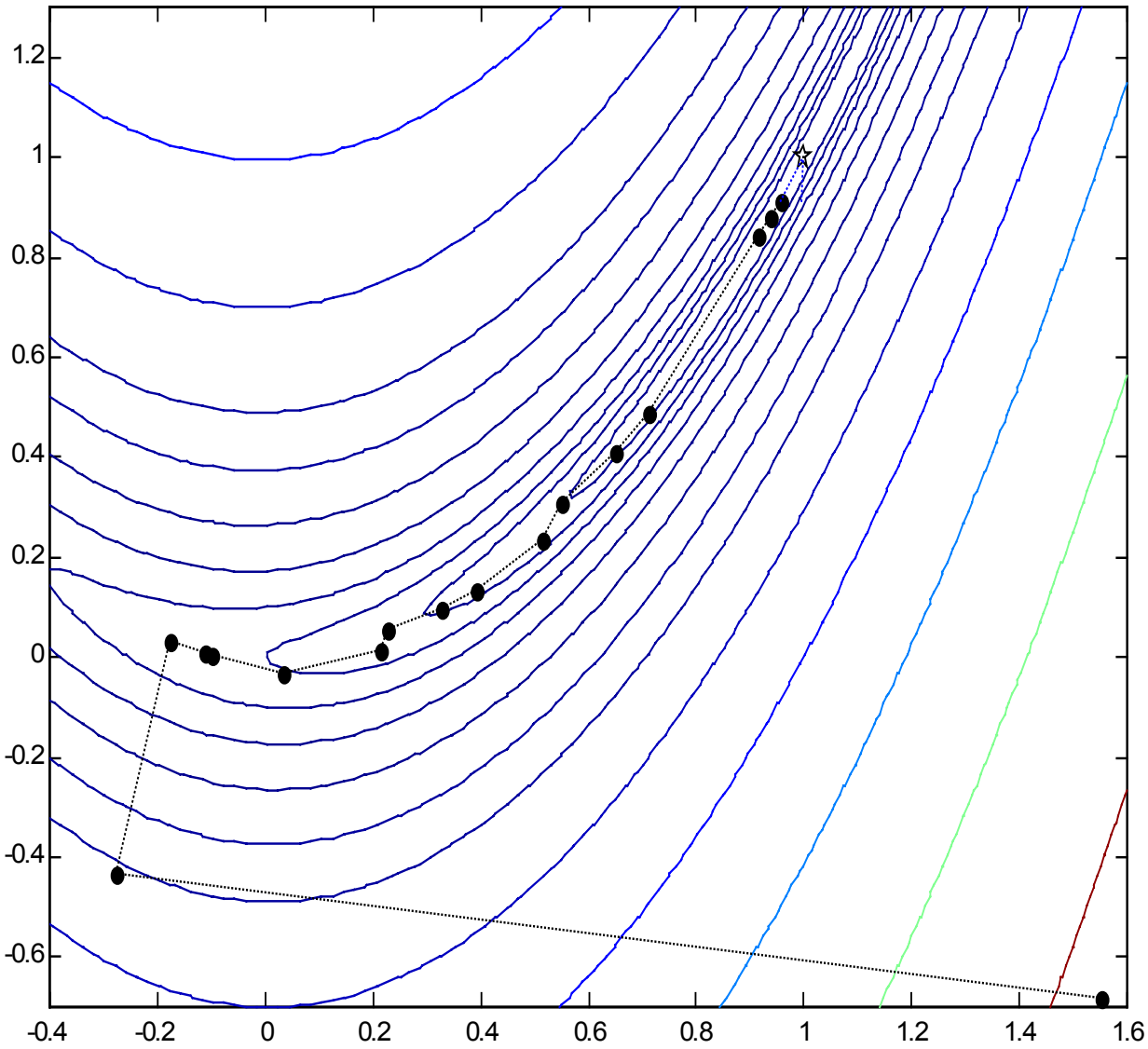
С. Д. ШТОВБА

Методи оптимізації в середовищі MatLab



Лабораторний практикум

УДК 519.7 (075)
Ш92

Р е ц е н з е н т и:
В. М. Михалевич, доктор технічних наук, професор
О. В. Бісікало, кандидат технічних наук, доцент
Д. І. Катєльніков, кандидат технічних наук

Рекомендовано до видання Ученою радою Вінницького державного технічного університету Міністерства освіти і науки України.

Штовба С.Д.
Ш92 **Методи оптимізації в середовищі MatLab.** Лабораторний практикум. Навчальний посібник. - Вінниця: ВДТУ, 2001.-56с.

