КОМПЬЮТЕРНАЯ МАТЕМАТИКА

УДК 519.85

П.І. СТЕЦЮК, О.В. ФЕСЮК, В.А. СИДОРУК

АЛГОРИТМИ РОЗВ'ЯЗАННЯ ЗАДАЧІ СЕПАРАБЕЛЬНОГО КВАДРАТИЧНОГО ПРОГРАМУВАННЯ

Розглянуто математичну модель задачі сепарабельного квадратичного програмування та методи розв'язання задачі за допомогою алгоритмів негладкої оптимізації. Описано програмні реалізації методів на основі модифікації r-алгоритму. Наведено результати обчислювальних експериментів з розв'язування квадратичних задач знаходження електричних навантажень енергоблоків теплових електростанцій ОЕС України.

Вступ. Функцію $f(x_1,\ldots,x_n)$ від n змінних називають сепарабельною, якщо її можна представити у вигляді $f(x_1,\ldots,x_n)=\sum_{i=1}^{n} f_i(x_i)$, де $f_i(x_i)$ – функція від однієї змінної, $i=1,\ldots,n$. Сепарабельне програмування (separable programming) – сукупність методів розв'язання задач нелінійного програмування з сепарабельною цільовою функцією та лінійною системою обмежень. Задачею сепарабельного нелінійного програмування є ELD-задача завантаження енергосистеми [1]. У ній цільова функція задає сумарні витрати умовного палива, необхідні для функціонування енергосистеми, та визначається незалежними один від одного навантаженнями енергоблоків.

У статті розглядаються алгоритми розв'язання задачі сепарабельного квадратичного програмування та їх застосування до квадратичних ELD-задач завантаження енергосистеми. У розділі 1 наведено опис задачі сепарабельного квадратичного програмування та алгоритмів її розв'язання за допомогою методів негладкої оптимізації. У розділі 2 наведено модифікацію *r*-алгоритму, яка покладена в основу програмних реалізацій послідовного та паралельного алгоритмів. У розділі 3 описано застосування алгоритмів для розв'язання ELD-задачі з квадратичною цільовою функцією. Приведено результати обчислювальних експериментів щодо задач знаходження електричних навантажень енергоблоків теплових електростанцій ОЕС України для різних за напруженістю графіків завантаження працюючих енергоблоків ТЕС.

1. Задача сепарабельного квадратичного програмування та її властивості. Розглянемо таку задачу квадратичного програмування: знайти

$$f^* = \min_{x\in E^n}\left\{ f(x) = \sum_{i=1}^{n}(c_i x_i^2 + d_i x_i + e_i)\right\} \tag{1}$$

при обмеженнях

$$b^{low} \le Ax \le b^{up}, \tag{2}$$

$$x^{low} \le x \le x^{up}, \tag{3}$$

де $c_i \ge 0$, $c_i \in R$, $d_i \in R$, $e_i \in R$, $i = 1,...,n$, $A \in R^{m\times n}$ – $m\times n$ -матриця, $b^{low} \in R^m$ і $b^{up} \in R^m$ – нижні (low) і верхні (up) границі, накладені на рядки-обмеження, $x^{low} \in R^n$ і $x^{up} \in R^n$ – нижні і верхні границі можливих значень змінних $x \in R^n$.

Задачу (1) – (3) можна записати у еквівалентній формі: знайти

$$f^* = \min_{x\in E^n}\sum_{i=1}^{n}(c_i x_i^2 + d_i x_i + e_i) \tag{4}$$

при обмеженнях

$$\sum_{j=1}^{n} a_{ij}x_j - b_i^{up} \le 0, \quad i = 1,\dots,m, \tag{5}$$

$$-\sum_{j=1}^{n} a_{ij}x_j + b_i^{low} \le 0, \quad i = 1,\dots,m, \tag{6}$$

$$x_i - x_i^{up} \le 0, \quad i = 1,\dots,n, \tag{7}$$

$$-x_i + x_i^{low} \le 0, \quad i = 1,\dots,n. \tag{8}$$

Задача (4) – (8) має n змінних та $M = 2m + 2n$ лінійних обмежень. Вона є частковим випадком задачі квадратичного програмування, що обумовлено сепарабельністю цільової функції. Задачу (4) – (8) та еквівалентну їй задачу (1) – (3) будемо називати задачами сепарабельного квадратичного програмування.

