所在邻域勘探强度的提高. 需要强调的是, $T_{\rm n}$只是抽样次数的上限, 并不是实际抽样次数, 种群中多数人工鱼在觅食过程中进行随机抽样的次数实际远小于$T_{\rm n}$次. 根据经验, $T_{\rm n}$合理的取值范围在3000至6000之间.

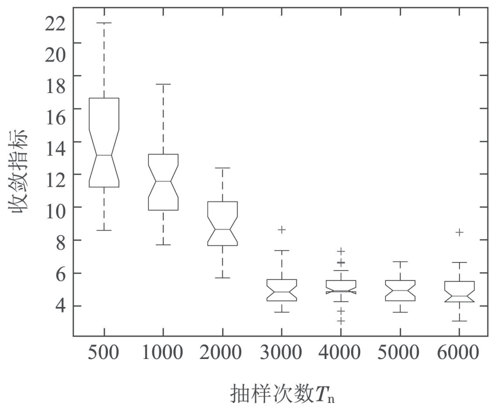


图 18 MO–AFSA的参数$T_{\rm n}$对算法收敛性能的影响

Fig. 18 Influence of parameter $T_{\rm n}$ of MO–AFSA on convergence performance

3) 随机步长范围[minStep, maxStep].

随机步长范围本质上是两个参数, 分别限定了人工鱼移动步长的上、下限. 为了验证随机步长范围对本文算法MO–AFSA收敛性能的影响, 仍以表7中F1为算例, 将算法在不同步长范围下的IGD分布情况以盒图形式绘制于图19. 可以看出, 算法在随机步长范围取$[1, 2]$, $[2, 4]$, $[3, 6]$时较好, 这与最优化理论的一般性结论相吻合, 即: 小步长的细粒度搜索有利于获得高质量解, 而大步长的粗粒度搜索则易使算法在极值区域震荡并停滞. 不过, 考虑到过小的步长可能减缓的收敛速度, 本文倾向于将步长范围设在区间$[3, 6]$, 即$\mathrm{minStep} = 3$和$\mathrm{maxStep} = 6$.

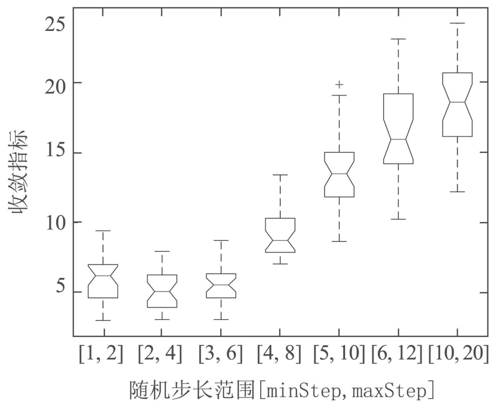


图 19 MO–AFSA的随机步长范围[minStep, maxStep]对算法收敛性能的影响

Fig. 19 Influence of stepsize range [minStep, maxStep] of MO–AFSA on convergence performance

4) 小生境容量$(2r + 1)$.

小生境的存在有助于提高局部种群的多样性. 图20以盒图形式展示了在小生境容量$(2r + 1)$的不同取

值下本文算法MO–AFSA对表7中F1算例进行求解时的IGD分布情况. 可以看出, 当取$(2r+1) \leqslant 7$时, 也即当$r=3$时, 算法收敛性能较好. 仿真中$Pn=40$, 因此小生境容量约为种群规模的1/6.

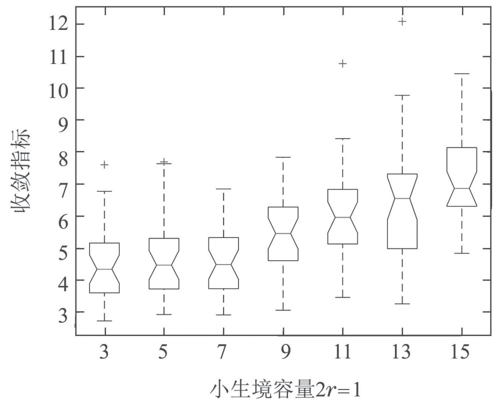


图 20 MO–AFSA的小生境容量$(2r+1)$对算法收敛性能的影响

Fig. 20 Influence of niche volume $(2r+1)$ of MO–AFSA on convergence performance

尽管本文提出的MO–AFSA在编码方法、行为算子的实现方法等诸多方面已显著区别于传统AFSA[21], 并且引入了蒙版、小生境等概念, 但并未颠覆传统AFSA的一般范式, 算法仍然通过个体底层行为的执行涌现群体智能, 因而其参数设置原则与一般AFSA并无显著不同. 此外, 提出的MO–AFSA对其参数并不十分敏感, 因此可在相对宽泛的范围内取值.

