

ЧИСЛЕННОЕ СРАВНЕНИЕ НЬЮТОНОВСКИХ МЕТОДОВ ДЛЯ ЗАДАЧ ОПТИМИЗАЦИИ С ИСЧЕЗАЮЩИМИ ОГРАНИЧЕНИЯМИ

А. Л. Погосян

Работа посвящена конкретной реализации предложенных ранее глобализованных методов активного множества и их численному сравнению с альтернативными подходами к решению задач оптимизации с исчезающими ограничениями.

1. Введение. Задача оптимизации с исчезающими ограничениями

Будем рассматривать следующую задачу оптимизации с исчезающими ограничениями (ЗОИО)

$$\begin{aligned} f(x) \rightarrow \min, \quad h(x) = 0, \quad g(x) \leq 0, \\ H_i(x) \geq 0, \quad G_i(x)H_i(x) \leq 0, \quad i = 1, \dots, s, \end{aligned} \quad (1)$$

где $f: \mathbb{R}^n \rightarrow \mathbb{R}$ — гладкая функция, а $h: \mathbb{R}^n \rightarrow \mathbb{R}^l$, $g: \mathbb{R}^n \rightarrow \mathbb{R}^m$, $G, H: \mathbb{R}^n \rightarrow \mathbb{R}^s$ — гладкие отображения. Название данного класса задач, впервые введенного в [7], связано с тем, что если в точке $x \in \mathbb{R}^n$ для некоторого индекса $i \in \{1, \dots, s\}$ имеет место $H_i(x) = 0$, то последнее ограничение в задаче (1) выполняется автоматически, т.е. «исчезает». Если же $H_i(x) > 0$, то последнее ограничение эквивалентно неравенству $G_i(x) \leq 0$.

В работах [7, 6] обсуждается применение ЗОИО для моделирования задач оптимального дизайна топологий механических конструкций. Ограничения ЗОИО обычно оказываются нерегулярными в решении, что создает трудности при анализе и численном решении задач этого класса (см. [2, 3, 7]). Специальным условиям оптимальности, а также результатам о чувствительности и методам релаксации для ЗОИО, использующим особую структуру этих задач, посвящены работы [7, 6, 10, 11, 12, 14].

Целью данной работы является конкретизация предложенных в [13] глобализованных методов активного множества для ЗОИО и их численное сравнение между собой, а также с альтернативными подходами, разработанными и протестированными ранее в [4] и [13].

Ниже приводятся некоторые обозначения, которые используются в данной работе. Всюду далее для конечного множества I под y_I понимается подвектор вектора y с компонентами y_i , $i \in I$.

Определим *ЗОИО-функцию Лагранжа* задачи (1):

$$\mathcal{L}(x, \mu) = f(x) + \langle \mu^h, h(x) \rangle + \langle \mu^g, g(x) \rangle - \langle \mu^H, H(x) \rangle + \langle \mu^G, G(x) \rangle,$$

где $x \in \mathbb{R}^n$ и $\mu = (\mu^h, \mu^g, \mu^H, \mu^G) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s$. Также введем обычную функцию Лагранжа задачи (1):

$$L(x, \lambda) = f(x) + \langle \lambda^h, h(x) \rangle + \langle \lambda^g, g(x) \rangle - \langle \lambda^H, H(x) \rangle + \sum_{i=1}^s \lambda_i^{GH} G_i(x) H_i(x),$$

где $x \in \mathbb{R}^n$ и $\lambda = (\lambda^h, \lambda^g, \lambda^G, \lambda^{GH}) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s$.

Введем отображение $\Phi: \mathbb{R}^n \times (\mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s) \rightarrow \mathbb{R}^n \times$

$$\mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s,$$

$$\Phi(x, \lambda) = \begin{pmatrix} \frac{\partial L}{\partial x}(x, \lambda) \\ h(x) \\ \left(\rho(\lambda_j^g, -g_j(x)), j = 1, \dots, m \right) \\ \left(\rho(\lambda_i^H, H_i(x)), i = 1, \dots, s \right) \\ \left(\rho(\lambda_i^{GH}, -G_i(x)H_i(x)), i = 1, \dots, s \right) \end{pmatrix},$$

где $\rho: \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ — функция дополнительности, т.е. такая функция, для которой равенство $\rho(a, b) = 0$ выполняется тогда и только тогда, когда $a \geq 0$, $b \geq 0$, $ab = 0$. В данной работе используются две функции дополнительности: функция естественной невязки $\rho(a, b) = \min\{a, b\}$ и функция Фишера–Бурмейстера $\rho(a, b) = \sqrt{a^2 + b^2} - a - b$. Соответствующие варианты отображения Φ будем обозначать Φ_{NR} (от английского «Natural Residual») и Φ_{FB} .

Нетрудно видеть, что $\|\Phi(x, \lambda)\|$ является невязкой системы Каруша–Куна–Таккера (ККТ)

$$\begin{aligned} \frac{\partial L}{\partial x}(x, \lambda) &= 0, \quad h(x) = 0, \\ \lambda^g &\geq 0, \quad g(x) \leq 0, \quad \langle \lambda^g, g(x) \rangle = 0, \\ \lambda^H &\geq 0, \quad H(x) \geq 0, \quad \langle \lambda^H, H(x) \rangle = 0, \\ \lambda^{GH} &\geq 0, \quad G_i(x)H_i(x) \leq 0, \quad i = 1, \dots, s, \\ \sum_{i=1}^s \lambda_i^{GH} G_i(x)H_i(x) &= 0 \end{aligned} \quad (2)$$

исходной задачи (1), характеризующей ее стационарные точки и отвечающие им множители Лагранжа.

Согласно обсуждению в [13, разд. 3], любые результаты, основанные на оценках расстояния, которые используют норму отображения Φ_{NR} (Φ_{FB}) остаются верными и для нормы Φ_{FB} (соответственно, Φ_{NR}). Всюду далее вместо Φ можно использовать любую из этих невязок.

