ТЕОРІЯ ОПТИМАЛЬНИХ РІШЕНЬ

УДК 519.8

Н.Г. ЖУРБЕНКО

АЛГОРИТМ ПОЛЯКА НА ОСНОВЕ АГРЕГАТНЫХ ε-СУБГРАДИЕНТОВ*

Предложена модификация субградиентного алгоритма Б.Т. Поляка. Алгоритм основан на использовании ε-субградиентов. Результаты численных расчетов показывают улучшение скорости сходимости алгоритма.

Введение. Субградиентный алгоритм Б.Т. Поляка [1] предназначен для решения задачи минимизации ограниченной снизу выпуклой функции $f(x)$ в n-мерном евклидовом пространстве R^n: $f^* = \min\{ f(x) | x \in R^n \}$. При этом предполагается, что оптимальное значение f^* известно и множество точек минимума $X^* \neq \varnothing$. Алгоритм определяется такой итеративной процедурой:

$$x_{k+1} = x_k - h_k g_k / |g_k|, \qquad (1)$$

где $h_k = \gamma(f(x_k) - f^*)/|g_k|$, $g_k \in \partial f(x_k)$ – множество субградиентов [2] в точке x_k, $0 < \gamma < 2$. Алгоритм Поляка реализует теоретическую оценку скорости сходимости $O\left(1/\sqrt{k}\right)$ (для алгоритмов с равномерной по размерности пространства переменных скоростью). Алгоритм Поляка, также как классический субградиентный алгоритм Н.З. Шора [2], чрезвычайно прост по численной реализации и до настоящего времени является эффективным средством для решения задач больших размерностей.

Хорошо известно [2], что малая скорость сходимости этих алгоритмов ярко проявляется для «овражных» функций (функций с сильно вытянутыми линиями уровня). В данной работе делается попытка модификации алгоритма Поляка для повышения его скорости сходимости для овражных функций. Алгоритм будет основан

*При поддержке Национальной академии наук Украины (тема 0114U001055).

на использовании ε-субградиентов, его скорости сходимости для овражных функций. Алгоритм будет основан на использовании ε-субградиентов.

1. Агрегатный ε-субградиент. (Содержание данного пункта соответствует работе [3]).

Для заданного числа $\bar{\varepsilon} \geq 0$ введем $\bar{\varepsilon}$-оптимальное множество $X^*(\bar{\varepsilon})$:

$$X^*(\bar{\varepsilon}) = \{x \in R^n \mid f(x) \leq f^* + \bar{\varepsilon}.$$

Произвольную точку $\tilde{x} \in X^*(\bar{\varepsilon})$ и значение функции $\tilde{f} = f(\tilde{x})$ будем называть решением задачи $\bar{\varepsilon}$-оптимизации ($\bar{\varepsilon}$-решением).

Мы будем использовать несколько отличное от классического определение ε-субградиента [4]. Вектор $g \in R^n$ называется $(\varepsilon, \tilde{f})$-субградиентом функции $f(x)$ в точке z, если для $\forall x$ выполняется неравенство:

$$f(x) \geq \tilde{f} + (g, x - z) - \varepsilon, \tag{2}$$

где $f(z) \geq \tilde{f} \geq f^*$; $\varepsilon \in R^1$. Значение $\tilde{f}$ на итерации алгоритма решения задачи $\bar{\varepsilon}$-оптимизации обычно равно полученному рекордному значению функции $f(x)$. Если значение f^* известно априори, то $\tilde{f} = f^*$.

Обычное классическое определение ε-субградиента соответствует определению $(\varepsilon, \tilde{f})$-субградиента (2) при $\tilde{f} = f(z)$; $\varepsilon \geq 0$. Обобщенный градиент функции $f(x)$ в точке z в классическом смысле [2] является $(0, f(z))$-субградиентом.

$G(\varepsilon, \tilde{f}, z)$, $\partial f(z)$ обозначим множество $(\varepsilon, \tilde{f})$-субградиентов и множество обобщенных градиентов функции $f(x)$ в точке z соответственно. В дальнейшем предполагается, что имеются алгоритмы вычисления $f(x)$ и $g(x) \in \partial f(x)$ в произвольной точке x.