В навчальному посібнику розглядаються питання, пов’язані з використанням методів розв’язання нелінійних безумовних та умовних задач оптимізації та задач лінійного програмування. Студентам пропонується провести порівняння ефективності різних методів оптимізації, виконати аналіз чутливості оптимального рішення та дослідити залежність часу оптимізації від розмірності задачі та якості розв’язку від кількості початкових точок. Програмною платформою для виконання лабораторних робіт є середовище MatLab.

Навчальний посібник написано для студентів вузів спеціальності «Системи управління і автоматики». Може бути корисним студентам спеціальностей «Інтелектуальні системи» та «Програмне забезпечення автоматизованих систем», науковим співробітникам та аспірантам, які використовують методи оптимізації в своїх дослідженнях.

УДК519.7 (075)

Зміст

Вступ

Необхідність постійного розв'язання задач оптимізації в інженерній діяльності обумовлює особливу позицію дисципліни «Методи оптимізації» в формуванні світогляду майбутніх інженерів. Незважаючи на велику кількість літератури по оптимізації студентам інженерних спеціальностей важко орієнтуватися в особливостях різнорідних задач, алгоритмах та методах. Це пов'язано з тим, що більшість книжок з оптимізації написано в стилі «математиками для математиків» з наголосом на строгості доведення тверджень та теорем, дослідженню збіжності методів та інших теоретичних аспектах. Найбільш цікавим для інженерів питанням практичного застосування математичного програмування та використання сучасних пакетів оптимізації приділено набагато менше уваги.

Головною метою навчального посібника є допомогти студентам набути практичних навичок використання математичного програмування і аналізу отриманих результатів та зорієнтуватися в особливостях оптимізації в пакеті MatLab. Матеріал посібника апробовано на протязі трирічного викладання дисципліни «Методи оптимізації» студентам факультету автоматики та комп'ютерних систем управління Вінницького державного технічного університету.

Для закріплення теоретичних положень та практичних навичок студентам пропонується після виконання кожної лабораторної роботи оформити звіт за такою схемою:

- титульний лист;
- індивідуальне завдання на виконання лабораторної роботи;
- необхідні теоретичні положення;
- результати комп'ютерного експерименту;
- обробка та узагальнення експериментальних даних;
- лістинги програм;
- висновки.

Автор висловлює подяку студенту групи 3АС97 Олексію Козачку за роботу по реалізації усіх комп'ютерних експериментів та оформлення рукопису.

Лабораторна робота №1

Дослідження залежності часу оптимізації від розмірності задачі

Теоретичні відомості

Оптимізація – це пошук найкращого рішення. В кожній задачі оптимізації використовуються такі поняття, як критерій, керовані змінні та цільова функція.

Критерій – це показник, який дозволяє визначити якість отриманого розв'язку задачі.

Керовані змінні – це такі параметри задачі, значення яких можна змінювати.

Цільова функція – це функція, що пов'язує керовані змінні та критерії таким чином, що дозволяє обчислити значення критерію при довільних значеннях керованих змінних.

Класична математична постановка задачі оптимізації полягає в тому, щоб знайти значення координат вектора керованих змінних $X = \{x_1, x_2, ..., x_n\}$, які забезпечують мінімум цільової функції

$$f(X) \to \min$$

та задовольняють обмеженням

$$g_i(X) \le 0, \quad i = \overline{1, r}$$

$$h_j(X) = 0, \quad j = \overline{1, s},$$

де $f(X)$ - цільова функція;

$g_i(X) \le 0 \ (i = \overline{1, r})$ - обмеження у вигляді нерівностей;

$h_j(X) = 0 \ (j = \overline{1, s})$ - обмеження у вигляді рівностей.

Задачі оптимізації класифікуються таким чином:

Скалярні та векторні

Якщо є лише один критерій, тоді ця задача відноситься до скалярної задачі оптимізації, а якщо є більше одного критерію, то це - задача векторної оптимізації. Усі реальні задачі є задачами векторної оптимізації. Відомі методи математичного програмування дозволяють коректно розв'язувати задачі скалярної оптимізації. Для того, щоб розв'зати задачу векторної оптимізації за допомогою методів математичного програмування, її перетворюють в задачу скалярної оптимізації. Існує три основних методи перетворення, що основані на:

1) введенні додаткових обмежень;
2) синтезі інтегрального критерію;
3) зміні масштабу постановки задачі.