2. Глобализованные методы активного множества

Напомним, что предложенный в [13, разд. 4] гибридный подход к глобализации сходимости методов активного множества (о ньютоновских методах активного множества для ЗО-ИО см. [2, 3]) использует метод внешней фазы, который, в принципе, может быть любым методом с разумными свойствами глобальной сходимости. Его роль — получить достаточно хорошее приближение к стационарной точке задачи (1) и отвечающему ей множителю Лагранжа. После каждого шага метода внешней фазы совершается попытка переключиться на шаги метода активного множества, которые принимаются в случае выполнения теста на линейное убывание невязки системы ККТ (2) задачи (1).

В [13, алгоритм 4.1], в тесте на линейное убывание, значение невязки системы ККТ (2) на каждом шаге сравнивается с ее значением на предыдущем шаге. Если впоследствии этот тест нарушается, то происходит возврат к последней точке, сгенерированной методом внешней фазы, из которой делается очередной шаг этого метода, и так далее. Таким образом, итерации, осуществляемые методом активного множества, могут впоследствии отбрасываться, и, конечно же, общая эффективность алгоритма во многом зависит от количества таких бесполезных вычислений.

В [13, алгоритм 4.2] реализован альтернативный подход, в котором нет необходимости отбрасывать сделанные ранее шаги активного множества. В нем значение невязки системы ККТ (2) сравнивается на каждом шаге не с предыдущим значением, а с наименьшим достигнутым ее значением на предыдущих итерациях.

Определим l_1 -точный штраф $\psi: \mathbb{R}^n \rightarrow \mathbb{R}$ для ограничений

задачи (1):

$$\begin{aligned}\psi(x) = \|h(x)\|_1 + \sum_{j=1}^m \max\{0, g_j(x)\} + \sum_{i=1}^s \max\{0, -H_i(x)\} + \\ + \sum_{i=1}^s \max\{0, G_i(x)H_i(x)\}.\end{aligned}$$

Этому штрафу отвечает семейство штрафных функций вида $\varphi_\beta: \mathbb{R}^n \rightarrow \mathbb{R}$,

$$\varphi_\beta(x) = f(x) + \beta\psi(x), \quad (3)$$

где $\beta > 0$ — параметр штрафа.

Следующие алгоритмы конкретизируют [13, алгоритм 4.1] и [13, алгоритм 4.2], соответственно, в том смысле, что на шаге внешней фазы они используют метод последовательного квадратичного программирования (ПКП) для исходной ЗОИО (1) с одномерным поиском для штрафных функций вида (3), в духе традиционного способа глобализации сходимости методов ПКП [5, алгоритм 5.4.1].

Алгоритм 1. Предварительный шаг. Фиксируем параметры $\theta, q, \varepsilon, \tau \in (0, 1)$ и $\beta > 0$. Полагаем $k = 0$ и выбираем $x^0 \in \mathbb{R}^n$ и $\lambda^0 = ((\lambda^h)^0, (\lambda^g)^0, (\lambda^H)^0, (\lambda^{GH})^0) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s$.

Шаг идентификации. Определяем множества индексов

$$\begin{aligned}I_+ &= I_+(x^k, \lambda^k) = \{i = 1, \dots, s \mid H_i(x^k) > \|\Phi(x^k, \lambda^k)\|^\theta\}, \\ I_0 &= I_0(x^k, \lambda^k) = \{i = 1, \dots, s\} \setminus I_+(x^k, \lambda^k), \\ I_{+-} &= I_{+-}(x^k, \lambda^k) = \\ &= \{i \in I_+(x^k, \lambda^k) \mid G_i(x^k) < -\|\Phi(x^k, \lambda^k)\|^\theta\}, \\ I_{+0} &= I_{+0}(x^k, \lambda^k) = I_+(x^k, \lambda^k) \setminus I_{+-}(x^k, \lambda^k), \\ I_{0+} &= I_{0+}(x^k, \lambda^k) = \{i \in I_0(x^k, \lambda^k) \mid G_i(x^k) > \|\Phi(x^k, \lambda^k)\|^\theta\}, \\ I_{0-} &= I_{0-}(x^k, \lambda^k) = \\ &= \{i \in I_0(x^k, \lambda^k) \mid G_i(x^k) < -\|\Phi(x^k, \lambda^k)\|^\theta\}, \\ I_{00} &= I_{00}(x^k, \lambda^k) = I_0(x^k, \lambda^k) \setminus (I_{0+}(x^k, \lambda^k) \cup I_{0-}(x^k, \lambda^k)).\end{aligned}$$

Если $k = 0$ или вычисленные на текущем шаге множества индексов не совпадают с соответствующими множествами на предыдущем шаге, либо $I_+ \cup I_0 \neq \{1, \dots, s\}$ или $I_{+0} \cup I_{+-} \neq I_+$ или $I_{0+} \cup I_{00} \cup I_{0-} \neq I_0$, то переходим к шагу внешней фазы.

Шаг метода активного множества. Если точка (x^k, λ^k) сгенерирована шагом внешней фазы, полагаем $\tilde{k} = k$, запоминаем (x^k, λ^k) и определяем $\mu^k = ((\mu^h)^k, (\mu^g)^k, (\mu^H)^k, (\mu^G)^k)$ по следующим формулам:

$$\begin{aligned} (\mu^h)^k &= (\lambda^h)^k, \quad (\mu^g)^k = (\lambda^g)^k, \\ (\mu_i^H)^k &= (\lambda_i^H)^k - (\lambda_i^{GH})^k G_i(x^k), \quad i \in I_{0+} \cup I_{0-}, \\ (\mu_{I_{00}}^H)^k &= (\lambda_{I_{00}}^H)^k, \quad (\mu_{I_+}^H)^k = 0, \\ (\mu_i^G)^k &= (\lambda_i^{GH})^k H_i(x^k), \quad i \in I_{+0}, \quad (\mu_{I_{+-} \cup I_0}^G)^k = 0. \end{aligned} \quad (4)$$

