Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский
Нижегородский государственный университет им. Н.И. Лобачевского»

На правах рукописи

СОВРАСОВ ВЛАДИСЛАВ ВАЛЕРЬЕВИЧ

Разработка и исследование способов распределения ресурсов в параллельных алгоритмах глобальной оптимизации

Направление подготовки
09.06.01 – Информатика и вычислительная
техника

Направленность подготовки
05.13.18 – Математическое моделирование,
численные методы и комплексы программ

**Научный доклад об основных результатах
научно-квалификационной работы (диссертации)**

Научный руководитель:
к. ф.-м. н., доцент,
Баркалов К. А.

Нижний Новгород
2021

Содержание

Введение

Задачи нелинейной глобальной оптимизации встречаются в различных прикладных областях [1; 2], и традиционно считаются одними из самых трудоёмких среди оптимизационных задач. Их сложность экспоненциально растёт в зависимости от размерности пространства поиска [3]. Зачастую глобальныя оптимизация при размерности пространства в 10 переменных сложнее, чем локальная оптимизация в существенно многомерном пространстве. Для последней может оказаться достаточно применения простейшего метода градиентного спуска или эвристических алгоритмов поиска по шаблону [4], в то время как чтобы *гаранитрованно* отыскать глобальный оптимум методам оптимизации приходится накапливать информацию о поведении целевой функции во всей области поиска [5—8].

Актуальность темы исследования

Цель диссертационной работы

Объект исследования –

Предмет исследования – м

Обоснование специальности

Научная новизна

Практическая ценность работы

Обоснованность и достоверность результатов

Реализация результатов работы

Структура и объемы работы

Квалификационная работа включает в себя введение, три главы, заключение и список литературы. Полный объём квалификационной работы составляет 35 страниц, включая 11 рисунков и 6 таблиц. Список литературы содержит 37 позиций.

Личный вклад автора

1. Постановки задачи глобальной оптимизации, обзор существующих методов решения

1.1. Задача глобальной оптимизации с функциональными ограничениями-неравенствами

В рамках данной работы будем рассматривать следующую постановку задачи глобальной оптимизации: найти глобальный минимум N-мерной функции $\varphi(y)$ в гиперинтервале $D = \{y \in \mathbf{R}^N : a_i \leqslant x_i \leqslant b_i, 1 \leqslant i \leqslant N\}$. Для построения оценки глобального минимума по конечному количеству вычислений значения функции требуется, чтобы скорость изменения $\varphi(y)$ в D была

ограничена. В качестве такого ограничения как правило принимается условие Липшица.

$$\varphi(y^*) = \min\{\varphi(y) : y \in D\} \tag{1}$$

$$|\varphi(y_1) - \varphi(y_2)| \leqslant L\|y_1 - y_2\|, y_1, y_2 \in D, 0 < L < \infty \tag{2}$$

Существуют различные методы, решающие рассмотренную многомерную задачу напрямую [5; 9], а также эффективные методы решения одномерных задач [8; 10]. В данной работе рассматривается одномерный метод, который применяется совместно со схемой редукции размерности. Классической схемой редукции размерности исходной задачи для алгоритмов глобальной оптимизации является использование разверток — кривых, заполняющих пространство [11].

$$\{y \in \mathbf{R}^N : -2^{-1} \leqslant y_i \leqslant 2^{-1}, 1 \leqslant i \leqslant N\} = \{y(x) : 0 \leqslant x \leqslant 1\} \tag{3}$$

Отображение вида (3) позволяет свести задачу в многомерном пространстве к решению одномерной ценой ухудшения ее свойств. В частности, одномерная функция $\varphi(y(x))$ является не Липшицевой, а Гёльдеровой:

$$|\varphi(y(x_1)) - \varphi(y(x_2))| \leqslant H|x_1 - x_2|^{\frac{1}{N}}, x_1, x_2 \in [0; 1],$$

где константа Гельдера H связана с константой Липшица L соотношением

$$H \leqslant 2L\sqrt{N+3} \tag{4}$$

Область допустимых аргументов целевой функции также может быть задана с помощью функциональных ограничений, что значительно усложняет задачу. Постановка задачи глобальной оптимизации в этом случае будет иметь следующий вид:

$$\varphi(y^*) = \min\{\varphi(y) : y \in D,\ g_j(y) \leqslant 0, 1 \leqslant j \leqslant m\} \tag{5}$$

Обозначим $g_{m+1}(y) = \varphi(y)$. Далее будем предполагать, что все функции $g_k(y), 1 \leqslant k \leqslant m+1$ удовлетворяют условию Липшица на гиперинтервале D.

1.2. Совместное решение серии задач глобальной оптимизации

Также будем интересоваться совместным решением серии из q задач вида (5):

$$\min\{\varphi_1(y), y \in D_1\}, \min\{\varphi_2(y), y \in D_2\}, ..., \min\{\varphi_q(y), y \in D_q\}. \tag{6}$$

Требуется получить приближённое решение во всех задачах их серии, потратив на это как можно меньшее количество испытаний. При решении серии задач (6) возникает проблема оптимального распределения ресурсов метода между ними. Будем считать, что метод оптимально распоряжается ресурсами, если при его остановке на некоторой итерации решение всех задач

получено с одинаковой точностью. Такого совйства можно добиться, потребовав от метода равномерной сходимости на всём множестве задач:

$$\exists \varepsilon > 0 : \forall s > 1, \forall i, j \in \{1, \ldots, q\} \; \frac{\|\tilde{y_i}(s)^* - y_i * \|_\infty}{\|\tilde{y_j}(s)^* - y_j^*\|_\infty} \leqslant \varepsilon, \tag{7}$$

где s это количество итераций метода оптимизации, $\tilde{y_i}(s)^*$ это приближение к решению, полученное методом в задаче i из множества (6) на итерации s.

1.3. Различные подходы к решению задач глобальной оптимизации

Одним из возможных способов классификации алгоритмов глобальной оптимизации является разделение их на детерминированные и стохастические. Стохастические алгоритмы в том или ином виде используют в своей работе случайные переменные и поэтому результат их работы, вообще говоря, меняется от запуска к запуску. Детерминированные методы работают по заранее установленным решающим правилам, результат их работы определяется только задачей и, возможно, некоторыми настраиваемыми параметрами.

1.3.1. Стохастические методы

В последнее время стали популярны различные стохастические алгоритмы глобально оптимизации, прежде всего эволюционные [12—14]. Они имеют довольно простую структуру, позволяют решать задачи большой размерности, однако обеспечивают глобальную сходимость только в вероятностном смысле. Общим недостатком стохастических методов является их зависимость от случайных переменных и, в силу характера сходимости, начальной точки, с которой начинается оптимизация (она может также генерироваться случайно). Примерами неэволюционных стратегий стохастической оптимизации являются случайный поиск [15], метод имитации отжига [16] и муравьиный алгоритм [17]. Стоит отметить, что стохастические методы поиска не требуют дифференцируемости или даже непрерывности от целевой функции задачи оптимизации, могут работать с зашумлёнными функциями.

1.3.2. Детерминированные методы

Стохастические методы обладают одним концептуальным недостатком – нельзя гарантировать их сходимость при каждом конкретном запуске процесса оптимизации. Для детерминированных методов есть возможность построить теорию сходимости, наложив ограничения на скорость изменения целевой функции в задаче (1). Если этого не сделать, то по любому конечному количеству значений целевой функции невозможно построить её модель с гарантированной нижней оценкой. Известным и широко представленным классом детерминированных методов являются методы липшицевой оптимизации, предполагающие Липшиц-непрерывность целевой функции в задаче оптимизации. В одномерном случае к таким методам можно отнести семейство характеристических методов [18], в которое входит, например, метод Пиявского [19]

и AGS [20]. Для их применение в случае многомерных задач используются схемы редукции размерности [20]. Методы диагональных [21] или симплексных [22] разбиений не требуют использования таких схем, однако более сложны в реализации, т.к. требуется эффектично хранить и модифицировать множество многомерных гиперинтервалов или симплексов.

Индексный алгоритм глобального поиска В качестве примера детерминированного характеристического метода приведём индексный алгоритм глобального поиска [8].

Принимая во внимание схему редукции размерности (3), будем при описании метода считать, что требуется найти глобальный минимум функции $\varphi(x), x \in [0;1]$, удовлетворяющей условию Гёльдера, при ограничениях $g_j(x)$, также удовлетворяющих этому условию на интервале $[0;1]$.

Рассматриваемый индексный алгоритм глобального поиска (IAGS) для решения одномерной задачи (5) предполагает построение последовательности точек x_k, в которых вычисляются значения минимизируемой функции или ограничений $z_k = g_s(x_k)$. Для учета последних используется индексная схема [8]. Пусть $Q_0 = [0;1]$. Ограничение, имеющее номер j, выполняется во всех точках области

$$Q_j = \{x \in [0;1] : g_j(x) \leq 0\},$$

которая называется допустимой для этого ограничения. При этом допустимая область D исходной задачи определяется равенством: $D = \cap_{j=0}^{m} Q_j$. Испытание в точке $x \in [0;1]$ состоит в последовательном вычислении значений величин $g_1(x), ..., g_\nu(x)$, где значение индекса ν определяется условиями: $x \in Q_j, 0 \leqslant j < \nu, x \notin Q_\nu$. Выявление первого нарушенного ограничения прерывает испытание в точке x. В случае, когда точка x допустима, т. е. $x \in D$ испытание включает в себя вычисление всех функций задачи. При этом значение индекса принимается равным величине $\nu = m + 1$. Пара $\nu = \nu(x), z = g_\nu(x)$, где индекс ν лежит в границах $1 \leqslant \nu \leqslant m + 1$, называется результатом испытания в точке x.

Такой подход к проведению испытаний позволяет свести исходную задачу с функциональными ограничениями к безусловной задаче минимизации разрывной функции:

$$\psi(x^*) = \min_{x\in[0;1]} \psi(x),$$
$$\psi(x) = \begin{cases} g_\nu(x)/H_\nu & \nu < M \\ (g_M(x) - g_M^*)/H_M & \nu = M \end{cases}$$

Здесь $M = \max\{\nu(x) : x \in [0;1]\}$, а $g_M^* = \min\{g_M(x) : x \in \cap_{i=0}^{M-1} Q_i\}$. В силу определения числа M, задача отыскания g_M^* всегда имеет решение, а если $M = m + 1$, то $g_M^* = \varphi(x^*)$. Дуги функции $\psi(x)$ гельдеровы на множествах $\cap_{i=0}^{j} Q_i, 0 \leq j \leq M - 1$ с константой 1, а сама $\psi(x)$ может иметь разрывы первого рода на границах этих множеств. Несмотря на то, что значения констант Гёльдера H_k и величина g_M^* заранее неизвестны, они могут быть оценены в процессе

решения задачи.

Множество троек $\{(x_k, \nu_k, z_k)\}, 1 \leqslant k \leqslant n$ составляет поисковую информацию, накопленную методом после проведения n шагов.

На первой итерации метода испытание проводится в произвольной внутренней точке x_1 интервала $[0; 1]$. Индексы точек 0 и 1 считаются нулевыми, значения z в них не определены. Пусть выполнено $k \geqslant 1$ итераций метода, в процессе которых были проведены испытания в k точках $x_i, 1 \leqslant i \leqslant k$. Тогда точка x^{k+1} поисковых испытаний следующей $(k+1)$-ой итерации определяются в соответствии с правилами:

Шаг 1. Перенумеровать точки множества $X_k = \{x^1, \ldots, x^k\} \cup \{0\} \cup \{1\}$, которое включает в себя граничные точки интервала $[0; 1]$, а также точки предшествующих испытаний, нижними индексами в порядке увеличения значений координаты, т.е.

$$0 = x_0 < x_1 < \ldots < x_{k+1} = 1 \tag{8}$$

и сопоставить им значения $z_i = g_\nu(x_i), \nu = \nu(x_i), i = \overline{1, k}$.

Шаг 2. Для каждого целого числа $\nu, 1 \leqslant \nu \leqslant m+1$ определить соответствующее ему множество I_ν нижних индексов точек, в которых вычислялись значения функций $g_\nu(x)$:

$$I_\nu = \{i : \nu(x_i) = \nu, 1 \leqslant i \leqslant k\}, 1 \leq \nu \leqslant m+1,$$

определить максимальное значение индекса $M = \max\{\nu(x_i), 1 \leq i \leq k\}$.

Шаг 3. Вычислить текущие оценки для неизвестных констант Гёльдера:

$$\mu_\nu = \max\{\frac{|g_\nu(x_i) - g_\nu(x_j)|}{(x_i - x_j)^{\frac{1}{N}}} : i, j \in I_\nu, i > j\}. \tag{9}$$

Если множество I_ν содержит менее двух элементов или если значение μ_ν оказывается равным нулю, то принять $\mu_\nu = 1$.

Шаг 4. Для всех непустых множеств $I_\nu, \nu = \overline{1, M}$ вычислить оценки

$$z_\nu^* = \begin{cases} \min\{g_\nu(x_i) : x_i \in I_\nu\} & \nu = M \\ -\varepsilon_\nu & \nu < M \end{cases}, \tag{10}$$

где вектор с неотрицательными координатами $\varepsilon_R = (\varepsilon_1, .., \varepsilon_m)$ называется вектором резервов.