Перший метод полягає в тому, що з множини критеріїв $W_1(X), W_2(X),\ldots, W_m(X)$ обирають один головний критерій, наприклад, $W_1(X)$, і прагнуть його обернути в максимум (мінімум), а решту критеріїв обмежують значеннями W_k. Тоді отримаємо скалярну задачу оптимізації з m-1 обмеженням:

$$W_1(X) \to \max,$$

$$W_k(X) \geq W_k\ ,\ k = \overline{2, m}$$

Недоліком цього методу є бінарне врахування неголовних критеріїв. В роботі [1] рекомендується розв'язувати подібні задачі методом "послідовних поступок". Припустимо, що критерії $W_k(X)$, $k = \overline{1, m}$ розташовані в порядку зменшення важливості. Спочатку шукається рішення, яке обертає в максимум перший (головний) критерій $W_1(X)= W_1^*$. Далі назначається деяка поступка ΔW_1, щоб максимізувати другий критерій $W_2(X)$. Накладемо на критерій $W_1(X)$ обмеження: зажадаємо, щоб він був не менше, ніж $W_1^* - \Delta W_1$, і при цьому обмеженні шукаємо рішення,

яке обертає в максимум $W_2(X)$. Далі знову назначаємо “поступку” в $W_2(X)$, ціною якої можна максимізувати $W_3(X)$ і т. д.

Другий метод використовується, коли нестачу одних критеріїв можна компенсувати надлишком решти. Міра “компенсації” визначається інтегральним критерієм. Найпростішим інтегральним критерієм є зважена сума часткових критеріїв:

$$W = \alpha_1 \cdot W_1 + \alpha_2 \cdot W_2 + ... + \alpha_n \cdot W_m,$$

де $\alpha_1, \alpha_2, \ldots, \alpha_m$ - вагові коефіцієнти, які відображають ступінь важливості відповідних критеріїв W_1, $W_2.,\ldots$, W_m. Недоліком методу є те, що вагові коефіцієнти не завжди вдається визначити об’єктивно, тому іноді їх доводиться назначати, виходячи з інтуїтивного представлення про ступені важливості критеріїв. Крім того в реальних задачах значення вагових коефіцієнтів може змінюватися в залежності від ситуації.

В роботі [1] це пояснюється таким прикладом. Людина з запізненням виходить з дому і міркує, яким транспортом скористатися, щоб потрапити на роботу без запізнення. Трамвай ходить часто, але їде повільно; автобус – швидше, але з великими інтервалами. Можна, звичайно взяти таксі, але це - дорого. Перед нами реальна задача з двома критеріями. Перший – середній очікуваний час запізнення W_1, який необхідно мінімізувати. Другий – очікувана вартість проїзду W_2, яку теж бажано зробити мінімальною. Тоді інтегральний критерій можна представити так:

$$W = \alpha_1 \cdot W_1 + \alpha_2 \cdot W_2$$

Але вагові коефіцієнти α_1, α_2 не можуть бути постійними. Наприклад, якщо людина отримала нещодавно догану за спізнення на роботу, тоді коефіцієнт при W_1 в нього, ймовірно, збільшиться, а на другий день після отримання зарплати, ймовірно, зменшиться коефіцієнт при W_2.

Третій метод полягає в зміні масштабу постановки задачі. Так при управлінні деяким підприємством виникає необхідність розглядати багато критеріїв. Якщо розширити масштаб постановки задачі, тобто в якості

об’єкта взяти всю галузь промисловості, то кількість критерій зменшиться, а в масштабі народного господарства всі критерії зводяться до одного глобального критерію – забезпечення мінімального часу, який потрібен для досягнення мети – необхідного рівня національного доходу держави;

Умовні та безумовні

Якщо існують обмеження на значення керованих змінних, тоді задача оптимізації є умовною, в протилежному випадку – безумовною;

Лінійні та нелінійні

Якщо цільова функція і всі обмеження – лінійні, то це задача лінійної оптимізації, а якщо цільова функція або хоча б одне обмеження нелінійне – то це задача нелінійної оптимізації;

Одновимірні та багатовимірні

Якщо вектор керованих змінних має лише одну координату, тоді задача оптимізації є одновимірною, в протилежному випадку – багатовимірною;

Одноекстремальні та багатоекстремальні

Одноекстремальні задачі – це такі задачі, в яких цільова функція f(x) на області допустимих значень має один екстремум, а в багатоекстремальних задачах оптимізації цільова функція f(x) на області допустимих значень має декілька екстремумів;

Неперервні та дискретні

Якщо керовані змінні задані на неперервній допустимій множині, тоді задача оптимізації є неперервною, а якщо на дискретній – то це задача дискретної оптимізації.

Збільшення кількості керованих змінних (розмірності) істотно ускладнює її рішення. А при деякому значенні починається різке зростання часу обчислення оптимуму (рис. 1). Таке явище в оптимізації називається проблемою “прокляття розмірності”.