Якщо $c_i > 0$ для всіх $i = 1,\dots,n$, то цільова функція (4) є строго опуклою. Це гарантує єдиний розв’язок задачі (4) – (8), якщо система лінійних обмежень (5) – (8) є сумісною. У випадку, коли $c_i = 0$ для деяких $i = 1,\dots,n$, то задача (4) – (8) може мати багато розв’язків. Це стосується також випадку, коли $c_i = 0$ для всіх $i = 1,\dots,n$ і задача (4) – (8) переходить у задачу лінійного програмування.

Задачу (4) – (8) можна розв’язати за допомогою стандартних програм розв’язання задач квадратичного програмування. Для цього можна використати такі відомі програми як Ipopt, Knitro, LANCELOT, LOQO, MINOS та інші. Для розв’язання задачі (4) – (8) можна застосувати методи мінімізації негладких опуклих функцій. Для цього задачу (4) – (8) потрібно звести до задачі безумовної мінімізації негладкої опуклої функції, що легко зробити за допомогою методу негладких штрафних функцій.

Нехай система обмежень (5) – (8) задовольняє умові Слейтера. Для задачі (4) – (8) позначимо $U^* = \{U_1^*, ..., U_m^*\}$ та $u^* = \{u_1^*, ..., u_m^*\}$ – оптимальні значення множників Лагранжа для обмежень (5) та (6), $V^* = \{V_1^*, ..., V_n^*\}$ та $v^* = \{v_1^*, ..., v_n^*\}$ – оптимальні значення множників Лагранжа для обмежень (7) та (8), а $P = \{P_1, P_2\}$ – вектор із двох додатних штрафних коефіцієнтів.

Теорема 1. Якщо $P_1 > \max\{U_1^*, ..., U_m^*, u_1^*, ..., u_m^*\}$ і $P_2 > \max\{V_1^*, ..., V_n^*, v_1^*, ..., v_n^*\}$, то задача мінімізації негладкої опуклої функції

$$F_P(x) = \min_{x \in E^n} \sum_{i=1}^{n} (c_i x_i^2 + d_i x_i + e_i) +$$

$$+ P_1 \sum_{i=1}^{m} \max\{0, \sum_{j=1}^{n} a_{ij} x_j - b_i^{up}\} + P_1 \sum_{i=1}^{m} \max\{0, -\sum_{j=1}^{n} a_{ij} x_j + b_i^{low}\} +$$

$$+ P_2 \sum_{i=1}^{n} \max\{0, x_i - x_i^{up}\} + P_2 \sum_{i=1}^{n} \max\{0, -x_i + x_i^{low}\} \qquad (9)$$

еквівалентна задачі (4) – (8) в тому сенсі, що множина мінімумів функції $F_P(x)$ збігається з множиною оптимальних розв'язків задачі (4) – (8).

Теорема 1 є наслідком теореми 4.2 [2, стор. 188], якщо для задачі (4) – (8) вибрати найпростіший варіант негладкої функції штрафу

$$p_i(t) = \begin{cases} 0, & \text{якщо } t \le 0, \\ c_i t, & \text{якщо } t > 0, \end{cases} \quad i = 1, ..., 2m + 2n,$$

а коефіцієнти c_i апроксимувати зверху максимальними значеннями оптимальних множників Лагранжа, які відповідають обмеженням (5), (6) та обмеженням (7), (8), відповідно.

На основі теореми 1 можна будувати різноманітні алгоритми для розв'язання задачі сепарабельного квадратичного програмування, вибираючи різні методи для мінімізації негладкої опуклої функції виду (9). При цьому досить впевнитися, що обмеження (5) – (8) задовольняють умові Слейтера. Така перевірка зводиться до задачі мінімізації негладкої опуклої функції.

