УДК 519.85

П.І. Стецюк, д-р фіз.-мат. наук,
О.В. Фесюк
Інститут кібернетики імені В.М. Глушкова НАН України, м. Київ,

ВИКОРИСТАННЯ r-АЛГОРИТМУ ДЛЯ РОЗВ'ЯЗАННЯ КВАДРАТИЧНОЇ ELD-ЗАДАЧІ

Анотація: Розглядається $r(\alpha)$-алгоритм з адаптивним кроком. Його програмна реалізація на мові octave застосована для розв'язання задачі оптимального добового завантаження енергосистеми (ELD-задача). Описано результати обчислювального експерименту для енергосистеми з 10, 12 та 40 енергоблоками.

Ключові слова: *r*-алгоритм, адаптивний крок, квадратична задача, негладкий штраф, програмне забезпечення.

Вступ. Одна із задач для оптимального завантаження енергосистеми в англомовній літературі має назву Economic Load Dispatch Problem (ELD-задача). В цій задачі задано набір енергоблоків, працюючих без відключень та їх характеристики. Потрібно знайти електричне навантаження кожного енергоблоку (тобто кількість електроенергії, яку генерує кожний енергоблок), щоб забезпечити мінімальні сумарні витрати на генерацію електроенергії та задовольнити потреби споживачів і ряд технологічних обмежень.

Одним з варіантів можливого розв'язання ELD-задач є прискорені субградієнтні методи з розтягом простору в напрямку різниці двох послідовних субградієнтів – так звані r-алгоритми [1]. Їх програмні реалізації виявились ефективними при мінімізації негладких опуклих функцій з яружною структурою поверхонь рівня. Octave-програма **ralgb5** [2] є однією з таких реалізацій, вона реалізує $r(\alpha)$-алгоритм з постійним α – коефіцієнтом розтягу простору та адаптивним регулюванням кроку в напрямку нормованого антисубградієнта у перетвореному просторі змінних.

В статті наведемо опис $r(\alpha)$-алгоритму та результатів роботи програми **ralgb5** для розв'язання квадратичних ELD-задач [3] (24 інтервали планового періоду, 10, 12, 40 енергоблоків). В задачі мінімізуються витрати умовного палива на генерацію електроенергії, які для кожного окремого енергоблоку описуються за допомогою опуклої квадратичної функції. Обмеження задачі описуються лінійними рівностями, які забезпечують виконання плану по генерації електричної енергії в інтервали планового періоду, та лінійними нерівностями, які визначають максимально допустимі границі можливого збільшення та зменшення потужності енергоблоку та діапазони можливих його електричних навантажень.

$r(\alpha)$-Алгоритми. Розглядається задача мінімізації опуклої функції $f(x)$, де $x \in E^n$ – вектор з n змінних. Мінімальне значення функції будемо позначати $f^* = f(x^*)$, $x^* \in X^*$. Вважаємо, що $f(x)$ має обмежену множину мінімумів X^*, тобто виконується умова $\lim_{\|x\| \to \infty} f(x) = +\infty$. Ця умова забезпечує коректність регулювання кроку в r-алгоритмах. Позначимо α – коефіцієнт розтягу простору, $\alpha > 1$.

$r(\alpha)$-Алгоритмом для мінімізації $f(x)$ називається ітеративна процедура знаходження послідовності n-вимірних векторів $\{x_k\}_{k=0}^{\infty}$ і послідовності $n \times n$-матриць $\{B_k\}_{k=0}^{\infty}$ за наступним правилом:

$$x_{k+1} = x_k - h_k B_k \xi_k, \quad B_{k+1} = B_k R_\beta(\eta_k), \quad k = 0, 1, 2, \ldots, \tag{1}$$

де

$$\xi_k = \frac{B_k^T g_f(x_k)}{\| B_k^T g_f(x_k) \|}, \quad h_k \ge h_k^* = \arg\min_{h \ge 0} f(x_k - h B_k \xi_k), \tag{2}$$

$$\eta_k = \frac{B_k^T r_k}{\| B_k^T r_k \|}, \quad \text{де} \quad r_k = g_f(x_{k+1}) - g_f(x_k). \tag{3}$$