Однією з задач сучасних методів оптимізації є розробка ефективних алгоритмів, що дозволяють відсунути стіну складності (рис. 1). В роботі [8] *ефективним алгоритмом* вважається такий алгоритм, складність якого (кількість ітерацій) описується поліноміальною функцією від розмірності задачі.

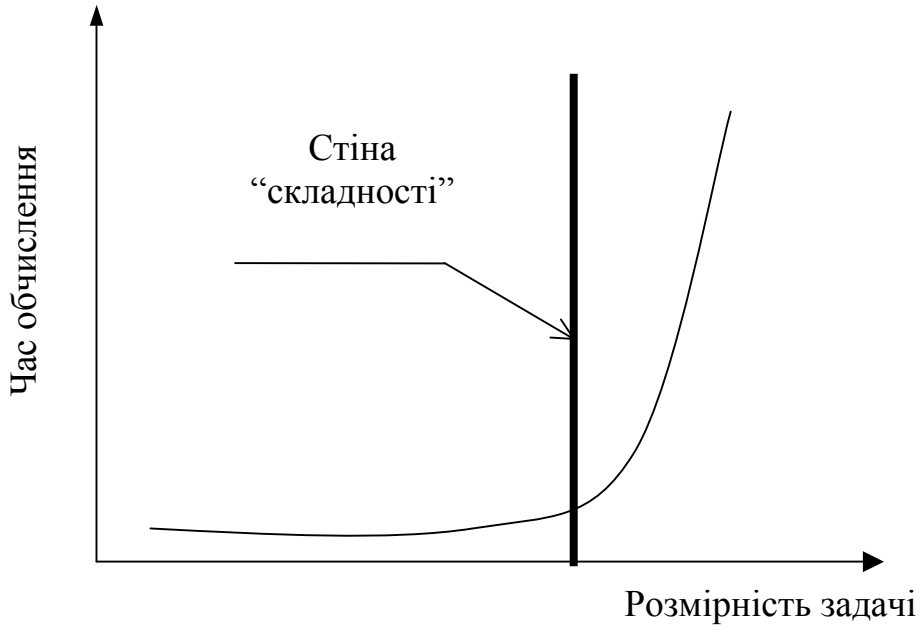


Рисунок 1 – Проблема “прокляття розмірності”

Завдання на роботу

1. Експериментально дослідити залежність часу розв’язання задачі оптимізації від кількості керованих змінних.
2. Підібрати аналітичну залежність, яка найкращим чином апроксимує отримані експериментальні дані.

Таблиця 1 – Варіанти завдань

Варіант	Цільова функція	Початкова точка	План експерименту
1	$y = \lvert x_1 + 5\rvert + \sum_{i=2}^{n}(x_i - i)^2$	$x_i^0 = i/2$ $i = \overline{1, n}$	n=2,4,6,8,10,15 20,25,30
2	$y = 100x_1^2 + \sum_{i=2}^{n}\lvert x_i - 2i\rvert$	$x_i^0 = 12$ $i = \overline{1, n}$	n=2,4,6,8,10,12 15,20,25
3	$y = \sum_{i=1}^{n}\frac{(x_i + i)^{2i}}{100i}$	$x_i^0 = i/n - 15$ $i = \overline{1, n}$	n=2,3,4,5,6,8, 10, 12, 15
4	$y = (x_1 - n)^6 + 0.2\lvert x_2 + 5\rvert + + \sum_{i=3}^{n}\frac{(x_i + 7i)^2}{i}$	$x_i^0 = 0$ $i = \overline{1, n}$	n=2,3,4,6,8,10, 15,20,25
5	$y = x_1^4 + \lvert(x_2 - 3)^3\rvert + \sum_{i=3}^{n}\frac{(x_i + 2i)^2}{i^2}$	$x_i^0 = 12$ $i = \overline{1, n}$	n=2,4,6,8,10,15 20,25,30
6	$y = \sum_{i=1,3,...,n}(x_i - 2i)^2 + \sum_{j=2,4,...,n}(x_j - 2j)^4$	$x_i^0 = 0$ $x_j^0 = 0$	n=2,3,4,5,6,8, 10, 15, 22
7	$y = (x_1 + n)^4 + \sum_{i=2}^{n}(x_i - i)^2$	$x_i^0 = i/2$ $i = \overline{1, n}$	n=2,3,4,6,8,10, 15,20,25
8	$y = \sum_{i=1}^{n}\frac{(i \cdot x_i - n)^{2i}}{10i^3}$	$x_i^0 = i/10$ $i = \overline{1, n}$	n=2,3,4,5,6,8, 10, 12, 15