Теорема 2. Система обмежень (5) – (8) задовольняє умові Слейтера, якщо негладка опукла функція

$$\Phi_P(x) = F_P(x) - \min_{x \in E^n} \sum_{i=1}^{n} (c_i x_i^2 + d_i x_i + e_i) =$$

$$= P_1 \sum_{i=1}^{m} \max\{0, \sum_{j=1}^{n} a_{ij} x_j - b_i^{up}\} + P_1 \sum_{i=1}^{m} \max\{0, -\sum_{j=1}^{n} a_{ij} x_j + b_i^{low}\} +$$

$$+ P_2 \sum_{i=1}^{n} \max\{0, x_i - x_i^{up}\} + P_2 \sum_{i=1}^{n} \max\{0, -x_i + x_i^{low}\} \qquad (10)$$

досягає мінімального значення $\Phi_P^* = \Phi_P(x^*) = 0$ при $P_1 > 0$ і $P_2 > 0$.

Для мінімізації негладких опуклих функцій $F_P(x)$ та $\Phi_P(x)$ можно використовувати подану далі модифікацію r-алгоритмів Шора (субградієнтних методів з розтягування простору в напрямку різниці двох послідовних субградієнтів).

2. *r*-алгоритм та програма ralgb5a [3]. Розглянемо задачу пошуку точки мінімуму деякої опуклої функції $f(x)$, $x \in R^n$. Мінімальне значення функції позначимо $f^* = f(x^*)$, де $x^* \in X^*$. Будемо вважати, що множина мінімумів X^* є обмеженою, тобто виконується умова $\lim\limits_{\|x\|\to\infty} f(x) = +\infty$, яка забезпечує коректність регулювання кроку в r-алгоритмах. Позначимо α – коефіцієнт розтягування простору, $\alpha > 1$.

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

$$x_{k+1} = x_k - h_k B_k \xi_k, \quad B_{k+1} = B_k R_\beta(\eta_k), \quad k = 0,1,2,\ldots, \tag{11}$$

де

$$\xi_k = \frac{B_k^T g_f(x_k)}{\left\|B_k^T g_f(x_k)\right\|}, \quad h_k \ge h_k^* = \arg\min_{h\ge 0} f(x_k - hB_k\xi_k), \tag{12}$$

$$\eta_k = \frac{B_k^T r_k}{\| B_k^T r_k \|}, \quad r_k = g_f(x_{k+1}) - g_f(x_k). \tag{13}$$

Тут x_0 – стартова точка; $B_0 = I_n$ – одинична $n \times n$-матриця[1]; h_k^* – величина кроку до точки мінімуму функції $f(x)$ у 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 та x_{k+1}. Якщо на ітерації k процесу (11) – (13) виконані деякі критерії (умови) зупинки, то вважаємо $k^* = k$, $x_k^* = x_k$ і закінчуємо роботу алгоритму.

Коментар. 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 $x_{k+1} = x_k - h_k B_k \xi_k$ домножити зліва на матрицю A_k, то отримаємо

$$y_{k+1} = A_k x_{k+1} = A_k x_k - h_k \xi_k = y_k - h_k \frac{B_k^T g_f(x_k)}{\left\|B_k^T g_f(x_k)\right\|} = y_k - h_k \frac{g_\varphi(y_k)}{\left\|g_\varphi(y_k)\right\|}, \tag{14}$$

---

[1]За матрицю B_0 часто вибирають діагональну матрицю D_n з додатними коефіцієнтами на діагоналі, за допомогою якої здійснюється масштабування змінних.

де вектор $g_\varphi(y_k) = B_k^T g_f(x_k)$ є субградієнтом функції $\varphi(y) = f(B_k y)$ у точці $y_k = A_k x_k$ простору змінних $y = A_k x$. Субградієнт функції $f(x)$ задовільняє нерівності

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

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

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

Сімейство $r(\alpha)$-алгоритмів визначається коефіцієнтом розтягування простору $\alpha > 1$ і послідовністю величин кроків $\{h_k\}_{k=0}^{\infty}$. Їх вибір (допускається $\alpha = \infty$, що відповідає $\beta = 0$) у поєднанні з вибором критеріїв зупинки визначають той чи інший варіант $r(\alpha)$-алгоритмів.

Регулювання кроку. Регулювання величини h_k задає певний спосіб реалізації одновимірного спуску в напрямку нормованого антисубградієнта функції $\varphi(y) = f(B_k y)$, який здійснюється в перетвореному просторі змінних $y = A_k x$ (див. формулу (14)). В r-алгоритмах використовуються два способи регулювання кроку [2, 4].