6 结论

本文聚焦于现有符号回归算法仅最小化拟合误差而忽略模型结构简化的问题, 提出了一种基于多目标人工鱼群算法的符号回归方法. 算法将拟合误差与模型复杂度同时作为目标函数对模型进行优化. 针对ORF串编码对语法树表达不稳定且在解码时需预先计算有效编码长度等缺点, 采用二叉堆结构对语法树进行存储与表达, 不仅使语法树的优良分支得以稳定地遗传和继承, 而且方便了解码. 本文在引入蒙版、邻域、小生境、拥挤度等概念的基础上, 定义了随机游动、觅食、追尾、逃脱等人工鱼行为算子. 通过详实的算例测试及横向对比, 验证了提出算法解决符号回归问题的有效性和优越性. 此外, 本文也对从Pareto前沿上选取折衷解的方法以及降低算法内存开销的方法进行了讨论. 本文研究适用于根据观测数据对变量间耦合关系进行辨识和构建, 从而发现系统运行规律或事物内在联系的数据驱动建模场景.

致谢 感谢我系2017级硕士研究生李星在SL–GEP算法实现上的部分仿真工作以及在Latex排版方面的热心协助.

参考文献:

[1] LI Yongming, WANG Liang, CHEN Xianglin, et al. Regression analysis of coincidence measurements for determinating the neutron emission rate of ^{252}Cf spontaneous fission. *Acta Physica Sinica*, 2018, 67(24): 242901-1 – 242901-6.
(李永明, 王亮, 陈想林, 等. ^{252}CF自发裂变中子发射率符合测量的回归分析. 物理学报, 2018, 67(24): 242901-1 – 242901-6.)

[2] XIE Yi, SONG Zhenhai, SHI Rian. Analysis and model of partial least squares of regression on form factor of deep-vee vessels. *Systems Engineering — Theory & Practice*, 2013, 33(6): 1628 – 1632.
(谢宜, 宋振海, 史日安. 排水型深V型船形因子的偏最小二乘回归分析与建模. 系统工程理论与实践, 2013, 33(6): 1628 – 1632.)

[3] ZHANG Dongming, ZHANG Xiaoyun, YANG Xiaobo, et al. Parameter analysis of vehicle-predestrian accidents in untypical contact state based on orthogonal tests and polynomial regression analysis. *Journal of Shanghai Jiao Tong University*, 2019, 53(1): 55 – 61.
(张东明, 张晓云, 杨小波, 等. 基于正交实验和多项式回归分析方法的非典型接触状态车人碰撞事故参数分析. 上海交通大学学报, 2019, 53(1): 55 – 61.)

[4] WANG Jianpei, HUANG Chunyue, LIANG Ying, et al. Termal stress analysis and optimization of BGA solder joint power load based on regressioin analysis and genetic algorithm. *Acta Electronica Sinica*, 2019, 47(3): 734 – 740.
(王建培, 黄春跃, 梁颖, 等. 基于回归分析和遗传算法的BGA焊点功率载荷热应力分析与优化. 电子学报, 2019, 47(3): 734 – 740.)

[5] LI Peiqiang, ZENG Xiaojun, LI Xinran, et al. Unified equivalent modeling of distributed generation using artificial neural network and its application in PSASP. *Power System Technology*, 2016, 40(4): 1224 – 1230.
(李培强, 曾小军, 李欣然, 等. 基于神经网络的分布式电源统一等效建模及其在PSASP中的应用. 电网技术, 2016, 40(4): 1224 – 1230.)

[6] CHEN Xiao, WANG Ning. Fuzzy recurrent neural network modeling based on chaos DNA genetic algorithm. *Control Theory & Applications*, 2011, 28(11): 1589 – 1594.
(陈霄, 王宁. 基于混沌DNA遗传算法的模糊递归神经网络建模. 控制理论与应用, 2011, 28(11): 1589 – 1594.)

[7] BARMPALEXIS P, KACHRIMANIS K, TSAKONAS A, et al. Symbolic regression via genetic programming in the optimization of a controlled release pharmaceutical formulation. *Chemometrics & Intelligent Laboratory Systems*, 2011, 107(1): 75 – 82.

[8] XU Jing, WANG Qiuwang, ZENG Min. Improvement of genetic programming symbolic regression and its application in heat exchanges. *Journal of Engineering Thermophysics*, 2012, 33(8): 1415 – 1418.
(徐婧, 王秋旺, 曾敏. 符号回归在换热关联式求解中的应用. 工程热物理学报, 2012, 33(8): 1415 – 1418.)