Продовження табл. 1

Варіант	Цільова функція	Початкова точка	План експерименту
9	$y = \sum_{i=1,3,...,n}(0.1x_i + 2i)^2 + \sum_{j=2,4,...,n}(x_j - 11n)^4$	$x_i^0 = 3,$ $x_j^0 = 3$	n=2,3,4,6,8,10, 15,20,25
10	$y = (x_1 - 67)^{2n} + \sum_{i=2}^{n} \frac{(x_i + 7i)^2}{i^3}$	$x_i^0 = n/2$ $i = \overline{1, n}$	n=2,3,4,6,8,9, 10, 12,15
11	$y = (x_n - 75)^4 + \sum_{i=1}^{n-1} \frac{(x_i - i)^2}{i^3}$	$x_i^0 = i$ $i = \overline{1, n}$	n=2,3,4,6,8,10, 15,20,25
12	$y = 500x_1^6 + \sum_{i=2}^{n} \lvert x_i - 3i \rvert$	$x_i^0 = i/2$ $i = \overline{1, n}$	n=2,4,6,8,10,15 20,25,30
13	$y = (x_1 + 7)^8 + \sum_{i=2}^{n} \frac{(0.01x_i - i)^2}{i}$	$x_i^0 = n$ $i = \overline{1, n}$	n=2,3,4,6,8,10, 15,20,25
14	$y = 0.01(x_1 - n)^6 + \sum_{i=2}^{n}(x_i + i)^2$	$x_i^0 = 12$ $i = \overline{1, n}$	n=2,3,4,6,8,10, 15,20,25
15	$y = \sum_{i=1}^{n} \frac{(x_i + i)^{2i}}{10i^3}$	$x_i^0 = i$ $i = \overline{1, n}$	n=2,3,4,5,6,8, 10, 12,15
16	$y = \sum_{i=1,3,...,n}(x_i + 2i)^2 + \sum_{j=2,4,...,n}(x_j - 12j)^4$	$x_i^0 = 0,$ $x_j^0 = 0$	n=2,3,4,5,6,8, 10, 15,20
17	$y = (x_1 - 67)^{2n} + 0.1\lvert x_2 + 5 \rvert + + \sum_{i=3}^{n} \frac{(x_i + 7i)^2}{i \cdot n}$	$x_i^0 = i$ $i = \overline{1, n}$	n=2,3,4,5,6,8, 10, 12,15

Вимоги до виконання роботи

1. Початкові дані для обчислювального експерименту вибрати з табл.1 згідно з варіантом.
2. Експерименти проводити в програмному середовищі MatLab з використанням бібліотеки Optimization.
3. Задачи оптимізації розв’язати, застосовуючи функцію fminu.
4. Час розв’язання задачі оптимізації визначити за допомогою функцій tic та toc.
5. Оптимальні значення параметрів апроксимувальної функції підібрати за методом найменших квадратів.
6. При побудові графічних зображень використовувати функцію plot.
7. Оформлення результатів обробки експерименту необхідно представити у вигляді графіка, як показано на рис. 2.

