Компьютерная Математика

УДК 519.85

А.Ф. ИЗМАИЛОВ, П.И. СТЕЦЮК, А. ФИШЕР

АЛГОРИТМ EMSHOR И ЕГО OCTAVE РЕАЛИЗАЦИЯ

Исследуется применение метода эллипсоидов для построения алгоритма нахождения приближения к точке минимума выпуклой функции: гарантируется нахождение такой точки, в которой значение функции отличается от минимального не более чем на заданную величину. Алгоритм является частным случаем субградиентных методов с растяжением пространства в направлении субградиента с коэффициентом, который зависит только от размерности пространства переменных. Он может быть ис-пользован для минимизации глад-ких и негладких выпуклых функций нескольких десятков переменных.

Введение. Метод эллипсоидов (МЭ) предложили в 1976 году Д.Б. Юдин и А.С. Немировский [1]. Они исходили из схемы последовательных отсечений и назвали МЭ модифицированным методом центрированных сечений. Независимо МЭ переоткрыл Н.З. Шор в 1977 году [2]. Он представил МЭ как частный случай методов с растяжением пространства в направлении субградиента, которые были предложены в 1969 – 1970 годы [3, 4]. В 1970 году Н.З. Шор был в одном шаге от открытия МЭ. В этом легко убедиться, если сравнить доказательство теоремы о сходимости МЭ из [2] с доказательством теоремы 2 из [4]. На статью [4] иногда ошибочно ссылаются как на статью, где впервые был предложен МЭ.

Далее рассмотрим алгоритм **emshor** (**e**llipsoid **m**ethod of **Shor**) и его реализацию на языке Octave для решения задачи безусловной минимизации выпуклой функции [5]. Алгоритм является частным случаем методов с растяжением пространства в направлении субградиента, где коэффициент растяжения зависит от размерности пространства переменных и не зависит от свойств выпуклой функции. Он может быть использован для гладких и негладких выпуклых функций нескольких десятков переменных.

Структура статьи следующая. В разделе 1 описан алгоритм **emshor** и две теоремы о его сходимости. В разделе 2 дано доказательство первой теоремы. В разделе 3 описана Octave реализация алгоритма и приведены результаты вычислительных экспериментов для негладких выпуклых функций, в том числе и для овражных (функций с сильно вытянутыми поверхностями уровня).

1. Алгоритм emshor. Пусть $f(x)$ – выпуклая функция, $x \in \mathbb{R}^n$. Обозначим ее минимальное значение $f^* = f(x^*)$ и предположим, что точка минимума x^* – единственная. Субградиент $g(x)$ удовлетворяет такому условию:

$$(x - x^*, g(x)) \geq f(x) - f^* \geq 0, \ \forall x \in \mathbb{R}^n. \quad (1)$$

Здесь, и везде далее, (x, y) – скалярное произведение векторов $x \in \mathbb{R}^n$ и $y \in \mathbb{R}^n$.

Алгоритм **emshor** предназначен для нахождения такой точки x_ε^*, для которой $f(x_\varepsilon^*) - f^* \leq \varepsilon$ и $\varepsilon > 0$ – заданное число. Он имеет следующий вид.

Шаг 0. Выбираем точку $x_0 \in \mathbb{R}^n$ и радиус r_0 такими, чтобы $\|x_0 - x^*\| \leq r_0$. Кроме того, выбираем $\varepsilon > 0$, полагаем $B_0 := I_n$ и $k := 0$.

Шаг 1. Если $\|B_k^\top g(x_k)\| r_k \leq \varepsilon$, то ОСТАНОВ: $k^* := k$, $x_\varepsilon^* := x_k$.

Шаг 2. Вычисляем $x_{k+1} := x_k - h_k B_k \xi_k$, где $\xi_k := \dfrac{B_k^\top g(x_k)}{\|B_k^\top g(x_k)\|}$, $h_k := \dfrac{1}{n+1} r_k$.

Шаг 3. Обновляем $B_{k+1} := B_k + \left(\sqrt{\dfrac{n-1}{n+1}} - 1 \right) (B_k \xi_k) \xi_k^\top$ и $r_{k+1} := \dfrac{n}{\sqrt{n^2 - 1}} r_k$.

Шаг 4. Устанавливаем $k := k + 1$ и переходим к шагу 1.