Пусть $g \in G(\varepsilon_1, \tilde{f}_1, z_1)$; $f^* \leq f_2 \leq f(z_2)$, тогда $g \in G(\varepsilon_2, \tilde{f}_2, z_2)$, где

$$\varepsilon_2 = \varepsilon_1 + \tilde{f}_2 - \tilde{f}_1 - (g, z_2 - z_1). \tag{3}$$

Таким образом, $(\varepsilon, \tilde{f})$-субградиент в некоторой точке z_1 является $(\varepsilon, \tilde{f})$-субградиентом в любой другой точке z_2. Формула (3) определяет правило пересчета параметров $(\varepsilon, \tilde{f})$-субградиента.

Пусть $P(z, \eta) = \{x \in R^n \mid (x - z, \eta) = 0\}$ плоскость, проходящая через точку z с нормалью $\eta \in R^n$, $|\eta| > 0$. $P^+(z, \eta) = \{x \in R^n \mid (x - z, \eta) \geq 0\}$ – полупространство, определяемое (секущей) плоскостью $P(z, \eta)$. Следующее утверждение соответствует обычному построению отсекающих плоскостей «со сдвигом»

для множества $\tilde{X}(\bar{\varepsilon}, \tilde{f})$. Пусть $g \in G(\varepsilon, \tilde{f}, z)$; $h = (\bar{\varepsilon} - \varepsilon)/|g|$; $\tilde{z} = z - hg/|g|$. Легко показать, что тогда $\tilde{X}(\bar{\varepsilon}, \tilde{f}) \subset P^+(x - \tilde{z}, -g)$.

Замечание. Если $\tilde{f} = f^*$ (задача с известным значением минимума), то субградиент $g \in \partial f(z)$ является (ε, f^*)-субградиент с $\varepsilon = -(f(z) - f^*)$. Для случая обычной задачи минимизации ($\bar{\varepsilon} = 0$, $X^* = \tilde{X}(\bar{\varepsilon}, \tilde{f})$) величина «сдвига» $h = (\bar{\varepsilon} - \varepsilon)/|g| = (f(z) - f^*)/|g|$ в точности соответствует шаговому множителю метода Поляка (1).

Легко видеть, что $P^+(x - \tilde{z}, -g)$ определяется неравенством: $(x - z, g) \leq$ $\leq -(\bar{\varepsilon} - \varepsilon)$, а вектор «сдвига» $\tilde{z} - z$ отсекающей плоскости можно определить как результат решения следующей задачи $\min\{1/2 |y|^2 : (y, g) \leq -\tilde{\varepsilon}\}$, где $\tilde{\varepsilon} = \bar{\varepsilon} - \varepsilon$. Обобщим этот результат для множества субградиентов. Пусть $g_i \in G(\varepsilon_i, \tilde{f}, z), i = 1, \ldots, m, \tilde{\varepsilon}_i = \bar{\varepsilon} - \varepsilon_i$. Тогда $\tilde{X}(\bar{\varepsilon}, \tilde{f})$ содержится в пересечении подпространств: $(x - z, g_i) \leq -\tilde{\varepsilon}_i$, (не исключается, что это пересечение пусто). Вектор сдвига y для множества ε-субградиентов $G(\varepsilon_i, \tilde{f}, z)$ из точки z определим решением следующей задачи:

$$\min 1/2 \| y \|^2, \tag{4}$$

$$(y, g_i) \leq -\tilde{\varepsilon}_i; i = 1, \ldots, m. \tag{5}$$

Если система (5) несовместна, то вектор y не определен.

Утверждение 1. Пусть система ограничений (5) – несовместна. Тогда $\tilde{f}$ – решение задачи $\bar{\varepsilon}$-оптимизации и выпуклая оболочка множества ε-субградиентов $G(\varepsilon_i, \tilde{f}, z)$ содержит 0.

Утверждение 2.

Пусть:

a) система (5) совместна;

b) y, λ – оптимальные значения прямых и двойственных переменных задачи (4 – 5);

c) $g = \sum_{i=1,m} \lambda_i g_i / \sum_{i=1,m} \lambda_i$; $\varepsilon = \sum_{i=1,m} \lambda_i \varepsilon_i / \sum_{i=1,m} \lambda_i$.

Тогда: $g \in G(\varepsilon, \tilde{f}, z)$, $y = -((\bar{\varepsilon} - \varepsilon)/|g|)g/|g|$. Если условие a) не выполнено, то, учитывая утверждение 1, положим $g = 0$.

Вектор g, определяемый в утверждении 2, будем называть агрегатным ε-субградиентом множества субградиентов $G(\varepsilon_i, \tilde{f}, z)$.