Вычисляем $x^{k+1} \in \mathbb{R}^n$ как стационарную точку задачи квадратичного программирования

$$\begin{aligned} \langle f'(x^k), x - x^k \rangle + \frac{1}{2} \left\langle \frac{\partial^2 \mathcal{L}}{\partial x^2}(x^k, \mu^k)(x - x^k), x - x^k \right\rangle &\rightarrow \min, \\ h(x^k) + h'(x^k)(x - x^k) &= 0, \\ g(x^k) + g'(x^k)(x - x^k) &\leq 0, \\ -H_{I_{0+} \cup I_{00}}(x^k) - H'_{I_{0+} \cup I_{00}}(x^k)(x - x^k) &= 0, \\ -H_{I_{0-}}(x^k) - H'_{I_{0-}}(x^k)(x - x^k) &\leq 0, \\ G_{I_{+0} \cup I_{00}}(x^k) + G'_{I_{+0} \cup I_{00}}(x^k)(x - x^k) &\leq 0, \end{aligned} \quad (5)$$

а $((\mu^h)^{k+1}, (\mu^g)^{k+1}, (\mu_{I_{0+} \cup I_{00}}^H)^{k+1}, (\mu_{I_{0-}}^H)^{k+1}, (\mu_{I_{+0} \cup I_{00}}^G)^{k+1}) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^{|I_{0+} \cup I_{00}|} \times \mathbb{R}_+^{|I_{0-}|} \times \mathbb{R}_+^{|I_{+0} \cup I_{00}|}$ — как множитель Лагранжа, отвечающий x^{k+1} . Определяем остальные компоненты множителя $\mu^{k+1} = ((\mu^h)^{k+1}, (\mu^g)^{k+1}, (\mu^H)^{k+1}, (\mu^G)^{k+1})$, полагая их равными нулю:

$$(\mu_{I_+}^H)^{k+1} = 0, \quad (\mu_{I_{+-} \cup I_{0+} \cup I_{0-}}^G)^{k+1} = 0. \quad (6)$$

Если существует $i \in I_{0+} \cup I_{0-}$ такой, что $G_i(x^{k+1}) = 0$ или существует $i \in I_{+0}$ такой, что $H_i(x^{k+1}) = 0$, то переходим к шагу внешней фазы.

Иначе полагаем

$$\nu_i^{k+1} = \max \left\{ 0, -\frac{(\mu_i^H)^{k+1}}{G_i(x^{k+1})} \right\}, \quad i \in I_{0+} \cup I_{0-}, \quad (7)$$

и определяем $\lambda^{k+1} = ((\lambda^h)^{k+1}, (\lambda^g)^{k+1}, (\lambda^H)^{k+1}, (\lambda^{GH})^{k+1})$ следующим образом:

$$\begin{aligned} (\lambda^h)^{k+1} &= (\mu^h)^{k+1}, \quad (\lambda^g)^{k+1} = (\mu^g)^{k+1}, \\ (\lambda_{I_+}^H)^{k+1} &= 0, \quad (\lambda_{I_{00}}^H)^{k+1} = (\mu_{I_{00}}^H)^{k+1}, \\ (\lambda_i^{GH})^{k+1} &= \max\{\nu_i^{k+1}, (\lambda_i^{GH})^k\}, \quad i \in I_{0+}, \\ (\lambda_i^{GH})^{k+1} &= \begin{cases} 0, & \text{если } (\lambda_i^{GH})^k \leq 0, \\ (\lambda_i^{GH})^k, & \text{если } 0 < (\lambda_i^{GH})^k < \nu_i^{k+1}, \\ \nu_i^{k+1}, & \text{если } (\lambda_i^{GH})^k \geq \nu_i^{k+1}, \end{cases} \quad i \in I_{0-}, \\ (\lambda_i^H)^{k+1} &= (\mu_i^H)^{k+1} + (\lambda_i^{GH})^{k+1} G_i(x^{k+1}), \quad i \in I_{0+} \cup I_{0-}, \\ (\lambda_i^{GH})^{k+1} &= \frac{(\mu_i^G)^{k+1}}{H_i(x^{k+1})}, \quad i \in I_{+0}, \quad (\lambda_{I_{+-}}^{GH})^{k+1} = 0, \\ (\lambda_i^{GH})^{k+1} &= \max\{0, (\lambda_i^{GH})^k\}, \quad i \in I_{00}. \end{aligned} \quad (8)$$

Если точка (x^{k+1}, λ^{k+1}) корректно определена и удовлетворяет условию

$$\|\Phi(x^{k+1}, \lambda^{k+1})\| \leq q \|\Phi(x^k, \lambda^k)\|,$$

увеличиваем k на 1 и переходим к шагу идентификации.

Шаг внешней фазы. Если точка (x^k, λ^k) сгенерирована шагом метода активного множества, полагаем $k = \tilde{k}$ и $(x^k, \lambda^k) = (x^{\tilde{k}}, \lambda^{\tilde{k}})$.

Шаг ПКП. Выбираем симметричную положительно определенную $n \times n$ -матрицу H_k . Вычисляем $\tilde{x}^{k+1} \in \mathbb{R}^n$ как ста-

ционарную точку задачи квадратичного программирования

$$\begin{aligned} \langle f'(x^k), x - x^k \rangle + \frac{1}{2} \langle H_k(x - x^k), x - x^k \rangle &\rightarrow \min, \\ h(x^k) + h'(x^k)(x - x^k) &= 0, \\ g(x^k) + g'(x^k)(x - x^k) &\leq 0, \\ -H(x^k) - H'(x^k)(x - x^k) &\leq 0, \\ G_i(x^k)H_i(x^k) + \langle H_i(x^k)G'_i(x^k) + G_i(x^k)H'_i(x^k), x - x^k \rangle &\leq 0, \\ i &= 1, \dots, s, \end{aligned}$$

а $\lambda^{k+1} = ((\lambda^h)^{k+1}, (\lambda^g)^{k+1}, (\lambda^H)^{k+1}, (\lambda^{GH})^{k+1}) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s$ — как отвечающий $\tilde{x}^{k+1}$ множитель Лагранжа. Полагаем $p^k = \tilde{x}^{k+1} - x^k$.