На каждой итерации алгоритма обновляется матрица B_k, которая связана с заменой переменных $x = B_k y$, где $y = A_k x$ – образ точки x в преобразованном пространстве переменных, $A_k = B_k^{-1}$. Очевидно, что обновление B-матрицы (см. шаг 3) требует $O(n^2)$ операций. Это обусловлено использованием оператора растяжения пространства $R_\alpha(\xi) : \mathbb{R}^n \to \mathbb{R}^n$, который определен как

$$R_\alpha(\xi) := I_n + (\alpha - 1)\xi\xi^\top, \quad (2)$$

где $\xi \in \mathbb{R}^n$, $\|\xi\| = 1$ – направление растяжения, а $I_n \in \mathbb{R}^{n \times n}$ – единичная матрица. Свойства оператора растяжения пространства подробно изучены в [6]. Полагая $\beta := 1/\alpha$ и обозначая оператор «обратного» растяжения как $R_\alpha^{-1}(\xi)$, имеем $R_\alpha^{-1}(\xi) = R_\beta(\xi)$ и $B_{k+1} = B_k R_\beta(\xi_k)$. Последнее выражение показывает роль оператора растяжения пространства для обновления B-матриц. Для обновления A-матриц имеем соотношение

$$A_{k+1} = R_\alpha(\xi_k) A_k, \quad (3)$$

которое следует из цепочки равенств

$$A_{k+1} = B_{k+1}^{-1} = \left(B_k R_\beta(\xi_k) \right)^{-1} = R_\beta^{-1}(\xi_k) B_k^{-1} = R_{1/\beta}(\xi_k) A_k = R_\alpha(\xi_k) A_k.$$

Теорема 1. Последовательность точек $\{x_k\}_{k=0}^{k^*}$, которую генерирует алгоритм **emshor**, удовлетворяет неравенствам

$$\left\|A_k(x_k - x^*)\right\| \le r_k, \quad A_k = B_k^{-1}, \quad k = 0,1,2,\ldots,k^*. \tag{4}$$

Доказательство теоремы 1 дано далее в разделе 2.

Множество точек x, удовлетворяющих неравенству $\| A_k(x_k - x) \| \le r_k$, является эллипсоидом E_k, содержащим точку x^*. Эллипсоид E_k имеет объем

$$vol(E_k) = \frac{v_0 r_k^n}{\det A_k}, \tag{5}$$

где v_0 – объем единичного n-мерного шара, $\det A_k$ – определитель матрицы A_k. Следовательно, скорость сходимости алгоритма **emshor** будет определяться отношением объема эллипсоида E_{k+1}, локализующего x^* на $(k+1)$-й итерации, к объему эллипсоида E_k, локализующего x^* на k-й итерации.

Теорема 2. На каждой итерации k, где $1 \le k \le k^*$, отношение объемов эллипсоидов $E_k = \left\{x : \|A_k(x_k - x)\| \le r_k\right\}$ и $E_{k-1} = \left\{x : \|A_{k-1}(x_{k-1} - x)\| \le r_{k-1}\right\}$, которые локализуют точку x^*, есть величина постоянная и равная

$$q_n = \frac{vol(E_k)}{vol(E_{k-1})} = \sqrt{\frac{n-1}{n+1}}\left(\frac{n}{\sqrt{n^2-1}}\right)^n = \left(\left(\frac{n}{n+1}\right)^{n+1}\left(\frac{n}{n+1}\right)^{n-1}\right)^{1/2} < \exp\left\{-\frac{1}{2n}\right\} < 1. \tag{6}$$

Если на итерации k^* выполняется $\left\|B_k^{\top} g(x_{k^*})\right\| r_{k^*} \le \varepsilon$, то $f(x_{k^*}) - f^* \le \varepsilon$.

Доказательство. Согласно (3) имеем $A_k = R_\alpha(\xi_k)A_{k-1}$ и для определителей матриц A_k и A_{k-1} справедливо $\det A_k = \det R_\alpha(\xi_{k-1}) \det A_{k-1}$. Учитывая (5) и то, что $\det R_\alpha(\xi) = \alpha$, найдем для (6) коэффициент уменьшения объема

$$q_n = \frac{vol(E_k)}{vol(E_{k-1})} = \frac{v_0 r_k^n \det A_{k-1}}{v_0 r_{k-1}^n \det A_k} = \left(\frac{r_k}{r_{k-1}}\right)^n \frac{\det A_{k-1}}{\det R_\alpha(\xi_{k-1}) \det A_{k-1}} = \sqrt{\frac{n-1}{n+1}}\left(\frac{n}{\sqrt{n^2-1}}\right)^n.$$

Доказательство неравенства в (6) можно найти в [7], стр. 77–78.