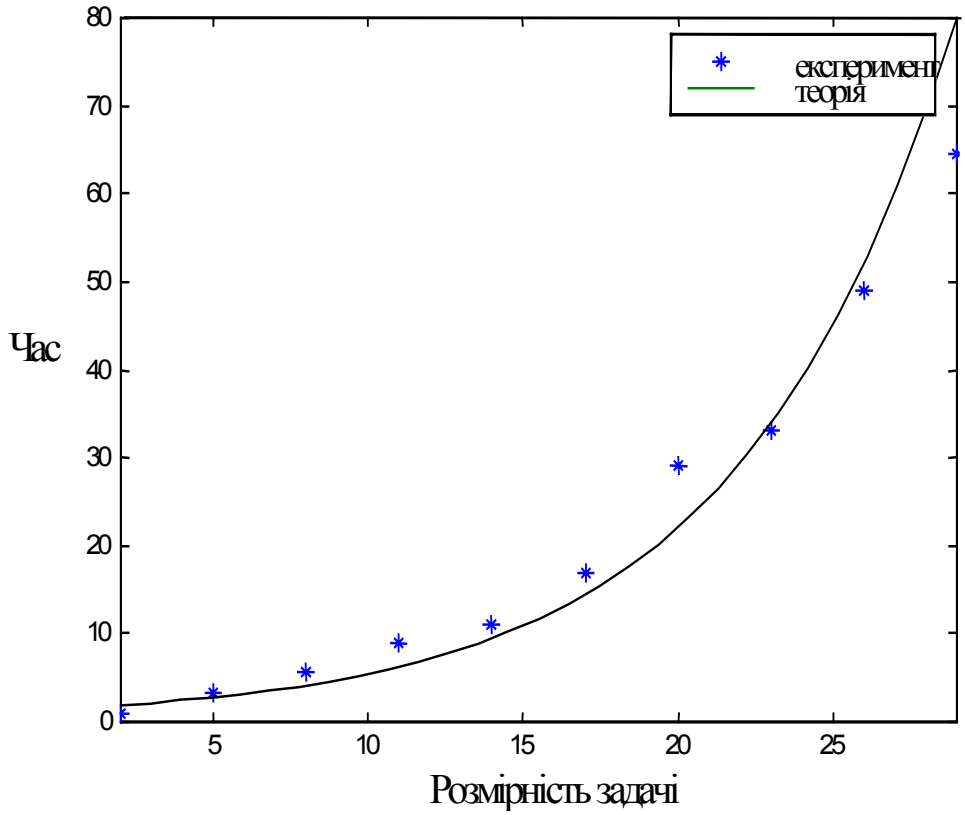


Рисунок 2 – Залежність часу оптимізації від розмірності задачі

Рекомендації по виконанню роботи та приклади

1. В даній лабораторній роботі необхідно дослідити залежність часу розв'язування задачі оптимізації від її розмірності. Ця задача відноситься до багатовимірної безумовної оптимізації, тому що кількість керованих зміних завжди більша однієї, а обмежень немає. Для розв'язування цієї задачи необхідно звернути увагу на максимальну кількість ітерацій, яка відводиться на розв'язання задачі. При збільшенні кількості керованих змінних може статися так, що для розв'язання задачі не вистачить ітерацій. Для того, щоб уникнути такої ситуації, необхідно збільшити значення options(14) (Дивись додаток А).

У другій частині лабораторної роботи необхідно підібрати таку криву залежності часу оптимізації від кількості керованих змінних, яка б найкращим чином описувала експериментальні дані. Експериментальні дані необхідно апроксимувати за методом найменших квадратів. В цьому випадку задача апроксимації ототожнюється з такою задачею оптимізації:

Знайти такі значення параметрів (φ_1, φ_1,…, φ_N) апроксимувальної функції $F(n,\varphi_1, \varphi_1,\ldots, \varphi_N)$, щоб

$$\sum_{r=1}^{b}(t_r - F(n_r,\varphi_1,\varphi_2,...,\varphi_n))^2 \to \min,$$

де r – номер експерименту;

t_r – час оптимізації за r-й експеримент;

n_r – розмірність задачи оптимізації в r-му експерименті;

$F(n_r,\varphi_1, \varphi_1,\ldots, \varphi_N)$ – теоретична апроксимувальна функція;

b – кількість експериментів.

2. Приклади та пояснення щодо розв'язання задач оптимізації в пакеті MatLab можно знайти в електронному довіднику, для запуску якого необхідно виконати такі дії. Demo → Optimization Toolbox → Command Line Demos → Tutorial. Основні функції, що входять в склад Optimization Toolbox, наведені також в додатку А.

3. Приклад програми для знаходження мінімуму функції:

$$y = |x_1 + 5| + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2$$

```
f='abs(x(1)+5)+(x(2)-2)^2+(x(3)-3)^2+(x(4)-4)^2';   % цільова функція
x0=[0 5 10 15];                                      % початкова точка
options=[];                                          % параметри оптимізації
tic                                                  % ввімкнення таймера
[Xopt options]=fminu(f,x0,options);                  % оптимізація
toc                                                  % вимкнення таймера
options(8)                                           % значення функції в точці
                                                     % Xopt
Xopt                                                 % координати оптимуму
```

Запитання для самоконтролю

1. Як перетворити задачу пошуку мінімуму в задачу пошуку максимуму?
2. За якими ознаками класифікують задачі оптимізації?
3. Як залежить час оптимізації від розмірності задачі?
4. Чи завжди при збільшення розмірності задачі збільшується час оптимізації?
5. Що таке ефективний алгоритм оптимізації?
6. Які чинники впливають на час розв’язання задачі оптимізації?

Лабораторна робота №2

Порівняльний аналіз методів безумовної оптимізації функції багатьох змінних

Теоретичні відомості

В даній лабораторній роботі розглядається задача знаходження мінімуму цільової функції f(X) при відсутності обмежень:

$$f(X) \to \min,$$

де $X = (x_1, x_2, ..., x_n)$ - вектор керованих змінних.