Тут x_0 – стартова точка; $B_0 = I_n$ – одинична $n \times n$-матриця[1]; h_k^* – величина кроку з умови мінімуму функції $f(x)$ в напрямку нормованого антисубградієнта в перетвореному просторі змінних; $R_\beta(\eta) = I_n + (\beta - 1)\eta\eta^T$ – оператор стиснення простору субградієнтів у нормованому напрямку η з коефіцієнтом $\beta = 1/\alpha < 1$; $g_f(x_k)$ і $g_f(x_{k+1})$ – субградієнти функції $f(x)$ в точках x_k і x_{k+1}. Якщо на деякій ітерації k виконані критерії (умови) зупинки, то $k^* = k$, $x_k^* = x_k$ і процес (1)–(3) зупиняється.

На кожній ітерації r-алгоритмів реалізується субградієнтний спуск для опуклої функції $\varphi(y) = f(B_k y)$ в перетвореному просторі змінних $y = A_k x$, де $A_k = B_k^{-1}$. Дійсно, якщо обидві частини формули $x_{k+1} = x_k - h_k B_k \xi_k$ помножити зліва на матрицю A_k, отримаємо

$$y_{k+1} = A_k x_{k+1} = A_k x_k - h_k \xi_k = y_k - h_k \frac{B_k^T g_f(x_k)}{\| B_k^T g_f(x_k) \|} = y_k - h_k \frac{g_\varphi(y_k)}{\| g_\varphi(y_k) \|}, \tag{4}$$

[1]Матрицю B_0 часто вибирають як діагональну матрицю D_n з додатними коефіцієнтами по діагоналі, за допомогою яких масштабуються змінні x

де вектор $g_\varphi(y_k) = B_k^T g_f(x_k)$ є субградієнтом функції $\varphi(y) = f(B_k y)$ в точці $y_k = A_k x_k$ простору змінних $y = A_k x$. Це легко бачити з того, що субградієнт функції $f(x)$ в точці x_k задовольняє нерівність

$$f(x) \geq f(x_k) + (g_f(x_k))^T (x - x_k) \quad \forall x \in E^n,$$

звідки, здійснюючи заміну змінних $x = B_k y$, отримуємо

$$\varphi(y) \geq \varphi(y_k) + (B_k^T g_f(x_k))^T (y - y_k) = \varphi(y_k) + (g_\varphi(y_k))^T (y - y_k) \; \forall y \in E^n.$$

Якщо $h_k = h_k^*$, то формула (4) означає пошук точного мінімуму функції $\varphi(y) = f(B_k y)$ в напрямку нормованого антисубградієнта в перетвореному просторі змінних $y = A_k x$, а якщо $h_k \approx h_k^*$, то – наближений пошук мінімуму функції.

Метод (1)–(3) називають B-формою r-алгоритмів; на кожній його ітерації коригується матриця, пов'язана із заміною змінних $x = By$. Ітерація методу вимагає приблизно $5n^2$ арифметичних операцій множення, які визначають обчислювальну трудомісткість ітерації (операції додавання не враховуємо через їх малий вклад у трудомісткість ітерації). При цьому $3n^2$ множень матриці на вектор $B_k \xi_k$, $B_k^T g_f(x_k)$ і $B_k^T r_k$, а $2n^2$ множень – для одновимірної корекції матриці $B_{k+1} = B_k R_{\beta_k}(\eta_k)$. Дійсно,

$$B_{k+1} = B_k R_{\beta_k}(\eta_k) = B_k (I_n + (\beta_k - 1)\eta_k \eta_k^T) = B_k + (\beta_k - 1)(B_k \eta_k)\eta_k^T,$$

звідки легко бачити, що обчислення вектора $\eta = B_k \eta_k$ потребує n^2 множень і стільки ж множень вимагає побудова однорангової матриці $\eta \eta_k^T$.

