УДК 519.853

МЕТОД ЛИНЕАРИЗАЦИИ И НЕГЛАДКАЯ ОПТИМИЗАЦИЯ

Э.И. НЕНАХОВ, Л.А. СОБОЛЕНКО

Рассматриваются две модификации нестандартного применения метода линеаризации к решению негладких оптимизационных задач. На основе модификации для задач обратно-выпуклого программирования разработан пакет прикладных программ Packing. Показана эффективность работы пакета и этой модификации на примерах различных задач упаковки и размещения объектов.

ВВЕДЕНИЕ

Метод линеаризации [1], используемый на протяжении многих лет, показал себя как один из самых эффективных методов решения общих задач математического программирования. При этом главным требованием к функциям, описывающим оптимизационную задачу, является их гладкость. Однако идеи, положенные в основу метода линеаризации, могут быть успешно использованы при решении многих других задач, которые не вкладываются в его первоначальную схему. Хотя метод создан для решения гладких задач оптимизации, он сразу нашел применение к решению некоторых минимаксных задач.

Для решения общей задачи выпуклого программирования в [2] разработан комбинированный метод, основанный на идеях трех методов: линеаризации, отсечения и точных штрафных функций. Комбинированный метод сходится при общих предположениях и позволяет оценить множители Лагранжа. При его модификации отбрасываются несущественные для решения задачи ограничения, что позволяет значительно уменьшить размерность решаемой на каждой итерации вспомогательной квадратичной задачи. Стабилизация штрафного коэффициента — основа быстрой сходимости построенных алгоритмов.

Комбинированный метод решения общей задачи выпуклого программирования с негладкими функциями показал свою эффективность на сложных тестовых задачах и достаточную конкурентоспособность по сравнению с другими методами оптимизации [2].

Еще одним классом негладких оптимизационных задач являются задачи обратно-выпуклого программирования. Эти задачи — многоэкстремальные. В работе [3] предложена эффективная модификация метода линеаризации для отыскания локальных экстремумов общей задачи обратно-

выпуклого программирования. Ответить на вопрос, является ли полученное решение глобальным экстремумом, в общей постановке крайне сложно. Однако численные результаты показали, что при решении отдельных конкретных задач получение глобального экстремума небезуспешно.

В общую схему задачи обратно-выпуклого программирования вкладываются многие формализованные математические модели задач упаковки и размещения различных объектов в пространстве R^n. Эти задачи широко применяются. Поэтому так велик к ним интерес многих специалистов, в том числе работающих в области численных методов оптимизации. В отличие от традиционных методов классической математики решения этих задач [4,5] при численном их решении появляется возможность значительно увеличивать размерности задач и снимать многие ограничения на формы и размеры объектов.

Существуют различные подходы к численному решению таких задач. Часто рассматриваются и решаются отдельные конкретные задачи упаковки [6–8]. В данной работе используется подход, основанный на формулировке проблемы упаковки в виде общей оптимизационной задачи обратно-выпуклого программирования и решении ее модифицированным методом линеаризации [3]. На основе этого метода разработан и реализован графический пакет программ Packing [9], с помощью которого пользователь имеет возможность, легко изменяя начальные положения рассматриваемых объектов, получать различные локальные решения и выбирать из них наилучшее. Удобная интерактивная среда Packing предоставляет в распоряжение пользователя не только численные результаты, но и визуальные изображения положений объектов (параллелепипедов, шаров и т. п.) в двумерном и трехмерном пространствах. Для пространств R^n, где $n > 3$, предоставляются только численные результаты.

Объем публикации не позволяет описать все множество решенных задач. Для наглядности графического представления в работе не приводятся примеры задач для пространств R^n, где $n > 3$. Представлены лишь несколько задач, которые показались авторам интересными.

КОМБИНИРОВАННЫЙ МЕТОД РЕШЕНИЯ ЗАДАЧИ ВЫПУКЛОГО ПРОГРАММИРОВАНИЯ

Рассмотрим задачу

$$\min_x \left\{ f(x) \mid f_j(x) \le 0;\ \ j = 1,...,m;\ \ x \in M \right\}, \qquad (1)$$

где $x \in R^n$, $f(x)$ и $f_j(x)$, $j = 1,...,m$ — выпуклые непрерывные функции; M — выпуклый многогранник. Через x^* обозначим решение задачи (1). При выполнении условия Слейтера x^* множители Лагранжа задачи (1) u_j, $j = 1,\ldots,m$, существуют.

Для произвольной выпуклой функции $f(x)$ ее субдифференциал в точке x существует. Обозначим его $\partial f(x)$. Элементы выпуклого компактного множества $\partial f(x)$ обозначим $f'(x)$. Предположим в дальнейшем, что

выбрано правило, которое ставит в соответствие точке x элемент $f'(x) \in \partial f(x)$. Исследуемые алгоритмы могут отличаться правилом выбора $f'(x)$, что влияет на эффективность конкретной реализации алгоритма, но от этого не зависит его сходимость.

Если определить выпуклую функцию $\phi(x) = \max\limits_j f_j(x)$, то задача (1) эквивалентна задаче

$$\min_x \{f(x) \mid \phi(x) \le 0; x \in M\}. \qquad (2)$$

Введем штрафную функцию $\Phi_N(x) = f(x) + N \max\{0, \phi(x)\}$, $N > 0$.

Лемма. Если число N достаточно велико, $N > \sum\limits_{j=1}^{m} u_j$, то задачи (1), (2) и

$$\min_x \{\Phi_N(x) \mid x \in M\} = \min_{x,\xi_1,\xi_2} \{\xi_1 + N\xi_2 \mid f(x) \le \xi_1, \phi(x) \le \xi_2, \xi_2 \ge 0; x \in M\} \qquad (3)$$

эквивалентны. Решение задачи (3) есть x^*, $\xi_1^* = f(x^*)$, $\xi_2^* = 0$.

Опишем алгоритм решения задачи (1). Пусть заданы исходная точка $x_1 \in M$ и число $N > 0$, а исходный массив X_1 состоит из единственной точки x_1.

