СБОРНИК **НАУЧНЫХ ТРУДОВ НГТУ. – 2018. – № 3–4 (93).** – 101–115

ОБРАБОТКА ИНФОРМАЦИИ

УДК 519.27(7) DOI: 10.17212/2307-6879-2018-3-4-101-115

ПОМЕХОУСТОЙЧИВЫЙ ВАРИАНТ АЛГОРИТМА ГРАВИТАЦИОННОГО ПОИСКА ГЛОБАЛЬНОГО МИНИМУМА*

В.С. ВОРОНОВ

630074, РФ, г. Красноярск, ул. академика Киренского, 26, Сибирский федеральный университет, Институт космических и информационных технологий, магистрант кафедры информатики. E-mail: voronov.volodya2013@yandex.ru

В работе рассматривается задача глобальной оптимизации многоэкстремальных функций с помощью алгоритма гравитационного поиска. Классический алгоритм является стохастическим и основан на гравитационном взаимодействии совокупностей масс (зондов) и законах движения. На основе сил определяются векторы скорости и ускорения каждого зонда для дальнейшего его перемещения. Данный алгоритм схож с методом роя частиц, так как базируется на развитии многоагентной системы. В ходе исследования классического алгоритма был выявлен недостаток, связанный с сильным снижением оценки вероятности попадания в глобальный экстремум при наличии аддитивной равномерно распределенной помехе. В связи с этим предложен модифицированный алгоритм. Улучшение основано на внедрении ядерной функции, что позволяет лучше выделять истинный экстремум при воздействии помехи, а также увеличить точность решения при ее отсутствии. Также предлагается использовать динамический закон изменения количества зондов, что повлечет сокращение количества измерений целевой функции по сравнению со статическим количеством точек. Для демонстрации улучшения качества поиска модифицированным алгоритмом в работе приведен пример тестовой функции, сконструированной по методу Бочарова–Фельдбаума и имеющей 10 экстремумов, один из которых является глобальным. Сравнительное исследование двух алгоритмов проводится в одинаковых условиях как без помехи, так и с постоянно нарастающей амплитудой, вплоть до 10-кратного превышения уровня шума над полезным сигналом. Описанные модификации также хорошо подходят для широкого круга алгоритмов как роевого интеллекта, так и других.

Ключевые слова: глобальная оптимизация, метод оптимизации, алгоритм, гравитационный поиск, аддитивная помеха, закон гравитации, ядерная функция, метод Бочарова–Фельдбаума, роевой интеллект, многоэкстремальная функция

* Статья получена 12 сентября 2018 г.

ВВЕДЕНИЕ

Задача поиска глобального экстремума целевой функции многих переменных [1] относится к классу сложных. Специфика глобальной оптимизации обусловлена решением практических задач: проектирование конструкций, протекание процессов, выбор оптимальных условий, режимов и параметров функционирования машин и механизмов [2]. Для данной проблемы характерны наличие большого количества переменных непрерывного, дискретного и смешенного типов, многоэкстремальный вид целевой функции, ее разрывный характер, ограниченность множества возможных значений искомых переменных. В реальных условиях воздействие помех при измерении целевой функции встречается повсеместно, будь то погрешность измерительного прибора или воздействие внешних условий. Это в совокупности с незначительной достоверной априорной информацией об объекте влечет за собой существенное ухудшение не только точности решения, но и вероятности нахождения глобального экстремума.

Однако большинство методов и алгоритмов глобальной оптимизации не имеет механизмов, позволяющих нивелировать воздействие помехи, что не позволяет использовать их в задачах, подразумевающих воздействие шума. В связи с этим борьба с помехами и разработка помехоустойчивых алгоритмов являются актуальными направлениями исследований.

Алгоритмы роевого интеллекта [3–5] являются одним из самых распространенных классов алгоритмов. Из данного класса был выбран алгоритм гравитационного поиска как перспективный вариант развития и широко применяемый инструмент в практических задачах (машинное обучение, выбор оптимальных конструкций и др.) [6, 7]. Он показывает лучшие оценку вероятности попадания в глобальный экстремум и скорость сходимости, чем алгоритм роя частиц или генетический алгоритм с вещественным кодированием [8], а также обладает простой структурой.