При першому способі величина h_k вибирається з умови $h_k = h_k^*$ ($h_k \approx h_k^*$), що означає точний (чи наближений) одновимірний пошук мінімуму функції, який гарантує, що r-алгоритми є монотонними (майже монотонними) по функції, яка мінімізується. Таке регулювання кроку використовуться в теоретично обгрунтованих модифікаціях $r(\alpha)$-алгоритмів, коли коефіцієнт α можна вибирати досить великим. Це дозволило показати, що граничний варіант r-алгоритмів (при $\beta = 0$) є проективним методом спряжених градієнтів, та обгрунтувати для граничного варіанта r-алгоритмів з відновленням квадратичної швидкості збіжності до точки мінімуму опуклої двічі неперервної диференційовної функції при деяких умовах гладкості та регулярності. Знайдено також такі умови для кусково-гладких опуклих функцій, при яких $r_\mu(\alpha)$-алгоритм збігається до точки мінімуму.

Другий спосіб регулювання кроку полягає у тому, що величина h_k налаштовується (адаптується) в процесі виконання одновимірного спуску, який завершується, як тільки знайдено субградієнт, що утворює негострий кут з субградієнтом, який визначає напрямок одновимірного спуску (умова завершення спуску за напрямом). Налаштування величини кроку h_k здійснюється за допомогою чотирьох параметрів: $h_0 > 0$ – величина початкового кроку (використовується на першій ітерації, а на кожній наступній уточнюється); q_1 ($q_1 \leq 1$) – коефіцієнт зменшення кроку (якщо умова завершення спуску за напрямком виконується за перший крок); q_2 ($q_2 \geq 1$) – коефіцієнт збільшення кроку. Через кожні n_h кроків одновимірного спуску ($h_h > 1$) величина кроку збільшується в q_2 раз. Оскільки припускається, що $\lim_{\|x\|\to\infty} f(x) = +\infty$, то після скінченної кількості кроків адаптивного спуску в напрямку нормованого антисубградієнта обов'язково виконується умова завершення спуску за напрямом.

Другий спосіб регулювання кроку називається адаптивним і використовується в практичних реалізаціях r-алгоритмів. Тут коефіцієнт α не рекомендується вибирати великим. Він має бути узгодженим з коефіцієнтом q_1, тому що вони обидва, хоча і по різному, впливають на зменшення величини кроку h_k. В результаті визначаються ті два послідовні субградієнти, розтягування за різницею яких зменшує степінь витягнутості функції у перетвореному просторі змінних. Цим пояснюється пришвидшена збіжність $r(\alpha)$-алгоритмів для яружних (з витягнутими поверхнями рівня) функцій. Кількість ітерацій $r(\alpha)$-алгоритму, необхідна для знаходження точки x_{k^*}, для якої $f(x_{k^*}) - f^* \leq \varepsilon$, емпірично оцінюється як $k^* = O(n \log(1/\varepsilon))$, де n – кількість змінних.

Octave-функция ralgb5a [3]. Програма **ralgb5a –** спрощена (для зручності використання) версія програми **ralgb5** [5, с. 383–386]. У ній зафіксовані два найбільш часто використовувані параметри $q_2 = 1.1$ і $n_h = 3$. Величина початкового кроку для чергової ітерації може максимально збільшуватися в 10^6 раз. У програмі **ralgb5a** використовується параметр **intp (int**erval for **p**rint**)**, який забезпечує друк інформації про хід процесу мінімізації через кожні **intp** ітерацій. Цей параметр дозволяє скоротити протокол друку роботи програми при мінімізації функції для сотень і тисяч змінних, коли кількість ітерацій оцінюється тисячами і десятками тисяч. Програма використовує параметри ε_x і ε_g для зупинки ітераційного процесу в точці $x_{k^*} \in [x_k, x_{k+1}]$, де $\|x_{k+1} - x_k\| \leq \varepsilon_x$ (зупинка за аргументом), або $\|g_f(x_{k^*})\| \leq \varepsilon_g$ (зупинка за нормою градієнта, яка використовується для гладких функцій). Використовуються також стандартна зупинка, якщо перевищено максимальну задану кількість ітерацій **maxitn,** та аварійна зупинка, яка сигналізує про те, що або функція $f(x)$ не є обмеженою знизу, або початковий крок h_0 занадто малий, і його треба збільшити.