Шаг одномерного поиска. Выбираем

$$\beta_k \geq \|\lambda^{k+1}\|_\infty + \bar{\beta}$$

и вычисляем

$$\Delta_k = \langle f'(x^k), p^k \rangle - \beta_k \psi(x^k).$$

Полагаем $\alpha = 1$. Если неравенство

$$\varphi_{\beta_k}(x^k + \alpha p^k) \leq \varphi_{\beta_k}(x^k) + \varepsilon \alpha \Delta_k \quad (9)$$

выполнено, полагаем $\alpha_k = \alpha$. Иначе заменяем α на $\tau\alpha$ и снова проверяем (9), и так далее, пока не будет выполнено (9).

Переход к следующей итерации. Полагаем

$$x^{k+1} = x^k + \alpha_k p^k,$$

увеличиваем k на 1 и переходим к шагу идентификации.

Алгоритм 2. Предварительный шаг. Фиксируем параметры $\theta, q, \varepsilon, \tau \in (0, 1)$ и $\bar{\beta} > 0$. Полагаем $k = 0$ и выбираем

$x^0 \in \mathbb{R}^n$ и $\lambda^0 = ((\lambda^h)^0, (\lambda^g)^0, (\lambda^H)^0, (\lambda^{GH})^0) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^s \times \mathbb{R}^s$. Полагаем

$$\sigma^{\text{rec}} = \|\Phi(x^0, \lambda^0)\|.$$

Шаг идентификации такой же, как в алгоритме 1.

Шаг метода активного множества. Если точка (x^k, λ^k) сгенерирована шагом внешней фазы, определяем множитель $\mu^k = ((\mu^h)^k, (\mu^g)^k, (\mu^H)^k, (\mu^G)^k)$ по формулам в (4).

Вычисляем $x^{k+1} \in \mathbb{R}^n$ как стационарную точку задачи квадратичного программирования (5), а $((\mu^h)^{k+1}, (\mu^g)^{k+1}, (\mu_{I_{0+} \cup I_{00}}^H)^{k+1}, (\mu_{I_{0-}}^H)^{k+1}, (\mu_{I_{+0} \cup I_{00}}^G)^{k+1}) \in \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^{|I_{0+} \cup I_{00}|} \times \mathbb{R}_+^{|I_{0-}|} \times \mathbb{R}_+^{|I_{+0} \cup I_{00}|}$ — как множитель Лагранжа, отвечающий x^{k+1} . Определяем остальные компоненты множителя $\mu^{k+1} = ((\mu^h)^{k+1}, (\mu^g)^{k+1}, (\mu^H)^{k+1}, (\mu^G)^{k+1})$, полагая, согласно (6), их равными нулю.

Если существует $i \in I_{0+} \cup I_{0-}$ такой, что $G_i(x^{k+1}) = 0$ или существует $i \in I_{+0}$ такой, что $H_i(x^{k+1}) = 0$, то переходим к шагу внешней фазы. Иначе определяем $\lambda^{k+1} = ((\lambda^h)^{k+1}, (\lambda^g)^{k+1}, (\lambda^H)^{k+1}, (\lambda^{GH})^{k+1})$ согласно (7), (8). Если точка (x^{k+1}, λ^{k+1}) корректно определена и удовлетворяет условию

$$\|\Phi(x^{k+1}, \lambda^{k+1})\| \leq q\sigma^{\text{rec}},$$

полагаем $\sigma^{\text{rec}} = \|\Phi(x^{k+1}, \lambda^{k+1})\|$, увеличиваем k на 1 и переходим к шагу идентификации.

Шаг внешней фазы.

Шаг ПКП такой же, как в алгоритме 1.

Шаг одномерного поиска такой же, как в алгоритме 1.

Переход к следующей итерации. Полагаем

$$x^{k+1} = x^k + \alpha_k p^k.$$

Если

$$\|\Phi(x^{k+1}, \lambda^{k+1})\| \leq \sigma^{\text{rec}},$$

полагаем $\sigma^{\text{rec}} = \|\Phi(x^{k+1}, \lambda^{k+1})\|$. Увеличиваем k на 1 и переходим к шагу идентификации.

Результат о глобальной сходимости алгоритма 1 в случае, когда вся траектория $\{(x^k, \lambda^k)\}$, начиная с некоторого достаточного большого k , сгенерирована на шаге метода активного множества, был получен в [13, теорема 4.1]. Аналогичный результат для алгоритма 2 в случае, когда шаг метода активного множества принимался бесконечно большое количество раз, был получен в [13, теорема 4.3]. Для алгоритма 1, помимо рассмотренного выше сценария поведения, может реализовываться лишь еще один случай, когда вся траектория сгенерирована шагом внешней фазы (так как все неудачные шаги метода активного множества отбрасываются), а для алгоритма 2 — когда все точки x^k траектории, начиная с некоторого достаточно большого индекса k , сгенерированы шагом внешней фазы. В обоих случаях методы наследуют стандартные свойства глобальной сходимости метода ПКП с одномерным поиском для штрафных функций вида (3) (см., например, [5, теорема 5.4.1]).

Результаты о квадратичной скорости сходимости алгоритмов 1 и 2 были получены соответственно в [13, теорема 4.2] и [13, теорема 4.4].

3. Численные результаты

В данном разделе приведены результаты численного сравнения алгоритмов 1 и 2 между собой и с некоторыми альтернативными методами на 23 примерах ЗОИО, взятых из всех известных автору публикаций, касающихся данного класса задач.