Если массив X_k и число N_k построены, то массив X_{k+1} и число N_{k+1} определяются по такому правилу. Вычисляем $\bar{x}_k$ из условия $\min\{\Phi_{N_k}(x_i) \mid x_i \in X_k\}$. Полагаем $N = N_k$ и решаем вспомогательную задачу квадратичного программирования в пространстве R^{n+2}

$$\min_{x,\xi_1,\xi_2} \frac{1}{2}\|x - \bar{x}_k\|^2 + \xi_1 + N\xi_2,$$

$$f(x_i) + (f'(x_i), x - x_i) \le \xi_1, \ i = 1, \ldots, k, \qquad (4)$$

$$\phi(x_i) + (\phi'(x_i), x - x_i) \le \xi_2, \ i = 1, \ldots, k, \quad \xi_2 \ge 0, \ x \in M.$$

Здесь и далее $\|.\|$ означает евклидову норму вектора, а (.,.) скалярное произведение двух векторов. Полагаем $X_{k+1} = X_k \cup \{x_{k+1}\}$,

$$N_{k+1} = \begin{cases} N_k, & \text{если} \quad \xi_2^k = 0, \\ 2N_k, & \text{если} \quad \xi_2^k > 0, \end{cases}$$

где x_{k+1}, ξ_1^k, ξ_2^k — решение задачи (4).

Начиная с некоторого достаточно большого номера k, N_k будет константой N и $\xi_2^k = 0$.

Кроме того, справедливо равенство

$$\lim_{k\to\infty} [\Phi_N(x_k) - (\xi_1^k + N\xi_2^k)] = 0.$$

Отсюда, для вектора $p_k = x_{k+1} - \bar{x}_k$ выполняется $\lim\limits_{k\to\infty} \|p_k\| = 0$. В предельной точке последовательности $\{x_k\} \subset M$ выполняются необходимые и

достаточные условия экстремума, а критерием останова итерационного процесса является $\|p_k\| \le \varepsilon$, $\varepsilon > 0$ [2].

Комбинированный метод может привести к большому накоплению информации, т.е. к неограниченному росту точек массива X_k. Некоторые из этих точек не играют роли в окончательном процессе нахождения решения задачи. Поэтому предлагается процедура отбрасывания несущественных для решения ограничений, которая позволяет существенно уменьшить размерность решаемой на каждой итерации квадратичной задачи (4).

Если для точки $x_i \in X_k$ выполняется неравенство

$$f(x_i) + (f'(x_i), x_{k+1} - x_i) \le \xi_1^k - \delta\|p_k\|^2, \ \delta > 0,$$

то при решении задачи (4) на всех последующих итерациях соответствующее ограничение в ней отбрасывается. Если же выполняется неравенство

$$\phi(x_i) + (\phi'(x_i), x_{k+1} - x_i) \le \xi_2^k - \delta\|p_k\|^2, \ \delta > 0,$$

то аналогично отбрасывается соответствующее ограничение в задаче (4). Но неравенства, соответствующие точкам x_{k+1} и $\bar{x}_k$, всегда участвуют в решении вспомогательной задачи.

Построенный комбинированный метод, в отличие от хорошо известных методов решения гладких задач, не содержат выбора шагового множителя вдоль направления спуска.

МЕТОД РЕШЕНИЯ ЗАДАЧИ ОБРАТНО-ВЫПУКЛОГО ПРОГРАММИРОВАНИЯ

Рассмотрим задачу

$$\max_x \{f_0(x) | f_i(x) \ge 0, \quad i = 1, \dots, m\}, \tag{5}$$

где $x \in R^n$, $f_i(x)$, $i = 0,1,\dots,m$ — выпуклые непрерывные функции. Несмотря на то, что функции $f_i(x)$ выпуклы, задача (5) не является выпуклой в обычном понимании, и нельзя утверждать о единственности ее решения. Она многоэкстремальная.

Пусть $D = \{x : f_i(x) \ge 0, \quad i = 1, \dots, m\}$ — допустимое множество задачи (5). Предположим, что начальная точка $x_0 \in D$. Если точка x_k уже построена, то новая точка вычисляется так:

$$x_{k+1} = x_k + p_k. \tag{6}$$

Вектор $p_k = p(x_k)$ определяет направление движения в итерационном процессе и является решением следующей задачи квадратичного программирования:

$$\min_{p \in R^n} \left\{ -(f_0'(x_k), p) + \frac{1}{2}\|p\|^2 \mid (f_i'(x_k), p) + f_i(x_k) \ge 0, \quad i = 1, \dots, m \right\}. \tag{7}$$

Существенно, что шаговый множитель в направлении движения не вычисляется, а сразу полагается равным 1.

В работе [3] доказано, что все точки итерационной последовательности (6) $x_k \in D$, $k = 1,\ldots$. В каждой из них вспомогательная задача (7) имеет единственное решение $p(x_k)$. При $k \to \infty$, $\|p_k\| \to 0$ и в любой предельной точке последовательности (6) выполняются необходимые условия экстремума и ограничения задачи (5). На основе этих утверждений итерационный процесс (6) останавливается, когда $\|p_k\| \le \varepsilon$, где ε — заданная точность решения исходной задачи (5).

Пусть $u^i(x_k)$, $i = 1,\ldots,m$ — множители Лагранжа, соответствующие вспомогательной задаче (7). Они являются решением задачи, двойственной к задаче (7).

$$\min_{u \in R^m} \left\{ -\frac{1}{2} \left\| f_i'(x_k) + \sum_{i=1}^{m} u^i f_i'(x_k) \right\|^2 + \sum_{i=1}^{m} u^i f_i(x_k) \mid \quad u^i \ge 0, \quad i = 1,\ldots,m \right\}. \quad (8)$$

Решения задач (7) и (8) связаны соотношением

$$p(x_k) = f_0'(x_k) + \sum_{i=1}^{m} u^i(x_k) f_i'(x_k). \quad (9)$$

Задача (8) приводится к виду

$$\min_{u \in R^m} \left\{ \ \phi(u) \mid u^i \ge 0, \quad i = 1,\ldots,m \right\}, \quad (10)$$

где $\phi(u) = -\frac{1}{2}(u, Cu) + (d, u)$; C — $m \times m$-симметричная матрица с элементами $c_{i,j} = (f_i'(x_k))^{\mathrm{T}} f_j'(x_k)$, $i, j = 1,\ldots,m$, а компоненты вектора d определяются соотношением $d^i = -(f_0'(x_k))^{\mathrm{T}} f_i'(x_k) + f_i(x_k)$, $i = 1,\ldots,m$. Здесь и далее в формулах буква T вверху означает знак транспонирования.