Код Octave-функції **ralgb5a** наведений у [3]. Якщо ітераційний процес запускається зі стартової точки x_0, то параметри $r(\alpha)$-алгоритму рекомендується вибирати наступними: $\alpha \in [2,4]$, $q_1 = 1.0$ (для негладких функцій), $q_1 = 0.8 \div 0.95$ (для гладких функцій), $h_0 \approx \|x_0 - x^*\|$ – оцінка відстані від стартової точки x_0 до точки мінімуму x^*. Як правило, використовуються такі параметри зупинки: $\varepsilon_x \sim 10^{-6}$, $\varepsilon_g \sim 10^{-12}$, **maxitn** $\sim 20\,n$. Тут параметр ε_g

використовується для гладких функцій, а параметр ε_x для негладких функцій. Якщо програма **ralgb5a** завершує роботу за умовою $\|x_{k+1} - x_k\| \le \varepsilon_x = 10^{-8}$, то цього цілком достатньо, щоб на $14-15$ порядків зменшити різницю між знайденим рекордним значенням квадратичної функції f_r і її мінімальним значенням f^*.

3. Квадратична ELD-задача та обчислювальні експерименти. 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 та верхня межі його допустимого електричного навантаження. Позначимо 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

$$f^* = f(x^*) = \min \sum_{t=1}^{T}\sum_{i=1}^{N}(c_i x_{i,t}^2 + d_i x_{i,t} + e_i) \tag{15}$$

при обмеженнях

$$\sum_{i=1}^{N} x_{i,t} = E_t,\ t = 1,\ldots,T, \tag{16}$$

$$x_{i,t} - x_{i,t-1} \le UR_i,\ t = 2,\ldots,T,\ i = 1,\ldots,N, \tag{17}$$

$$x_{i,t-1} - x_{i,t} \le DR_i,\ t = 2,\ldots,T,\ i = 1,\ldots,N, \tag{18}$$

$$P_i^{low} \le x_{i,t} \le P_i^{up},\ i = 1,\ldots,N,\ t = 1,\ldots,T, \tag{19}$$

де цільова функція (15) мінімізує витрати умовного палива, c_i, d_i, e_i – задані параметри, такі що $c_i > 0$ та $d_i \ge 0$ для $i = 1,\ldots,N$. Для розв'язання опуклих квадратичних ELD-задач можна використовувати r-алгоритм [6].

З урахуванням заданої точності щодо виконання плану навантажень енергоблоків задачу (15) – (19) можна звести до наступної задачі сепарабельного квадратичного програмування: знайти

$$f^* = f(x^*) = \min\left\{ f(x) = \sum_{t=1}^{T}\sum_{i=1}^{N}(c_i x_{i,t}^2 + d_i x_{i,t} + e_i) \right\} \tag{20}$$

при обмеженнях

$$E_t - \varepsilon \le \sum_{i=1}^{N} x_{i,t} \le E_t + \varepsilon, \quad t = 1,\ldots,T, \tag{21}$$

$$-DR_i \le x_{i,t} - x_{i,t-1} \le UR_i, \quad i = 1,\ldots,N,\ t = 2,\ldots,T, \tag{22}$$

$$P_i^{low} \le x_{i,t} \le P_i^{up}, \quad i = 1,\ldots,N,\ t = 1,\ldots,T, \tag{23}$$

де ε – достатньо мала величина, з точністю до якої потрібно виконати планове навантаження енергосистеми у кожному інтервалі планового періоду.

Для розв'язання задачі (20) – (23) розроблено програмне забезпечення, яке реалізує послідовний та паралельний алгоритми, де для мінімізації функцій $F_P(x)$ та $\Phi_P(x)$ (теореми 1 та 2) використовується octave-програма **ralgb5a.** Алгоритми реалізовано на мові Octave з використанням бібліотеки OpenBLAS та на мові C++ з використанням технології Nvidia CUDA (гібридний алгоритм). Програмне забезпечення використовує вхідні файли для керування параметрами задачі (20) – (23), для вибору параметрів $r(\alpha)$-алгоритму з адаптивним кроком, для визначення даних про енергоблоки та графік завантаження енергосистеми.