$r(\alpha)$-Алгоритм з адаптивним кроком. Одним з ефективних зарекомендував себе $r(\alpha)$-алгоритм з адаптивним регулюванням кроку. Це варіант r-алгоритмів, в якому величина кроку h_k налаштовується в процесі виконання одновимірного спуску в напрямку нормованого антисубградієнта в перетвореному просторі змінних за допомогою параметрів h_0, q_1, n_h, q_2. Тут h_0 – величина початкового кроку (вона використовується на 1-й ітерації, а на кожній наступній ітерації ця величина уточнюється); q_1 – коефіцієнт зменшення кроку ($q_1 \leq 1$), якщо умова завершення спуску у напрямку виконується за один крок; q_2 – коефіцієнт збільшення кроку ($q_2 \geq 1$); натуральне число n_h задає кількість зроблених кроків одновимірного спуску ($n_h > 1$), після якої величина кроку збільшуватиметься в q_2 раз. Умова завершення спуску у деякому напрямку виконується, як тільки виявлена точка x_{k+1}, для якої $(x_{k+1} - x_k)^T g_f(x_{k+1}) \geq 0$. Оскільки передбачається, що $\lim\limits_{\|x\| \to \infty} f(x) = +\infty$, то після скінченної кількості кроків адаптивного спуску в напрямку нормованого антисубградієнта обов'язково виконається умова завершення спуску.

Ітеративний процес в $r(\alpha)$-алгоритмі з адаптивним кроком триває до виконання деякого критерію зупинки, де ключову роль відіграють параметри ε_x і ε_g. Вони визначають дві умови зупинки: метод зупиняється в точці x_{k+1}, якщо виконана умова $\| x_{k+1} - x_k \| \leq \varepsilon_x$ (зупинка за аргументом); метод зупиняється в точці x_{k+1}, якщо виконана умова $\| g_f(x_{k+1}) \| \leq \varepsilon_g$ (зупинка за нормою субградієнта, яка використовується для гладких функцій). Використовуються ще дві можливі умови зупинки методу: стандартна зупинка, якщо перевищено максимальну кількість ітерацій, і аварійна зупинка, яка сигналізує про те, що або функція $f(x)$ необмежена знизу, або початковий крок h_0 занадто малий, і його потрібно збільшити.

Якщо $r(\alpha)$-алгоритм з адаптивним регулюванням кроку застосовувати для мінімізації негладких функцій, то рекомендується наступний вибір значень параметрів: $\alpha = 2 \div 4$, $h_0 = 1.0$, $q_1 = 1.0$, $q_2 = 1.1 \div 1.2$, $n_h = 2 \div 3$. Якщо відома апріорна оцінка відстані від початкової точки x_0 до точки мінімуму x^*, то доцільно, щоб величина початкового кроку h_0 приблизно дорівнювала $\| x_0 - x^* \|$. При мінімізації негладких функцій рекомендовані такі ж значення параметрів, за винятком $q_1 = 0.8 \div 0.95$. Це обумовлено тим, що додаткове подрібнення кроку сприяє підвищенню точності пошуку мінімуму функції у напрямку і при мінімізації гладких функцій забезпечує більш швидку швидкість збіжності. При такому виборі параметрів, як правило, кількість кроків у напрямку рідко перевищує два, а за n ітерацій точність за значенням функції поліпшується в три-п'ять разів. Параметри зупинки $\varepsilon_x, \varepsilon_g \sim 10^{-6} \div 10^{-5}$ при мінімізації опуклої функції, навіть істотно яружної структури, забезпечують знаходження точки x_k^*, для якої значення опуклої функції є досить близьким до оптимального: $(f(x_k^*) - f^*) \big/ (|f^*| + 1) \sim 10^{-6} \div 10^{-5}$ – для негладких функцій і $\sim 10^{-12} \div 10^{-10}$ – для гладких функцій.

Квадратична ELD-задача та програма ralgb5. Нехай енергосистема складається з N енергоблоків, які працюють паралельно. Для кожного енергоблоку i, $i = 1,...,N$, задані P_i^{low} і P_i^{up} – нижня і верхня межі його допустимого електричного навантаження. Нехай T – тривалість планового періоду. Для кожного інтервалу t, $t = 1,...,T$, задано E_t – планове електричне навантаження енергосистеми. Параметри DR_i (UR_i) задають допустимі значення на послідовне зменшення (збільшення) навантаження для *i*-го енергоблоку.