Алгоритм гравитационного поиска основан на законе гравитационного взаимодействия и законах движения. Он использует теорию ньютоновской физики, где агентами выступают совокупности масс. При этом каждый агент, находясь в изолированной системе, может обмениваться информацией с другими агентами путем силы притяжения, зависящей от массы (основана на значении целевой функции) и расстояния между ними.

В статье представлены следующие разделы. В разделе 1 приведен перечень некоторых обозначений, которые используются в статье. Раздел 2 посвящен краткому описанию стандартного алгоритма гравитационного поиска. В разделе 3 описывается модификация алгоритма глобальной оптимизации

для работы в условиях воздействия помех. В разделе 4 приведен аналитический вид и графическое представление тестовой функции. Заключительный раздел включает результаты тестирования.

1. ПОСТАНОВКА ЗАДАЧИ

Пусть дана целевая функция $f(x)$ и некая область X, где находится единственный глобальный минимум, тогда задачу оптимизации можно записать следующим образом:

$$f(x) = \min_{x \in X}.$$

Здесь $f(x)$ – целевая функция, X – прямоугольная область поиска глобального экстремума, $x = (x^{(1)}, x^{(2)}, \ldots, x^{(d)}, \ldots, x^{(n)})$ – точка, расположенная в области поиска X (n – размерность задачи оптимизации, т. е. количество переменных).

2. КЛАССИЧЕСКИЙ АЛГОРИТМ ГРАВИТАЦИОННОГО ПОИСКА

Оригинальный алгоритм представляет собой последовательность шагов [8]:

1) инициализация системы зондов случайным образом в области поиска:

$$x_{i,0}^{(d)} = u_i^{(d)}(h^{(d)} - l^{(d)}) + l^{(d)}, \quad d = \overline{1,n}, \quad i = \overline{1,N}, \quad u_i^{(d)} \in [0,1],$$

здесь d – номер параметра; $h^{(d)}$ – ограничение сверху; $l^{(d)}$ – ограничение снизу; u_i – значение случайной величины, имеющей равномерный закон распределения; N – количество точек;

2) расчет значений целевой функции (функции качества) для каждой точки $f_i = f(x_{i,t})$;

3) расчет значения гравитационной постоянной:

$$G_t = G_0 \exp\left(-\alpha \frac{t}{T}\right), \quad \alpha > 0, \tag{1}$$

где α – свободный параметр алгоритма; t – номер текущей итерации; T – общее число итераций; G_0 – начальное значение гравитационной постоянной, настраиваемый параметр алгоритма;

4) нахождение масс (весов) зондов осуществляется по следующей формуле:

$$m_{i,t} = \frac{\mu_{i,t}}{\sum_{i=1}^{N} \mu_{i,t}}, \quad \mu_{i,t} = \frac{f(x_{i,t}) - \hat{f}_{\max,t}}{\hat{f}_{\min,t} - \hat{f}_{\max,t}}, \quad i = \overline{1,N}, \quad t = \overline{1,T}, \tag{2}$$

здесь $\hat{f}_{\max,t}$ – максимальное значение оптимизируемой функции на текущей итерации; $\hat{f}_{\min,t}$ – минимальное значение оптимизируемой функции на текущей итерации;

5) вычисление сил гравитационного взаимодействия между зондами и нахождение результирующей силы для каждого зонда:

$$F_{i,j,t}^{(d)} = G_t \frac{m_{i,t} m_{j,t}}{r_{i,j,t} + \varepsilon}\left(x_{j,t}^{(d)} - x_{i,t}^{(d)}\right), \quad r_{i,j,t} = \sqrt{\sum_{d=1}^{n}\left(x_{j,t}^{(d)} - x_{i,t}^{(d)}\right)^2},$$

$$F_{i,t}^{(d)} = \sum_{i=1, i\neq j}^{N} u_i^{(d)} F_{i,j,t}^{(d)}, \quad i, j = \overline{1,N}, i \neq j, \quad t = \overline{1,T},$$

здесь ε – малая константа или машинный ноль, т. е. минимальное числовое значение, для которого выполняется условие $1 + \varepsilon > 1$, например, для языка программирования Python 3.6 для чисел типа float $\varepsilon \approx 2{,}2 \cdot 10^{-16}$;