Шаг 5. Для каждого интервала $(x_{i-1}; x_i), 1 \leqslant i \leqslant k$ вычислить характеристику

$$R(i) = \begin{cases} \Delta_i + \frac{(z_i - z_{i-1})^2}{(r_\nu \mu_\nu)^2 \Delta_i} - 2\frac{z_i + z_{i-1} - 2z_\nu^*}{r_\nu \mu_\nu} & \nu = \nu(x_i) = \nu(x_{i-1}) \\ 2\Delta_i - 4\frac{z_{i-1} - z_\nu^*}{r_\nu \mu_\nu} & \nu = \nu(x_{i-1}) > \nu(x_i) \\ 2\Delta_i - 4\frac{z_i - z_\nu^*}{r_\nu \mu_\nu} & \nu = \nu(x_i) > \nu(x_{i-1}) \end{cases} \tag{11}$$

где $\Delta_i = (x_i - x_{i-1})^{\frac{1}{N}}$. Величины $r_\nu > 1, \nu = \overline{1, m}$ являются параметрами алгоритма. От них зависят произведения $r_\nu \mu_\nu$, используемые при вычислении характеристик в качестве оценок неизвестных констант Гёльдера.

Шаг 6. Выбрать наибольшую характеристику:

$$t = \underset{1 \leqslant i \leqslant k+1}{\arg\max} R(i) \tag{12}$$

Шаг 7. Провести очередное испытание в середине интервала $(x_{t-1}; x_t)$, если индексы его концевых точек не совпадают: $x^{k+1} = \frac{1}{2}(x_t + x_{t-1})$. В противном случае провести испытание в точке

$$x^{k+1} = \frac{1}{2}(x_t + x_{t-1}) - \text{sgn}(z_t - z_{t-1})\frac{|z_t - z_{t-1}|^n}{2r_\nu \mu_\nu^n}, \nu = \nu(x_t) = \nu(x_{t-1}),$$

а затем увеличить k на 1.

Алгоритм прекращает работу, если выполняется условие $\Delta_t \leqslant \varepsilon$, где $\varepsilon > 0$ есть заданная точность. В качестве оценки глобально-оптимального решения выбираются значения

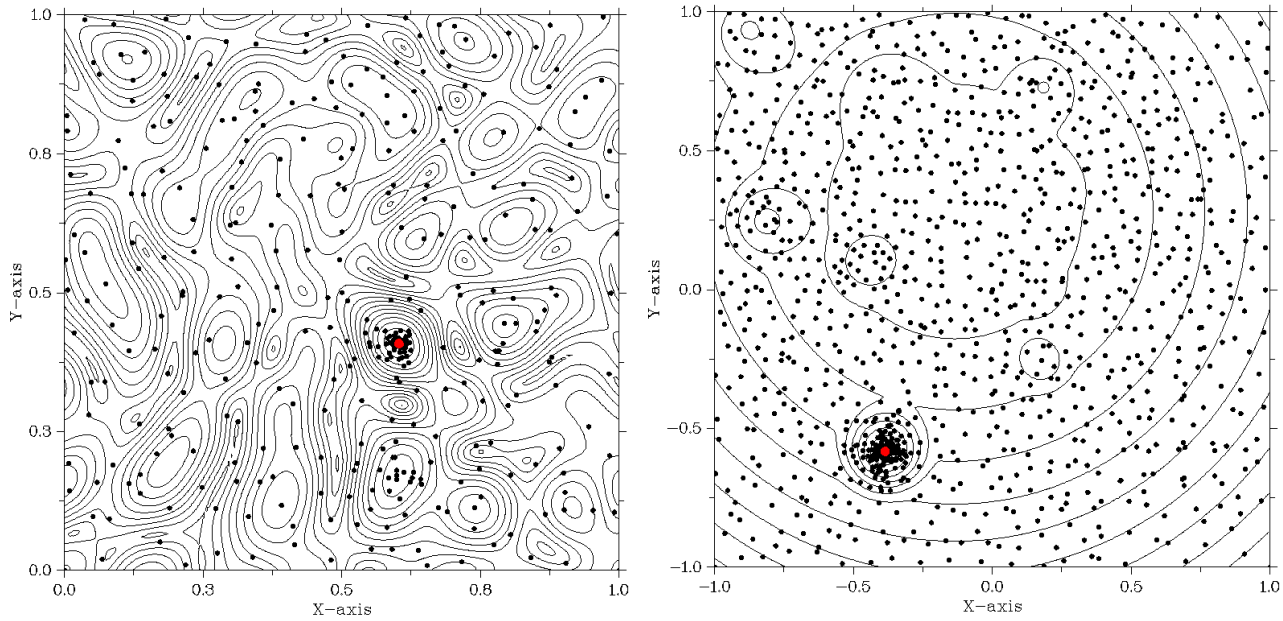
$$\varphi_k^* = \min_{1 \leqslant i \leqslant k} \varphi(x_i), x_k^* = \underset{1 \leqslant i \leqslant k}{\arg\min} \varphi(x_i) \tag{13}$$

Теорема о сходимости для этого метода подробно рассмотрена в [8], а также будет рассматриваться в главе 5.2. Здесь отметим лишь, что путём увеличения r и при $\varepsilon = 0$ можно гарантированно добиться сходимости, если все функции задачи (5) удовлетворяют условию Липшица на гиперкубе из (1).

2. Подходы к сравнению алгоритмов глобальной оптимизации

Основным подходом к сравнению между собой методов оптимизации является запуск этих методов на наборах тестовых задач и последующее сравнение собранных в процессе запуска метрик, характеризующие различные аспекты работы алгоритмов. Исключая некоторые специальные случаи, когда требуется определить метод, лучше других себя ведущий на небольшом наборе специфичных задач или вообще на одной задаче с параметром. Существуют два способа построить тестовый набор задач: собрать небольшое множество известных из литературы тестовых задач [23] или исплользовать рандомизированные генераторы задач, позволяющие генерировать потенциально любое количество задач [24; 25]. В [26] подробно рассмотрены недостатки и достоинства каждого из подходов. Известных тестовых задач мало, поэтому используя только их не всегда можно достаточно полно выявить особенности работы различных методов. В то же время генераторы задач позовляют создать любое их количество, однако полученные задачи могут обладать скрытой общей структурой, благоприятной для некоторых из тестируемых методов, либо обладать малой вариативностью. Поскольку в рамках работы рассматриваются алгоритмы для решения существенно многоэкстремальных задач, будем производить все сравнения на задачах, полученных с помощью нескольких генераторов достаточно таких задач. Это сгладит проблему потенциально низкой вариативности задач, полученных одним генератором. Также в одном специальном случае рассмотрим решение одной известной задачи из литературы [27].

В рамках данной работы будем использовать два механизма генерации тестовых задач [24; 25] [1]. Тестовые задачи, получаемые данными механизмами, имеют абсолютно разную природу (см. линии уровня на Рис. 1), что увеличивает достоверность результатов сравнения алгоритмов оптимизации.



(a) Пример линий уровня функции, порождённой F_{GR} **(b)** Пример линий уровня функции, порождённой GKLS

Рис. 1. Линии уровня и точки испытаний ИАГП в двух синтетических задачах без ограничений

Обозначим набор двумерных задач, полученный с помощью генератора [25] как F_{GR}. Генератор F_{GR} не позволяет контролировать размерность и сложность задач, а также количество локальных экстремумов. Каждая тестовая задача определяется формулой:

$$\varphi(y) = \sqrt{\left(\sum_{i=1}^{7}\sum_{j-1}^{7} A_{ij}g_{ij}(y) + B_{ij}h_{ij}(y)\right)^2 + \left(\sum_{i=1}^{7}\sum_{j-1}^{7} C_{ij}g_{ij}(y) - D_{ij}h_{ij}(y)\right)^2}$$

где

$$y \in [0;1]^2,$$

$$g_{ij} = \sin(i\pi y_1)\sin(j\pi y_2),$$

$$h_{ij} = \cos(i\pi y_1)\cos(j\pi y_2),$$

где коэффициенты $A_{ij}, B_{ij}, C_{ij}, D_{ij}$ генерируются случайно с равномерным в интервале $[-1;1]$ распределением. Получаемые функции являются существенно многоэксремальными. В данной работе будем использовать 100 функций, сненерированных с помощью F_{GR}.

Геренатор GKLS [24] позволяет получать тестовые задачи заданной размерности с заданным количеством локальных экстремумов. Также этот генератор позволяет варьировать сложность задачи, изменяя размер области притяжения глобального минимума. Это достигается за

[1] Программные реализации генераторов доступны в исходных кодах по адресу `https://github.com/sovrasov/global-optimization-test-problems`

счёт модификации параболоида $g(x) = \|x - T\| + t$ в шаровых окрестностях некоторых случайно сгенерированных точек $M_i, i = \overline{1, m}$. В точках M_i располагаются локальные минимумы со значениями, превосходящими значение $g(T) = t$. В [28] приводятся параметры генератора для получения наборов по 100 задач двух уровней сложности (Simple and Hard) размерностей 2, 3, 4 и 5. Как и авторы генератора GKLS, будем использовать эти параметры и таким образом получим 800 тестовых задач.

Перечисленные выше генераторы порождают задачи без нелинейных ограничений, поэтому в дополнение к ним будем использовать систему GCGen [2] [29], которая позволяющет генерировать задачи с ограничениями на основе произвольных нелинейных функций. При использовании этой системы совместно с генераторами, такиим как GKLS, порождённые ими функции выступают как в роли целевой функции, так и в роли ограничений.

С помощью GCGen были сгенерированы следующие наборы задач с ограничениями:

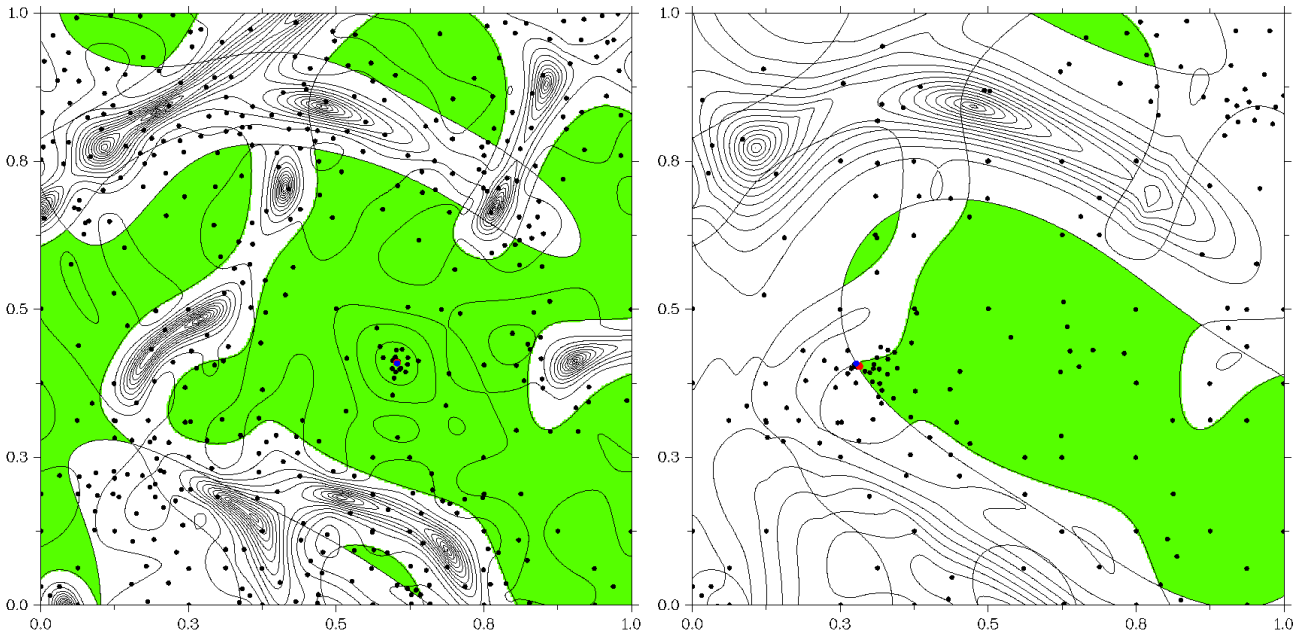
- 100 двумернух задач на основе GKLS Simple 2d с двумя ограничениями;
- 100 трёхмерных задач на основе GKLS Simple 3d с двумя ограничениями;
- 20 четырёхмерных задач на основе GKLS Simple 4d с двумя ограничениями;
- смешанный набор, стостоящий из 50 задач на основе GKLS Simple 2d и 50 задач на основе F_{GR}. Каждая задача имеет два функциональных ограничения. Этот набор позвоялет проверить наличие у метода оптимизации свойства равномерной сходимости при решении множества задач в постановке (6).