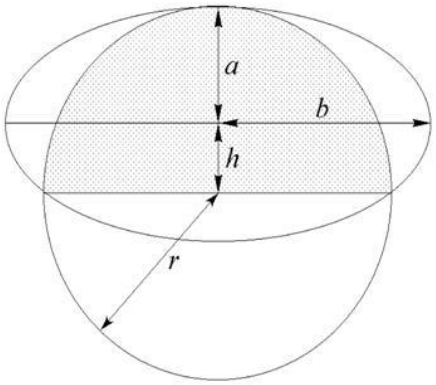
Тот факт, что условие $r_k \left\|B_k^T g(x_{k^*})\right\| \le \varepsilon$ позволяет найти точку x_{k^*}, для которой $f(x_{k^*}) - f^* \le \varepsilon$, следует из цепочки соотношений

$$r_k \ge \left\|B_k^{-1}(x_k - x^*)\right\| \ge \left(B_k^{-1}(x_k - x^*), \frac{B_k^T g(x_k)}{\left\|B_k^T g(x_k)\right\|}\right) = \frac{(x_k - x^*, g(x_k))}{\left\|B_k^T g(x_k)\right\|} \ge \frac{f(x_k) - f^*}{\left\|B_k^T g(x_k)\right\|},$$

которое, учитывая неравенство (1), выполняется для любого $x_k \in \mathbb{R}^n$. Теорема 2 доказана.

Отметим, что метод эллипсоидов базируется на использовании эллипсоида минимального объема, содержащего полушар радиуса r в $\mathbb{R}^n$ $(n \ge 2)$. Такой эллипсоид имеет сжатую форму в направлении нормали к гиперплоскости, определяющей полушар. Параметры эллипсоида показаны на рисунке,

где a – длина малой полуоси эллипсоида, b – длина больших полуосей эллипсоида (количество таких полуосей равно $n-1$), h – расстояние от центра шара до центра эллипсоида в направлении его малой полуоси.



$$a = r\frac{n}{n+1}$$

$$b = r\frac{n}{\sqrt{n^2-1}}$$

$$h = r\frac{1}{n+1}$$

РИСУНОК. Эллипсоид минимального объема, содержащий полушар в $\mathbb{R}^n$

Объем эллипсоида минимального объема равен $v_e = v_0 ab^{n-1}$, объем шара равен $v_b = v_0 r^n$, где v_0 – объем единичного шара в $\mathbb{R}^n$. Следовательно, коэффициент уменьшения объема равен

$$\frac{v_e}{v_b} = \left(\frac{a}{r}\right)\left(\frac{b}{r}\right)^{n-1} = \left(\frac{a}{b}\right)\left(\frac{b}{r}\right)^{n} = \sqrt{\frac{n-1}{n+1}}\left(\frac{n}{\sqrt{n^2-1}}\right)^{n} = q_n.$$

Чтобы преобразовать эллипсоид минимального объема в новый шар, достаточно пространство переменных растянуть в направлении малой полуоси с коэффициентом $\alpha = b/a = \sqrt{(n+1)/(n-1)}$. Это реализуется с помощью $R_\alpha(\xi)$, где направление ξ в формуле (2) совпадает с направлением малой полуоси эллипсоида.

Если $X = \mathbb{R}^n$ является исходным пространством переменных, то в преобразованном пространстве переменных $Y = R_\alpha(\xi)X$, мы получаем новый шар радиуса b, который содержит решение задачи, если его содержал шар в исходном пространстве. Повторяя эту же процедуру, но для нового шара в преобразованном пространстве получаем метод эллипсоидов. Для этого на шаге 2, в преобразованном пространстве $Y_k = B_k^{-1}X$ рассчитывается направление малой полуоси эллипсоида минимального объема, содержащего шар радиуса r_k, и выполняется переход в его центр. Вычисленное направление используется для следующего растяжения пространства, которое осуществляется на шаге 3 с помощью обновления матрицы B_{k+1}. В результате растяжения пространства получаем шар радиуса r_{k+1} в следующем преобразованном пространстве: $Y_{k+1} = B_{k+1}^{-1}X$.

2. *Доказательство теоремы 1.* Доказательство проведем по аналогии с доказательством, которое для метода эллипсоидов было предложено Н.З. Шором [2]. При доказательстве будем использовать соотношение

$$R_\alpha^T(\xi)R_\alpha(\xi) = R_{\alpha^2}(\xi), \tag{7}$$

которое следует из свойств оператора растяжения (2), см. [6], стр. 68 – 69.

Доказательство теоремы 1 проведем методом индукции по k. Для $k = 0$ неравенство (4) переходит в $\|x_0 - x^*\| \le r_0$ и выполняется в силу выбора x_0 и r_0 на шаге 0 алгоритма. Предположим, что неравенство (4) выполняется для $k = \bar{k}$. Докажем его выполнение для $k = \bar{k}+1$.