Очевидно, что задача (10) имеет более простые ограничения по сравнению с задачей (7). Поэтому в рассматриваемом пакете программ решается задача (10), а вектор направления p_k вычисляется по формуле (9).

Метод решения задачи (10), реализованный в Packing, подробно описан в работе [10]. Это конечный метод, который относится к методам активного набора. Очень кратко суть его состоит в следующем.

Пусть $J = \{i = 1,\ldots,m \mid u^i \ge 0\}$ — индексное множество ограничений задачи (10), а $I = \{i \in J \mid u^i = 0\}$ — множество ее активных ограничений в текущей точке.

Опишем кратко действия на одной итерации, начиная с допустимой точки $u_0^i \ge 0$, $i \in J$. Вычисляем $I(u_0)$ и градиент $\phi'(u_0)$. Если $\frac{\partial \phi(u_0)}{\partial u^i} \ge 0$, для всех $i \in I(u_0)$, а $\frac{\partial \phi(u_0)}{\partial u^i} = 0$ для всех $i \notin I(u_0)$, то u_0 — решение задачи, так как выполняются необходимые и достаточные условия минимума задачи (10). Если для какого-то индекса $i \in I(u_0)$, $\frac{\partial \phi(u_0)}{\partial u^i} < 0$ либо суще-

ствуют индексы, для которых $\frac{\partial \phi(u_0)}{\partial u^i} \neq 0$, $i \notin I(u_0)$, то определяется множество $I' = \left\{ i \in I(u_0) \,|\, \frac{\partial \phi}{\partial u^i} \geq 0 \right\}$.

Удерживая нулевыми компоненты вектора u_0 по $i \in I'$, применяем известный метод сопряженных градиентов по остальным компонентам этого вектора. При этом контролируется величина шагового множителя в направлении метода сопряженных градиентов, чтобы новая точка оставалась допустимой. Через конечное число шагов будет найдена точка u_{k+1} такая, что в ней $\phi(u)$ достигает минимума при условии $u^i = 0,\ i \in I'$, либо новая допустимая точка, для которой $I(u_{k+1}) \supset I'$. В обоих случаях точка u_{k+1} считается начальной, и процесс повторяется.

ПРИМЕРЫ ЗАДАЧ

Все представленные в работе задачи решены с использованием пакета программ Packing. С помощью меню, системы окон и панели инструментов пакета пользователь указывает тип задачи, который определяется ее формализованной математической моделью, задает начальное положение рассматриваемых объектов и указывает их размеры. Некоторые начальные данные, например, размеры и начальное положение искомого оптимального объекта пакет устанавливает сам по умолчанию. Эти данные пользователь может отредактировать или оставить, если они для него приемлемы. При этом автоматически происходит графическое отображение объектов на экране компьютера. Затем по указанию пользователя пакет решает задачу. Численные результаты решения поступают в выходной файл пакета, а графическое изображение решения задачи отображается на экране компьютера. При описании задач здесь для графического представления результатов решения будем использовать средства Packing.

1. Задача определения шара наименьшего радиуса, объемлющего заданные шары различных радиусов. Пусть имеется m шаров $S_{r_i}(x_i)$, $i = 1,...,m$, с центрами $x_i \in R^n$ и радиусами r_i. Требуется найти шар наименьшего радиуса, объемлющий данные шары. Если R — искомый радиус объемлющего шара, а x — его неизвестный центр, то формально задачу можно записать так:

$$\min_{R,x,x_i} \begin{cases} R \,|\, \left\| x_i - x_j \right\| \geq r_i + r_j, \quad i < j, \quad i, j = 1,...,m, \\ \left\| x_i - x \right\| \leq R - r_i, \qquad\qquad i = 1,...,m. \end{cases} \tag{11}$$

Здесь первые C_m^2 (число сочетаний из m по 2) ограничений задачи представляют условие не пересечения заданных шаров, а вторая группа из m ограничений — условие не выхода заданных шаров за границы искомого шара.

В результате решения задачи (11) мы должны получить значения координат центров оптимального расположения упаковываемых шаров, координаты центра и радиус объемлющего шара. Отсюда и из (8) видно, что размерности задачи определяются по следующим формулам: число неизвестных равно $n_1 = n + 1 + m \times n$, число ограничений — $m_1 = C_m^2 + m$.

Рассмотрим сначала плоский случай в пространстве R^2. Пусть дано девять кругов одинакового радиуса и один круг, радиус которого в два раза больше. Нужно найти круг минимального радиуса, объемлющий эти круги. Следовательно, задача имеет 23 неизвестных и 55 ограничений. Пусть радиус седьмого круга равен 1, а для остальных кругов $r_i = 0{,}5$; $i = 1\ldots6, 8\ldots10$. Числовые значения координат центров кругов в начальном положении и в решении приведены в табл. 1.