Генератор GKLS [24] позволяет получать функции заданной размерности и с заданным количеством экстремумов. В сочетании с GCGen были порождены два множества по 100 задач размерности 2 и 3. Каждая из задач имеет два ограничения. Также с целью демонстрации того, что свойства метода сохраняются при существенно разных свойствах задач был сгенерирован смешанный класс, состоящий из 50 задач с двухмерными функциями GKLS и 50 задач с функциями F_{GR}. На рис. 2 представлены примеры линий уровня рассматриваемых задач. Допустимая область закрашена.

Будем считать, что тестовая задача решена, если метод выполняет испытание y^k в δ-окрестности глобального минимума y^*, т.е. $\left\|y^k - y^*\right\|_{inf} \leq \delta = \alpha \left\|b - a\right\|_{inf}$, где a и b это правая и левая границы гиперкуба из (1), а α это относительная точность. Если указанное соотношение не выполнено до исчерпания лимита на испытания, то задача считается нерешённой. Лимит на количество испытаний и α заданы для каждого класса задач в отдельности в соответствии с размерностью задач и их сложностью (см. Таблицу 1).

Будем рассматривать среднее количество испытаний, затраченное на решение одной задачи, и количество решенных задач как характеристики эффективности метода оптимизации на

[2] Исходный код системы доступен по ссылке `https://github.com/UNN-ITMM-Software/GCGen`



(a) Решение внутри допустимой области **(b)** Решение на границе допустимой области

Рис. 2. Линии уровня и точки испытаний ИАГП в двух синтетических задачах

заданном классе задач. Чем меньше среднее количество испытаний на задачу, тем быстрее метод сходится к решению, а значит и меньше обращается к потенциально трудоёмкой процедуре вычисления ограничений и целевой функции задачи. Количество решённых задач характеризует надёжность метода. Чтобы сделать рассматриваемые характеристики независимыми друг от друга, будем вычислять среднее количество испытаний, принимая во внимание только решённые задачи.

Среднее число испытаний на задачу в некоторых случаях не позволяет получить полной картины о поведении численного метода оптимизации на рассматриваемом множестве задач. Например, если метод тратит много ресурсов на решение небольшого подмножества задач из выборки, то это невозможно понять, глядя только на среднее число испытаний. Как дополнительный критерий сравнения методов будем использовать операционную характеристику [25]. Операционная характеристика представляет из себя кривую на плоскости (K, P), где K это среднее количество испытаний, произведённое методом до выполнения критерия остановки, а P – доля задач из выборки, решённых не более чем за K испытаний. Если при заданном K операционная характеристика одного метода лежит выше, чем операционная характеристика другого, то это означает, что при фиксированных затратах на поиск, первый метод с большей вероятностью найдёт решение. При заданном P если характеристика одного метода лежит левее, чем характеристика другого, то при одинаковой надёжности, первый метод в среднем потратит меньше испытаний на поиск, чем второй.

Таблица 1. Лимит на количество испытаний и относительная точность в критерии остановки для различных классов задач

Problems class	Trials limit	α
F_{GR}	5000	0.01
GKLS 2d Simple	8000	0.01
GKLS 2d Hard	9000	0.01
GKLS 3d Simple	15000	0.01
GKLS 3d Hard	25000	0.01
GKLS 4d Simple	150000	$\sqrt[4]{10^{-6}}$
GKLS 4d Hard	250000	$\sqrt[4]{10^{-6}}$
GKLS 5d Simple	350000	$\sqrt[5]{10^{-7}}$
GKLS 5d Hard	600000	$\sqrt[5]{10^{-7}}$

3. Методы редукции размерности

В этом разделе рассмотрим более подробно упомянутый ранее метод редукции размерности с помощью кривых Пеано, заполняющих пространство, и сравним между собой разные модификации различных кривых. Кроме прямой редукции размерности с помощью кривых типа Пеано, существует и способ, сводящий одну задачу оптимизации к множеству вложенных задач оптимизации: многошаговая схема [20]:

$$\min_{y\in D}\varphi(y) = \min_{y_1\in[a_1,b_1]}\ldots\min_{y_N\in[a_N,b_N]}\varphi(y_1,\ldots,y_N). \tag{14}$$

В рамках многошаговой схемы каждое вычисление целевой функции по переменной y_1 во внешней задаче оптимизации влечёт за собой проведение ещё $N-1$ вложенной оптимизации. Недостатками данной схемы является её низкая экономичность в плане количества обращений к целевой функции и отсутствие теоретической гаранитии сходимости IAGS при наличии функциональных ограничений (в некоторых случаях целевые функции во вложенных задачах перестают удовлетворять условию Липшица). В качестве обобщения многошаговой схемы было предложено вложенную оптимизацию вести не по одиночным переменным, а по блокам переменных [30]. К многомерным вложенным задачам в таком случае можно применить редукцию размерности с помощью кривых типа Пеано, поэтому выявление наиболее эффективного типа кривой имеет смысл и для ускорения сходимости в случае блочной многошаговой схемы.

3.1. Инъективная развёртка

Для редукции размерности задач оптимизации в рамках информационно-статистического подхода применяются кривые типа Пеано, заполняющие пространство (развёртки) [8; 11; 20; 31]. Такие кривые отображают отрезок $[0,1]$ на N-мерный гиперкуб D.

После редукции размерности исходная многомерная задача (1) преобразуется в одномерную задачу следующего вида:

$$\varphi(y(x^*)) = \min\{\varphi(y(x)) : x \in [0,1]\}. \tag{15}$$

Стоит отметить, что подстановка развёртки в Липшицеву функцию из (1) порождает одномерную функцию $\varphi(y(x))$, которая удовлетворяет условию Гёльдера:

$$|\varphi(y(x_1)) - \varphi(y(x_2))| \leq H|x_1 - x_2|^{\frac{1}{N}}, x_1, x_2 \in [0,1], \tag{16}$$

где константа H удовлетворяет неравенству $H \leqslant 2L\sqrt{N+3}$, L – константа Липшица из (2), и N – размерность задачи (1).

На Рис. 3 приведена одномерная функция, полученная после применения развёртки к параболоиду вращения $\varphi(y) = y_1^2 + y_2^2$. Точка с координатами $(0;0)$ имеет несколько одномерных прообразов и глобальный минимум целевой функции в ней расщепился на три глобальных минимума одномерной функции, полученной в результате редукции. Этот пример демонстрирует последствия потери информации об окрестности точки $(0;0)$ при редукции размерности.

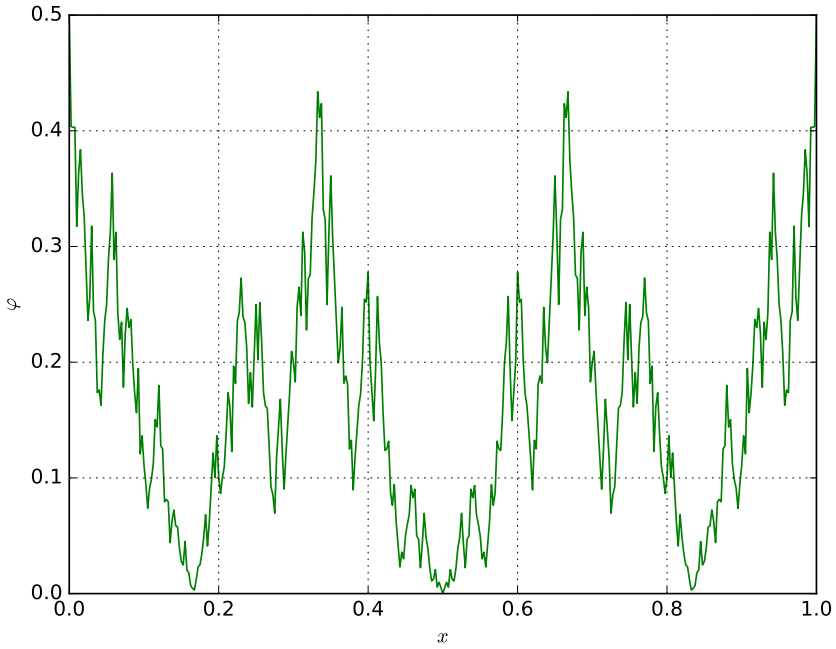


Рис. 3. Пример одномерной функции, порождённой развёрткой

Алгоритмы численного построения аппроксимаций кривой Пеано приведены в [8].

Вычислительная схема алгоритма, применяющего кривые Пеано для редукции размерности, выглядит следующим образом:

- Алгоритм оптимизации выполняет минимизацию редуцированной одномерной функции $\varphi(y(x))$ из (15),

- После нахождения точки следующего испытания x, её многомерный образ y вычисляется с использованием развёртки $y(x)$,
- Значение исходной многомерной функции $\varphi(y)$ вычисляется в точке $y \in D$,
- Вычисленное значение $z = \varphi(y)$ используется как значение одномерной редуцированной функции $\varphi(y(x))$ в точке x.

3.2. Сдвиговые развёртки

Применение кривой типа Пеано для редукции размерности приводит к потере локальной информации об окрестности многомерных точек в пространстве $\mathbb{R}^N$ (см. [32]). Одним из способов частично решить эту проблему является использование множества развёрток [32]:

$$Y_L(x) = \left\{y^0(x),\ y^1(x), ...,\ y^L(x)\right\} \tag{17}$$

вместо одной кривой Пеано $y(x)$ (см. [8; 32; 33]).

Такой набор отображений может быть получен путём сдвига исходной развёртки $y^0(x)$ на $2^{-l}, 0 \le l \le L$ по каждой из координат. Каждая из развёрток определена на своём гиперкубе $D_l = \left\{y \in R^N : -2^{-1} \le y_i + 2^{-l} \le 3 \cdot 2^{-1},\ 1 \le i \le N\right\},\ 0 \le l \le L$.

На Рис. 4а пунктирной линией обозначен образ интервала $[0, 1]$, полученный с помощью $y^0(x),\ x \in [0, 1]$,. Поскольку гиперкуб D из (1) принадлежит пересечению семейства гиперкубов D_l, необходимо ввести дополнительное ограничение:

$$g_0(y) = \max\left\{|y_i| - 2^{-1} :\ 1 \le i \le N\right\}, \tag{18}$$

тогда гиперкуб D можно представить в виде

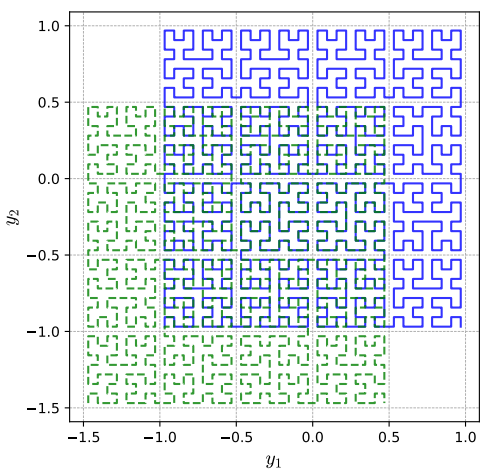
$$D = \left\{y^l(x) :\ x \in [0, 1],\ g_0(y^l(x)) \le 0\right\},\ 0 \le l \le L,$$

т.е. $g_0(y) \le 0$ если $y \in D$, иначе $g_0(y) > 0$. Следовательно, любая точка $y \in D$ имеет прообраз $x^l \in [0, 1]$, определяемый соответствующей развёрткой $y^l(x),\ 0 \le l \le L$.

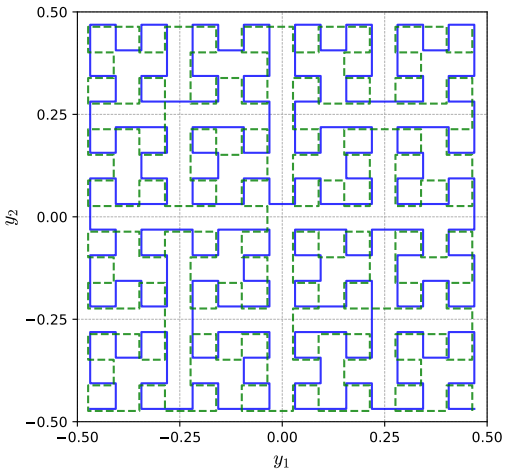
Таким образом, каждая развёртка $y^l(x),\ 0 \le l \le L$, порождает свою собственную задачу вида (1), имеющую расширенную (по сравнению с D) область поиска D_l и дополнительное ограничение-неравенство с левой частью вида (18)

$$\min\left\{\varphi(y^l(x)) : x \in [0, 1],\ g_j(y^l(x)) \le 0,\ 0 \le j \le m\right\},\ 0 \le l \le L. \tag{19}$$

При этом, L копий метода AGS решают каждую из указанных задач, на каждой итерации обмениваясь многомерными точками следующего испытания и добавляя их прообразы в свою поисковую информацию. Такая схема имеет смысл только при наличии L параллельных вычислительных ядер или процессоров.



