Оптимизция параметров нечеткого аппроксиматора на основе алгоритма стаи ласточек

А. О. Слезкин

Научный руководитель И. А. Ходашинский, профессор каф. КИБЭВС

Томский государственный университет систем управления и радиоэлектроники

Кафедра комплексной информационной безопасности электронно-вычислительных систем

Проект ГПО КИБЭВС-1211 – Нечеткие системы

Введение

Нечеткие системы давно и успешно применяются в таких проблемных областях, как классификация, аппроксимация, интеллектуальный анализ данных, распознавание образов, прогнозирование и управление, в задачах принятия решений. Они встроены в огромное количество промышленных изделий, начиная с роботов и систем управления электропоездами, и заканчивая такими потребительскими товарами, как фото и видеокамеры, кондиционеры, стиральные машины и др. Достоинствами нечетких систем является невысокая стоимость разработки, гибкость, интуитивно понятная логика функционирования.

Целью работы является реализация метаэвристического алгоритма стаи ласточек (Swallow Swarm Optimization) [1] для оптимизации нечеткого аппроксиматора типа синглтон и сравнение результатов с результатами других алгоритмов [2] на наборах данных KEEL.

Постановка задачи

В данной работе рассматривается нечеткий аппроксиматор типа синглтон.

Построение нечеткой системы осуществляется на основе таблицы наблюдений:

$$T = \{t_1, t_2, \dots, t_M\},$$

где $t_k = (\boldsymbol{x}_k, y_k)$ — строка таблицы наблюдений, $\boldsymbol{x}_k = (x_1, x_2, \dots, x_n)$ — вектор значений входных переменных, y_k — выходное значение, M — число наблюдений, n — число входных переменных.

На основе таблицы наблюдений на области определения каждой входной переменной формируются нечеткие термы.

Исходя из разбиения переменных на нечеткие термы формируется база правил нечеткой системы. Нечеткое правило в нечеткой системе типа синглтон имеет вид:

$$IF\ x_1 = A_{1j} AND\ x_2 = A_{2j} AND \dots AND\ x_n = A_{nj}\ THEN\ y = r_j,$$

где r_j — действительное число, которым оценивается выход y, $j = \overline{1 \dots m}$, m — количество правил, A_{ij} — терм j-го правила i-ой переменной [3].

На основе сформированной базы правил нечеткая система осуществляет отображение F: $\Re^n \longrightarrow \Re$, определяемое формулой:

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{m} \mu_{A_{1i}}(x_1)\mu_{A_{2i}}(x_2)\dots\mu_{A_{ni}}(x_n) r_i}{\sum_{i=1}^{m} \mu_{A_{1i}}(x_1)\mu_{A_{2i}}(x_2)\dots\mu_{A_{ni}}(x_n)},$$

где $\mathbf{x} = [x_1, \dots, x_n] \in \Re^n$, $F(\mathbf{x}) \in \Re$, $\mu_{A_{ij}}$— функция принадлежности, указывающая степень принадлежности x_i к множеству A_{ij}.

Нечеткий аппроксиматор может быть представлен как:

$$y\ =\ f(\mathbf{x}, \boldsymbol{\theta}, \mathbf{r}),$$

где $\mathbf{x}$ — входной вектор $\boldsymbol{\theta}$ — вектор параметров антецедентов, N — число параметров, описывающих одну функцию принадлежности, y — скалярный выход системы, $\mathbf{r} =$

$(r_1, \dots, r_m)$ — вектор параметров консеквентов. В качестве численного критерия адекватности модели используется среднеквадратическая функция ошибки, вычисляющаяся по формуле:

$$MSE(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{r}) = \frac{1}{2} * \frac{\sum_{k=1}^{M} \left(y_k - f(\mathbf{x}_k, \boldsymbol{\theta}, \mathbf{r})\right)^2}{M} \ .$$

Описание алгоритма

Алгоритм стаи ласточек является продолжением идеи роящихся частиц и основан на поведении стаи ласточек в привычной им среде обитания.

Главная идея алгоритма состоит в разделении всех частиц на три вида:

- частицы лидеры,
- частицы исследователи,
- бесцельные частицы.

Роли зависят от положения частицы. Считается, что чем меньше значение фитнесс-функции *MSE*, тем лучше положение.

Частицы лидеры делятся на глобального лидера и локальных лидеров. Глобальным лидером является частица с самым лучшим положением в проблемном пространстве. Локальные лидеры — это l следующих за глобальным лидером частиц. Частицы исследователи — k худших частиц популяции. Все остальные частицы — исследователи.

Частицы лидеры во время итерации не двигаются, а служат ориентиром для исследователей.

Частицы исследователи изучают проблемное пространство между ближайшим локальным лидером и глобальным лидером. Для изменения их положения используются формулы:

$$\boldsymbol{\theta}_{e_{i+1}} = \boldsymbol{\theta}_{e_i} + \mathbf{V}_{i+1},$$

$$\mathbf{V}_{i+1} = \mathbf{VHL}_{i+1} + \mathbf{VLL}_{i+1},$$

$$\mathbf{VHL}_{i+1} = \mathbf{VHL}_i + \alpha_{HL} * rand(0{,}1) * \left(\boldsymbol{\theta}_{e_{best}} - \boldsymbol{\theta}_{e_i}\right) + \beta_{HL} * rand(0{,}1) * (\mathbf{HL} - \boldsymbol{\theta}_{e_i}),$$

$$\mathbf{VLL}_{i+1} = \mathbf{VLL}_i + \alpha_{LL} * rand(0{,}1) * \left(\boldsymbol{\theta}_{e_{best}} - \boldsymbol{\theta}_{e_i}\right) + \beta_{LL} * rand(0{,}1) * (\mathbf{LL}_i - \boldsymbol{\theta}_{e_i}),$$