Для сравниваемых методов используются следующие аббревиатуры:

SQP-BFGS-AS-BS — алгоритм 1;

SQP-BFGS-AS-REC — алгоритм 2;

SQP-BFGS-AS-BFGS-BS — алгоритм 1, в котором матрицы гессе ЗОИО-функции Лагранжа $\frac{\partial^2 \mathcal{L}}{\partial x^2}(x^k, \mu^k)$ аппроксимируются посредством квазиньютоновской формулы Бroyдена–Флетчера–Голдфарба–Шанно (БФГШ) с модификацией Пауэлла (см., [15, стр. 536, 537]);

SQP-BFGS-AS-BFGS-REC — алгоритм 2 с такой же аппроксимацией $\frac{\partial^2 \mathcal{L}}{\partial x^2}(x^k, \mu^k)$, как в SQP-BFGS-AS-BFGS-BS;

SQP-BFGS — алгоритм 1 или алгоритм 2 без шага метода активного множества, т.е. обычный квазиньютоновский (БФГШ с модификацией Пауэлла) метод ПКП с одномерным поиском для l_1 -точной штрафной функции вида (3), применяемый к задаче (1);

SNM-AS — гибридный алгоритм, комбинирующий в себе метод активного множества и полугладкий метод Ньютона для уравнения $\Phi_{\text{FB}}(x, \lambda) = 0$ с одномерным поиском для функции качества $\varphi_{\text{FB}}(x, \lambda) = \frac{1}{2} \|\Phi_{\text{FB}}(x, \lambda)\|^2$;

Lifted-ssSQP-BFGS — квазиньютоновский (БФГШ с модификацией Пауэлла) полугладкий метод ПКП с одномерным поиском для l_1 -точной штрафной функции, применяемый к поднятой ЗОИО (о поднятых ЗОИО и деталях реализации данного метода см. [4]).

В алгоритмах 1 и 2 везде, кроме шага идентификации, использовалось отображение $\Phi = \Phi_{\text{FB}}$, а на шаге идентификации использовалось отображение $\Phi = \Phi_{\text{NR}}$. На шаге ПКП этих алгоритмов матрицы H_k вычислялись по формуле БФГШ с модификацией Пауэлла [15, стр. 536, 537].

Значения параметров алгоритмов 1 и 2, а также соответствующих им параметров других алгоритмов были выбраны следующим образом: $\theta = 0.5$, $q = 0.5$, $\varepsilon = 10^{-4}$, $\tau = 0.5$ и $\bar{\beta} = 1$.

На каждой итерации параметр штрафа β_k вычислялся по

формуле

$$\beta_k = \|\lambda^{k+1}\|_\infty + \bar{\beta}.$$

Такой выбор привлекателен своей простотой и хорошо работает на практике, поскольку допускает уменьшение больших значений штрафного параметра, которые могут возникать на ранних итерациях. О более сложных и практичных правилах выбора β_k см., например, [15, разд. 18.3], [8, разд. 17.1].

Все вычисления проводились в среде Matlab. Для решения квадратичных подзадач использовался встроенный в Matlab солвер quadprog. Для всех методов, кроме **Lifted-ssSQP-BFGS**, использовался критерий остановки

$$\|\Phi_{\text{FB}}(x^k, \lambda^k)\| < 10^{-6}.$$

Lifted-ssSQP-BFGS останавливался, когда аналогичная невязка системы ККТ поднятой ЗОИО становилась меньше 10^{-6} .

Запуск считался неудачным, если требуемая точность не была достигнута после 500 итераций, или если в процессе дробления параметр длины шага α в (9) становился меньше, чем 10^{-17} , или если метод оказывался неспособным выполнить очередную итерацию по той или иной причине.

Для каждого тестового примера осуществлялось 100 запусков алгоритмов из случайно сгенерированных начальных точек (одних и тех же для всех алгоритмов). Прямые начальные точки x^0 генерировались в кубе с центром в (известном для каждого теста) решении $\bar{x}$, с ребрами куба равными 20. Начальные значения двойственных переменных для всех алгоритмов генерировались аналогично прямым, но в кубе с центром в нуле, и с дополнительными ограничениями неотрицательности для компонент, отвечающих ограничениям-неравенствам.

В случае удачного запуска, сходимость к оптимальному значению целевой функции объявлялась в тех случаях, когда

модуль разности оптимального и достигнутого на последней итерации значений оказывался меньше, чем 10^{-4} .

Ниже приводятся результаты сравнения между собой методов SQP-BFGS-AS-BS, SQP-BFGS-AS-REC, SQP-BFGS-AS-BFGS-BS, SQP-BFGS-AS-BFGS-REC и SQP-BFGS с целью выявления наилучшего среди них и его дальнейшего сравнения с альтернативными алгоритмами SNM-AS и Lifted-ssSQP-BFGS.

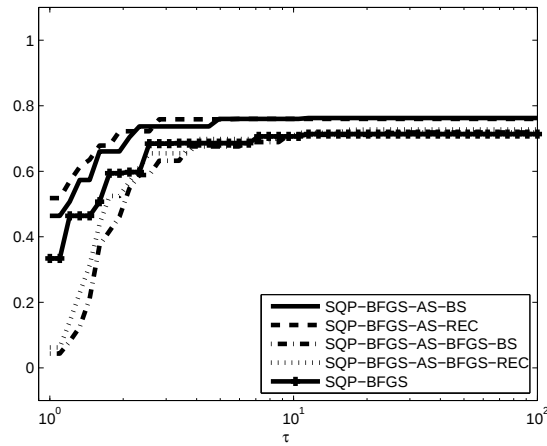


Рис. 1: Внешние итерации.

Рис. 1 дает представление об относительном среднем количестве внешних итераций на один успешный запуск, где под внешними итерациями понимаются запуски quadprog. Результаты представлены в виде так называемых «performance profiles», подобный агрегированный способ представления результатов вычислительного эксперимента был впервые предложен в [9]. Для каждого алгоритма приводится график функции, значение которой в $\tau \in [1, +\infty)$ есть отнесенная к общему количеству задач в наборе сумма долей успешных запусков для

тех задач, для которых результат (в данном случае среднее количество итераций) алгоритма был хуже (в данном случае больше) наилучшего (среди пяти алгоритмов) не более, чем в τ раз. С известной долей условности, это значение можно интерпретировать как вероятность того, что для задачи из данного набора результат запуска данного алгоритма будет хуже наилучшего не более, чем в τ раз. При этом считается, что результат неудачного запуска в бесконечное число раз хуже, чем результат любого успешного запуска. Точные формулы для функций, графики которых представлены в «performance profiles» используемого здесь типа, можно найти в [1, 13]. Значение такой функции при $\tau = 1$ можно интерпретировать как характеристику «чистой» эффективности алгоритма, т.е. вероятность того, что алгоритм покажет наилучший результат, а значение при $\tau = +\infty$ — как характеристику «чистой» робастности алгоритма, т.е. вероятность успешности запуска.