(a) Две сдвиговые развёртки, определённые на гиперкубах D_0 и D_1



(b) Две вращаемые развёртки на одной плоскости

Рис. 4. Различные развёртки, построенные с низкой плотностью

3.3. Вращаемые развёртки

Схема построения множества развёрток (здесь и далее будем называть, развёртки, порождённые ей сдвиговыми или S-развёртками), описанная в Секции 3.2, позволяет сохранять часть информации о близости точке в многомерном пространстве и, таким образом, обеспечивает более точную (по сравнению с одной развёрткой) оценку константы Гёльдера в процессе оптимизации. Однако, однако, этот подход имеет недостаток в виде наличия дополнительного ограничения что затрудняет построение эффективных реализаций алгоритма оптимизации на базе S-развёрток (см. конец Секции 3.6).

Чтобы обойти слложности, возникающие при работе с S-развёртками и одновременно сохранить часть инорrмации об окрестностях точек N-мерном вещественном пространстве, была редложена ещё одна схема построения множества развёрток. Эта схема предполагает вместо получения группы развёрток делать сдвиг не сдвиг кривой Пеано по диагонали гиперкуба, а её вращение относиельно центра координат [31]. На Рис. 4b представлены две развёртки, аппроксимирующие привую Пеано при $N = 2$. Получая новые развёртки путём отражения относительно осей координат, можно породить множество развёрток мощностью до 2^N. При этом дополнительное ограничение $g_0(y)$ из (18), необходимое для получения набора S-развёрток, отсутствует.

3.4. Неинъективная развёртка

Как уже было сказано в секции 3.2, потеря информации о близости точек в многомерном пространстве может быть частично скомпенсирована использованием множественных отображений $Y_L(x) = \{y^1(x), ..., y^L(x)\}$. Однако, сама по себе кривая типа Пеано сохраняет в себе часть этой информации: она не является инъективым отображением, поэтому имея один образ

$y(x) \in \mathbb{R}^N$, можно получить несколько несколько отличных x прообразов $t_j \in [0,1], t_j \neq x$, которые затем могут быть добавлены в поисковую информацию индексного метода.

Кривая типа Пеано, используемая в (15) для редукции размерности, определяется через предельный переход, поэтому не может быть вычислена непосредственно. При численной оптимизации используется некоторое её приближение, являющееся инъективной кусочно-линейной кривой. В [20] было предложено неинъективное отображение равномерной сетки на отрезка $[0,1]$ на равномерную сетку в гиперкубе D. Каждый многомерный узел может иметь до 2^N одномерных прообразов. На рис. 5b крестиками обозначена сетка в пространесве $\mathbb{R}^2$, для двух узлов которой указаны соответствующие им одномерное прообразы из $[0,1]$ (отмечены квадратами и кругами). Каждый указанный узел имеет по 3 прообраза.

Недостатком неинъективной развёртки является потенциально большое количетво прообразов (до 2^N).

3.5. Гладкая развёртка

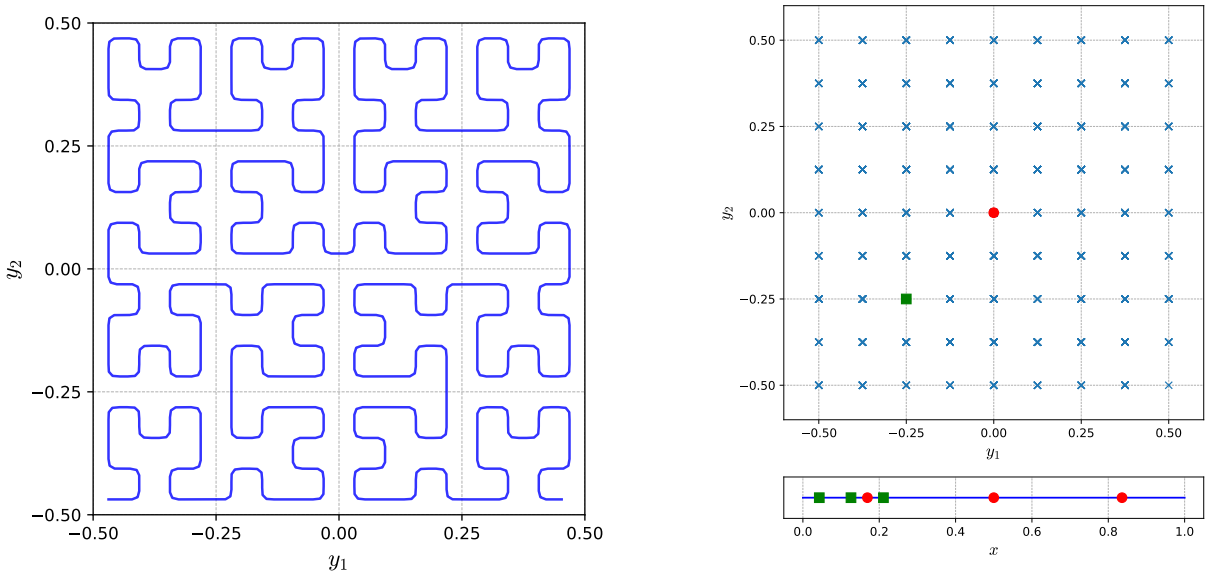
Рассмотренные в предыдущих пунктах способы построения развертки строят кривую $y(x)$, которая не является гладкой (см. рис. 4a). Отсутствие гладкости может негативно сказаться на свойствах редуцированной одномерной функции $\varphi(y(x))$, т.к. гладкая кривая более качественно передает информации о возрастании/убывании исходной функции. На основе исходного алгоритма построения негладкой развертки было предложен обобщенный алгоритм [34], позволяющий строить гладкую развёртку. Для иллюстрации на рис. 5a изображена гладкая развертка в двумерном случае. Недостатком гладкой развёртки является в несколько раз большая сложность вычисления по сравнению с кусочно-линейными кривыми (требуется вычислять нелинейные гладкие функции). Причём с ростом точности аппроксимации и размерности количество интервалов гладкости увеличивается и сложность вычисления кривой нарастает.

3.6. Сравнение развёрток

Сравнение скорости сходимости IAGS с различными типами развёткок проводилось в соответствии с методикой, описанной в Секции 2. С целью понять, обладает ли какой-либо из перечисленных ранее типов развёрток существенным преимуществом над другими, были построены операционные характеристики индексного метода с различными типами развёрток на наборах задач GKLS 2d Simple и GKLS 3d Simple.

Во всех экспериментах параметр плотности построения развёрток $m = 12$. Минимальное значение пераметра надёжности r было найдено для каждого типа развёртки перебором по равномерной сетке с шагом 0.1.

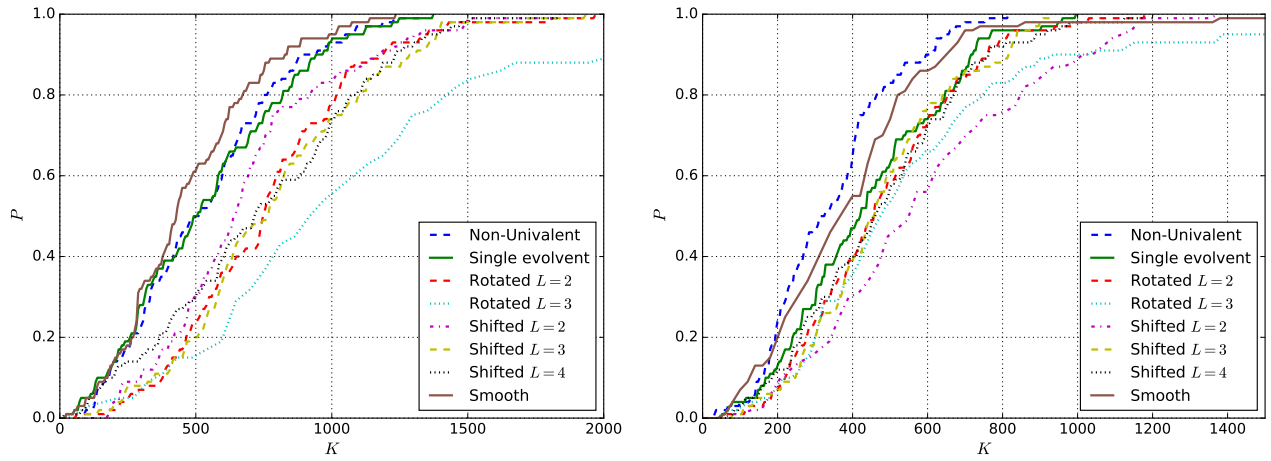
На классе GKLS 2d Simple при минимальном r неинъективная и гладкая развёртка обеспечивают более быструю сходимость (рис. 6b). То же самое, наблюдается и при $r = 5.0$ (рис. 6a). В последнем случае сдвиговая и вращаемая развёртки начинают отставать от остальных, т.к.



(a) Гладкая развёртка **(b)** Неинъективная развёртка

Рис. 5. Различные развёртки, построенные с низкой плотностью

значение $r = 5.0$ является завышенным для них.

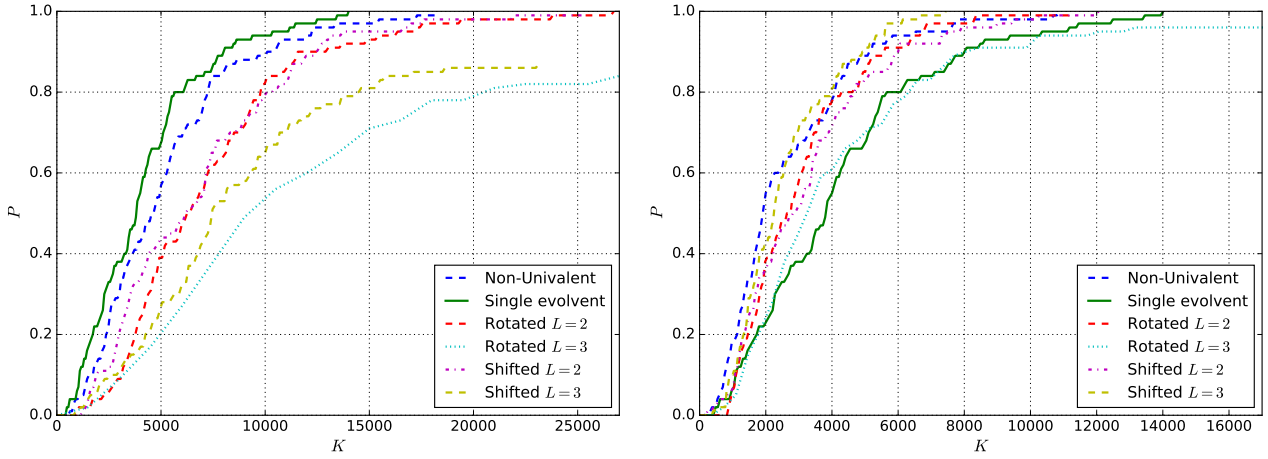


(a) $r = 5.0$ **(b)** Minimal r

Рис. 6. Операционные характеристики на классе GKLS 2d Simple

На классе GKLS 2d Simple при минимальном r неинъективная и множественные развёртки имеют значительное преимущество над единственной развёрткой (рис. 7b). Значение $r = 4.5$ является завышенным для вращаемой и сдвиговой развёрток (рис. 7a).

Накладные расходы при использовании сдвиговой развёртки. Во всех представленных выше экспериментах при построении операционной характеристики учитывалось количество вычислений целевой функции из класса GKLS, однако в случае сдвиговой развёртки индексный метод решает задачу с ограничением g_0 из (18). В точках, где g_0 нарушено, значение целевой функции не вычисляется. Эти точки, тем не менее, хранятся в поисковой информации,



(a) $r = 4.5$ **(b)** Minimal r

Рис. 7. Операционные характеристики на классе GKLS 3d Simple class

Таблица 2. Среднее количество вычислений g_0 и φ при решении задач класса GKLS 3d Simple с помощью сдвиговой развёртки

L	$calc(g_0)$	$calc(\varphi)$	$\frac{calc(g_0)}{calc(\varphi)}$ ratio
2	96247.9	6840.14	14.07
3	153131.0	7702.82	19.88

создавая дополнительные расходы вычислительных ресурсов. В таблице 2 приведено среднее количество обращение к g_0 и целевой функции. При $L = 3$ ограниение g_0 вычисляется почти в 20 раз чаще, чем целевая функция φ, т. е. 95% всей поисковой информации приходится на вспомогательные точки. Такие накладные расходы приемлемы при решении задач малой размерности с трудоёмкими целевыми функциями, но при росте размерности и общего количества испытаний выгоднее использовать другие типы развёрток.

3.7. Итоги сравнения

В результате проведённого сравнения различных способов редукции размерности, основанных на отображениях типа кривой Пеано, можно сделать вывод о том, что для построения базовой весиии AGS с практической стороны выгоднее использовать единственную кривую Пеано, кроме случаев низкой размерности или чрезвычайно трудоёмкого вычисления целевой функции. Из множественных отображений оптимальным выбором будет использование вращаемых развёрток. Поскольку результаты, полученные для алгоритма с одной развёрткой непосредственно распространяются на алгоритмы с вращаемыми развёртками, далее будем рассматривать только случай единственной кривой Пеано.