Учитывая соотношение (7) и то, что из (3) следует $A_{\bar{k}+1} = R_\alpha(\xi_{\bar{k}})A_{\bar{k}}$, имеем цепочку равенств

$$\begin{aligned}
&\| A_{\bar{k}+1}(x_{\bar{k}+1} - x^*)\|^2 = (A_{\bar{k}+1}(x_{\bar{k}+1} - x^*), A_{\bar{k}+1}(x_{\bar{k}+1} - x^*)) = \\
&= (R_\alpha(\xi_{\bar{k}})A_{\bar{k}}(x_{\bar{k}+1} - x^*), R_\alpha(\xi_{\bar{k}})A_{\bar{k}}(x_{\bar{k}+1} - x^*)) = \\
&= (A_{\bar{k}}(x_{\bar{k}+1} - x^*), R_\alpha^T(\xi_{\bar{k}})R_\alpha(\xi_{\bar{k}})A_{\bar{k}}(x_{\bar{k}+1} - x^*)) = \\
&= (A_{\bar{k}}(x_{\bar{k}+1} - x^*), R_{\alpha^2}(\xi_{\bar{k}})A_{\bar{k}}(x_{\bar{k}+1} - x^*)) = \\
&= (A_{\bar{k}}(x_{\bar{k}+1} - x^*), A_{\bar{k}}(x_{\bar{k}+1} - x^*)) + (\alpha^2 - 1)(\xi_{\bar{k}}, A_{\bar{k}}(x_{\bar{k}+1} - x^*))^2 = \\
&= \| A_{\bar{k}}(x_{\bar{k}+1} - x^*)\|^2 + (\alpha^2 - 1)(\xi_{\bar{k}}, A_{\bar{k}}(x_{\bar{k}+1} - x^*))^2,
\end{aligned}$$

которую запишем в виде соотношения

$$\| A_{\bar{k}+1}(x_{\bar{k}+1} - x^*)\|^2 = \| A_{\bar{k}}(x_{\bar{k}+1} - x^*)\|^2 + (\alpha^2 - 1)(\xi_{\bar{k}}, A_{\bar{k}}(x_{\bar{k}+1} - x^*))^2. \tag{8}$$

Далее, расшифруем оба слагаемых в правой части (8), используя соотношение

$$A_{\bar{k}}(x_{\bar{k}+1} - x^*) = A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}\xi_{\bar{k}}, \tag{9}$$

которое, учитывая, что $A_{\bar{k}} = B_{\bar{k}}^{-1}$ и очередная точка в алгоритме вычисляется по формуле из шага 2, следует из цепочки равенств

$$A_{\bar{k}}(x_{\bar{k}+1} - x^*) = A_{\bar{k}}(x_{\bar{k}} - h_{\bar{k}}B_{\bar{k}}\xi_{\bar{k}} - x^*) = A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}A_{\bar{k}}B_{\bar{k}}\xi_{\bar{k}} = A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}\xi_{\bar{k}}.$$

Первое слагаемое в правой части (8) можно записать в виде равенства

$$\| A_{\bar{k}}(x_{\bar{k}+1} - x^*)\|^2 = \| A_{\bar{k}}(x_{\bar{k}} - x^*)\|^2 - 2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}} - x^*), \xi_{\bar{k}}\right) + h_{\bar{k}}^2, \tag{10}$$

которое с учетом (9) и того, что $\| \xi_{\bar{k}} \| = 1$, следует из цепочки равенств

$$\begin{aligned}
&\| A_{\bar{k}}(x_{\bar{k}+1} - x^*)\|^2 = \| A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}\xi_{\bar{k}}\|^2 = \left(A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}\xi_{\bar{k}}, A_{\bar{k}}(x_{\bar{k}} - x^*) - h_{\bar{k}}\xi_{\bar{k}}\right) = \\
&= \left(A_{\bar{k}}(x_{\bar{k}} - x^*), A_{\bar{k}}(x_{\bar{k}} - x^*)\right) - 2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}} - x^*), \xi_{\bar{k}}\right) + h_{\bar{k}}^2(\xi_{\bar{k}}, \xi_{\bar{k}}) = \\
&= \| A_{\bar{k}}(x_{\bar{k}} - x^*)\|^2 - 2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}} - x^*), \xi_{\bar{k}}\right) + h_{\bar{k}}^2 \| \xi_{\bar{k}} \|^2 = \\
&= \| A_{\bar{k}}(x_{\bar{k}} - x^*)\|^2 - 2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}} - x^*), \xi_{\bar{k}}\right) + h_{\bar{k}}^2.
\end{aligned}$$