Геометрический смысл агрегатного ε-субградиента: агрегатный ε-субградиент принадлежит выпуклой оболочке множества субградиентов $G(\varepsilon_i, \tilde{f}, z)$ и обеспечивает максимальный сдвиг отсекающей плоскости.

Утверждение 3. Пусть для всех субградиентов $\varepsilon_i = \varepsilon < \bar{\varepsilon},\ i = 1,\ldots,m.$ Тогда агрегатный ε-субградиент – это вектор наименьшей длины выпуклой оболочки множества субградиентов ($g = Nr\{g_1,\ldots,g_m\}$).

Таким образом, для множества ε-субградиентов $G(\varepsilon_i, \tilde{f}, z)$ с одинаковыми значениями ε_i агрегатный ε-субградиент совпадает с кратчайшим вектором выпуклой оболочки $G(\varepsilon_i, \tilde{f}, z).$ Именно этот вектор обычно используется при построении ε-субградиентных методов оптимизации. Введенное выше определение агрегатного ε-субградиента полезно в следующих смыслах. Во-первых, при решении задачи (3 – 4) может оказаться, что некоторое значение $\lambda_i > 0$ даже если соответствующее значение $\varepsilon_i > \bar{\varepsilon}.$ Обычно такие ε-субградиенты при решении задачи $\bar{\varepsilon}$-оптимизации не используются. Во-вторых, при использовании $g = Nr\{g_1,\ldots,g_m\}$ все ε-субградиенты при выборе направления сдвига из точки z оказываются «равноправными» по отношению к значениям ε_i: вектор $g = Nr\{g_1,\ldots,g_m\}$ не зависит от ε-параметров. Однако значения ε-параметров существенны для локализации $\bar{\varepsilon}$-решения.

Использование введенного агрегатного ε-субградиента позволяет получать модификации ε-субградиентных методов минимизации по обычной схеме их построения. В следующих пунктах агрегатный ε-субградиент используется для построения модификации алгоритма Поляка.

Отметим, что приведенное выше определение $(\varepsilon, \tilde{f})$-субградиента можно рассматривать как удобный (по мнению автора) способ отслеживания локализации множества $X^*(\bar{\varepsilon})$ с учетом накопленного множества отсекающих плоскостей.

2. Численная схема алгоритма Поляка на основе агрегатного ε-субградиента. Поскольку для рассматриваемой задачи оптимальное значение f^* задано, то в дальнейшем значение параметра $\tilde{f}$ для $(\varepsilon, \tilde{f})$-субградиентов полагаем равным f^*: $\tilde{f} = f^*.$ Кроме того, рассматривается обычная задача минимизации: $\bar{\varepsilon} = 0.$ Так как субградиент $g \in \partial f(z)$ является (ε, f^*)-субградиентом с $\varepsilon = -(f(z) - f^*),$ то $h = (f(z) - f^*)/|g|.$ Величина сдвига отсекающей плоскости для такого субградиента (как отмечалось выше) соответствует значению шага алгоритма Поляка (1) при $\gamma = 1.$

По поводу параметра γ отметим следующее. В алгоритме Поляка выбор значения $\gamma \in (0,2)$ для произвольной выпуклой функции обеспечивает «фейеровское» свойство алгоритма относительно $|x_k - x^*|$: $|x_{k+1} - x^*| < |x_k - x^*|$, для $\forall x^* \in X^*$. Наше определение (ε, f^*)-субградиента фактически основано на следующем требовании: ни одна точка множества из X^* не должна отсекаться: $X^* \subset P^+(x - \tilde{z}, -g)$. Для $\gamma = 1$ это обеспечивается для произвольной выпуклой функции. Но для конкретной функции это требование может выполняться и для больших значений γ. Например, для квадратичной функции указанное условие выполняется для $\gamma = 2$. Нетрудно видеть, что учет значения параметра γ можно осуществить следующим образом: субградиент $g \in \partial f(z)$ является (ε, f^*)-субградиент с $\varepsilon = -\gamma(f(z) - f^*)$.

При описании численной схемы алгоритма будет использоваться множество $SetGrad = \{Grad[1], Grad[2], \ldots Grad[size]\}$. Здесь: $size$ – число элементов множества $SetGrad$; $Grad[l]$ – элемент l множества $SetGrad$, определяющий информацию о ε-субградиенте: вектор g, значение параметра ε, номер итерации алгоритма k, на которой он был вычислен. Для обозначения этих компонент элемента $Grad[l]$ будет использоваться следующая форма (в стиле указателей языка C++): $SetGrad[l] -> g$, $SetGrad[l] -> \varepsilon$, $SetGrad[l] -> k$. В приведенной далее численной схеме алгоритма, элемент $Grad[1]$ будет содержать информацию об агрегатном субградиенте.