где $\boldsymbol{\theta}_{e_i}$— положение i-ой частицы-исследователя, $\mathbf{HL}$ — положение глобального лидера, $\mathbf{LL}_i$ — положение ближайшего до $\boldsymbol{\theta}_{e_i}$ локального лидера, $\boldsymbol{\theta}_{e_{best}}$— лучшее положение $\boldsymbol{\theta}_{e_i}$, α_{HL}, β_{HL} — параметры, контролирующие скорость приближения к $\mathbf{HL}$, α_{LL}, β_{LL} — параметры, контролирующие скорость приближения к $\mathbf{LL}$, $\mathbf{V}_i$ — вектор скорости $\boldsymbol{\theta}_{e_i}$, $\mathbf{VHL}$ — вектор скорости частицы относительно $\mathbf{HL}$, $\mathbf{VLL}$ — вектор скорости частицы относительно LL.

Бесцельные частицы перемещаются в случайное место проблемного пространства. Для этого используется формула:

$$\boldsymbol{\theta}_{o_{i+1}} = \boldsymbol{\theta}_{o_i} + rand(\{-1; 1\}) * \frac{rand(min_s, max_s)}{1 + rand(0{,}1)},$$

где $\boldsymbol{\theta}_{o_i}$— положение i-ой бесцельной частицы, min_s, max_s — границы исследуемого пространства.

После каждой итерации частицы снова сортируются по возрастанию их фитнесс-функции и их роли переопределяются. После выполнения условия останова алгоритма, его работа завершается и возвращается положение частицы с наименьшим значением фитнесс-функции.

Возникшие при реализации проблемы

В ходе программной реализации данного алгоритма были обнаружены некоторые его проблемы. Коэффициенты α_{HL}, β_{HL} ,α_{LL}, β_{LL} получались слишком большими, что приводило к хаотичному движению частиц, вместо приближения их к глобальному минимуму. Второй проблемой стало то, что использование оригинальной формулы алгоритма для изменения положения бесцельных частиц приводило к выходу этих частиц, в некоторый момент, за пределы исследуемого пространства.

Решение возникших проблем

Формула движения бесцельных частиц была изменена на:

$$\boldsymbol{\theta}_{o_{i+1}} = rand(\{0,5; 1,5\}) * \mathbf{VSS}$$

$$\mathbf{VSS} = \frac{\sum_{i=1}^{N-k} \boldsymbol{\theta}_i}{N-k}$$

где $\boldsymbol{\theta}_i$ — i-ая частица популяции, N — количество всех частиц в популяции, k — количество бесцельных частиц.

Коэффициеты, контролирующие скорость приближения к лидерам, были приравнены к константам, подобранным опытным путем.

Полученные результаты

Для проведения тестирования работы алгоритма были выбраны следующие наборы данных из репозитория KEEL: plastic, ele-2, delta-ail, friedman, dee. Размер популяции был равен 50, количество локальных лидеров – 3, количество бесцельных частиц – 4, количество итераций – 100. После тестирования было проведено сравнение полученных результатов с результатами работы других алгоритмов (табл. 1) [2].

Таблица 1 – Сравнение результатов работы алгоритмов

Data	ANFIS-SUB		TSK-IRL		Linear-LMS		LEL-TSK		METSK-HD		SSO	
	trn	test	trn	test	trn	test	trn	test	trn	test	trn	test
pla	1,011	1,504	1,090	1,146	1,166	1,172	1,032	1,188	1,057	**1,136**	1,343	1,433
ele-2	8208	8525	17024	19786	13361	13541	2928	3752	2270	**3192**	17149	15589
delail	0,973	1,484	1,321	1,419	1,478	1,478	1,193	1,760	1,190	**1,402**	1,903	1,868
fried	0,085	3,158	0,433	1,419	3,653	435	0,322	**1,070**	1,075	1,888	1,566	2,089
dee	3087	2083	0,545	882,02	0,081	**0,085**	0,662	0,628	0,030	0,103	0,0824	0,0888

Заключение

Хоть ни по одному из опробованных наборов данных и не удалось получить самого лучшего результата, используя алгоритм стаи ласточек, уже сейчас, полученные результаты лучше результатов работы одних алгоритмов и сравнимы с результатами других. На это так же повлияло и то, что некоторые части алгоритма пришлось самостоятельно исправлять, ввиду их плохой работоспособности. Из вышесказанного можно сделать вывод, что при дальнейших улучшениях алгоритма и использовании его для оптимизации более сложных систем, чем аппроксиматор типа синглтон(например нечеткого аппроксиматора типа Такаги-Сугено), можно ожидать заметного улучшения выходных результатов.

Список литературы

1. Neshat M., Sepidnam G., Sargolzaei M. Swallow swarm optimization algorithm: a new method to optimization // Neural Computing and Application. – 2013. V. 23, N. 2. – P. 429–454.
2. Gacto M.J., Galende M., Alcalá R., Herrera F. METSK-HDe: A multiobjective evolutionary algorithm to learn accurate TSK-fuzzy systems in high-dimensional and large-scale regression problems // Information Sciences. – 2014. – V. 276. – P. 63–79.
3. Ходашинский И.А. Идентификация нечетких систем: методы и алгоритмы // Проблемы управления. – 2009. – № 4. – С. 15–23.