[9] YANG Chunyu, LI Heng, CHE Zhiyuan. Energy consumption modeling and parameter identification for double-motor driven coal mine belt conveyers. *Control Theory & Applications*, 2018, 35(3): 335 – 341.
(杨春雨, 李恒, 车志远. 煤矿双电机驱动带式输送机的能耗建模与参数辨识. 控制理论与应用, 2018, 35(3): 335 – 341.)

[10] KOZA J. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambrige, MA: MIT Press, 1992.

[11] SOULE T, FOSTER J A, DICKINSON J. Code growth in genetic programming. *Proceedings of the 1st Annual Conference on Genetic Programming*. Cambridge, MA: MIT Press, 1996.

[12] FERREIRA C. Gene expression programming: A new adaptive algorithm for solving problems. *Complex Systems*, 2001, 13(2): 87 – 129.

[13] LI X, ZHOU C, Xiao W M, et al. Prefix gene expression programming. *Genetic and Evolutionary Computation Conference(GECCO)’05*. Washington, DC, USA: ACM Press, 2005.

[14] ZHONG J H, ONG Y S, CAI W T. Self-learning gene expression programming. *IEEE Transactions on Evolutionary Computation*, 2016, 20(1): 65 – 80.

[15] PENG Y Z, YUAN C A, QIN X, et al. An improved gene expression programming approach for symbolic regression problems. *Neurocomputing*, 2014, 137(15): 293 – 301.

[16] LIU Q, ODAKA T, KUROIWA J, et al. Application of an artificial fish swarm algorithm in symbolic regression. *IEICE Transactions on Information and Systems*, 2013, E96-D(4): 872 – 885.

[17] KARABOGA D, OZTURK C, KARABOGA N, et al. Artificial bee colony programming for symbolic regression. *Information Sciences*, 2012, 209(22): 1 – 15.

[18] QI F, MA Y, LIU X, et al. A hybrid genetic programming with particle swarm optimization. *International Conference in Swarm Intelligence*. Berlin Heidelberg: Springer, 2013: 11 – 18.

[19] HOLLAND J. *Adaptation in Natural and Artificial Systems*. Ann Arbor, USA: University of Michigan Press, 1992.

[20] JIANG Dazhi, WU Zhijian, KANG Lishan, et al. New method used in gene expression programming: GRCM. *Journal of System Simulation*, 2006, 18(6): 1466 – 1468.
(姜大志, 吴志健, 康立山. 基因表达式程序设计的GRCM方法. 系统仿真学报, 2006, 18(6): 1466 – 1468.)

[21] LI Xiaolei, SHAO Zhijiang, QIAN Jixin. An optimizing method based on autonomous animats: fish-swarm algorithm. *System Engineering Theory and Practice*, 2002, 22(11): 32 – 38.
(李晓磊, 邵之江, 钱积新. 一种基于动物自治体的寻优模式:鱼群算法. 系统工程理论与实践, 2002, 22(11): 32 – 38.)

[22] XIA Lirong, LI Runxue, LIU Qiyu, et al. An adaptive multi-objective particle swarm optimization algorithm based on dynamic AHP and its application. *Control and Decision*, 2015, 30(2): 215 – 221.
(夏立荣, 李润学, 刘启玉, 等. 基于动态层次分析的自适应多目标粒子群优化算法及其应用. 控制与决策, 2015, 30(2): 215 – 221.)

[23] COELLO C A C, PULIDO G T, LECHUGA M S. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 2004, 8(3): 256 – 279.

[24] FERREIRA C. Gene expression programming in problem solving. *Soft Computing and Industry*. London: Springer, 2002: 635 – 653.

[25] GONG Jie, TANG Changjie, XU Kaikuo, et al. CC–GEP: a novel algorithm for gene expression programming based on cluster competition. *Journal of Sichuan University (Natural Science Edition)*, 2010, 47(3): 530 – 536.
(巩杰, 唐常杰, 徐开阔, 等. CC–GEP: 基于聚类竞争的基因表达式编程新算法. 四川大学学报(自然科学版), 2010, 47(3): 530 – 536.)

[26] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multiobjective genetic algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 2002, 6(2): 182 – 197.

作者简介:

刘 庆 博士, 讲师, 主要研究方向为数据驱动建模、群智能优化, E-mail: liuqing@xaut.edu.cn;

任海鹏 博士, 教授, 主要研究方向为复杂系统分析与控制、混沌无线通信, E-mail: renhaipeng@xaut.edu.cn;

姚俊良 博士, 讲师, 主要研究方向为大规模MIMO信道建模、混沌理论及其在无线通信中的应用, E-mail: yaojunliang@xaut.edu.cn;

刘 龙 博士, 教授, 主要研究方向为机器视觉、机器学习, E-mail: liulong@xaut.edu.cn.