6) вычисление ускорения (3) и скорости (4) для каждого зонда:

$$a_{i,t}^{(d)} = \frac{F_{i,t}^{(d)}}{m_i}, \tag{3}$$

$$v_{i,t+1}^{(d)} = u_i v_{i,t}^{(d)} + a_{i,t}^{(d)}, \quad v_{i,0}^{(d)} = 0, \quad d = \overline{1,n}, \quad i = \overline{1,N}; \tag{4}$$

7) расчет новых координат зондов производится по следующей формуле:

$$x_{i,t+1}^{(d)} = x_{i,t}^{(d)} + v_{i,t+1}^{(d)};$$

8) если новое положение точки x_i оказывается вне области X, то вернуть точку можно одним из двух вариантов: либо заново случайно расположить в области X, либо расположить на границе:

$$x_{i,t+1}^{(d)} = \begin{cases} l^{(d)}, & x_{i,t+1}^{(d)} \le l^{(d)}, \\ h^{(d)}, & x_{i,t+1}^{(d)} \ge h^{(d)}; \end{cases}$$

9) повторение шагов 2–8 до выполнения условия останова.

3. МОДИФИЦИРОВАННЫЙ АЛГОРИТМ

Для борьбы с аддитивной помехой, имеющей равномерное распределение и присутствующей в измерениях целевой функции, были протестированы следующие модификации:

а) добавление буфера для измерений различной длины и использование в качестве значения целевой функции среднего значения по буферу;

б) добавление буфера и использование медианы;

в) добавление ядерной функции [9] и использование в качестве аргумента усредненного значения функции;

г) различные сочетания данных способов.

Практика показывает, что лучшим способом для борьбы с помехой является ядерная функция. Несмотря на то что добавление буфера увеличивает количество измерений, данный подход дает результат гораздо хуже, нежели введение ядерной функции. Кроме этого, подходы (а) и (б) ухудшают результаты при использовании в любом сочетании вместе с ядерной функцией, в связи с чем далее будет описан метод интеграции ядерной функции.

Примерами ядерных функций являются:

а) линейные, параболические, кубические и им подобные функции:

$$p_s(g) = (1 - g^r)^s, \quad r = 1, 2, 3, \ldots, \quad s > 0; \tag{5}$$

б) экспоненциальные:

$$p_s(g) = \exp(-sg), \quad s > 0; \tag{6}$$

в) гиперболические:

$$p_s(g) = g^{-s}, \quad s > 0; \tag{7}$$

г) степенные:

$$p_s(g) = s^{-g}, \quad s > 0.$$

Графическое представление ядер приведено на рис. 1.

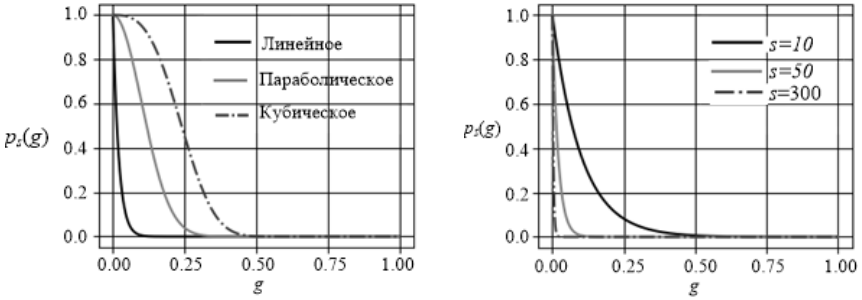

Рис. 1. Ядерные функции (5) при $s = 50$ и экспоненциальная ядерная функция (6) соответственно

Для тестирования модифицированного алгоритма использовались ядра (5), (6) и (7).

Ядра внедрялись в алгоритм на этапе нахождения масс (весов) зондов. Таким образом, выражение (2) можно записать так:

$$m_{i,t} = \frac{p_s\left(\mu_{i,t}\right)}{\sum_{i=1}^{N} p_s\left(\mu_{i,t}\right)}, \quad \mu_{i,t} = \frac{f(x_i) - \hat{f}_{\min,t}}{\hat{f}_{\max,t} - \hat{f}_{\min,t}}, \quad i = \overline{1,N},$$

где $p_s(\mu_{i,t})$ – значение ядерной функции для i-й точки.