Из рис. 1 можно видеть, что по количеству внешних итераций SQP-BFGS-AS-REC несколько эффективнее остальных методов. В плане робастности лидерами являются алгоритмы SQP-BFGS-AS-REC и SQP-BFGS-AS-BS.

На рис. 2 в аналогичной форме представлены данные об относительном среднем количестве внутренних итераций на один успешный запуск, где под внутренними итерациями понимаются решения линейных систем (решение задачи линейного программирования на шаге инициализации начальной допустимой точки в quadprog тоже принималось за одну внутреннюю итерацию). Полученная картина аналогична рис. 1.

На рис. 3 представлены данные об относительном среднем количестве вычислений значений целевой функции на один успешный запуск. Здесь можно видеть, что SQP-BFGS по эффективности явно уступает другим методам. В остальном же картина аналогична рис. 1.

На рис. 4 представлены данные об относительном сред-

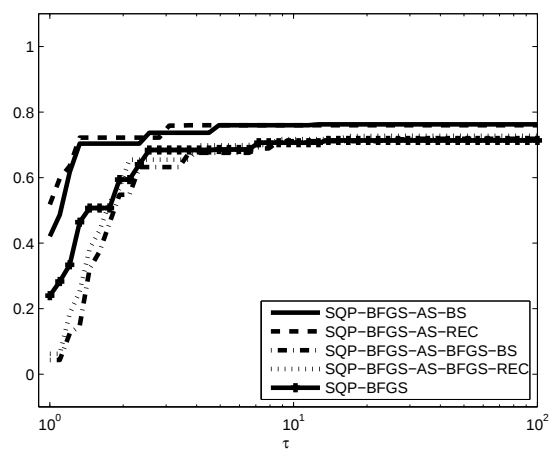


Рис. 2: Внутренние итерации.

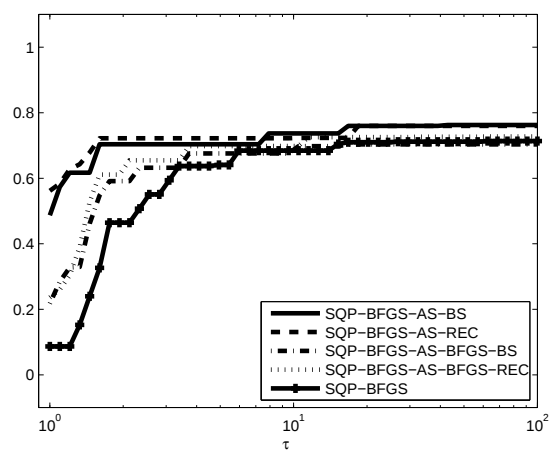


Рис. 3: Вычисления целевой функции.

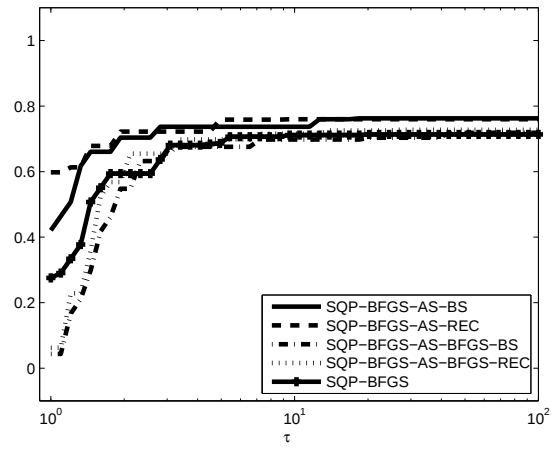


Рис. 4: Вычисления ограничений.

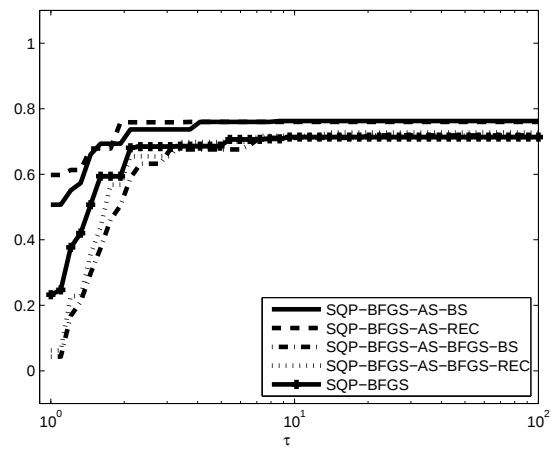


Рис. 5: Вычисления производных.

нем количестве вычислений значений функций, задающих ограничения, на один успешный запуск. Аналогично, на рис. 5 представлены данные по количеству вычислений производных целевой и ограничивающих функций. Во всех случаях полученная картина аналогична рис. 1.

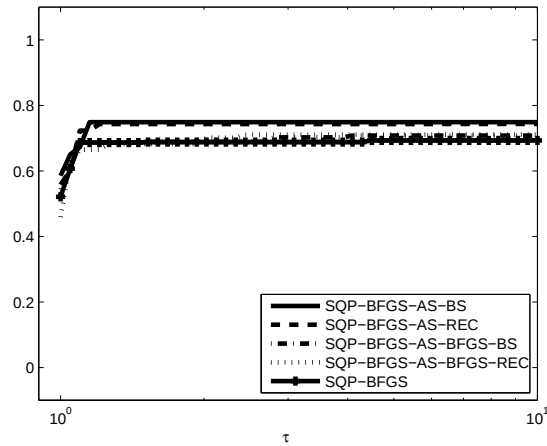


Рис. 6: Сходимость к оптимальному значению.