Вычислительная схема рассматриваемых алгоритмов состоит в следующем.

0-й шаг алгоритма.

Задаем значения параметров алгоритма: $m \geq 2$ и γ. Значение параметра m определяет максимальное число ранее вычисленных (ε, f^*)-субградиентов (включая агрегатный субградиент), которые используются на итерации алгоритма.

Выбираем начальное приближение x_0. Вычисляем:

1) $g(x_0) \in \partial f(x_0)$ (субградиент в точке x_0); $\varepsilon(x_0) = -\gamma(f(x_0) - f^*)$;

2) $SetGrad[1] -> g = g(x_0)$, $SetGrad[1] -> \varepsilon = \varepsilon(x_0)$,
$SetGrad[1] -> k = 0$,
$size = 1$.

Пусть на шаге k алгоритма $(k = 0,1,2,\ldots)$ получены: x_k, $SetGrad$, $size$.

$(k+1)$-ая итерация алгоритма $(k = 0,1,2,\ldots)$.

1) $g = SetGrad[1] -> g$; $\varepsilon = SetGrad[1] -> \varepsilon$; ($g$ – агрегатный субградиент, ε – значение его ε-параметра);

2) $h_{k+1} = -\varepsilon / |g|$;

3) $x_{k+1} = x_k - h_{k+1} g / |g|$;

4) $g(x_{k+1}) \in \partial f(x_{k+1})$ (субградиент в точке x_{k+1}); $\varepsilon(x_{k+1}) = -\gamma(f(x_{k+1}) - f^*)$;

5) Пересчет ε-параметров субградиентов множества *SetGrad*;
(пересчет ведется в соответствии с формулой (3): $\varepsilon := \varepsilon - (g, x_{k+1} - x_k)$);

6) если $size < m$, то { $size := size + 1$; $SetGrad[size] -> g = g(x_{k+1})$;
$SetGrad[size] -> \varepsilon = \varepsilon(x_{k+1})$; $SetGrad[size] -> k = k + 1$; *goto* 7 };

6') если $size = m$, то {определить номер $kDelete$, $kDelete > 1$ удаляемого субградиента из множества *SetGrad*; $SetGrad[kDelete] -> g = g(x_{k+1})$;
$SetGrad[kDelete] -> \varepsilon = \varepsilon(x_{k+1})$; $SetGrad[kDelete] -> k = k + 1$; *goto* 7;
(ввод нового субградиента на место удаленного);

7) вычислить агрегатный субградиент g_{agr} множества *SetGrad* и значение его параметра ε_{agr}. Положить $SetGrad[1] -> g = g_{agr}$;
$SetGrad[1] -> \varepsilon = \varepsilon_{agr}$; $SetGrad[1] -> k = k + 1$;
(вычисление агрегатного субградиента основано на решении задачи (4 – 5)).

Переходим к $(k+2)$-у шагу алгоритма или прекращаем работу при выполнении критерия останова.

Приведенная численная схема определяет конкретный алгоритм при уточнении процедуры отсева субградиентного множества *SetGrad* (п. 6')).

Свойства агрегатного алгоритма обеспечивают справедливость следующего утверждения, в точности соответствующего работе [1].

Теорема 1. Пусть множество точек минимума $X^* \neq \varnothing$. Тогда $x_k \to x^* \in X^*$; $\underline{\lim} \sqrt{k}(f(x_k) - f^*) = 0$.

3. *Результаты решения тестовых задач.*

Приведем результаты численных исследований следующего варианта процедуры отсева субградиентного множества *SetGrad*. Из множества накопленных субградиентов удаляется наиболее «старый»: $kDelete = \arg\min\{SetGrad[i] -> k \mid i = 2,3,\ldots m;\}$.