Если параметр μ_i оставить, как в формуле (2), то необходимо брать возрастающие ядерные функции.

Кроме внедрения ядерной функции очевидно, что на начальных этапах работы алгоритму требуется больше информации об исследуемом объекте. По мере того как точки будут стягиваться к экстремуму, информации будет требоваться всё меньше. Исходя из этого можно предложить изменять количество точек динамически по мере работы алгоритма. Примером функции,

описывающей изменение количества точек, может служить следующая невозрастающая кусочно-заданная функция:

$$N_t = \begin{cases} 500, \ 0 \le t \le 30, \\ N_{t-1} - 2, 30 < t \le 230, \\ 100, \ 230 < t \le 400, \\ N_{t-1} - 1, \ 400 < t \le 450, \\ 50, \ t > 450, \end{cases} \tag{8}$$

где N_t – количество точек на итерации t.

Для более легкого и короткого задания закона изменения точек можно взять невозрастающую на отрезке [0,1] функцию, например, одно из ядер (5), (6) или (7). Тогда функцию можно представить в следующем виде:

$$\hat{N}_t = p_k\left(\frac{t}{T}\right)(N_0 - N_T) + N_T, \quad N_t = \lfloor \hat{N}_t \rfloor, \quad N_0 > N_T. \tag{9}$$

Здесь N_0 – количество точек на 0-й итерации, а N_T – количество точек на итерации с номером T (конечной), $\lfloor \cdot \rfloor$ – операция округления вниз до ближайшего целого. Вместо коэффициента s в ядерной функции теперь используется параметр k, который определяет, как быстро будет убывать количество точек.

Законы изменения количества точек (8) и (9) позволяют получать большее количество информации об объекте на первых итерациях и уменьшать количество точек при уменьшении области поиска в ходе работы алгоритма. При этом отбрасываются зонды, имеющие худший вес. Это позволяет улучшить результаты в среднем на 5…10 %, а также сократить объем вычислений по сравнению с алгоритмом, использующим постоянное количество точек.

Целочисленную функцию, определенную на отрезке [0, T], а также границы, которые в выражении (8) определены номером итерации алгоритма, можно подбирать экспериментальным путем.

4. ТЕСТОВЫЙ ПРИМЕР

Для тестирования алгоритмов были сконструирована функция по методу Бочарова–Фельдбаума [10]:

$$f(x_1, x_2) = \min\{\varphi_k(x_1, x_2), \ k = \overline{1,10}\}, \tag{10}$$

где

$$\varphi_1 = 6|x_1 + 2|^{0.6} + 6|x_2 - 4|^{1.6},$$
$$\varphi_2 = 6|x_1|^{1.6} + 7|x_2|^2 + 3,$$
$$\varphi_3 = 6|x_1 - 4|^{1.1} + 7|x_2 - 4|^{0.6} + 5,$$
$$\varphi_4 = 5|x_1 - 4|^{1.1} + 5|x_2|^{1.8} + 6,$$
$$\varphi_5 = 5|x_1 + 2|^{0.5} + 5|x_2|^{0.5} + 7,$$
$$\varphi_6 = 5|x_1|^{1.3} + 5|x_2 + 2|^{1.3} + 8,$$
$$\varphi_7 = 4|x_1 + 4|^{0.8} + 3|x_2 - 2|^{1.2} + 9,$$
$$\varphi_8 = 2|x_1 - 2|^{0.9} + 4|x_2 + 4|^{0.3} + 10,$$
$$\varphi_9 = 6|x_1 - 2|^{1.1} + 4|x_2 - 2|^{1.7} + 11,$$
$$\varphi_{10} = 3|x_1 + 4|^{1.2} + 3|x_2 + 2|^{0.5} + 12.$$

Функция (10) имеет десять несимметрично расположенных минимумов. Глобальный экстремум находится в точке (–2; 4) со значением 0. Остальные девять локальных экстремумов расположены (в порядке возрастания значения функции) в точках: (0; 0), (4; 4), (4; 0), (–2; 0), (0; –2), (–4; 2), (2; –4), (2; 2), (4; 2). С ними можно ознакомиться на графике линий равных уровней (рис. 2 и 3). Также ограничим область поиска экстремумов областью с размером $[-6, 6]^2$.