За допомогою розробленого програмного забезпечення проведено тестові розрахунки для квадратичних ELD-задач завантаження енергоблоків. Для експериментів були вибрані три графіки з різними за складністю навантаженнями енергоблоків теплових електростанцій ОЕС України за інформацією з [7, 8]. Найлегшим до виконання є перший графік, який відповідає фактичному навантаженню 24 енергоблоків ТЕС, яке мало місце 6 червня 2017 року. Для другого тесту вибрано графік середньої важкості, який відповідає фактичному навантаженню за 19 жовтня 2016 року, до якого залучено 33 енергоблоки. Найважчим до виконання є третій графік, до якого залучено 40 енергоблоків. Він наведений в табл. 1 та відповідає фактичному навантаженню теплових електростанцій ОЕС України за 30 січня 2017 року.

ТАБЛИЦЯ 1. Фактичне навантаження ТЕС ОЕС України за 30 січня 2017 року

Година	1	2	3	4	5	6	7	8	9	10	11	12
Навантаження	7155	6696	6953	7158	6918	6868	8056	8747	9162	9585	9628	9655
Година	13	14	15	16	17	18	19	20	21	22	23	24
Навантаження	9720	9841	9945	10009	10065	10082	10033	10037	10042	10002	8980	8743

Експерименти з послідовним алгоритмом проводилися на персональному комп'ютері з процесором Intel® Xeon® CPU E5-1607 / 3.00GHz×4, операційною системою Ubuntu 14.04 LTS та за допомогою GNU Octave версії 4.2.1 з використанням бібліотеки OpenBLAS. В табл. 2 наведено результати обчислень для кожного графіку навантаження з 20 та 40 хвилинними інтервалами. Тут N – кількість енергоблоків, T – кількість згенерованих періодів у проміжку від 7 години ранку до 24 години вечора, відповідно до фактичного графіка навантаження, коли енергоблоки не вимикались, $N \times T$ – кількість змінних задачі. У стовпчику $time$ вказано час у секундах, який затрачено на пошук оптимального розв'язку.

Для покриття графіків навантаження вибирались енергоблоки, які працювали у відповідні дні. Верхня межа навантаження та використання умовного палива ($fuel$) взяті з [7, 8], нижня межа допустимого навантаження енергоблоків визначалася згідно [9]. Коефіцієнт e_i визначався діленням величини $fuel$ на 3; d_i дорівнює e_i поділеному на avr_i – середнє значення навантаження енергоблоку i; c_i дорівнює відповідно d_i / avr_i . Параметр DR відповідає плановій допустимій зміні навантаження на хвилину [10], яку помножено на 20 або 40, UR – неплановій допустимій зміні, яка помножена відповідно на 20 або 40.

ТАБЛИЦЯ 2. Результати обчислювальних експериментів

Графік завантаження	Розмірність задачі			*time*
	N	T	$N \times T$	(sec)
08 липня 2017	24	24	576	9,25
	24	29	696	16,64
19 жовтня 2016	33	24	792	19,6
	33	49	1617	103,5
30 січня 2017	40	24	960	28,54
	40	50	2000	153,48

Обчислювальні експерименти із застосуванням гібридного алгоритму [11] на основі технології NVidia CUDA показали пришвидшення часу розв'язання задач у 8 – 10 раз. Це дозволяє розв'язувати вказані квадратичні ELD-задачі завантаження енергоблоків в оперативному режимі (до 30 секунд) за допомогою сучасного персонального комп'ютера.

Висновки. В роботі наведено математичну модель задачі сепарабельного квадратичного програмування та описано методи її розв'язання за допомогою алгоритмів негладкої оптимізації. Описано алгоритмічне та програмне забезпечення на основі модифікації *r*-алгоритму та наведено результати обчислювальних експериментів для квадратичних ELD-задач з різними (за складністю) режимами навантажень енергоблоків теплових електростанцій ОЕС України. Аналіз результатів проведених експериментів показує, що програмне забезпечення можна використовувати для оперативного планування навантаження енергоблоків теплових електростанцій ОЕС України.