На рис. 6 под «результатом» алгоритма понимается обратная величина к количеству сходимостей к оптимальному значению целевой функции. Заметим, что результат здесь считается равным $+\infty$, если алгоритм ни разу не сошелся к оптимальному значению целевой функции, и такие случаи вносят дополнительный вклад в общее число неудачных запусков, несколько понижая характеристики робастности алгоритмов по сравнению с предыдущими «performance profiles». Здесь лидерами являются SQP-BFGS-AS-BS и SQP-BFGS-AS-REC.

Таким образом, на первом этапе сравнения наблюдается положительный эффект от использования фазы метода ак-

тивного множества. Однако квазиньютоновская версия этой фазы оказывается не особо полезной.

Второй этап посвящен сравнению SQP-BFGS-AS-REC, лидера первого этапа, с SNM-AS и Lifted-ssSQP-BFGS. Все приведенные ниже рисунки построены аналогично предыдущим.

Прежде всего отметим, что в SNM-AS, в отличие от остальных методов, на шаге внешней фазы решается линейная система, а не задача квадратичного программирования. Поэтому под внешними итерациями SNM-AS понимаются как запуски quadprog на шагах метода активного множества, так и решения линейных систем на шагах внешней фазы.

Из рис. 7 видно, что наиболее эффективным является алгоритм SQP-BFGS-AS-REC, однако наиболее робастным является Lifted-ssSQP-BFGS. На рис. 8 можно видеть несколько иную картину: здесь наиболее эффективным является SNM-AS. На рис. 9–11 картина аналогична рис. 7.

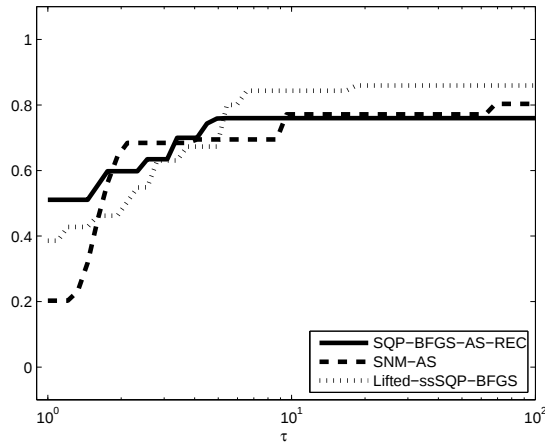


Рис. 7: Внешние итерации.

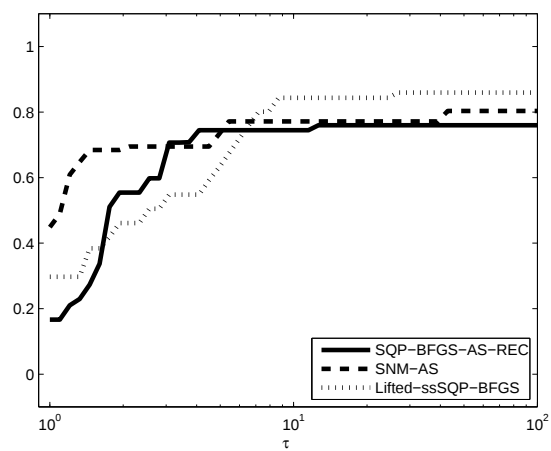


Рис. 8: Внутренние итерации.

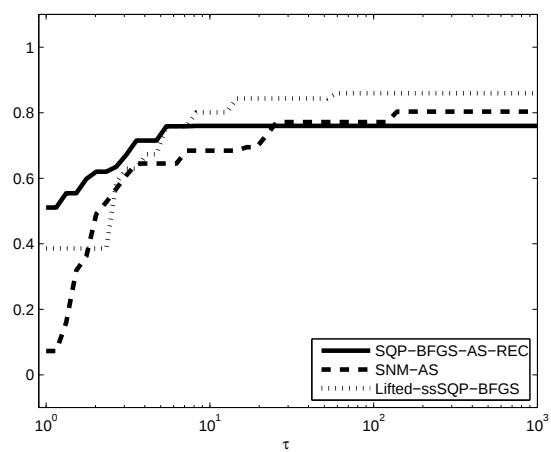


Рис. 9: Вычисления ограничений.

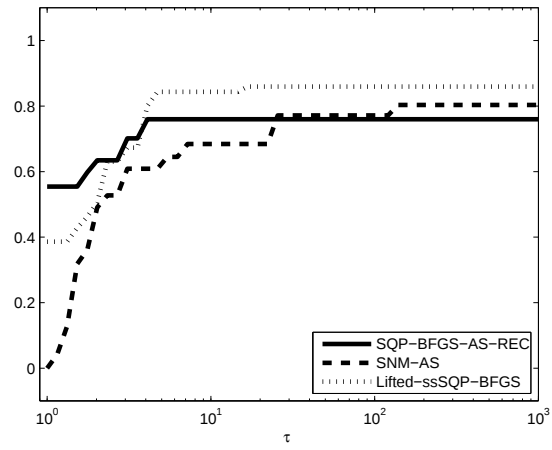


Рис. 10: Вычисления производных.

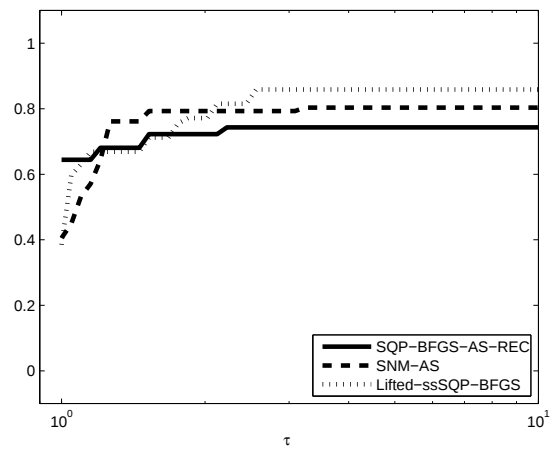


Рис. 11: Сходимость к оптимальному значению.