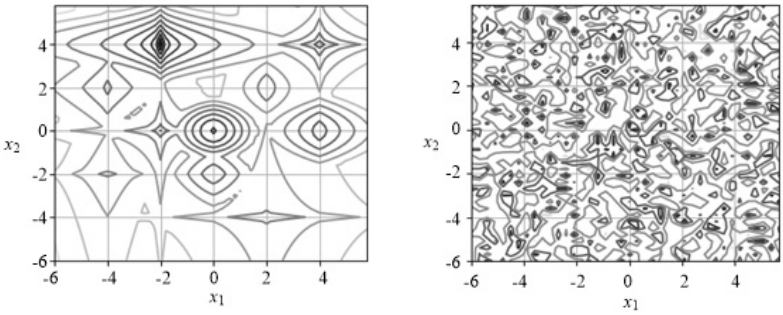

Рис. 2. Линии равных уровней функции (10) без помех и при 1000 % помехе соответственно

Минимальное значение функции в области поиска равно $f(-2, 4) = 0$, а максимальное – 27. Следовательно, амплитуда сигнала будет равна $A_s = (0 + 27)/2 = 13.5$. Отношение шум/сигнал будет определяться как $k_{SN} = A_n / A_s$, где A_n – амплитуда шума. Тогда аддитивная помеха будет воздействовать на функцию следующим образом:

$$f_n(x_1, x_2) = f(x_1, x_2) + uA_n, \quad u = U(-1,1),$$

где $f_n(x_1, x_2)$ – значение функции, подвергшееся воздействию шума; u – случайная величина, имеющая равномерный закон распределения.

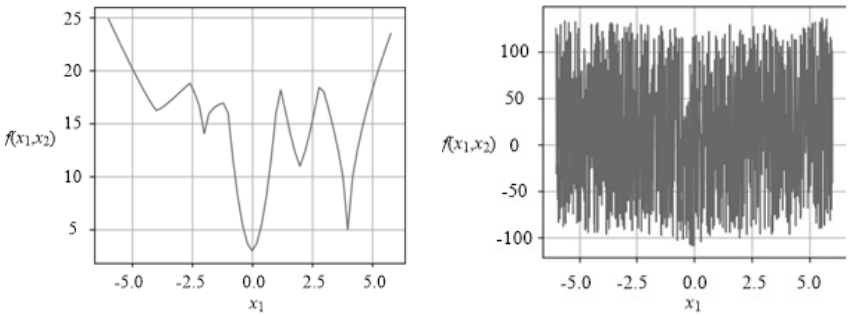

Рис. 3. Срезы функции (10) без помехи и с 1000 % помехой при $x_2 = x_1$ соответственно

Если $A_n = A_s$, то помеха будет 100 %, а если $A_n = 10A_s$, то 1000 %.

5. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Тестирование алгоритмов проводилось при следующих настройках:

а) количество итераций – 500;

б) количество точек для стандартного GSA – 200;

в) закон изменения количества точек для модифицированного GSA (NR-GSA) – (8);

г) закон изменения гравитационной постоянной – (1);

д) $G_0 = 100$;

е) $\alpha = 20$;

ж) степень влияния расстояния на силу притяжения равна единице;

з) ядерная функция для NR-GSA – (6);

и) коэффициент ядра $s = 500$;

к) при выходе зонда за допустимую область поиска он заново располагается в ее пределах случайным образом;

л) помеха является аддитивной, подчиняющейся равномерному закону распределения величиной;

м) критерием останова выступает достижение максимального количества итераций (T).

Качество работы алгоритмов будем оценивать с помощью оценки вероятности попадания в глобальный минимум $\left(\hat{P}_\delta\right)$. Данная величина рассчитывается по результатам 100 экспериментов как отношение удачных (результат работы алгоритма отличается от истинного не более чем на заданную константу δ) к общему числу экспериментов.

Зададим размер δ-окрестности как относительную величину. Если размер δ-окрестности в размерности задачи будет равен 0.50, тогда размер δ-окрестности в относительных единицах будет равен отношению заданного размера (0.50) и общего размера области поиска для конкретной переменной, которую можно определить по наложенным на нее ограничениям $(-6 \le x_1,\ x_2 \le 6)$. Рассчитаем размер δ-окрестности для x_1 и x_2 при желаемой погрешности 0.50, он будет равен 0.50/12 или 0.042. По аналогии размеры определены для величин 0.25 и 0.10.