Дана задача відноситься до безумовної оптимізації. Методи, які застосовують для розв'язування задач безумовної оптимізації, можна поділити на три широких класи:

1) методи нульового порядку, які базуються на обчисленні тільки цільової функції;

2) методи першого порядку, в яких використовується значення частинних похідних першого порядку;

3) методи другого порядку, які використовують значення частинних похідних другого порядку.

До **методів нульового порядку** відноситься пошук за симплексом (не плутати з симплекс методом в лінійному програмуванні), метод покоординатного спуску Гауса-Зейделя, випадковий пошук та інші. Розглянемо основні ідеї означених методів.

Пошук за симплексом [4] полягає в тому, що при пошуку оптимуму в просторі незалежних змінних будується регулярний симплекс і обчислюються значення цільової функції в його вершинах. Регулярний симплекс в n-вимірном просторі представляє собою багатогранник, який має n+1 рівновіддалених вершин. Наприклад, у випадку двох змінних

симплексом є рівнобічний трикутник; в тривимірному просторі симплекс представляє собою тетраедр. Після побудови симплекса визначається вершина, у якій цільова функція має найбільше значення. На рис. 3 це вершина з номером 1.

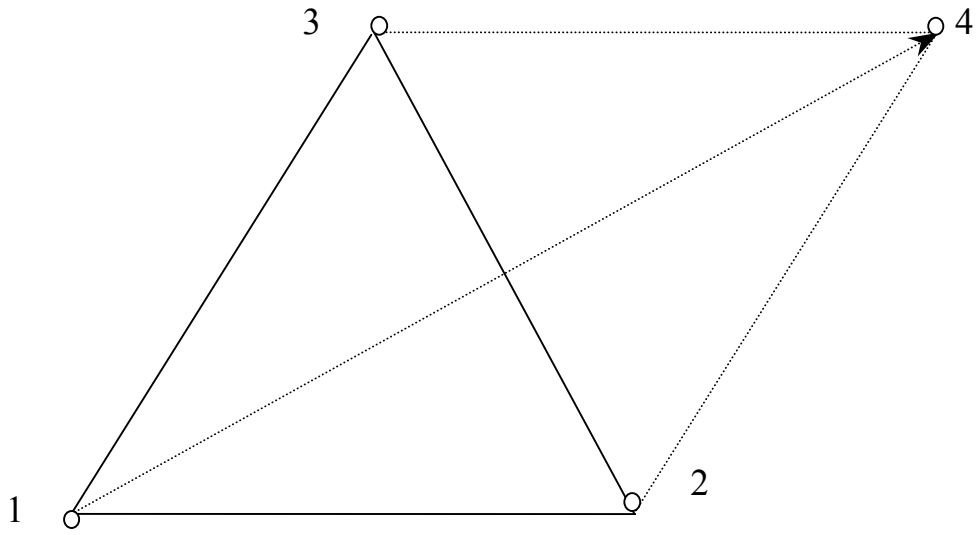


Рисунок 3 – Ілюстрація ідеї симплексного методу

Далі знайдена вершина проекціюється через центр ваги інших вершин симплекса (точки 2 і 3) в нову вершину (точка 4). Точки 2, 3 та 4 використовуються в якості вершини нового симплекса. Таким чином, трикутник немов би перевертається через сторону з найменшим значенням цільової функції. При пошукові мінімуму використовуються такі два правили.

Правило “накриття” точки мінімуму: якщо вершина, якій відповідає найбільше значення цільової функції, побудована на попередній ітерації, то замість неї береться вершина з меншим значенням цільової функції.

Правило циклічного руху: якщо деяка вершина симплекса не виключається протягом багатьох ітерацій, то необхідно зменшити розмір симплекса.

Пошук завершується, коли розмір симплекса або різниця між значеннями функції в вершинах симплекса стає достатньо малою.

Недоліком цього методу є велика кількість ітерацій; крім того він не завжди забезпечує розв'язування задачі.

Ідея методу *покоординатного спуску* [2] полягає в тому, що спочатку робиться пробний крок в напрямі, що паралельний одній з координатних осей і обчислюється значення цільової функції. Якщо значення цільової функції зменшується, то рух продовжується далі в цьому ж напрямку, а якщо функція збільшується, то ми повертаємося назад і робимо пробний крок в іншому напрямку. І так продовжується до тих пір, доки не буде знайдена оптимальна точка (рис. 4). Особливість цього методу полягає в тому, що пошук оптимуму проводиться виключно паралельно координатним осям. На початку пошуку вибирається великий крок і перевіряється значення функції по всіх напрямках. Якщо буде зростання функції по всіх напрямках, то необхідно зменшити крок.

Недоліком цього методу є обмежені можливості при пошукові оптимуму. Метод застосовують тоді, коли залежність між змінними x_1, x_2,…, x_n практично відсутня.