4. Сравнение методов глобальной оптимизации

В предыдущем разделе был рассмотрен вопрос о выборе базовой схемы редукции размерности размерности задач оптимизации вида 1 для алгоритма IAGS. В этом разделе рассмотрим сравнение алгоритма глобального поиска с одной развёрткой типа кривой Пеано и других алгоритмов глобальной оптимизации, основанных на кардинально различных принципах. Такое сравнение позволит выяснить, перспективен ли базовый алгоритм глобального поиска для дальнейших модификаций. Как и ранее, в качестве тестовых задач для сравнения, будем рассматривать задачи без ограничений, поскольку большинство рассматриваемых методов не предусматривают специально разработанных схем для их эффективного учёта. Обозначим алгоритм глобального поиска без индексной схемы как AGS. Также рассмотрим схему контроля параметра надёжности r в AGS, позволяющую в некоторых случаях снизить зависимость метода от его выбора.

4.1. Методы глобальной оптимизации для сравнения

- **Multi Level Single Linkage** [35]. MLSL является улучшенным вариантом мультистартовой схемы [36]. Алгоритм сэмплирует стартовые точки, равномерно распределённые по области посика, и производит локальную оптимизацию из них. В сравнении со стандартной мультистартовой схемой MLSL использует кластеризацию и набор эвристических правил для избежания повторных локальных спусков в уже обнаруженные лакальные минимумы.

- **DIRECT** [5]. Метод является детерминированным и рекурсивно разделяет область поиска, формируя дерево гиперпрямоугольников. DIRECT использует значения целевой функции и оценку константы Липшица (2) для получения оценки перспективности дальнейшего разбиения каждого из гиперинтервалов.

- **Locally-biased DIRECT (DIRECT*l*)** [37]. Вариация метода DIRECT, которая меньше уделяет внимания гиперинтервалам с низкой оценкой перспективности. Эта вариация сходится быстрее на задачах с небоьлшим количеством локальных минимумов, но в сложных случаях может не найти глобальный.

- **Dual Simulated Annealing** [16]. Этот стохастический метода является комбинацией классическойго метода имитации отжига (CSA [38]) и быстрого метода имитации отжига (FSA [39]), совмещенной с применением локальной оптимизации. Метод сходится гораздо быстрее, чем CSA и FSA.

- **Differential Evolution** [12]. DE является адаптацией оригинального генетического алгоритма к непрерывному пространству поиска.

- **Controlled Random Search** [40]. CRS производит испытания в случайно сгенерированных начальных точках и определяет следующую точку для испытания на основе симплекса, построенного из случайно выбранных точек более ранних испытаний. CRS не считается эволюционным алгоритмом, хотя хранит точки, похожие на популяцию, и выполняет случайные трансформации над ними, по аналогии с применением оператора мутации.

- **StoGO** [41]. StoGO разделяет пространство поиска на гиперпрямогульники и использует метод ветвей и границ, вычисляя верхние оценки минимумов в подобластях с помощью локальной оптимизации.

Все перечисленные алгоритмы доступны в исходных кодах как компоненты широко распространённых программных пакетов. DIRECT, DIRECTl, CRS, MLSL и StoGO являются компонентами библиотеки NLOpt [42]. Реализации DE и DSA распространяются в пакете SciPy [43] для языка Python.

4.1.1. Контроль параметра надёжности в AGS

Параметр r из (9) непосредственно влияет на глобальную сходимость AGS (см. [8], Глава 8): при достаточно большом значении r метод гарантированно сходится ко всем глобальным минимумам целевой функции. В то же время, согласно (11), при бесконечно большом значении r AGS превращается в полный перебор по равномерной сетке.

В идеальном случае, для обеспечения наибольшей скорости сходимости оценка константы Липшица из (9) не должна быть слишком завышенной, но на практике реальное значение L из (2) неизвестно и приходится либо брать заведомо большое значение r, либо проводить несколько запусков AGS с разными параметрами. Чтобы в некотрой степени решить обозначенную проблему выбора r будем использовать следующую схему:

- совершить q итераций AGS при $r = r_{max}$;

- совершить q итераций AGS при $r = r_{min}$;

- повторять предыдущие шаги до схоимости или исчерпания лимита итараций.

В указанном алгоритме $r_{min} < r_{max}$, $q > 1$. Вместо одного параметра r теперь необходимо выбирать 3, однако, согласно численным экспериментам, сделать это проще, чем найти оптимальное значение r. Интуитивно практическую эффективность предложенной схемы можно объяснить тем, что теперь работа метода происходит в двух режимах: глобальный поиск при $r = r_{max}$ и локальный при $r = r_{min}$. Если во время фазы глобального посика метод приблизился к глобальному минимуму, то во время следующей фазы оценка глобального минимума будет быстро уточнена. Если двух фаз не хватило, то процесс продолжается. Таким образом выбирается более оптимальный компромисс между локальной и глобальной оптимизацией. Далее будем обозначать метод, использующий описанную схему как AGS-AR.

Таблица 3. Параметры методов оптимизации для различных тестовых задач

	AGS	CRS	DE
F_{GR}	$r = 3$	popsize=150	mutation=(1.1,1.9), popsize=60
GKLS 2d Simple	$r = 4.6$	popsize=200	mutation=(1.1,1.9), popsize=60
GKLS 2d Hard	$r = 6.5$	popsize=400	mutation=(1.1,1.9), popsize=60
GKLS 3d Simple	$r = 3.7$	popsize=1000	mutation=(1.1,1.9), popsize=70
GKLS 3d Hard	$r = 4.4$	popsize=2000	mutation=(1.1,1.9), popsize=80
GKLS 4d Simple	$r = 4.7$	popsize=8000	mutation=(1.1,1.9), popsize=90
GKLS 4d Hard	$r = 4.9$	popsize=16000	mutation=(1.1,1.9), popsize=100
GKLS 5d Simple	$r = 4$	popsize=25000	mutation=(1.1,1.9), popsize=120
GKLS 5d Hard	$r = 4$	popsize=30000	mutation=(1.1,1.9), popsize=140

4.2. Результаты численных экспериментов

Сравнение алгоритмов оптимизации было проведено по методике, описанной в Секции 2 и на описанных там же классах тестовых задач.

Результаты решения методами задач глобальной оптимизации различной сложности напрямую зависят от выбранных параметров методав. В большинстве случаев авторы программных реализаций ориентируются на задачи средней сложности. Чтобы получить приемлемые результаты на существенно многоэкстремальных задачах, параметры некоторых методов были выбраны слудующим образом:

- в AGS-AR параметр чередования локальной и глобальной фаз q был выбран равным $50 \cdot \log_2(N-1) \cdot N^2$, а $r_{min} = 3,\ r_{max} = 2 \cdot r_{min}$;
- в DIRECT и DIRECTl параметр $\epsilon = 10^{-4}$;
- в SDA параметр $visit = 2.72$.

Остальные параметры варьировались в зависимости от набора тестовых задач (см. Таблицу 3). Для AGS минимальное значение параметра r такое, что все задачи заданного класса решаются, было получено перебором по равномерной сетке с шагом 0.1.

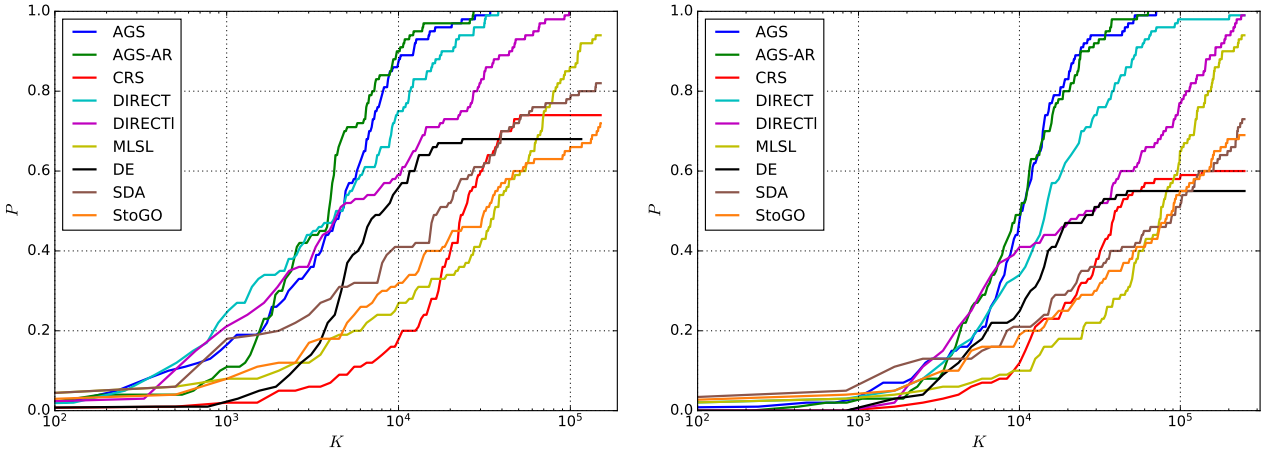
Результаты запуска рассматриваемых методов оптимизации на наборах тестовых задач представлены в таблицах 4, 5. DIRECT, AGS, AGS-AR продемонстрировали лучшую скорость сходимости на всех классах, при этом AGS-AR уступает DIRECT на двухмерных задачах класса Simple, но имеет преимущество на задачах класса Hard. Как можно видеть из Table 5, детерминированные методы (AGS, AGS-AR, DIRECT и DIRECTl) оказались более надёжными, т.е. решили больше тестовых задач. Среди стохастических методов самую высокую надёжность показали MLSL и SDA.

Таблица 4. Среднее количество испытаний, затраченное методами оптимизации при решении задач различных классов

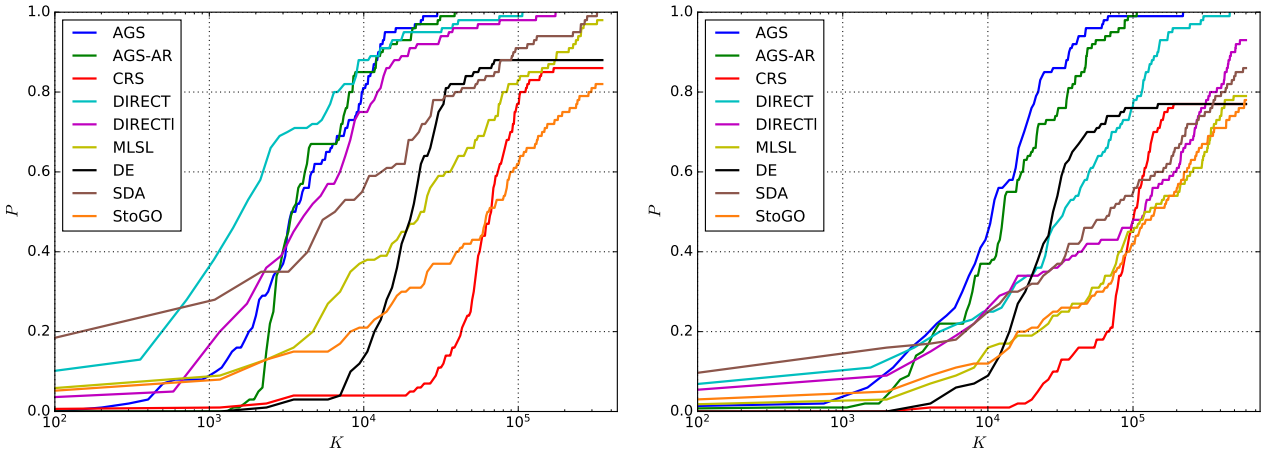
	AGS	AGS-AR	CRS	DIRECT	DIRECTl	MLSL	SDA	DE	StoGO
F_{GR}	193.1	248.3	400.3	**182.2**	214.9	947.2	691.2	1257.3	1336.8
GKLS 2d Simple	254.9	221.6	510.6	**189.0**	255.2	556.8	356.3	952.2	1251.5
GKLS 2d Hard	**728.7**	785.0	844.7	985.4	1126.7	1042.5	1637.9	1041.1	2532.2
GKLS 3d Simple	1372.1	1169.5	4145.8	**973.6**	1477.8	4609.2	2706.5	5956.9	3856.1
GKLS 3d Hard	3636.1	**1952.1**	6787.0	2298.7	3553.3	5640.1	4708.4	6914.3	7843.2
GKLS 4d Simple	5729.8	**4919.1**	19883.6	7328.8	15010.0	41484.8	22066.0	6271.2	29359.2
GKLS 4d Hard	13113.4	**12860.1**	27137.4	22884.4	55596.1	80220.1	68048.0	12487.6	58925.5
GKLS 5d Simple	**5821.5**	6241.3	62921.7	5966.1	10795.5	52609.2	34208.8	20859.4	69206.8
GKLS 5d Hard	**17008.6**	21555.1	87563.9	61657.3	148637.8	138011.8	115634.6	26850.0	141886.5

Таблица 5. Количество задач оптимизации, решённых методами при заданном лимите испытаний (общее количество задач в каждом наборе равно 100)

	AGS	AGS-AR	CRS	DIRECT	DIRECTl	MLSL	SDA	DE	StoGO
F_{GR}	100	100	76	100	100	97	96	96	67
GKLS 2d Simple	100	100	85	100	100	100	100	98	90
GKLS 2d Hard	100	97	74	100	100	100	93	85	77
GKLS 3d Simple	100	100	75	100	100	100	89	86	44
GKLS 3d Hard	100	100	72	100	99	100	88	77	43
GKLS 4d Simple	100	100	74	100	100	94	82	68	72
GKLS 4d Hard	100	100	60	99	99	94	73	55	69
GKLS 5d Simple	100	100	86	100	100	98	100	88	82
GKLS 5d Hard	100	100	77	100	93	79	86	77	78



(a) 4d Simple **(b)** 4d Hard



(c) 5d Simple **(d)** 5d Hard

Рис. 8. Операционные характеристики методов при решении задач из классов GKLS 4d и 5d.