Учитывая соотношение (9), для квадрата скалярного произведения во втором слагаемом правой части (8) имеем следующую цепочку равенств:

$$\left(A_{\bar{k}}(x_{\bar{k}+1}-x^*),\xi_{\bar{k}}\right)^2=\left(A_{\bar{k}}(x_{\bar{k}}-x^*)-h_{\bar{k}}\xi_{\bar{k}},\xi_{\bar{k}}\right)^2=$$

$$=\left(\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)-h_{\bar{k}}(\xi_{\bar{k}},\xi_{\bar{k}})\right)^2=\left(\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)-h_{\bar{k}}\|\xi_{\bar{k}}\|^2\right)^2=$$

$$=\left(\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)-h_{\bar{k}}\right)^2=\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)^2-2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)+h_{\bar{k}}^2.$$

Следовательно, квадрат данного скалярного произведения можно записать в виде

$$\left(A_{\bar{k}}(x_{\bar{k}+1}-x^*),\xi_{\bar{k}}\right)^2=\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)^2-2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)+h_{\bar{k}}^2. \qquad (11)$$

Подставляя равенства (10) и (11) в (8) имеем

$$\|A_{\bar{k}+1}(x_{\bar{k}+1}-x^*)\|^2=\|A_{\bar{k}}(x_{\bar{k}+1}-x^*)\|^2+(\alpha^2-1)(\xi_{\bar{k}},A_{\bar{k}}(x_{\bar{k}+1}-x^*))^2=\|A_{\bar{k}}(x_{\bar{k}}-x^*)\|^2-$$

$$=2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)+h_{\bar{k}}^2+(\alpha^2-1)\left(\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)^2-2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)+h_{\bar{k}}^2\right)=$$

$$=\|A_{\bar{k}}(x_{\bar{k}}-x^*)\|^2-2\alpha^2h_{\bar{k}}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)+(\alpha^2-1)\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)^2+\alpha^2h_{\bar{k}}^2=$$

$$=\|A_{\bar{k}}(x_{\bar{k}}-x^*)\|^2-\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\left(2\alpha^2h_{\bar{k}}-(\alpha^2-1)\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\right)+\alpha^2h_{\bar{k}}^2,$$

откуда с учетом равенств $h_k=\dfrac{1}{n+1}r_k$ и $\alpha=\sqrt{\dfrac{n+1}{n-1}}$ получаем

$$\|A_{\bar{k}+1}(x_{\bar{k}+1}-x^*)\|^2=\|A_{\bar{k}}(x_{\bar{k}}-x^*)\|^2-$$
$$-\frac{2}{n-1}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\left(r_{\bar{k}}-\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\right)+\frac{1}{n^2-1}r_{\bar{k}}^2. \qquad (12)$$

Далее, для определения знака произведения

$$\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\left(r_{\bar{k}}-\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\right),$$

входящего в правую часть соотношения (12), оценим знаки обоих его сомножителей. Первый сомножитель будет неотрицательным. Так как из (1) имеем $(x-x^*,g(x))\ge 0$ для всех $x\in\mathbb{R}^n$, его легко оценить следующим образом:

$$\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)=\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\frac{B_{\bar{k}}^Tg(x_{\bar{k}})}{\|B_{\bar{k}}^Tg(x_{\bar{k}})\|}\right)=\frac{1}{\|B_{\bar{k}}^Tg(x_{\bar{k}})\|}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),B_{\bar{k}}^Tg(x_{\bar{k}})\right)=$$

$$=\frac{1}{\|B_{\bar{k}}^Tg(x_{\bar{k}})\|}\left(B_{\bar{k}}A_{\bar{k}}=(x_{\bar{k}}-x^*),g(x_{\bar{k}})\right)=\frac{1}{\|B_{\bar{k}}^Tg(x_{\bar{k}})\|}\left(x_{\bar{k}}-x^*,g(x_{\bar{k}})\right)\ge 0.$$

Учитывая, что

$$0\le\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\le\left\|A_{\bar{k}}(x_{\bar{k}}-x^*)\right\|\le r_{\bar{k}},$$

второй сомножитель оценивается следующим образом:

$$r_{\bar{k}}-\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\ge 0.$$

Из неотрицательности обоих сомножителей следует, что

$$\frac{2}{n-1}\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\left(r_{\bar{k}}-\left(A_{\bar{k}}(x_{\bar{k}}-x^*),\xi_{\bar{k}}\right)\right)\ge 0. \qquad (13)$$

Далее, учитывая (13) и то, что $\| A_{\bar{k}}(x_{\bar{k}} - x^*) \| \le r_{\bar{k}}$, соотношение (12) перепишем в виде

$$\| A_{\bar{k}+1}(x_{\bar{k}+1} - x^*) \|^2 \le \| A_{\bar{k}}(x_{\bar{k}} - x^*) \|^2 + \frac{1}{n^2-1} r_{\bar{k}}^2 \le r_{\bar{k}}^2 + \frac{1}{n^2-1} r_{\bar{k}}^2 = \left(\frac{n^2}{n^2-1} \right) r_{\bar{k}}^2,$$

откуда имеем неравенство

$$\| A_{\bar{k}+1}(x_{\bar{k}+1} - x^*) \|^2 \le r_{\bar{k}}^2 \left(\frac{n}{\sqrt{n^2-1}} \right)^2 = r_{\bar{k}+1}^2,$$

из которого следует справедливость неравенства (4) при $k = \bar{k} + 1$. Теорема 1 доказана.