1. Стецюк П.І., Лиховид О.П., Фесюк О.В. NLP-програми для ELD-задач завантаження енергосистеми. *Компьютерная математика.* 2016. № 2. С. 142 – 150.
2. Шор Н.З. Методы минимизации недифференцируемых функций и их приложения. К.: Наук. думка, 1979. 200 с.
3. Стецюк П.И., Фишер А. *r*-Алгоритмы Шора и octave-функция ralgb5a. Тези Міжнародної наукової конференції *«Сучасна інформатика: проблеми, досягнення та перспективи розвитку»*, присвяченої 60-річчю заснування Інституту кібернетики імені В.М. Глушкова НАН України (м. Київ, 13 – 15 грудня 2017 р.). К.: Ін-т кібернетики імені В.М. Глушкова НАН України, 2017. С. 143 – 146.
4. Стецюк П.И. Теория и программные реализации *r*-алгоритмов Шора. *Кибернетика и системный анализ*. 2017. № 5. С. 43 – 57.
5. Стецюк П.И. Методы эллипсоидов и *r*-алгоритмы. Кишинэу. Эврика, 2014. 488 с.
6. Стецюк П.І., Фесюк О.В. Використання *r*-алгоритму для розв'язання квадратичної ELD-задачі. *Математичне та комп'ютерне моделювання*. Серія: фізико-математичні науки. 2017. № 15. С. 225 – 231.
7. ДП «Енергоринок» [Електронний ресурс]: http://www.er.gov.ua/.

8. ДП НЕК «Укренерго» [Електронний ресурс]: http://www.ukrenergo.energy.gov.ua/Pages/main.aspx.
9. Яндульський О.С., Стелюк А.О., Лукаш М.П. Автоматичне регулювання частоти та перетоків потужності в енергосистемах: навч. посіб. / під загальною редакцією д.т.н. О.С. Яндульського. Київ: НТУ України «КПІ», 2010. 88 с.
10. РД 34.25.504 (НР 34-70-113-86). Нормы предельно допустимых скоростей изменения нагрузки при работе энергоблоков 160–800 МВт в регулировочном диапазоне.
11. Стецюк П.І., Хіміч О.М., Сидорук В.А. Реалізація *r*-алгоритму на графічних процесорах. *Компьютерная математика*. 2016. № 2. С. 100 – 109.

Одержано 30.11.2017

П.И. Стецюк, А.В. Фесюк, В.А. Сидорук

АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ СЕПАРАБЕЛЬНОГО КВАДРАТИЧНОГО ПРОГРАММИРОВАНИЯ

Рассмотрена математическая модель задачи сепарабельного квадратичного программирования и методы решения задачи с помощью алгоритмов негладкой оптимизации. Описаны программные реализации методов на основе модификации *r*-алгоритма. Приведены результаты вычислительных экспериментов для решения квадратичных задач нахождения электрических нагрузок энергоблоков тепловых электростанций ОЭС Украины.

P.I. Stetsyuk, O.V. Fesiuk, V.A. Sidoruk

ALGORITHMS FOR SOLVING A SEPARABLE QUADRATIC PROGRAMMING PROBLEM

A mathematical model of the problem of separable quadratic programming and the methods for solving the problem using nonsmooth optimization algorithms are given. Software implementations of the methods based on modification of *r*-algorithm are described. Computational experiment results for the quadratic problems of finding the electrical loads for power units of thermal power plants of the Ukrainian IPS are presented.

Про авторів:

Стецюк Петро Іванович,
доктор фізико-математичних наук, завідувач відділу
Інституту кібернетики імені В.М. Глушкова НАН України,
E-mail: stetsyukp@gmail.com

Фесюк Олександр Володимирович,
провідний інженер-програміст
Інституту кібернетики імені В.М. Глушкова НАН України,
E-mail: sasha.fesyuk@gmail.com

Сидорук Володимир Антонович,
кандидат фізико-математичних наук, науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України.
E-mail: wolodymyr.sydoruk@gmail.com