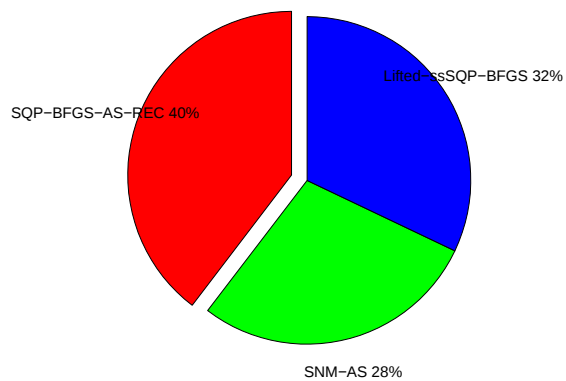


Рис. 12: Достигнуто наилучшее значение целевой функции.

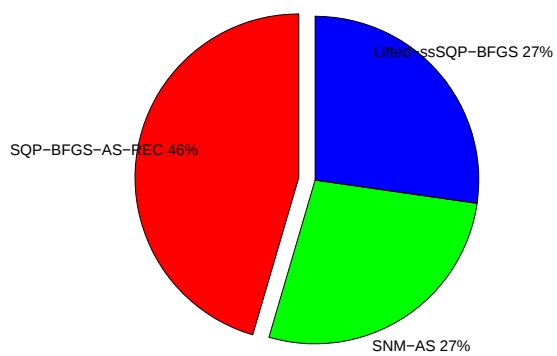


Рис. 13: Достигнуто не худшее значение целевой функции.

Круговые диаграммы на рис. 12 и 13 призваны дать некоторое представление о способности алгоритмов достигать меньших (относительно других алгоритмов) значений целевой функции в случае успешных запусков: в таких случаях полученное прямое приближение является допустимым, или почти допустимым, и поэтому достигнутое значение целевой функции может рассматриваться как еще одна разумная характеристика поведения алгоритма. Диаграммы на рис. 12 и 13 получены следующим образом. Для каждого алгоритма и каждой задачи вычисляется среднее достигнутое значение целевой функции на один успешный запуск; если это значение минимально (среди трех алгоритмов), то алгоритм относится в категорию «лучший», а если максимально, то в категорию «худший». При этом среднее достигнутое значение целевой функции алгоритма считалось равным минимальному (максимальному), если оно отличалось от минимального (соответственно, максимального) менее, чем на 10^{-4} . Заметим, что для некоторых задач алгоритм мог быть отнесен в обе указанные категории, если все три алгоритма имели одинаковые средние достигнутые значения целевой функции. После определения количества попаданий каждого алгоритма в каждую из категорий, эти числа суммируются для каждой категории, и доля каждого алгоритма в полученной сумме отображается на рис. 12 и 13. По указанным характеристикам SQP-BFGS-AS-REC ведет себя несколько лучше остальных.

Полученная неоднозначная картина свидетельствует о том, что у каждого из рассматриваемых подходов есть свои преимущества и недостатки.

Работа выполнена при финансовой поддержке РФФИ (проект 10-01-00251).

Литература

1. *Дарьина А.Н., Измаилов А.Ф.* Полугладкий метод Ньютона для задачи квадратичного программирования с простыми ограничениями // Ж. вычисл. матем. и матем. физ. 2009. Т. 49, № 10. С. 1785–1795.
2. *Измаилов А.Ф., Погосян А.Л.* Условия оптимальности и ньютоновские методы для задач оптимизации с исчезающими ограничениями // ЖВМиМФ. 2009. Т. 49, № 7. С. 1184–1196.
3. *Измаилов А.Ф., Погосян А.Л.* О методах активного множества для задач оптимизации с исчезающими ограничениями // Тематический сборник "Теоретические и прикладные задачи нелинейного анализа". М.: ВЦ РАН, 2009. С. 18–49.
4. *Измаилов А.Ф., Погосян А.Л.* Полугладкий метод последовательного квадратичного программирования для поднятых задач оптимизации с исчезающими ограничениями // ЖВМиМФ. В печати.
5. *Измаилов А.Ф., Солодов М.В.* Численные методы оптимизации. 2-е изд., перераб. и доп. М.: Физматлит, 2008.
6. *Achtziger W., Hoheisel T., Kanzow C.* A smoothing-regularization approach to mathematical programs with vanishing constraints // Preprint 284 / Institute of Mathematics, University of Würzburg. Würzburg, 2008.
7. *Achtziger W., Kanzow C.* Mathematical programs with vanishing constraints: optimality conditions and constraint qualifications // Math. Program. 2007. V. 114, № 1. P. 69–99.

8. *Bonnans J.F., Gilbert J.Ch., Lemaréchal C., Sagastizábal C.* Numerical Optimization: Theoretical and Practical Aspects. Second edition. Berlin: Springer-Verlag, 2006.
9. *Dolan E., Moré J.* Benchmarking optimization software with performance profiles. // Math. Program. 2002. V. 91, № 2. P. 201–213.
10. *Hoheisel T., Kanzow C.* First- and second-order optimality conditions for mathematical programs with vanishing constraints // Appl. of Math. 2007. V. 52. P. 495–514.
11. *Hoheisel T., Kanzow C.* Stationarity conditions for mathematical programs with vanishing constraints using weak constraint qualifications // J. Math. Anal. Appl. 2008. V. 337. P. 292–310.
12. *Hoheisel T., Kanzow C.* On the Abadie and Guignard constraint qualifications for mathematical programs with vanishing constraints // Optimization. 2009. V. 58, № 4. P. 431–448.
13. *Izmailov A.F., Pogosyan A.L.* Active-set Newton methods for mathematical programs with vanishing constraints // Comput. Optim. Appl. Submitted.
14. *Izmailov A.F., Solodov M.V.* Mathematical programs with vanishing constraints: optimality conditions, sensitivity, and a relaxation method // J. Optim. Theory Appl. 2009. V. 142, № 3. P. 501–532.
15. *Nocedal J., Wright S.J.* Numerical Optimization. Second edition. New York: Springer, 2006.