Операционные характеристики методов (Рис. 8a, 8b, 8c, 8d) показывают, что AGS и AGS-AR быстрее других методов достигают 100% надёжности. Также на классе GKLS 5d Simple DIRECT имеет скорость сходимости выше, чем у других методов, но есть несколько сложных задач, решение которые решаются существенно дольше других, что влияет на среднее количество испытаний.

Чувствительность AGS и AGS-AR к выбору параметров. Чтобы оценить степень влияния настроек методов на скорость сходимости AGS и AGS-AR, были проведены эксперименты на классе задач GKLS 5d Simple со следующими параметрами:

- AGS при $r = 4$ (как в Таблице 3);
- AGS при $r = 6$;
- AGS-AR с параметрами, указанными в начале Секции 4.2 ($q = 50 \cdot \log_2(4) \cdot 25 = 2500$, $r_{min} = 3,\ r_{max} = 2 \cdot r_{min}$);

- AGS-AR при $r_{max} = 8$ and и остальными параметрами как в предыдущем эксперименте;
- AGS-AR при $q = 1000$ и другими параметрами как в начале Секции 4.2.

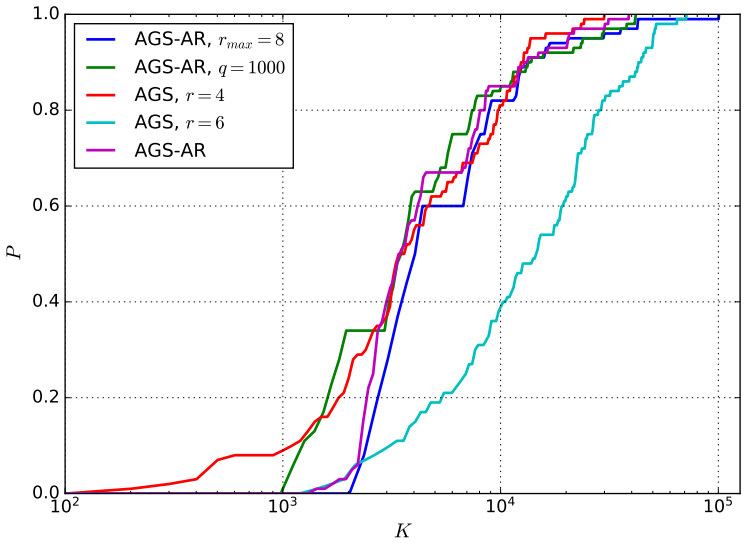


Рис. 9. Операционные характеристики AGS и AGS-AR на классе GKLS 5d Simple с различными параметрами.

Операционные характеристики, полученные в экспериментах, описанных выше, представлены на Рис. 9. AGS при $r = 6$ (голубая кривая) показывает самую медленную сходимость, что говорит о высокой чувствительности AGS к выбору значения r. Поскольку AGS-AR используе то же самое значение r как AGS при $r = 6$, операционные характеристики этих методов идентичны до точки $K = 2500$. После этой границы AGS-AR переключается на $r = 3$ и количество решённых задач начинает быстро возрастать, вплоть до начала следующей фазы глобального поиска при $K = 5000$. Интервалы, на которых AGS-AR работает с $r = r_{max}$ видны на операционных характеристиках как плато. Изменение r и q влияет на операционнух характеристику AGS-AR незнаичтельно. Это наблюдение подтверждает надёжность предложенной модификации AGS с переменным параметром r.

4.3. Итоги сравнения методов

В результате сравнения разлиных методов оптимизации, были сделаны следующие выводы:

- предложенная модификация AGS, AGS-AR менее чувствительна к параметрам и сходится так же быстро, как AGS с заранее подобранными под класс задач параметрами;
- AGS-AR продемонстрировал надёжность и скорость сходимости на уровне удргого детеримнированного метода, DIRECT, и превзошёл на рассмотрнных тестовых задачах многие другие методы, реализации которых так же доступны в исходных кодах;

- на рассмотренных существенно многоэкстремальных задачах малой размерности стохастические методы оптимизации существенно уступают в скорости сходимости и надёжности детерминированным.

Кроме того, реализация алгоритма AGS-AR на языке C++, выполненная в ходе данной работы, была добавлена в состав библиотеки алгоритмов оптимизации с открытым исходным кодом NLOpt [3].

5. Алгоритм, решающий множество задач

В данной главе рассмотрим построение алгоритма, решающего серию задач с нелинейными ограничениями из (6). Задачи, подобные рассмартриваемой, могут возникать, например, при скаляризации задачи многокритериальной оптимизации методом свёртки критериев или при решении задач смешанного целочисленного программирования [44] путём перебора всех возможных значений целочисленных параметров и дальнейшей оптимизации по вещественным. Как уже было отмечено ранее, желаемым свойством метода, решающего совокупность задач, является равномерная сходимость (7). В этой главе учитывая результаты, полученные в предыдущих главах, будем рассматривать IAGS с одной развёрткой как базовый вариант метода оптимизации, подходящий для широкого круга задач. Далее рассмотрим его модификацию для решения задачи (6) и докажем теорему о достаточных условиях сходимости полученного метода. Кроме того, проведём численные эксперименты, показывающие наличние равномерной сходимости и результаты примениения к методу известного подхода распараллеливания по характеристикам [8].

5.1. Описание алгоритма

По аналогии с подходом, описанном в [45], для решения серии задач (6) будем использовать q синхронно работающих копий IAGS с тем лишь отличием, что на шаге 6 при выборе интервала с наилучшей характеристикой, выбор будет осуществляться из всех интервалов, которые породили на данный момент q копий IAGS. Если наибольшая характеристика соответствует задаче i, то выполняется шаг 7 в копии метода с номером i, а остальные копии метода простаивают. Таким образом, на каждой итерации испытание проводится в задаче, наиболее перспективной с точки зрения характеристик (11), что позволяет динамически распределять ресурсы метода между задачами. Обозначим метод, решеющий множество задач с ограничениями, как MIAGS.

Параллельная модификация метода не отличается от рассматриваемой в [45] и заключается в выборе p интервалов на шаге 6 и выполнения p испытаний параллельно на следующем шаге. При этом все ресурсы метода в рамках итерации могут быть направлены как на одну, так и

[3] `https://github.com/stevengj/nlopt`

на $l \leqslant p$ задач одновременно (в зависимости от того, какой из задач принадлежат выбранные методам интервалы).

5.2. Условия сходимости

Достаточные условия сходимости метода IAGS в случае $q = 1$ приведены в [8], рассмотрим их подробнее:

Теорема 1. *(Достаточные условия сходимости IAGS) Пердположим, что следующие утверждения справедливы:*

1. $D \neq \emptyset$*, задача (5) имеет решение.*
2. *Функции* $g_j(y) \leqslant 0, 1 \leqslant j \leqslant m+1$*, липшицевы в области* D *с соответствующими константами* L_i *(здесь* $g_{m+1}(y) = \varphi(y)$*).*
3. *Для достаточно больших* k *из (8), значения* μ_ν *из (9) удовлетворяют неравенствам:*

$$r_\nu \mu_\nu > 2^{3-1/N} L_\nu \sqrt{N+3},\ 1 \leqslant \nu \leqslant m+1. \tag{20}$$

Тогда любая предельная точка $\overline{y}$ *последовательности* $\{y_k\} = \{y(x_k)\}$*, сгенерированная IAGS при решении задачи (5), является допустимой и удовлетворяет условиям*

$$\varphi(\overline{y}) = \inf\{\varphi(y^k) : g_i(y^k) \leqslant 0, 1 \leqslant i \leqslant m, k = 1, 2, \dots\} = \varphi(y^*). \tag{21}$$

Замечание 1. *Из соотношения между константами Гёльдера и Липшица из неравенства (4) следует, что параметр* r_ν *из (11) должен удовлетворять условию*

$$r_\nu > 2^{2-1/N}. \tag{22}$$

Теорема 2. *(О сходимости MIAGS) Пусть условия 1-3 Теоремы 1 верны для каждой задачи* $i,\ 1 \leqslant i \leqslant q$ *из (6), т.е. каждая из задач может быть решена AGS. Тогда в процессе решения* q *задач MIAGS сгенерирует* q *бесконечных последовательностей* $\{y_i^k\},\ 1 \leqslant i \leqslant q$*, таких, что*

$$\varphi_i(\overline{y_i}) = \inf\{\varphi(y_i^k) : g_j^i(y_i^k) \leqslant 0, 1 \leqslant j \leqslant m_i, k = 1, 2, \dots\} = \varphi_i(y_i^*).$$

Доказательство. Рассмотрим две случайно выбранные задачи из множества (6)

$$\begin{aligned} &\min\{\varphi(y) : y \in D_1,\ g_j^\varphi(y) \leqslant 0, 1 \leqslant j \leqslant m_1\}, \\ &\min\{\psi(y) : y \in D_2,\ g_j^\psi(y) \leqslant 0, 1 \leqslant j \leqslant m_2\}. \end{aligned} \tag{23}$$

Обозначим характеристики интервалов (11) в первой задаче как $R_\varphi(i)$ и во второй задаче как $R_\psi(j)$. Принимая во внимание обозначения, имеем:

$$\begin{aligned} R_\varphi(t_\varphi) &= \max_{1 \leqslant i \leqslant k} R_\varphi(i), \\ R_\psi(t_\psi) &= \max_{1 \leqslant j \leqslant s} R_\psi(j), \end{aligned} \tag{24}$$

где k обозначает количество испытаний, сделанных при решении первой задачи, а s это количество испытаний, сделанных при решении второй. Последовательность точек испытаний в первой задаче обозначим $\{v^k\}$, во второй задаче $\{u^s\}$. Значения $z^k = g^{\varphi}_{\nu}(v_k), \nu = \nu(v_k)$ соответствуют точкам $\{v^k\}$, а значения $w^s = g^{\psi}_{\nu}(u_s), \nu = \nu(u_s)$ соответствуют точкам $\{u^s\}$.

Когда две задачи решаются одновременно, метод выбирает интервал для следубщего испытаний в соответствии с условием:

$$R(t) = \max\{R_{\varphi}(i), R_{\psi}(j)\}. \tag{25}$$

Пусть алгоритм решает две задачи и счётчик испытаний равен $l = k+s, l = 0, 1, 2, \ldots$. Тогда, поскольку Теорема 1 справедлива для обеих задач, как минимум одна из последовательностей $\{v^k\}$ или $\{u^s\}$ будет бесконечной (пусть это $\{v^k\}$). Если доказать, что обе последовательности бесконечны, то это будет означать наличие сходимости в обеих задачах.

Рассмотрим предельную точку $\overline{v} \in [v_{i-1}, v_i]$, где $i = i(k)$. Индексы v_{i-1} , v_i могут быть равны или различаться, но в силу сходимости, при достаточно больших k будут стабильны. В первом случае алгоритм будет испльзовать первую ветку правила (11), иначе он будет применять одну из оставшихся ветвей.

Рассмотрим сначала первый случай: из (9) следует, что

$$\frac{|z_i - z_{i-1}|}{\Delta_i} \leqslant \mu_{\varphi,\nu}.$$

Принимая во внимание это неравенстов, получим верхнюю оценку:

$$\frac{(z_i - z_{i-1})^2}{r_{\nu}^2 \mu_{\varphi,\nu}^2 \Delta_i} = \frac{(z_i - z_{i-1})^2 \Delta_i}{(r_{\nu}\mu_{\varphi,\nu}\Delta_i)^2} \leqslant \frac{\Delta_i}{r_{\nu}^2}.$$

Таким образом, при использовании первой ветки правила (11) справедливо неравенство

$$R_{\varphi}(i) \leqslant \Delta_i(1 + \frac{1}{r_{\nu}^2}) - \frac{2(z_i + z_{i-1} - 2z_{\nu}^*)}{r_{\nu}\mu_{\varphi,\nu}}. \tag{26}$$

Поскольку $\overline{v}$ является предельной точкой последовательности $\{v^k\}$ и $\varphi(y(\overline{v})) \leqslant z_{\nu}^*, \nu = m+1$ или $z_{\nu}^* = 0, \nu < m+1$ и $z_{i-1}, z_i \to 0$ при $k \to \infty$:

$$\Delta_i \to 0, z_i + z_{i-1} - 2z_{\nu}^* \to 0. \tag{27}$$

Во втором случае (когда применяется одна из оставшихся ветвей правила (11)) имеем:

$$R_{\varphi}(i) = 2\Delta_i - 4\frac{z_i - z_{\nu}^*}{r_{\nu}\mu_{\varphi,\nu}}.$$

Если $z_{\nu}^* \neq 0$, то $z_i - z_{\nu}^* \geqslant 0$ и

$$R_{\varphi}(i) = 2\Delta_i - 4\frac{z_i - z_{\nu}^*}{r_{\nu}\mu_{\varphi,\nu}} \leqslant 2\Delta_i. \tag{28}$$

Иначе, поскольку $\overline{v}$ это допустимая точка, $z_i \to 0$ при $k \to \infty$.