Ниже приведены результаты тестов двух алгоритмов при различной точности и без влияния помех.

Таблица 1

Результаты тестирования без помехи при разной точности

Алгоритм	δ	$\hat{P}_\delta$	Минимальное значение	Координата
Standard GSA	0.042	1.00	0.0598	[–1.9777; 4.0292]
	0.021	0.98		
	0.008	0.64		
NR-GSA	0.042	1.00	0.0485	[–2.0001; 4.0266]
	0.021	1.00		
	0.008	0.80		

Стоит учесть, что если объект не подвержен помехам, не стоит использовать большое количество точек. Оценка вероятности для алгоритма гравитационного поиска слабо зависит от этого параметра. Оценка вероятности ведет себя стабильно при $N \ge 30$, показывая результат в 0.95–1.00. Для достижения приемлемого результата (оценка вероятности равна 0.95–1.00) для большинства задач достаточно использовать 50 точек при 500 итерациях, дальнейшее увеличение количества зондов нецелесообразно.

В табл. 2 приведены результаты тестирования модифицированного алгоритма (NR-GSA) с различными ядрами и при разном уровне помех.

Таблица 2

Результаты тестирования NR-GSA с различными ядрами при помехах

Уровень помех / Ядро	$\hat{P}_{\delta}$						
	100 %	200 %	300 %	400 %	500 %	600 %	700 %
Линейное	1.00	0.98	0.96	0.87	0.81	0.81	0.81
Параболическое	0.99	0.94	0.91	0.89	0.84	0.83	0.76
Кубическое	1.00	0.92	0.89	0.86	0.83	0.77	0.73
Экспоненциальное	1.00	0.97	0.93	0.94	0.88	0.87	0.83
Степенное	0.97	0.93	0.87	0.85	0.75	0.71	0.67
Гиперболическое	0.38	0.27	0.22	0.16	0.17	0.13	0.07

Результаты, приведенные в табл. 2, показывают, что для данной тестовой задачи, а также для похожих задач наиболее подходящими являются линейное и экспоненциальное я́дра. Данные ядра обладают наиболее медленной скоростью ухудшения результата. Самые худшие показатели оценки вероятности принадлежат гиперболическому ядру. Это может объясняться тем, что скорость его убывания по отношению к остальным является самой медленной.

Кроме этого, стоит отметить, что линейное и экспоненциальное я́дра слабо подвержены варьированию коэффициента s. Многочисленные тесты показывают, что вышеупомянутые ядерные функции дают стабильный результат при $s \geq 40$. Все остальные ядра сильно чувствительны к значению параметра s. При их использовании приемлемого результата можно добиться при $s \geq 300$.

Ниже приведено сравнение алгоритмов GSA и NR-GSA (с использованием экспоненциального ядра).

Из рис. 4 видно, что модифицированный алгоритм позволяет улучшить оценку вероятности попадания в глобальный минимум более чем в два раза (при десятикратной помехе) по сравнению с базовым алгоритмом.

Для наглядного представления влияния коэффициента s и уровня помехи на оценку вероятности при использовании экспоненциального ядра построена тепловая карта (рис. 5).

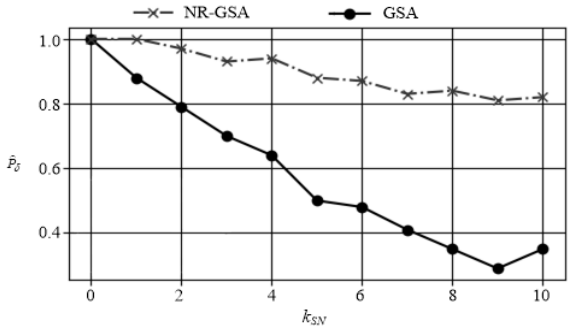

Рис. 4. Зависимость оценки вероятности попадания в окрестность глобального минимума от уровня помех для стандартного алгоритма гравитационного поиска (GSA) и его модификации (NR-GSA)