3. Octave программа emshor. Алгоритм **emshor** реализован одноименной программой на языке Octave [5]. Она использует octave-функцию вида **function [f, g] = calcfg (x)**, которая вычисляет значение функции $f = f(x)$ и ее субградиент $g = g(x)$ в точке x. Эта функция готовится пользователем и может иметь произвольное имя, которое поддерживает синтаксис Octave. Код программы **emshor**, включающий краткие комментарии, приведен далее.

```
# Входные параметры:
#   calcfg - имя функции для вычисления f и g
#   x0 - стартовая точка, x0(1:n)
#   rad - радиус шара, который содержит точку минимума
#   epsf, maxitn - параметры останова (точн., макс. итер.)
#   intp - интервал печати (через каждые intp итераций)
# Выходные параметры:
#   x - найденное приближение к точке минимума, x(1:n)
#   f - значение функции f в точке x
#   itn - количество выполненных итераций
#   ist - код останова (1 = epsf, 4 = maxitn)
function [x,f,itn,ist] = emshor(calcfg,x0,rad,           #row01
                                epsf,maxitn,intp);
dn=double(length(x0)); beta=sqrt((dn-1.d0)/(dn+1.d0));   #row02
x=x0; radn=rad; B=eye(length(x));                        #row03
for (itn = 0:maxitn)                                     #row04
   [f, g1] = calcfg(x); g=B'*g1; dg=norm(g);             #row05
   if(radn*dg < epsf) ist = 1;  return; endif            #row06
   xi=(1.d0/dg)*g; dx = B * xi;                          #row07
   hs=radn/(dn+1.d0); x -= hs * dx;                      #row08
   B += (beta - 1) * B * xi * xi';                       #row09
   radn=radn/sqrt(1.d0-1.d0/dn)/sqrt(1.d0+1.d0/dn);      #row10
   if(mod(itn,intp)==0)                                  #row11
     printf("itn %4d  f %14.6e\n",itn,f);                #row12
   endif                                                 #row13
endfor                                                   #row14
ist = 4;                                                 #row15
endfunction                                              #row16
```

В программе итерационный процесс выполняется в цикле **for** (строки 04 – 14), где строки 05 – 06 выполняют шаг 1 алгоритма **emshor**, строки 07 – 08 – шаг 2, а строки 09 – 10 – шаг 3. После каждых intp итераций в цикле **for** выводятся промежуточные результаты (см. строки 11 – 13). Программа **emshor** завершается выполнением одного из двух условий: 1) найдена точка x_ε^* – такая, что $f(x_\varepsilon^*) \le f^* + \varepsilon$ (**ist=1,** см. строку 06); 2) достигается **maxitn** – максимальное количество итераций (**ist=4,** см. строки 04 и 15).

Работу программы **emshor** продемонстрируем на примерах минимизации двух выпуклых кусочно-линейных функций

$$f_1(x) = \sum_{i=1}^{n} i\left|x_i - 1\right| \quad \text{и} \quad f_2(x) = \sum_{i=1}^{n} 2^{i-1}\left|x_i - 1\right|, \tag{14}$$

для которых $f_1^* = f_1(x^*) = f_2^* = f_2(x^*) = 0$, $x^* = (1,1,\ldots,1)^T$. Здесь функция $f_2(x)$ является овражной и степень вытянутости поверхностей уровня для нее определяется отношением максимального коэфициента при $\left|x_i - 1\right|$ к минимальному. Если $n = 20$, то минимальный коэфициент равен $(2)^0 = 1$, а максимальный – $(2)^{19} \approx 5.24288\text{e+}05$.

В табл. 1 и 2 приведены результаты работы программы **emshor** для стартовой точки $x_0 = (0,0,\ldots,0)^T$ и двух значений радиусов шара, в котором локализована точка минимума, $r_0 \in \{5,\ 500\}$. Первый радиус $r_0 = 5$ выбран такого же порядка, как и расстояние от центра шара до точки минимума, а второй радиус $r_0 = 500$ выбран сильно (а именно, в сто раз) завышенным по отношению к расстоянию до точки минимума. Проведены расчеты для $n \in \{5,\ 10,\ 15,\ 20\}$ при различных значениях $\varepsilon \in \{10^{-5},\ 10^{-6},\ 10^{-10}\}$. Вычисления проводились на компьютере Pentium 2.5GHz в системе Windows XP с помощью GNU Octave версии 3.0.0.