В качестве тестовых задач рассматривались задачи минимизации двух следующих функций:

$$f_1(x) = \sum_{i=1,n} \rho_n^{i-1} |x_i|; \quad f_2(x) = \sum_{i=1,n} \rho_n^{i-1} x_i^2,$$

где параметр ρ_n выбирался в зависимости от размерности задачи n по формуле $\rho_n = 10^{3/(n-1)}$. Таким образом, степень вытянутости линий уровня («овражности») функций определяется значением параметра $\rho_n^{n-1} = 10^3$, она одинакова для всех функций независимо от числа переменных. Начальная точка

$x_i = 1.0,\ i = 1,2,...n.$ Критерий останова: $f_k \le 10^{-6}$, где f_k – значение функции на итерации останова k.

Результаты решения тестовых задач минимизации функций $f_1(x)$, $f_2(x)$ приведены в табл. 1 и 2 соответственно, где приняты следующие обозначения: n – размерность пространства переменных; m – максимальное число ранее вычисленных субградиентов, которые используются на итерации алгоритма. В таблицах указываются номера итерации, на которых алгоритм прекратил работу. Данные для $m = 1$ соответствуют алгоритму Поляка. Прочерк в колонке означает, что заданная точность решения не достигнута за 10000 итераций.

ТАБЛИЦА 1

$n\backslash m$	1	2	20	40	60	80	100	120
2	–	2	2	2	2	2	2	2
10	–	937	22	22	22	22	22	22
100	–	11129	264	268	257	265	287	298

ТАБЛИЦА 2

$n\backslash m$	1	2	20	40	60	80	100	120
2	21	2	2	2	2	2	2	2
10	3413	13	10	10	10	10	10	10
100	3570	113	115	100	87	69	69	69

Результаты решения тестовых задач показывают существенное повышение скорости сходимости алгоритма при минимизации овражных функций. Однако следует учитывать, что трудоемкость итерации алгоритма увеличивается с ростом параметра алгоритма m (число используемых на итерации ранее вычисленных субградиентов). Для $m = 2$ трудоемкость итерации алгоритма фактически такая же, как в алгоритма Б.Т. Поляка: вычисление агрегатного субградиента выполняется по простым аналитическим соотношениям. Поэтому для задач большой размерности целесообразно использование, данного простейшего варианта алгоритма.

В программной реализации алгоритма вычисление агрегатного ε-субградиента основывается на решении двойственной к (4 – 5) задаче. При этом используется специальные меры по ее нормировке. Это связано с тем, что точность решения двойственной задачи необходимо увеличивать по мере приближения к точке минимума (по мере уменьшения шага h_{k+1}).

Заключение. В работе приведен простейший вариант рассматриваемого семейства алгоритмов. В этом варианте используется только субградиенты, вычисленные в точках, соответствующих шагу алгоритма Поляка. Нетрудно

привести примеры функций (даже в одномерном случае), для которых приведенный алгоритм в точности совпадет с алгоритмом Б.Т. Поляка. Фактически приведенный вариант алгоритма отличается от алгоритма Б.Т. Поляка [1] только тем, что множество *SetGrad* содержит агрегатные $\varepsilon-$субградиенты. Однако можно использовать более сложные процедуры генерации множества ε-субградиентов *SetGrad*. Такие процедуры могут, например, основываться на алгоритме одномерной оптимизации (аналогично работе [3]). Кроме того, можно использовать более сложные процедуры удаления «устаревших» субградиентов.

М.Г. Журбенко

АЛГОРИТМ ПОЛЯКА НА ОСНОВІ АГРЕГАТНИХ ε-СУБГРАДІЄНТІВ

Запропоновано модифікацію субградієнтного алгоритму Б.Т. Поляка. Алгоритм заснований на використанні ε-субградієнтів. Результати чисельних розрахунків показують поліпшення швидкості збіжності алгоритму.

N.G. Zhurbenko

POLJAK'S ALGORITHM ON BASE OF AGGREGATE ε-SUBGRADIENTS

A modification of B.T.Poljak's subgradient algorithm is proposed. The algorithm is based on application of ε-subgradients. The numerical results show the convergences rate improvement of the algorithm.

1. *Поляк Б.Т.* Минимизация негладких функционалов // Вычислительная математика и математическая физика. – 1969.– № 3. – С. 509 – 521.
2. *Шор Н.З.* Методы минимизации недифференцируемых функций и их применение. – Киев: Наук.думка, 1979. – 200 с.
3. *Журбенко Н.Г.* Об одном классе методов минимизации с преобразованием пространства // Методы решения экстремальных задач. – 1996. – С. 68 – 80.
4. *Lemarechal C., Mifflin K.* Nonsmooth Optimization.Oxford: Pergamon Press, 1978. – 180 p.

Получено 22.03.2015