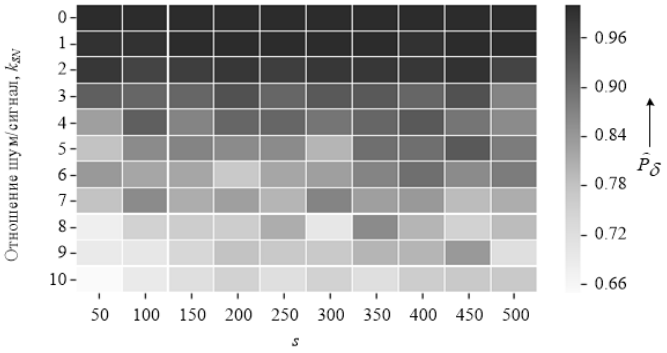

Рис. 5. Зависимость оценки вероятности попадания в окрестность глобального минимума от уровня помех и степени ядра s

Из рис. 5 видно, что для хорошей работы экспоненциальное ядро слабо реагирует на изменение коэффициента s, однако при $s < 300$ оценка вероятности немного ниже, чем при $s \geq 300$.

Общая динамика показывает, что при небольших уровнях помех $(0 \leq k_{SN} \leq 2)$ оценка вероятности близка к единице, но при увеличении интенсивности шума $\hat{P}_\delta$ начинает снижаться. Также видно, как высокий коэффициент $s \geq 350$ замедляет снижение $\hat{P}_\delta$ в диапазоне $3 \leq k_{SN} \leq 7$.

Если сравнить рис. 4 и столбец тепловой карты рис. 5 при $s = 500$, можно увидеть, что они идентичны.

ЗАКЛЮЧЕНИЕ

В последние годы появляется всё больше алгоритмов глобальной оптимизации, однако проблеме оптимизации в условиях воздействия помех уделяется мало внимания. В данной статье предлагается модификация одного популярного алгоритма глобальной оптимизации. Модификация заключается во введении ядерной функции в алгоритм, что позволяет улучшить его устойчивость к аддитивной, равномерно распределенной помехе и добиться приемлемого результата даже в условиях сильной зашумленности основного сигнала. Для оценки модифицированного алгоритма она была протестирована со стандартным алгоритмом в идентичных условиях на многоэкстремальной тестовой функции. Предложенная в статье модификация подходит для широкого круга алгоритмов класса роевого интеллекта, и не только.

В данном направлении и дальше необходимо совершенствовать алгоритмы, например, придать стабильности при высоких уровнях помех, а также сокращать количество измерений целевой функции.

СПИСОК ЛИТЕРАТУРЫ

1. *Horst R., Tuy H.* Global optimization: deterministic approaches. – 3rd ed. – Berlin: Springer, 1996. – 729 p.

2. *Растригин Л.А.* Адаптация сложных систем. – Рига: Зинатне, 1981. – 375 с.

3. *Kennedy J., Eberhart R.C.* Particle swarm optimization // Proceedings of IEEE International Conference on Neural Networks. – 1995. – Vol. 4. – P. 1942–1948.

4. *Karaboga D.* An idea based on honey bee swarm for numerical optimization: technical report TR-06 / Erciyes University, Engineering Faculty, Computer Engineering Department. – Kayseri, Türkiye, 2005.

5. *Castro L.N. de, Timmis J.* Artificial immune systems: a new computational intelligence approach. – London: Springer, 2003. – 364 p.

6. *Shams M., Rashedi E., Hakimi A.* Clustered-gravitational search algorithm and its application in parameter optimization of a low noise amplifier // Applied Mathematics and Computation. – 2015. – Vol. 258. – P. 436–453.

7. *Yildiz B.S., Lekesiz H., Yildiz A.R.* Structural design of vehicle components using gravitational search and charged system search algorithms // Component Design. – 2016. – Vol. 1. – P. 79–81.

8. *Rashedi E., Nezamabadi-pour H., Saryazdi S.* GSA: a gravitational search algorithm // Information Sciences. – 2009. – Vol. 179. – P. 2232–2248.

9. *Рубан А.И.* Глобальная оптимизация методом усреднения координат: монография. – Красноярск: ИПЦ КГТУ, 2004. – 303 с.