В данных таблицах приведены следующие характеристики работы программы **emshor**: количество сделанных итераций (itn), значения функций $f_1(x_{itn})$ и $f_2(x_{itn})$ на последних итерациях, затраченное время ($time$) в секундах. Табл. 1 характеризует вычислительные затраты программы для минимизации «неовражной» функции $f_1(x)$ при двух значениях точностей – $\varepsilon = 10^{-5}$ и $\varepsilon = 10^{-10}$. В табл. 2 дано сравнение вычислительных затрат программы для минимизации «неовражной» функции $f_1(x)$ и вычислительных затрат программы для минимизации овражной функции $f_2(x)$ при одном и том же значении точности $\varepsilon = 10^{-6}$.

ТАБЛИЦА 1

	$r_0 = 5, \quad \varepsilon = 10^{-5}$			$r_0 = 5, \quad \varepsilon = 10^{-10}$		
n	*itn*	$f_1(x_{itn})$	*time*	*itn*	$f_1(x_{itn})$	*time*
5	710	1.6e – 06	0.24	1256	2.7e – 12	0.40
10	3090	8.6e – 07	0.97	5423	9.2e – 12	1.76
15	7257	1.7e – 08	2.33	12505	7.4e – 13	4.13
20	13131	5.0e – 07	4.28	22510	2.4e – 12	7.53
	$r_0 = 500, \quad \varepsilon = 10^{-5}$			$r_0 = 500, \quad \varepsilon = 10^{-10}$		
n	*itn*	$f_1(x_{itn})$	*time*	*itn*	$f_1(x_{itn})$	*time*
5	956	1.6e – 06	0.32	1530	3.6e – 12	0.48
10	4042	7.9e – 08	1.28	6293	9.0e – 12	2.00
15	9337	5.3e – 07	3.00	14561	5.6e – 12	4.71
20	16951	4.8e – 07	5.54	26085	3.0e – 12	8.57

ТАБЛИЦА 2

	$r_0 = 5, \quad \varepsilon = 10^{-6}$			$r_0 = 5, \quad \varepsilon = 10^{-6}$		
n	*itn*	$f_1(x_{itn})$	*time*	*itn*	$f_2(x_{itn})$	*time*
5	821	1.6e-07	0.27	873	1.1e-07	0.27
10	3598	8.5e – 08	1.13	3829	7.2e – 08	1.22
15	8279	2.2e – 09	2.65	9641	6.4e – 08	3.11
20	15031	4.3e – 08	4.91	18711	4.3e – 08	6.16
	$r_0 = 500, \quad \varepsilon = 10^{-6}$			$r_0 = 500, \quad \varepsilon = 10^{-6}$		
n	*itn*	$f_1(x_{itn})$	*time*	*itn*	$f_2(x_{itn})$	*time*
5	1069	1.6e – 07	0.35	1080	1.6e – 07	0.34
10	4469	8.3e – 08	1.41	4810	9.3e – 08	1.53
15	10328	2.3e – 09	3.31	11741	6.1e – 08	3.79
20	18719	4.4e – 08	6.11	22434	4.3e – 08	7.38

Из таблиц видно, что с помощью программы **emshor** можно находить достаточно точные приближения к точке минимума овражной выпуклой функции нескольких десятков переменных. Например, если $n = 20$, то для этого нужно несколько секунд на современных персональных ЭВМ. При этом количество итераций слабо зависит от точности ε и r_0 – радиуса шара локализации точки минимума. Кроме того, скорость сходимости алгоритма практически не зависит от овражности (степени вытянутости поверхностей уровня) выпуклой функции.

Выводы. В работе описан алгоритм **emshor** (**e**llipsoid **m**ethod of **Shor**) и его реализация на языке Octave для решения задачи безусловной минимизации выпуклой функции. Алгоритм находит такое приближение к точке минимума,

для которого значение функции отличается от минимального не более, чем на заданную величину. Алгоритм построен на основе метода эллипсоидов и является частным случаем методов с растяжением пространства в направлении субградиента, где коэффициент растяжения зависит только от размерности пространства переменных и не зависит от свойств выпуклой функции.

Octave реализация алгоритма **emshor** может быть использована для минимизации гладких и негладких выпуклых функций нескольких десятков переменных. Она позволяет находить достаточно точные приближения к точке минимума овражной выпуклой функции, и на это для функций двадцати переменных требуется несколько секунд на современных ПЭВМ. Алгоритм можно использовать для очень точного решения негладких задач при реализации методов декомпозиции для блочных задач линейного программирования с малым количеством связывающих переменных или связывающих ограничений.