Нехай $x_{i,t}$ – невідоме електричне навантаження *i*-го енергоблоку в інтервалі t. Опукла квадратична ELD-задача має вигляд: знайти

$$f^* = f(x^*) = \min \sum_{t=1}^{T} \sum_{i=1}^{N} (a_i x_{i,t}{}^2 + b_i x_{i,t} + c_i) \tag{5}$$

при обмеженнях

$$\sum_{i=1}^{N} x_{i,t} = E_t, t = 1,\ldots,T, \tag{6}$$

$$x_{i,t} - x_{i,t-1} \le UR_i, t = 2,\ldots,T, i = 1,\ldots,N, \tag{7}$$

$$x_{i,t-1} - x_{i,t} \le DR_i, t = 2,\ldots,T, i = 1,\ldots,N, \tag{8}$$

$$P_i^{low} \le x_{i,t} \le P_i^{up}, i = 1,\ldots,N, t = 1,\ldots,T, \tag{9}$$

де цільова функція (5) мінімізує витрати умовного палива, a_i, b_i, c_i – задані параметри, такі що $a_i > 0$ та $b_i \ge 0$ для $i = 1,\ldots,N$.

Алгоритм розв'язання задачі (5)–(9) є наступним. За допомогою методу негладких штрафних функцій задача (5)–(9) зводиться до задачі безумовної мінімізації негладкої опуклої функції

$$\begin{aligned} F(x) = {} & \sum_{i=1}^{N} \sum_{t=1}^{T} (a_i x_{i,t}{}^2 + b_i x_{i,t} + c_i) + Q_1 \sum_{t=1}^{T} | \sum_{i=1}^{N} x_{i,t} - E_t | + \\ & + Q_2 \sum_{t=2}^{T} \sum_{i=1}^{N} \max\{0, x_{i,t} - x_{i,t-1} - UR_i, x_{i,t-1} - x_{i,t} - DR_i\} + \\ & + Q_3 \sum_{t=1}^{T} \sum_{i=1}^{N} \max\{0, x_{i,t} - P_i^{up}, P_i^{low} - x_{i,t}\}, \end{aligned} \tag{10}$$

де Q_1, Q_2, Q_3 – штрафні множники, які відповідають за порушення обмежень (6), (7) та (8), (9), відповідно. Для мінімізації функції $F(x)$ використовується octave-програма **ralgb5** [3, с. 383–386], яка реалізує $r(\alpha)$-алгоритм з адаптивним кроком. Octave-функція **calcfg(x)**, яка для програми **ralgb5** обчислює значення негладкої функції (10) та її субградієнта в точці x, реалізована за допомогою алгебраїчних операцій системи GNU Octave.

Результати експериментів для квадратичних ELD-задач (24 інтервали планового періоду, 10, 12 та 40 енергоблоків) наведено в таблиці. Обчислення проводилися на комп'ютері з процесором AMD FX-8350 Eight-Core Processor з частотою 4 ГГц (час виконання програми в секундах наведено у колонці "t_1") та комп'ютері AMD Athlon X2 Dual-Core QL-64×2 з частотою 2.1 ГГц (колонка "t_2") з операційними системами Ubuntu та Octave 4.0.2. Для функції $F(x)$ використовувались наступні штрафні множники: $Q_1 = Q_2 = Q_3 = 1000$. Параметри програми **ralgb5** вибирались наступними: $\alpha = 4$, $h_0 = 500$, $q_1 = 0.95$, $q_2 = 1.1$, $n_h = 3$, $\varepsilon_g = 10^{-6}$, $\varepsilon_x \in \{10^{-6}, 10^{-5}, 10^{-4}\}$. У таблиці через *calls* позначено кількість викликів функції **calcfg(x)**, які затрачено на k^* ітерацій $r(\alpha)$-алгоритму.

Таблиця.