10. *Бочаров И.Н., Фельдбаум А.А.* Автоматический оптимизатор для поиска минимального из нескольких минимумов (глобальный оптимизатор) // Автоматика и телемеханика. – 1962. – № 3. – С. 289–301.

Воронов Владимир Сергеевич, магистрант кафедры информатики Сибирского федерального университета, Институт космических и информационных технологий. E-mail: voronov.volodya2013@yandex.ru

DOI: 10.17212/2307-6879-2018-3-4-101-115

Noise resistant a gravity search algorithm*

V.S. Voronov

Siberian Federal University, 26 Academician Kirensky street, Krasnoyarsk, 630074, Russian Federation, undergraduate student, e-mail: voronov.volodya2013@yandex.ru

The paper considers the problem of global optimization of multi-extremal functions using the gravitational search algorithm. The classical algorithm is stochastic and is based on the gravitational interaction of masses and the laws of motion. In the course of the study of the classical algorithm, a shortcoming was revealed. It is associated with a strong decrease in the probability of falling into the global extremum in the presence of additive noise. In connection with which a modified algorithm is proposed. The improvement is based on the introduction of nuclear function. This makes it possible to better determine the global extremum under the influence of noise. It is also proposed to use the dynamic law of changing the number of probes. This will reduce the number of measurements of the objective function. An example of a test function is given for testing algorithms. It has 10 extremes, one of which is global. A comparative study of the two algorithms is conducted under the same conditions.

Keywords: global optimization, optimization method, algorithm, Gravitational Search Algorithm, additive interference, law of gravity, nuclear function, swarm intelligence, multiextremal function

REFERENCES

1. Horst R., Tuy H. *Global optimization:* d*eterministic approaches*. 3rd ed. Berlin, Springer, 1996. 729 p.

2. Rastrigin L.A. *Adaptatsiya slozhnykh sistem* [Adaptation of complex systems]. Riga, Zinatne Publ., 1981. 375 p. (In Russian).

* *Received 12 September 2018.*

3. Kennedy J., Eberhart R.C. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 1995, vol. 4, pp. 1942–1948.

4. Karaboga D. *An idea based on honey bee swarm for numerical optimization*. Technical Report TR-06. Erciyes University, Engineering Faculty, Computer Engineering Department. Kayseri, Türkiye, 2005.

5. Castro L.N. de, Timmis J. *Artificial immune systems: a new computational intelligence approach*. London, Springer, 2003. 364 p.

6. Shams M., Rashedi E., Hakimi A. Clustered-gravitational search algorithm and its application in parameter optimization of a low noise amplifier. *Applied Mathematics and Computation*, 2015, vol. 258, pp. 436–453.

7. Yildiz B.S., Lekesiz H., Yildiz A.R. Structural design of vehicle components using gravitational search and charged system search algorithms. *Component Design*, 2016, vol. 1, pp. 79–81.

8. Rashedi E., Nezamabadi-pour H., Saryazdi S. GSA: a gravitational search algorithm. *Information Sciences*, 2009, vol. 179, pp. 2232–2248.

9. Ruban A.I. *Global'naya optimizatsiya metodom usredneniya koordinat* [Global optimization by a method of averaging of coordinates]. Krasnoyarsk, KSTU Publ., 2004. 303 p.

10. Bocharov I.N., Fel'dbaum A.A. Avtomaticheskii optimizator dlya poiska minimal'nogo iz neskol'kikh minimumov (global'nyi optimizator) [Automatic optimizer for search minimum fromseveral minima (global optimizer)]. *Avtomatika i telemekhanika – Automation and Remote Control*, 1962, no. 3, pp. 289–301. (In Russian).

Для цитирования:

Воронов В.С. Помехоустойчивый вариант алгоритма гравитационного поиска глобального минимума // Сборник научных трудов НГТУ. – 2018. – № 3 (93). – С. 101–115. – DOI: 10.17212/2307-6879-2018-3-4-101-115.

For citation:

Voronov V.S. Pomekhoustoichivyi variant algoritma gravitatsionnogo poiska global'nogo minimuma [Noise resistant a gravity search algorithm]. *Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university*, 2018, no. 3 (93), pp. 101–115. DOI: 10.17212/2307-6879-2018-3-4-101-115.