Таблица 1. Центры кругов в начальном положении и в точках решения

Номер круга	x_0^1	x_0^2	x_*^1	x_*^2
0	0	0	3,17	3,91
1	1	3	1,70	3,61
2	3	2	3,15	2,41
3	3	6	2,73	5,34
4	5	4	4,65	3,66
5	6	1	4,11	2,74

Номер круга	x_0^1	x_0^2	x_*^1	x_*^2
6	7	7	4,48	4,64
7	3	4	3,17	3,91
8	4	8	3,73	5,30
9	2	6	1,92	4,74
10	1	2	2,21	2,75

В табл. 1 нулевой номер определен для внешнего круга, а дальше идут номера упаковываемых кругов. Для координаты x^i нижний индекс 0 дан для начальных данных, а «*» — для результатов решения. Этот принцип заполнения таблиц будет сохранен и при описании остальных задач. Начальное значение для радиуса объемлющего круга Packing установил $R_0 = 10{,}40$. Это значение R_0 вполне допустимо, так как все круги в начальном положении оказываются внутри объемлющего круга. Таким образом, предположение в алгоритме $x_0 \in D$ удовлетворяется. Графические отображения кругов в начальном положении и положение кругов в точках решения, данные пакетом, показаны на рис. 1. Начальное положение — левая часть рисунка, правая — положение кругов в решении. Здесь внешний объект (в данном случае круг) по ука-

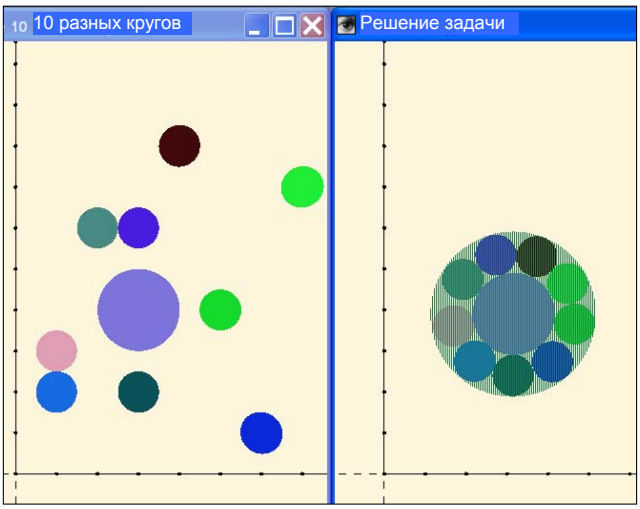

Рис. 1. Упаковка 10-ти кругов разных радиусов в круг минимального радиуса

занию пользователя отображается только справа. Внешний объект Packing всегда подается сероватым прозрачным фоном. Радиус наименьшего круга, объемлющего данные круги, $R_* = 2{,}00$. Из рис. 1 очевидно, что получено оптимальное решение с точностью $\|p_k\| \le 0{,}5 \cdot 10^{-2}$ на 26-й итерации.

Приведем пример упаковки 21-го шара одинакового радиуса в шар наименьшего радиуса в трехмерном пространстве. Размерности этой задачи следующие: число неизвестных — 67, ограничений — 231. Пусть радиусы шаров r_i, $i = 1,\ldots,21$ равны 0,5, а начальное значение радиуса искомого шара R_0 равно 12,71. Числовые значения координат центров шаров в начальном положении и в решении приведены в табл. 2.

Таблица 2. Центры шаров в начальном положении и в точках решения

Номер шара	x_0^1	x_0^2	x_0^3	x_*^1	x_*^2	x_*^3
0	0	0	0	2,58	3,23	2,04
1	2	2	5	2,55	2,37	2,94
2	3	6	1	2,24	4,40	1,83
3	2	4	6	2,00	3,18	3,13
4	6	5	6	3,00	3,22	3,21
5	7	3	2	3,56	2,58	1,61
6	6	7	2	2,88	4,04	1,15
7	8	7	6	3,67	3,56	2,55
8	6	4	4	3,47	2,58	2,61
9	5	7	5	3,21	4,30	2,06
10	7	6	3	3,67	3,57	1,55

Номер шара	x_0^1	x_0^2	x_0^3	x_*^1	x_*^2	x_*^3
11	4	4	3	2,76	3,26	2,24
12	3	7	5	1,74	3,97	2,57
13	5	6	5	2,68	4,12	2,90
14	–3	3	2	1,39	3,06	2,35
15	3	0	0	2,86	2,01	2,05
16	0	5	0	1,53	3,76	1,63
17	3	3	0	3,08	3,09	0,90
18	–2	–3	–3	2,48	2,33	1,18
19	–3	0	–5	2,14	3,43	0,89
20	0	–3	4	1,89	2,20	2,20
21	0	–3	4	1,61	2,76	1,42

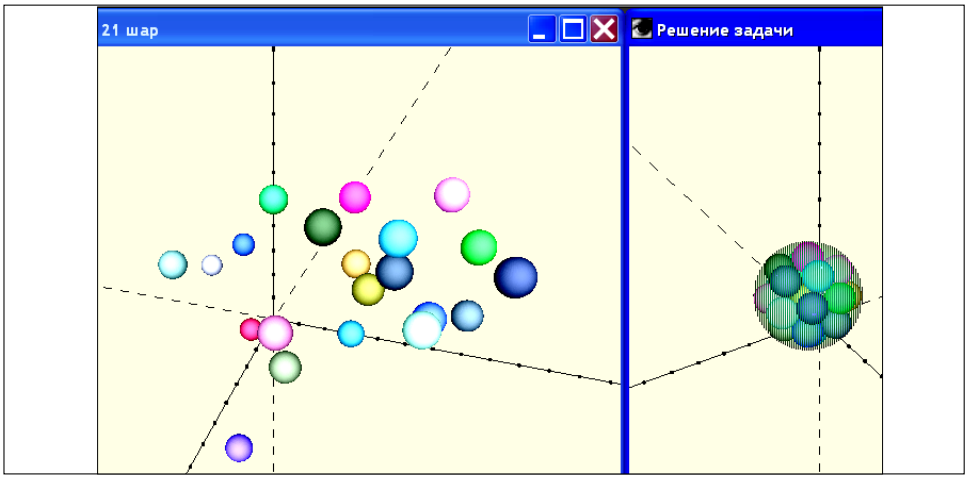

Рис. 2. Упаковка 21-го шара в шар минимального радиуса

На рис. 2 дано графическое отображение шаров: слева — начальное положение, справа — положение шаров в решении и объемлющий их шар. Радиус объемлющего шара $R_* = 1{,}74$. Число итераций, используемых мето-

дом, — 194. На рисунке видно, что внешний шар очень плотно объемлет заданные шары.