Следует отметить, что алгоритм **emshor** можно использовать для нахождения одной из точек множества X^* – множества минимумов выпуклой функции $f(x)$. Для этого достаточно начальный радиус r_0 выбрать таким, чтобы в шаре радиуса r_0 с центром в точке x_0 находилась хотя бы одна из точек, принадлежащих множеству X^*. Если гарантировать, что в шаре содержится множество X^* целиком, то тогда полученный на каждой итерации алгоритма **emshor** эллипсоид будет локализовать множество X^* целиком. Это означает, что аппроксимационный эллипсоид, полученный на последней итерации алгоритма, можно эффективно использовать для постоптимального анализа граничных или других точек множества X^*. При этом потребуется меньшее количество дополнительных отсечений, так как в расчет принимается вся полученная в ходе итерационного процесса информация о локализации множества X^*; она содержится в построенном на последней итерации аппроксимационном эллипсоиде.

Работа выполнена при финансовой поддержке Volkswagen Foundation (грант No 90 306), грантов РФФИ 17-01-00125 и 19-51-12003 ННИО_а (А.Ф. Измаилов), гранта НАН Украины 0118U005227 (П.И. Стецюк).

О.Ф. Ізмаілов, П.І. Стецюк, А. Фішер

АЛГОРИТМ EMSHOR ТА ЙОГО OCTAVE РЕАЛІЗАЦІЯ

Досліджується застосування методу еліпсоїдів для побудови алгоритму знаходження наближення до точки мінімуму опуклої функції: гарантується знаходження такої точки, в якій значення функції відрізняється від мінімального не більше, ніж на задану величину. Алгоритм є окремим випадком субградієнтних методів з розтягом простору в напрямку субградієнта з коефіцієнтом, який залежить тільки від розмірності простору змінних. Він може бути використаний для мінімізації гладких і негладких опуклих функцій декількох десятків змінних.

A.F. Izmailov, P.I. Stetsyuk, A. Fischer

EMSHOR ALGORITM AND ITS OCTAVE IMPLEMENTATION

The application of the ellipsoid method for constructing an algorithm for finding an approximation to a minimum point of a convex function is investigated: the algorithm guarantees finding such a point at which the value of the function differs from the minimum by no more than a specified value. The algorithm is a special case of subgradient methods with space dilation in the direction of the subgradient with a coefficient that depends only on the dimension of the space of variables. It can be used to minimize smooth and non-smooth convex functions of several tens of variables.

Список литературы

1. Юдин Д.Б., Немировский А.С. Информационная сложность и эффективные методы решения выпуклых экстремальных задач. *Экономика и математические методы*. 1976. Вып. 2. С. 357 – 369.
2. Шор Н.З. Метод отсечения с растяжением пространства для решения задач выпуклого программирования. *Кибернетика*. 1977. № 1. С. 94 – 95.
3. Шор Н.З., Билецкий В.И. Метод растяжения пространства для ускорения сходимости в задачах овражного типа. Тр. семинара Науч. совета АН УССР по кибернетике *Теория оптимальных решений.* Киев. 1969. № 2. С. 3 – 18.
4. *Шор Н.З.* Использование операции растяжения пространства в задачах минимизации выпуклых функций. *Кибернетика*. 1970. № 1. С. 6 – 12.
5. Стецюк П.І., Івлічев А.В. Метод еліпсоїдів та Octave-програма emshor. Міжнародний науковий симпозіум «І*нтелектуальні рішення*». Теорія прийняття рішень: праці Міжнар. школи-семінару, 15–20 квітня 2019 р. У.: УНУ, 2019. С. 119 – 120.
6. Шор Н.З. Методы минимизации недифференцируемых функций и их приложения. Киев: Наукова думка, 1979. 200 с.
7. Grötschel M., Lovász L., Schrijver A. Geometric algorithms and combinatorial optimization. Berlin: Springer-Verlag, 1988. 362 p.

Получено 18.04.2019

Об авторах:

Измаилов Алексей Феридович,
доктор физико-математических наук, профессор
Московского государственного университета им. М.В. Ломоносова,
E-mail: izmaf@cs.msu.ru

Стецюк Петр Иванович,
доктор физико-математических наук, заведующий отделом
Института кибернетики имени В.М. Глушкова НАН Украины,
E-mail: stetsyukp@gmail.com

Фишер Андреас,
профессор, директор Института вычислительной математики
технического университета Дрездена.
E-mail: Andreas.Fischer@tu-dresden.de