Результати експериментів для квадратичної ELD-задачі

ε_x	$n = N \times T$	$f(x_k^*)$	k^*	*calls*	t_1	t_2
1.e-6	240=10*24	420540	4427	9198	6.73	17.1
1.e-5			3879	8063	5.87	14.9
1.e-4			3248	6719	4.96	12.6
1.e-6	288=12*24	463391	5150	10475	10	25.4
1.e-5			3944	8155	7.56	19.2
1.e-4			3420	7062	6.54	16.6
1.e-6	960=40*24	3021209	13437	26272	180	456.8
1.e-5			13313	26038	178.7	445
1.e-4			12446	24380	164.5	417.9

В таблиці наведено результати обчислень для трьох ELD-задач з добовими погодинними графіками $T = 24$ для $N = 10$, $N = 12$ та $N = 40$ енергоблоків, яким відповідають задачі мінімізації функції $F(x)$ від $n = 240$, $n = 288$ та $n = 960$ змінних. Так, для найбільшої ELD-задачі з $N = 40$, де погодинний графік $E = E^{(1)}$ та параметри енергоблоків взяті з [3], час розв'язання при $\varepsilon_x = 10^{-6}$ склав 180 секунд, а при $\varepsilon_x = 10^{-4}$ – 164,5 секунди, що на 8.6% менше ніж при $\varepsilon_x = 10^{-6}$. При цьому для всіх трьох ε_x знайдено значення цільової функції, яке дорівнює 3 021 209 та співпадає зі знайденим за допомогою відомих програм з NEOS-сервера [3]. Для ELD-задач менших розмірів ($n = 240$, $n = 288$) результати розрахунків в таблиці вказують на те, що зменшення точності ε_x на два порядки дозволяє майже на 30% швидше отримати їх розв'язок.

Висновки: Проведені для опуклих квадратичних ELD-задач експерименти показали, що $r(\alpha)$-алгоритм з адаптивним кроком за прийнятний час та з необхідною точністю дозволяє знайти оптимальне завантаження енергосистеми, в якій для закриття балансу потужностей залучаються до 40 енергоблоків ТЕС (теплових електростанцій). Програмне забезпечення на основі програми **ralgb5** буде ефективним для енергосистеми з кількістю 10–12 енергоблоків ТЕС (виконання програми вимагає не більше 10 секунд на 8-ядерному та не більше 25 секунд на 2-ядерному комп'ютерах). Якщо енергосистема включає до 40 енергоблоків ТЕС, то час виконання програми на 8-ядерному комп'ютері буде прийнятним (до трьох хвилин). Його можна пришвидшити на 30-50 відсотків за рахунок вибору параметрів для програми **ralgb5**, а в 10-12 разів за допомогою технології CUDA з використанням графічних процесорів [4].

Список використаних джерел

1. *Шор Н.З.* Методы минимизации недифференцируемых функций и их приложения. Киев: Наук. думка, 1979. 200 с.
2. *Стецюк П.И.* Методы эллипсоидов и *r*-алгоритмы. Кишинэу, Эврика, 2014. 488 с.
3. *Стецюк П.І., Лиховид О.П., Фесюк О.В.* NLP-програми для ELD-задач завантаження енергосистеми. *Комп'ютерна математика*. 2016. № 2. С. 142 – 150.
4. *Стецюк П.І., Хіміч О.М., Сидорук В.А.* Реалізація *r*-алгоритму на графічних процесорах. *Комп'ютерна математика*. 2016. № 2. С. 100 – 109.

Annotation: $r(\alpha)$-algorithm with adaptive step size adjustment is considered. It's program implementation in octave language was used for solving the problem of optimal daily power system load (ELD-problem). The results of computational experiments for power system with 10, 12 and 40 units are described.

Key words: r-algorithm, adaptive step, quadratic problem, non-smooth penalty, software.

Article title: Use of r-algorithm for solving quadratic ELD-problem

Відомості про авторів:

Стецюк Петро Іванович, доктор фізико-математичних наук, старший науковий співробітник, Інститут кібернетики імені В.М. Глушкова НАН України, stetsyukp@gmail.com

Фесюк Олександр Володимирович, Інститут кібернетики імені В.М. Глушкова НАН України, sasha.fesyuk@gmail.com