2. Задача упаковки шаров в параллелепипед с минимальной суммой сторон. Требуется упаковать шары $S_{r_i}(x_i)$, $i=1,...,m$, с центрами $x_i \in R^n$ и радиусами r_i в параллелепипед, сумма сторон которого минимальна. Если неизвестные стороны параллелепипеда обозначить ξ^k, $k=1,...,n$, то формально задача сведется к оптимизационной задаче

$$\min_{\xi^k,\, x_i} \left\{ \begin{array}{ll} \sum\limits_{k=1}^{m} \xi^k \left| \left\| x_i - x_j \right\| \ge r_i + r_j, & i<j,\ i,j=1,...,m, \\ r_i \le x_i^k \le \xi^k - r_i, & i=1,...,m,\ k=1,...,n. \end{array} \right\}. \qquad (12)$$

Здесь формулы для определения размерностей задачи имеют вид $n_1 = n + m \times n$, $m_1 = C_m^2 + 2m \times n$.

Эту задачу рассмотрим на примере упаковки 16-ти кругов одинакового радиуса в прямоугольник с минимальной суммой сторон. Тогда $n_1 = 34$, $m_1 = 184$. Пусть радиусы $r_{i,}$ $i=1,...,16$, заданных кругов равны 0,5, а начальные значения для сторон прямоугольника, в который упаковываются шары, $\xi_0^1 = 25$, $\xi_0^2 = 26$. Координаты центров начального расположения заданных 16-ти кругов x_0^i, $i=1,2$, и их координаты в решении x_*^i, $i=1,2$, приведены в табл. 3.

Таблица 3. Центры кругов в начальном положении и в точках решения

Номер круга	x_0^1	x_0^2	x_*^1	x_*^2
1	7	1	3,5	1,5
2	3	8	1,5	3,5
3	1	10	0,5	3,5
4	2	2	1,5	0,5
5	4	3	2,5	2,5
6	6	3	3,5	3,5
7	4	2	2,5	1,5
8	5	1	2,5	0,5

Номер круга	x_0^1	x_0^2	x_*^1	x_*^2
9	9	0	3,5	0,5
10	3	6	1,5	2,5
11	4	4	2,5	3,5
12	0,5	0,5	0,5	0,5
13	9	2,5	3,5	2,5
14	1	6	0,5	2,5
15	1	4	0,5	1,5
16	3	5	1,5	1,5

Размеры полученного прямоугольника с минимальной суммой сторон, содержащего заданные круги, такие: $\xi_*^1 = 4{,}0$, $\xi_*^2 = 4{,}0$. Для получения этого решения с заданной точностью $\varepsilon = 0{,}01$ описанному методу решения задачи обратно-выпуклого программирования потребовалось выполнить 61 итерацию. Графическое отображение начала и окончания итерационного процесса (6), (7) дано на рис. 3. В левой части рисунка изображены круги в начальном положении, справа — круги в точках решения и полученный оптимальный прямоугольник, в который они упакованы.

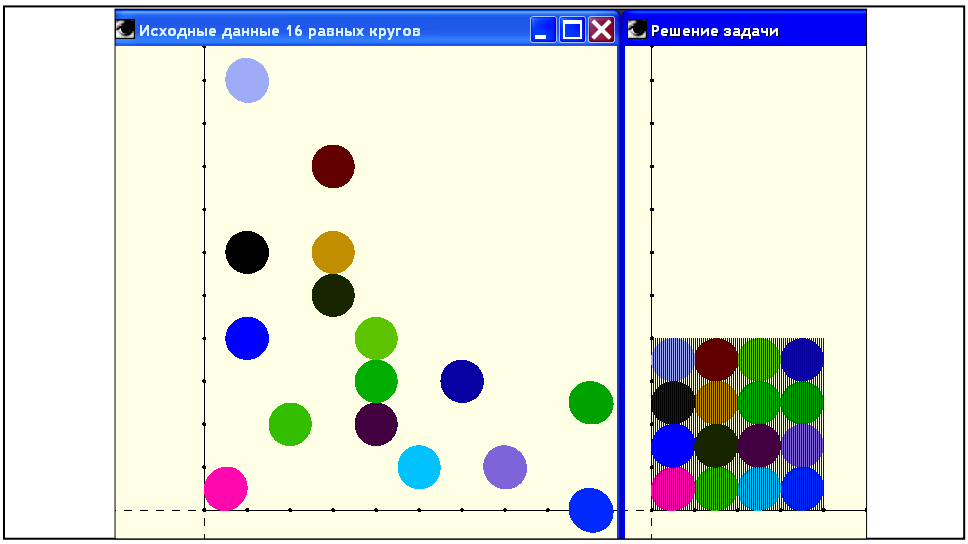

Рис. 3. Упаковка 16-ти кругов в параллелепипед с минимальной суммой сторон

3. Задача оптимальной упаковки параллелепипедов. Пусть параллелепипед в пространстве R^n задается с помощью центра $x \in R^n$ и вектора полуосей $a \in R^n$, т.е. параллелепипед $C_a(x)$ определен системой неравенств

$$C_a(x) = \left\{ y \in R^n \mid \left\| y^i - x^i \right\| \le a^i, \quad i = 1,\dots,n \right\}. \tag{13}$$

Рассмотрим непересекающиеся параллелепипеды. Чтобы выразить условие не пересечения двух параллелепипедов $C_a(x)$ и $C_b(y)$, $x, y \in R^n$, в работе [11] введено понятие квазирасстояния с помощью функции

$$\rho_\varepsilon(C_a(x), C_b(y)) = \sum_{k=1}^{n} f_\varepsilon\left(\left| x^k - y^k \right| - (a^k + b^k) \right). \tag{14}$$

Если обозначить $t = | x^k - y^k | - (a^k + b^k)$, то $f_\varepsilon(t) = \begin{cases} t, & t \ge 0, \\ \varepsilon t, & t < 0, \end{cases}$ $\varepsilon \in (0,1)$. Два параллелепипеда $C_a(x)$ и $C_b(y)$ не пересекаются, когда $\rho_\varepsilon\left(C_a(x), C_b(y)\right) \ge 0$.