Из (26) (27) (28) для сколь угодно малого $\delta > 0$ существует достаточно большое значение k такое, что

$$R_\varphi(i) \leqslant \delta. \tag{29}$$

Пусть $\alpha = \max\{\nu(u) : u \in \{u^s\}\}$. Т.к. α это текущий максимальный индекс в поисковой последовательности $\{u^s\}$, то в соответствии с правилом (10), $\exists j : w^*_\alpha = w_j$.

Если $\nu(w_{j-1}) = \nu(w_j) = \alpha$, то применяется первая ветка правила (11) и

$$\begin{aligned} &R_\psi(j) = \Delta_j + \tfrac{(w_j - w_{j-1})^2}{r^2_\alpha \mu^2_{\psi,\alpha} \Delta_j} - \tfrac{2(w_j + w_{j-1} - 2w^*_\alpha)}{r_\alpha \mu_{\psi,\alpha}} \geqslant \\ &\geqslant \Delta_j - \tfrac{2(w_j + w_{j-1} - 2w^*_\alpha)}{r_\alpha \mu_{\psi,\alpha}} = \Delta_j - \tfrac{2\Delta_j (w_{j-1} - w_j)}{r_\alpha \mu_{\psi,\alpha} \Delta_j} \geqslant \\ &\geqslant \Delta_j - \tfrac{2\Delta_j}{r_\alpha} = \Delta_j \left(1 - \tfrac{2}{r_\alpha}\right). \end{aligned} \tag{30}$$

Если $\nu(w_{j-1}) \neq \nu(w_j) = \alpha$, то $\nu(w_{j-1}) < \alpha$ и применяется третья ветка правила (11):

$$R_\psi(j) = 2\Delta_j - 4\frac{w_j - w^*_\alpha}{r_\alpha \mu_{\psi,\alpha}} = 2\Delta_j > 0. \tag{31}$$

Принимая во внимание Зачечание 1, (30), (31), можно сделать вывод о том, что существует интервал с положительной характеристикой характеристикой $R_\psi(j) > 0$. В то же время, выполняется (29) и неравенство

$$R_\psi(j) > R_\varphi(i)$$

будет справедливым при достаточно большом k. Таким образом, следующее испытание будет совершено в интервале, соответствующему второй задаче с целевой функцией $\psi(y)$, и последовательность $\{v^s\}$ будет бесконечной.

Поскольку были рассмотрены две произвольные задачи из q задач, теорема верна для любой пары задач рассматриваемого множества. По индукции, теорема также верна и для всего множества. □

Теорема 2 утанавливает только достаточные условия сходимости IAGS, наличие равномерной сходимости проверим численно в следующем разделе.

5.3. Результаты численных экспериментов

В качестве тестовых задач для MIAGS рассмотрим наборы задач с ограничениями, которые описаны в Секции 2. Кроме того, рассмотрим одну многокритериальную задачу, которая сводится к решению совокупности задач вида (6) с помощью свёртки критериев.

При оценке качества метода и его реализации кроме ускорения от распараллеливания по тиерациям и по времени выполнения, также будем принимать во внимание среднее максимальное расстояние (в смысле l_{inf}-нормы) текущей оценки оптимума до его реального положения, вычисленное на множестве задач (6): D_{avg} и D_{max}. Динамика этих величин в процессе оптимизации показывает, насколько равномерно метод распределяет ресурсы между задачами. Если

кривые D_{avg} и D_{max} ведут себя одинаково, то метод идеально распределяет вычислительные ресурсы между задачами, в противном случае, чем больше разница между ними, тем более неравномерно происходит распределение.

Реализация параллельного метода была выполнена на языке C++ с использованием технологии OpenMP для распареллеливания процесса проведения испытаний на общей памяти. Все вычислительные эксперименты проведены на машине со следующей конфигурацией: Intel Core i7-7800X, 64GB RAM, Unubtu 16.04 OS, GCC 5.5 compiler.

5.3.1. Результаты решения сгенерированных задач

Результаты решения тестовых задач последовательной и параллельной версией модифицированного IAGS для решения множества задач представлены в таблице 6. Для всех двухмерных классов задач параметр $r = 4.7$. В случае трехмерных задач $r = 4.7,\ \varepsilon_\nu = 0.1$. Кроме того, для трёх- и четырёх- мерных задач была применена техника ε-резервирования из [8] Глава 8.3 с $\varepsilon = 0.1$. Это позволило сократить время экспериментов, ускорив сходимость метода. Следуя изначальному предположению о высокой трудоёмкости проведения испытаний, во всех экспериментах в целевые функции и ограничения была внесена дополнительная вычислительная нагрузка так, чтобы время одного обращения к функции задачи было равно примерно 1 мс.

Из Таблицы 6 видно, что ускорение по итерациям $S_i = \frac{iters(p=1)}{iters(p=i)}$ растет линейно с увеличением числа потоков p, в то время, как ускорение по времени $S_t = \frac{time(p=1)}{time(p=i)}$ увеличивается не так быстро, что говорит о неидеальной реализации алгоритма. Увеличить реальное ускорение, верхней границей для которого является S_i, возможно путем оптимизации взаимодействий между копиями IAGS и это планируется сделать в ходе будущей работы.

Для того, чтобы показать равномерную сходимость все тестовые задачи были также решены IAGS в режиме решения отдельных задач. На рис. 10 указаны графики величин средних и максимальных расстояний от реальных оптимумов до их текущих оценок при решении серии из задач, порожденных двумя разными генераторами, по отдельности (сплошная кривая) и совместно (пунктирная кривая). Несмотря на значительную разницу в структуре задач, MIAGS гораздо быстрее уменьшает максимальное и среднее отклонения оценок от действительных оптимумов, чем это происходит при раздельном решении задач. Это говорит о наличии равномерной сходимости по всему множеству совместно решаемых задач. При этом в случае последовательного решения задач величина D_{max} имеет наибольшее значение вплоть до решения последней задачи.

5.3.2. Пример решения многокритериальной задачи

Для демонстрации эффективности подхода к балансировке нагрузки рассмотрим пример, в котором множество задач вида (6) порождено в результате скаляризации многокритериальной задачи оптимизации с ограничениями.

Таблица 6. Результаты экспериментов на наборах синтетических задач

Класс задач	p	Количество итераций	Время, с	S_i	S_t
GCGen GKLS Simple 2d & F_{GR}	1	51434	90.20	-	-
	2	25698	56.96	2.00	1.58
	4	13015	36.67	3.95	2.46
	6	8332	26.85	6.17	3.36
GCGen GKLS Simple 2d	1	59066	97.53	-	-
	2	29060	60.56	2.04	1.61
	4	14266	38.92	4.14	2.51
	6	9436	29.53	6.26	3.30
GCGen GKLS Simple 3d	1	782544	1117.55	-	-
	2	397565	752.92	1.97	1.48
	4	208073	526.67	3.76	2.12
	6	142089	445.45	5.50	2.51
GCGen GKLS Simple 4d	1	14021720	15806.6	-	-
	2	6313070	7254.85	2.22	2.18
	4	3479344	4932.55	4.03	3.20
	6	2783339	3955.38	5.04	3.99

Рассмотрим тестовую задачу, предложенную в [27]:

$$Minimize \begin{cases} f_1(y) = 4y_1^2 + 4y_2^2 \\ f_2(y) = (y_1 - 5)^2 + (y_2 - 5)^2 \end{cases} y_1 \in [-1; 2], y_2 \in [-2; 1]$$

$$s.t. \tag{32}$$

$$\begin{cases} g_1(y) = (y_1 - 5)^2 + y_2^2 - 25 \leqslant 0 \\ g_2(y) = -(y_1 - 8)^2 - (y_2 + 3)^2 + 7.7 \leqslant 0 \end{cases}$$

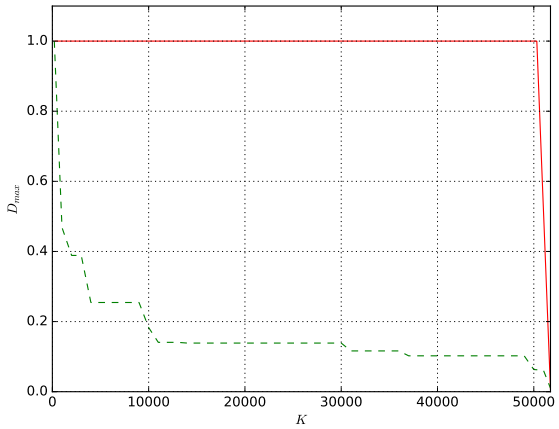
Будем использовать свертку Гермейера для скаляризации задачи (32). После свертки скалярная целевая функция имеет вид:

$$\varphi(y, \lambda_1, \lambda_2) = \max\{\lambda_1 f_1(y), \lambda_2 f_2(y)\}, \tag{33}$$

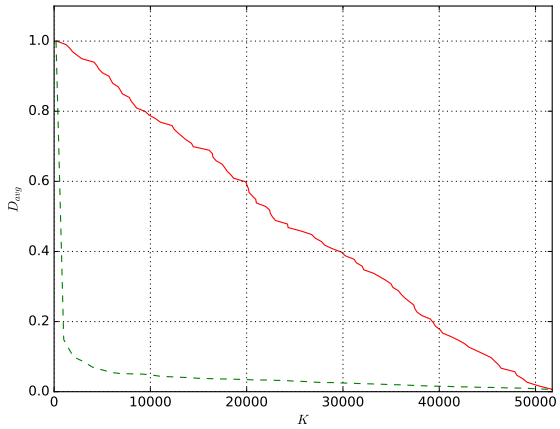
где $\lambda_1, \lambda_2 \in [0, 1]$, $\lambda_1 + \lambda_2 = 1$. Перебирая все возможные коэффициенты свертки, можно найти все множество парето-оптимальных решений в задаче (32). Для численного построения множества Парето выберем 100 наборов коэффициентов (λ_1, λ_2) таких, что $\lambda_1^i = ih$, $\lambda_2^i = 1 - \lambda_1^i$, $h = 10^{-2}, i = \overline{1, 100}$.

В качестве ограничения на вычислительные ресурсы был выбран лимит в 2500 испытаний. Множество вспомогательных скалярных задач решалось двумя способами:

- каждая задача решается отдельно с помощью IAGS с установленным лимитом в 25 испытаний. Таким образом, вычислительные ресурсы равномерно распределены между задачами;



(a) D_{max}

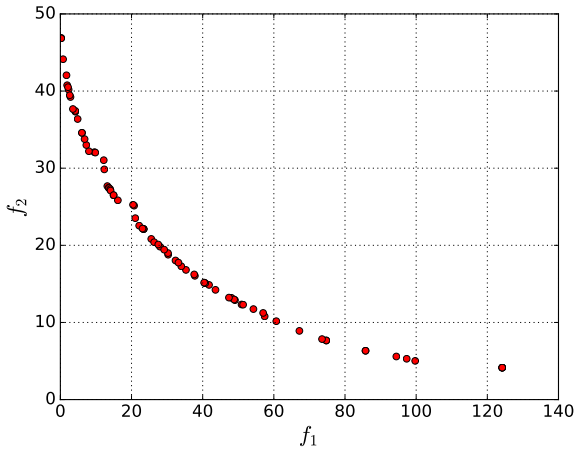


(b) D_{avg}

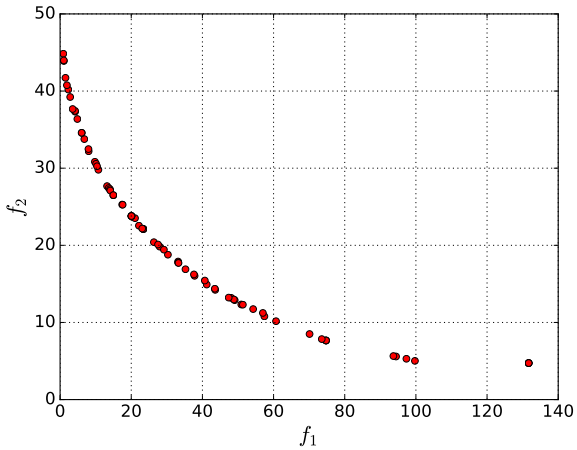
Рис. 10. Динамика величин D_{avg} и D_{max} в процессе решения множества двухмерных задач, порождённых двумя разными генераторами GKLS и F_{GR}

- все задачи решаются одновременно с помощью обобщенного IAGS с установленным лимитом в 2500 испытаний.