При фиксированных значениях a и b $\rho_\varepsilon(x,y) = \rho_\varepsilon\left(C_a(x), C_b(y)\right)$ является выпуклой положительно однородной функцией аргументов x, y. Следовательно, можно найти ее субградиенты.

Сформулируем задачу оптимальной упаковки параллелепипедов.

Пусть имеется m непересекающихся параллелепипедов $C_{a_j}(x_j)$, $j = 1,\dots,m$, и контейнер $C_A = \{ y \in R^n : \ 0 < y^k \le A^k, \quad k = 1,\dots,n \}$, в который их нужно уложить. Здесь A^k — компоненты вектора $A \in R^n$, определяющего размеры контейнера. Предполагается, что параллелепипеды можно перемещать только параллельным сдвигом. Обозначим ξ^k, $k = 1,\dots,n$, величины

выхода параллелепипедов за границу контейнера по каждой из сторон A^k, $k=1,...,n$. Тогда получим следующую оптимизационную задачу:

$$\min_{\xi^k, x_i} \left\{ \begin{array}{ll} \sum_{k=1}^{n} \xi^k \mid \sum_{k=1}^{n} f_\varepsilon\left(\mid x_i^k - x_j^k \mid -(a_i^k + a_j^k) \right) \geq 0, & i<j,\ i,j=1,...,m, \\ x_i^k - a_i^k \geq 0, & i=1,...,m,\ k=1,..,n, \\ x_i^k + a_i^k \leq A^k + \xi^k, & i=1,...,m,\ k=1,..,n. \end{array} \right\} \quad (15)$$

Замечание. Из формул (15) следует, что для величин ξ^k, $k=1,...,n$, допускаются как положительные, так и отрицательные значения. Это позволяет не только найти центры оптимальной укладки параллелепипедов, но и размеры самого контейнера, если они неизвестны. Для этого достаточно задать значения A^k,, $k=1,...,n$, произвольно, а затем, решив задачу, скорректировать их на основе полученных значений ξ^k, $k=1,...,n$.

Размерности задачи (15) определяются по формулам $n_1 = n + m \times n$, а число ограничений $m_1 = C_m^2 + 2m \times n$.

В качестве примера рассмотрим задачу в трехмерном пространстве. Пусть задан контейнер следующих размеров: $A^1 = 3$, $A^2 = 2$, $A^3 = 3$ и 10 непересекающихся параллелепипедов различной формы и ориентации. Размеры параллелепипедов и координаты их центров в начальном положении указаны в табл. 4, соответствующие функции ρ_ε положительны. Их нужно уложить в контейнер таким образом, чтобы сумма величин, обозначающих нарушение границ контейнера, была минимальной.

Таблица 4. Размеры параллелепипедов и их центры в начальном положении

Номер паралл.	x_0^1	x_0^2	x_0^3	a^1	a^2	a^3
1	3	3	4	0,5	1	0,5
2	6	3	5	0,5	1	0,5
3	5	2	1	1	0,5	0,5
4	6	4	3,5	0,5	0,5	0,5
5	6	4	0,5	0,5	0,5	0,5

Номер паралл	x_0^1	x_0^2	x_0^3	a^1	a^2	a^3
6	4	5	0	1	1	0,5
7	6	0	0	0,5	0,5	0,5
8	1,5	1	5	0,5	1	0,5
9	2	2	2	0,5	1	0,5
10	4	4	2	0,5	0,5	0,5

Решение на рис. 4 справа получено с точностью $\|p_k\| \leq 0,009$ на 32-й итерации алгоритма (6), (7). При этом пакету программ Packing дано задание изображать внешний объект (в данном случае контейнер). В левой части рис. 4 виден контейнер и параллелепипеды в начальном положении. Вначале ни один параллелепипед полностью не расположен внутри контейнера. Следовательно, условие $x_0 \in D$ не выполнено. Однако это не является препятствием для метода, поскольку в точке x_0 оказалась разрешимой вспомогательная задача (7), и тогда сходимость алгоритма обеспечена [3].

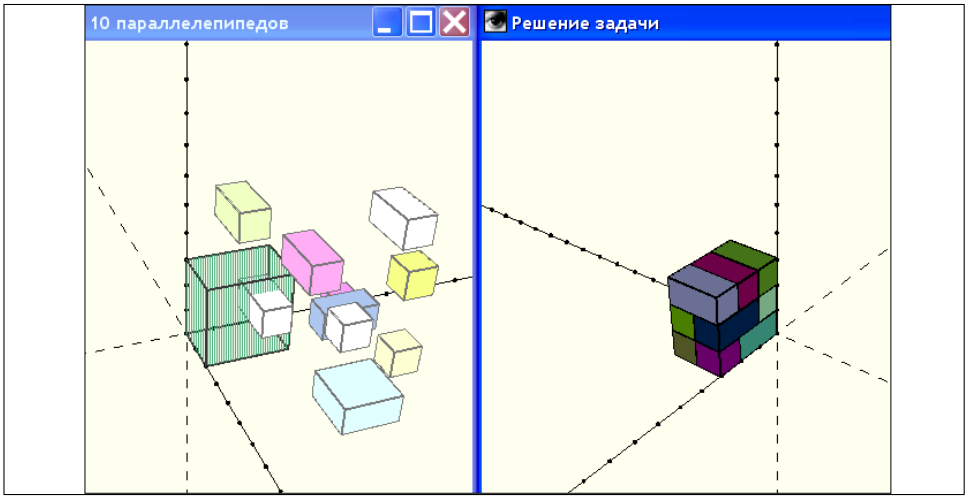

Рис. 4. Упаковка 10-ти разных параллелепипедов в заданный контейнер

В правой части рис. 4 все параллелепипеды находятся внутри контейнера, а $\xi_*^1 = 0,059$; $\xi_*^2 = -0,098$; $\xi_*^3 = 0,106$. Значения координат центров параллелепипедов в решении даны в табл. 5.

Таблица 5. Оптимальные значения координат центров параллелепипедов

Номер паралл.	x_*^1	x_*^2	x_*^3
1	1,53	1,02	2,54
2	2,56	1,02	2,56
3	2,05	0,50	1,52
4	2,56	1,52	1,54
5	2,56	1,52	0,51

Номер паралл.	x_*^1	x_*^2	x_*^3
6	1,04	1,02	0,50
7	2,56	0,50	0,50
8	0,50	1,00	2,56
9	0,51	1,02	1,53
10	1,53	1,52	1,52

4. Задача оптимальной упаковки грузов в самолет. Пусть имеется m грузов для перевозок самолетом. Известны массы грузов m_i, $i = 1,...,m$, и центр тяжести самолета, определяемый вектором $x_* \in R^n$. Для обеспечения нормального маневра самолета грузы необходимо уложить в имеющийся в самолете контейнер так, чтобы в результате увеличения нагрузки смещение известного центра тяжести самолета x_* было минимальным.

Предположим, что грузы и контейнер самолета являются параллелепипедами. Совместим начало координатных осей с точкой x_*. Если x_* не совпадает с центром контейнера, то полуоси контейнера не равны между собой. Пусть $A^k \geq 0$, $k = 1,...,n$ — полуоси контейнера, которые соответствуют отрицательным осям координат, а $B^k \geq 0$, $k = 1,...,n$, положительным. Грузы задаются своими центрами x_i, $i = 1,...,m$, и полуосями a_i, $i = 1,...,m$, т.е. $S_{a_i}(x_i) = \{y_i \in R^n : |y_i^k - x_i^k| \leq a_i^k,\ i = 1,...,m,\ k = 1,...,n\}$. В пространстве грузы можно перемещать только параллельным сдвигом. Обозначим ξ_i^k,

$i=1,2$, $k=1,...,n$, величины выхода грузов за границы контейнера по каждой из сторон и введем в рассмотрение вектор α с компонентами, α^k, $k=1,...,n$, которые могут принимать значения 0 или 1.Тогда формально задача об упаковке грузов в самолет сводится к следующей оптимизационной задаче:

$$\min_{x_i,\xi_i^k}\left\{\begin{array}{c} \sum_{k=1}^{n}(C_1^k\xi_1^k+C_2^k\xi_2^k)+C\sum_{k=1}^{n}\left(\frac{\sum_{i=1}^{m}m_i x_i^k}{\sum_{i=1}^{m}m_i}-x_*^k\right)^2, \\ x_i^k-a_i^k\ge A^k-\alpha^k\xi_1^k,\quad k=1,...,n,\quad i=1,...,m, \\ x_i^k+a_i^k\le B^k+\xi_2^k,\quad k=1,...,n,\quad i=1,...,m, \\ \rho_\varepsilon(S_{a_i}(x_i),S_{a_j}(x_j))=\sum_{k=1}^{n}f_\varepsilon(|x_i^k-x_j^k|-(a_i^k+a_j^k))\ge 0, \\ i<j,\quad i,j=1,..,m, \end{array}\right\} \qquad (16)$$

где C, C_i^k, $i=1,2$, $k=1,...,n$ — весовые коэффициенты. Первые две группы ограничений показывают, что грузы не должны выходить за границы контейнера, а условия не пересечения грузов задаются с помощью функции квазирасстояния $\rho_\varepsilon(x,y)=\rho_\varepsilon(S_a(x),S_b(y))$.

Алгоритм (6), (7) применим для любых $x\in R^n$. Поскольку грузы упаковываются в самолет, то $n=3$. Если начало координатной оси совместить с проекцией x_* на плоскость x^1, x^2, т.е. в (16) положить $A^n=C_1^n=\alpha^n=0$, то грузы будут укладываться на пол контейнера вокруг проекции центра тяжести. Размерности задачи (16) определяются по формулам $n_1=2n+$ $+m\times n$, $m_1=2m\times n+C_m^2$.

В качестве примера рассмотрим задачу упаковки восьми грузов на пол контейнера в самолет типа ИЛ-76. Задача имеет 30 неизвестных и 76 ограничений. Пусть размеры контейнера в данной системе координат будут $A^1=9$; $A^2=1{,}73$; $A^3=0$; $B^1=9$; $B^2=1{,}73$; $B^3=3{,}2$. Размеры грузов и значения координат их центров в начальном положении указаны в табл. 6. Координаты центра тяжести $x_*=(0;0;1{,}6)$, массы грузов m_i, $i=1,...,m$ имеют значения $m_1=m_2=3$, $m_3=m_4=m_8=2$, $m_5=1$, $m_6=m_7=2{,}5$. Весовые коэффициенты принимают значения $C_1^3=0$, $C=C_i^k=1$, $i,k=1,2$.

Таблица 6. Размеры грузов и их центры в начальном положении

Номер груза	x_0^1	x_0^2	x_0^3	a^1	a^2	a^3
1	–2	–15	–4	5	0,5	0,5
2	–6	–6	4	5	0,5	0,5
3	7	9	–5	4,5	0,5	0,25
4	5	–5	–6	4,5	0,5	0,25

Номер груза	x_0^1	x_0^2	x_0^3	a^1	a^2	a^3
5	5	6	4,5	3	0,5	0,25
6	–6	5	–3	3	0,5	0,25
7	–8	–4	–8	3	0,5	0,25
8	7	–4	6	2	0,5	0,25