В обоих случаях параметр $r = 4$.



(a) IAGS, раздельное решение задач



(b) IAGS для множества задач

Рис. 11. Численные оценки множества Парето в задаче (32), полученные после 2500 испытаний

На рис. 11 представлены графики решений, полученных каждым из методов. Все графики качественно совпадают с указанным в [27] (авторы не предоставили другой информации и решениях для сравнения). Видно, что на рис. 11а кривая Парето имеет вогнутости, что не соответствует решению, указанному в [27] и означает нехватку ресурсов для решения некоторых из вспомогательных задач. Для оценки качества решения также был вычислен показатель

$Spacing(SP)$ [46], характеризующий плотность точек аппроксимации множества Парето.

$$SP(S) = \sqrt{\frac{1}{|S|-1}\sum_{i=1}^{|S|}(\overline{d}-d_i)^2},\ \overline{d} = mean\{d_i\}, d_i = \min_{s_i,s_j \in S: s_i \neq s_j} ||F(s_i) - F(s_j)||_1,\ F = (f_1, f_2).$$

В случае отдельного решения задач $SP_{single} = 0.984$, а при решении задач методом с балансировкой нагрузки $SP_{multi} = 0.749$, что говорит о более качественном приближении решения.

5.4. Итоги

В данном разделе была реализована поддержка нелинейных ограничений в алгоритме, решающем множество задач глобальной оптимизации в совокупности и распределяющего свои ресурсы так, чтобы обеспечивать равномерную сходимость во всех задачах в совокупности. Доказано теорема о достаточных условиях сходимости полученного метода. Свойство равномерной сходимости проверено с помощью численного эксперимента. Также в ходе численных экспериментов была оценена эффективность выполненной параллельной реализации метода и показана эффективность совместного решения множества задач на примере получения Парето фронта в многокритериальной задаче с нелинейными ограничениями.

Заключение

В рамках квалификационной работы была поставлена цель выбрать оптимальную модификацию алгоритма глобального поиска (AGS) и на его основе просроить метод для одновременного решения множества задач глобальной оптимизации с нелинейными ограничениями.

В ходе работы получены следующие результаты:

1.

Список литературы

1. *Kvasov D. E.*, *Sergeyev Y. D.* Lipschitz global optimization methods in control problems // Automation and Remote Control. — 2013. — Vol. 74, no. 9. — P. 1435–1448.
2. SVM Regression Parameters Optimization Using Parallel Global Search Algorithm / K. Barkalov, A. Polovinkin, I. Meyerov, S. Sidorov, N. Zolotykh // Parallel Computing Technologies: 12th International Conference, PaCT 2013, St. Petersburg, Russia, September 30 - October 4, 2013. Proceedings / ed. by V. Malyshkin. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. — P. 154–166.
3. *Vavasis S. A.* Complexity Issues in Global Optimization // Handbook of Global Optimization / ed. by R. Horst, P. M. Pardalos. — Boston, MA : Springer US, 1995. — P. 27–41.
4. *Torczon V.* On the convergence of pattern search algorithms // SIAM Journal on Optimization. — 1997. — Т. 9, № 1. — С. 1—25.
5. *Jones, D.R.* The direct global optimization algorithm // The Encyclopedia of Optimization. — Springer, Heidelberg, 2009. — С. 725—735. — DOI: 10.1007/978-0-387-74759-0_128.

6. *Paulavivcius, R., Zilinskas, J., Grothey, A.* Parallel branch and bound for global optimization with combination of Lipschitz bounds // Optim. Method. Softw. — 1997. — Т. 26, № 3. — С. 487—498. — DOI: 10.1080/10556788.2010.551537.

7. *Evtushenko Y.*, *Posypkin M.* A deterministic approach to global box-constrained optimization // Optim. Lett. — 2013. — Т. 7. — С. 819—829. — DOI: 10.1007/s11590-012-0452-1.

8. *Strongin R.G., Sergeyev Ya.D.* Global optimization with non-convex constraints. Sequential and parallel algorithms. — Dordrecht : Kluwer Academic Publishers, 2000. — DOI: 10.1007/978-1-4615-4677-1.

9. *Sergeyev Y.*, *Kvasov D.* Deterministic Global Optimization. — 01.2017. — DOI: 10.1007/978-1-4939-7199-2.

10. *Норкин В. И.* О методе Пиявского для решения общей задачи глобальной оптимизации // Ж. вычисл. матем. и матем. физ. — 1992. — Т. 32, вып. 7. — С. 992—1006.

11. *Sergeyev, Y.D., Strongin, R.G., Lera, D.* Introduction to global optimization exploiting space-filling curves. — Springer, New York : Springer Briefs in Optimization, 2013. — DOI: 10.1007/978-1-4614-8042-6.

12. *Storn R.*, *Price K.* Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces // Journal of Global Optimization. — 1997. — Дек. — Т. 11, № 4. — С. 341—359. — ISSN 1573-2916. — DOI: 10.1023/A:1008202821328.

13. *Schluter M.*, *Egea J. A.*, *Banga J. R.* Extended ant colony optimization for non-convex mixed integer nonlinear programming // Computers & Operations Research. — 2009. — Т. 36, № 7. — С. 2217—2229. — ISSN 0305-0548. — DOI: 10.1016/j.cor.2008.08.015.

14. *Kennedy J.*, *Eberhart R.* Particle swarm optimization // Proceedings of ICNN'95 - International Conference on Neural Networks. Т. 4. — 11.1995. — 1942—1948 vol.4. — DOI: 10.1109/ICNN.1995.488968.

15. *Rastrigin L. A.* About Convergence of Random Search Method in Extremal Control of Multi-Parameter Systems // Avtomat. i Telemekh. — 1963. — Нояб. — Т. 24. — С. 1467—1473.

16. Generalized simulated annealing algorithm and its application to the Thomson model / Y. Xiang, D. Sun, W. Fan, X. Gong // Physics Letters A. — 1997. — Т. 233, № 3. — С. 216—220. — DOI: 10.1016/S0375-9601(97)00474-X.

17. *Hu X.-M.*, *Zhang J.*, *Li Y.* Orthogonal Methods Based Ant Colony Search for Solving Continuous Optimization Problems // Journal of Computer Science and Technology. — 2008. — Янв. — Т. 23. — С. 2—18. — DOI: 10.1007/s11390-008-9111-5.

18. *Городецкий С.Ю. Г. В.* Нелинейное программирование и многоэкстремальная оптимизация. Учебное пособие. — Нижний Новгород : Изд-во ННГУ, 2007. — С. 489.

19. *Пиявский С. А.* Один алгоритм отыскания абсолютного экстремума функций // Ж. вычисл. матем. и матем. физ. — 1972. — Т. 12, вып. 4. — С. 888—896.

20. *Strongin R.* Numerical Methods in Multiextremal Problems (Information-StatisticalAlgorithms). — Moscow: Nauka (In Russian), 1978.

21. *Sergeyev Y.*, *Kvasov D.* A deterministic global optimization using smooth diagonal auxiliary functions // Communications in Nonlinear Science and Numerical Simulation. — 2015. — Т. 21, № 1—3. — С. 99—111.

22. *Paulavičius R.*, *Zilinskas J.* Simplicial Lipschitz optimization without the Lipschitz constant // Journal of Global Optimization. — 2014. — Май. — Т. 59. — С. 23—40. — DOI: 10.1007/s10898-013-0089-3.

23. *Kämpf J.*, *Wetter M.*, *Robinson D.* A comparison of global optimisation algorithms with standard benchmark functions and real-world applications using EnergyPlus // Journal of Building Performance Simulation. — 2010. — Июнь. — Т. 3. — DOI: 10.1080/19401490903494597.

24. *Gaviano, M., Kvasov, D.E., Lera, D., Sergeev, Ya.D.* Software for generation of classes of test functions with known local and global minima for global optimization // ACM Transactions on Mathematical Software. — 2003. — Т. 29, № 4. — С. 469—480. — DOI: 10.1145/962437.962444.

25. *Гришагин В.А.* Операционные характеристики некоторых алгоритмов глобального поиска // Проблемы статистической оптимизации. — Рига : Зинатне, 1978.

26. *Beiranvand V.*, *Hare W.*, *Lucet Y.* Best practices for comparing optimization algorithms // Optimization and Engineering. — 2017. — Дек. — Т. 18, № 4. — С. 815—848. — DOI: 10.1007/s11081-017-9366-1.

27. *To T. B.*, *Korn B.* MOBES: A Multiobjective Evolution Strategy for Constrained Optimization Problems. — 1999. — Март.

28. *Lera D.*, *Sergeyev Y.* An information global minimization algorithm using the local improvement technique // J. Glob. Optim. — 2010. — Т. 48, № 1. — С. 99—112. — DOI: 10.1007/s10898-009-9508-x.

29. A flexible generator of constrained global optimization test problems / V. Gergel, K. Barkalov, I. Lebedev, M. Rachinskaya, A. Sysoyev //. Т. 2070. — 02.2019. — С. 020009. — DOI: 10.1063/1.5089976.

30. *Gergel V.P., Barkalov K.A., and Sysoyev A.V.* A novel supercomputer software system for solving time-consuming global optimization problems // Numerical Algebra, Control & Optimization. — 2018. — Т. 8, № 1. — С. 47—62. — DOI: 10.3934/naco.2018003.

31. *Strongin, R.G., Gergel, V.P., Barkalov, K.A.* Parallel methods for global optimization problem solving // Journal of instrument engineering. — 2009 (In Russian). — Т. 52. — С. 25—33.

32. *R. G. Strongin*. Algorithms for multi-extremal mathematical programming problems employing a set of joint space-filling curves // J. Glob. Optim. — 1992. — Т. 2. — С. 357—378.

33. *R. G. Strongin*. Parallel multi-extremal optimization using a set of evolvents // Comp. Math. Math. Phys. — 1991. — Т. 31, № 8. — С. 37—46.

34. *Goryachih, A.* A Class of Smooth Modification of Space-Filling Curves for Global Optimization Problems // Models, Algorithms, and Technologies for Network Analysis. NET 2016. — Springer, Cham, 2017. — С. 152—159.

35. *Kan A. H. G. R.*, *Timmer G. T.* Stochastic global optimization methods part II: Multi level methods // Math. Program. — 1987. — Т. 39. — С. 57—78. — DOI: 10.1007/BF02592071.

36. *Hickernell F. J.*, *Yuan Y.-x.* A Simple Multistart Algorithm for Global Optimization. — 1997.

37. *Gablonsky, J.M., Kelley, C.T.* A locally-biased form of the DIRECT algorithm // J. Glob. Optim. — 2001. — Т. 21, № 1. — С. 27—37. — DOI: 10.1023/A:1017930332101.

38. *Kirkpatrick S.*, *Jr D.*, *Vecchi M.* Optimization by Simulated Annealing // Science. — 1983. — Янв. — Т. 220. — С. 671—680. — DOI: 10.1142/9789812799371_0035.

39. *Szu H.*, *Hartley R.* Fast simulated annealing // Physics Letters A. — 1987. — Т. 122, № 3. — С. 157—162. — ISSN 0375-9601. — DOI: https://doi.org/10.1016/0375-9601(87)90796-1. — URL: https://www.sciencedirect.com/science/article/pii/0375960187907961

40. *Price W. L.* Global optimization by controlled random search // Journal of Optimization Theory and Applications. — 1983. — Июль. — Т. 40, № 3. — С. 333—348. — DOI: 10.1007/BF00933504.

41. *Madsen K.*, *Zertchaninov S.* A New Branch-and-Bound Method for Global Optimization. — 1998.

42. *Johnson S. G.* The NLopt nonlinear-optimization package. — URL: http://ab-initio.mit.edu/nlopt ; [Online; accessed <24.12.2018>].

43. SciPy: Open source scientific tools for Python / E. Jones, T. Oliphant, P. Peterson [и др.]. — 2001–. — URL: http://www.scipy.org/ ; [Online; accessed <24.12.2018>].

44. *Gergel V.*, *Barkalov K.*, *Lebedev I.* A Global Optimization Algorithm for Non-Convex Mixed-Integer Problems // Learning and Intelligent Optimization / под ред. R. Battiti, M. Brunato, I. Kotsireas, P. M. Pardalos. — Cham : Springer International Publishing, 2019. — С. 78—81. — ISBN 978-3-030-05348-2.

45. *Barkalov K.*, *Strongin R.* Solving a Set of Global Optimization Problems by the Parallel Technique with Uniform Convergence // J. of Global Optimization. — Norwell, MA, USA, 2018. — Май. — Т. 71, № 1. — С. 21—36. — ISSN 0925-5001. — DOI: 10.1007/s10898-017-0555-4.

46. *Riquelme N.*, *Von Lucken C.*, *Baran B.* Performance metrics in multi-objective optimization // 2015 Latin American Computing Conference (CLEI). — 10.2015. — С. 1—11. — DOI: 10.1109/CLEI.2015.7360024.