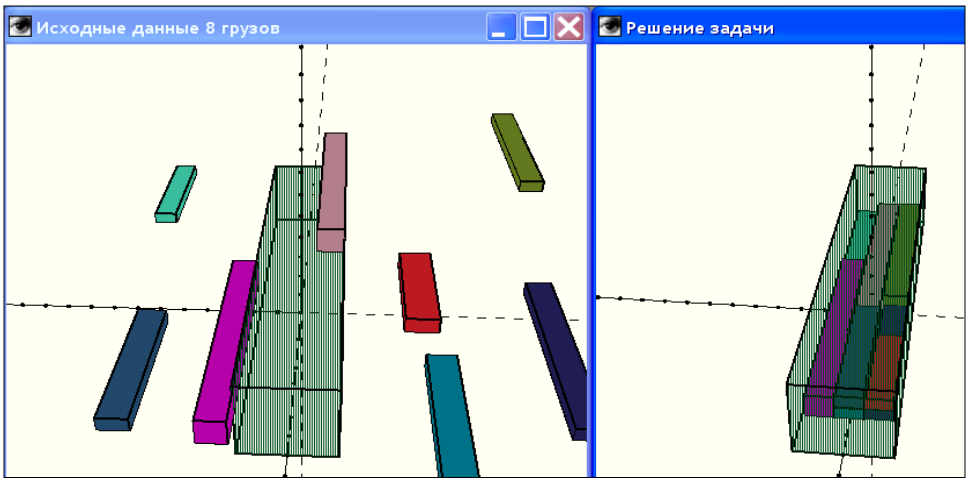

Рис. 5. Размещение 8-ми грузов в контейнер самолета

На рис. 5 дано графическое отображение положения грузов в начале (слева) и в конце (справа) итерационного процесса (6). Видно, что сначала все грузы лежат вне контейнера, а в конце — внутри. Полученные численные результаты также свидетельствуют об этом. В точках решения имеем $\xi_1^1 = -0{,}051$; $\xi_1^2 = -0{,}183$; $\xi_1^3 = 0$; $\xi_2^1 = -1{,}934$; $\xi_2^2 = -0{,}117$; $\xi_2^3 = -2{,}130$. То, что $\xi_1^3 = 0$, означает, что грузы упакованы на пол контейнера. Значения остальных ξ_i^k отрицательны, следовательно, все грузы находятся внутри контейнера. Для нормального маневра самолета смещение центра тяжести самолета x_*, соответствующее оси x^3, не является столь существенным, как смещения по осям x^1 и x^2. В нашем случае смещение по координате x^1 равно $0{,}7 \cdot 10^{-6}$, по координате x^2 — $0{,}3 \cdot 10^{-2}$, по x^3 — 0,485. Следовательно, упакованные грузы практически не изменили координат центра тяжести самолета по осям x^1, x^2. Координаты центров грузов в решении задачи приведены в табл.7. Для получения решения с точностью $\|p_k\| \le 0{,}2 \cdot 10^{-2}$ алгоритму потребовалось выполнить 109 итераций.

Таблица 7. Координаты центров грузов в точках решения

Номер груза	x_*^1	x_*^2	x_*^3
1	2,07	1,11	0,50
2	–3,95	0,05	0,50
3	2,53	–1,05	0,25
4	–4,45	–1,05	0,78

Номер груза	x_*^1	x_*^2	x_*^3
5	4,07	0,04	0,82
6	4,07	0,02	0,25
7	–5,95	1,13	0,25
8	5,07	–1,05	0,80

ВЫВОДЫ

1. На примерах задач выпуклого и обратно-выпуклого программирования показана возможность использования в негладкой оптимизации извест-

ного метода линеаризации, разработанного Б.Н. Пшеничным для решения гладких оптимизационных задач.

2. Предложен пакет программ Packing для решения задач упаковки и размещения различных объектов в конечномерном пространстве R^n, разработанный на основе формализованных математических моделей задач упаковки и модификации метода линеаризации.

3. Приведены расчеты конкретных задач упаковки и размещения объектов, которые демонстрируют эффективность работы описанной модификации метода линеаризации и разработанного на его основе пакета прикладных программ Packing.

ЛИТЕРАТУРА

1. *Пшеничный Б.Н.* Метод линеаризации. — М.: Наука, 1983. — 136 с.
2. *Пшеничный Б.Н., Ненахов Э.И., Кузьменко В.Н.* Комбинированный метод решения общей задачи выпуклого программирования // Кибернетика и системный анализ. — 1998. — № 4. — С. 121–133.
3. *Пшеничный Б.Н., Соболенко Л.А.* Метод линеаризации для обратно-выпуклого программирования // Кибернетика и системный анализ. — 1995. — № 6. — С. 86–97.
4. *Тот Л.Ф.* Расположения на плоскости, на сфере и в пространстве. — М.: Физматгиз, 1958. — 363 с.
5. *Конвей Дж., Слоэн Н.* Упаковка шаров, решетки и группы. — М.: Мир, 1990. — **1,2.** — 791 с.
6. *Стоян Ю.Г., Придатко Д.И.* Упаковка различных круговых цилиндров в параллелепипеде // Доп. НАН України. — 2004. — № 4. — С. 27–32.
7. *Pinter J.D.* Nonlinear optimization with GAMS/LGO // J. of Global Optimization. — 2007. — **38**, № 1. — P. 79–101.
8. *Guanglu Zhou, Kim-Chuan Ton, Jie Sun*. Efficient Algorithms for the Smallest Enclosing Ball Problem // Computational Optimization and Applications. — 2005. — **30**, № 2. — P. 147–160.
9. *Соболенко Л.А.* Комплекс программ Packing для решения задач упаковки и размещения объектов // Тези доп. 13-ї Міжнар. наук.-техн. конф. —Вінниця, 25–28 вересня 2006 р. — Вінниця, УНІВЕРСУМ. — 2006. — С. 362.
10. *Пшеничный Б.Н., Данилин Ю.М.* Численные методы в экстремальных задачах. — М.: Наука, 1975. — 320 с.
11. *Пшеничный Б.Н., Соболенко Л.А.* Метод обратно-выпуклого программирования и укладка параллелепипедов // Кибернетика и системный анализ. — 1996. — № 3. — С.16–26.

Поступила 08.02.2008