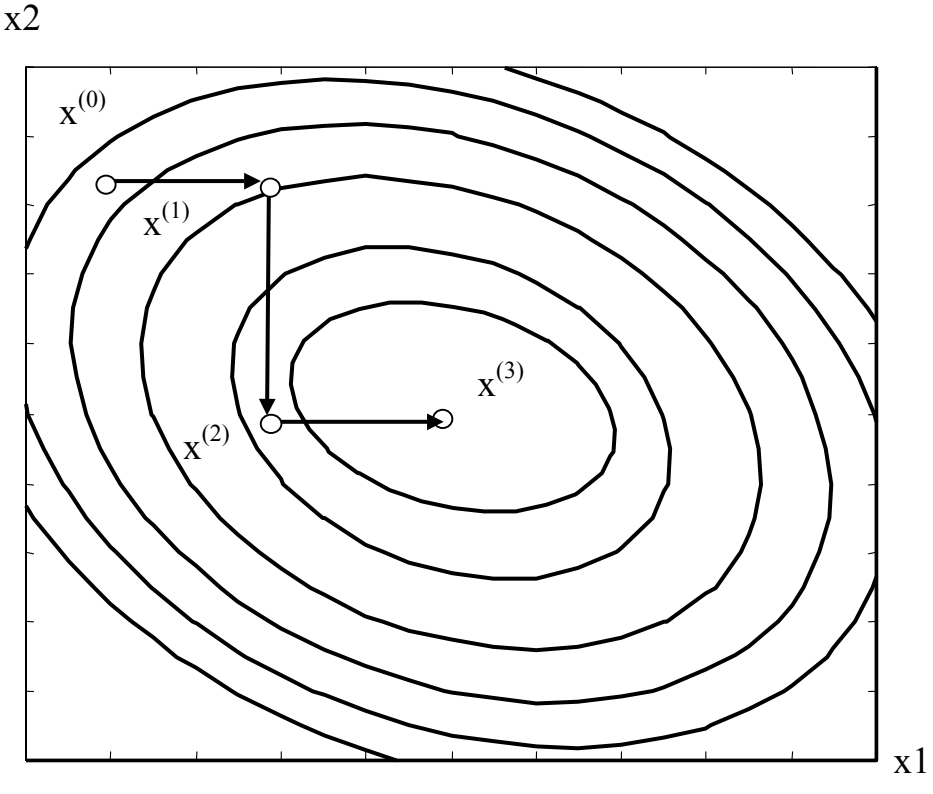


Рисунок 4 – Ілюстрація покоординатного спуску

Ідея *випадкового пошуку* [4] полягає в тому, що вибір напрямку руху здійснюється випадково. Якщо цільова функція зменшується, то рух у вибраному напрямку продовжується, в протилежному випадку необхідно повернутися на один крок назад та знову випадково вибрати напрямок пошуку і т.д.

Усі випадкові методи пошуку реалізуються за ітераційною формулою:

$$X^{(k+1)} = X^{(k)} + \xi^{(k)},$$

де k – номер ітерації;

$\xi^{(k)}$ - випадкова величина.

Популярність методів випадкового пошуку пояснюється їхньою простотою та широкими можливостями для користувачей самому модифікувати методи.

До **методів першого порядку** відносяться градієнтний метод Коші, метод найшвидшого спуску та інші. Особливість цих методів – це використання градієнта. *Градієнтом функції* f(X) називається вектор, величина якого визначає швидкість зміни функції f(X), а напрямок збігається з напрямком найбільшого зростання цієї функції. Вектор, який протилежний за напрямком градієнту, називається *антиградієнтом*.

Всі методи першого порядку базуються на ітераційній процедурі, яка реалізується за формулою:

$$X^{(k+1)} = X^{(k)} + \lambda^{(k)} \cdot \nabla f(X^{(k)}),$$

де k - номер ітерації;

$X^{(k)}$ – поточне наближення до розв’язку;

$\lambda^{(k)}$ –параметр, який характеризує довжину кроку;

$\nabla f(X^{(k)})$ – напрямок пошуку в n-вимірному просторі керованих зміних.

Градієнтний метод [5] представляє собою послідовність кроків, кожний з яких складається з двох операцій:

1) визначення напрямку найбільшої крутизни спуску, тобто напрямок антиградієнта функції f(X);

2) переміщення в вибраному напрямку на задану відстань.

Характер траєкторій при градієнтному методі показано на рис 5. Особливість траєкторії полягає в тому, що кожний крок здійснюється в напрямку, який перпендикулярний лініям однакового рівня (ізолініям) цільової функції.

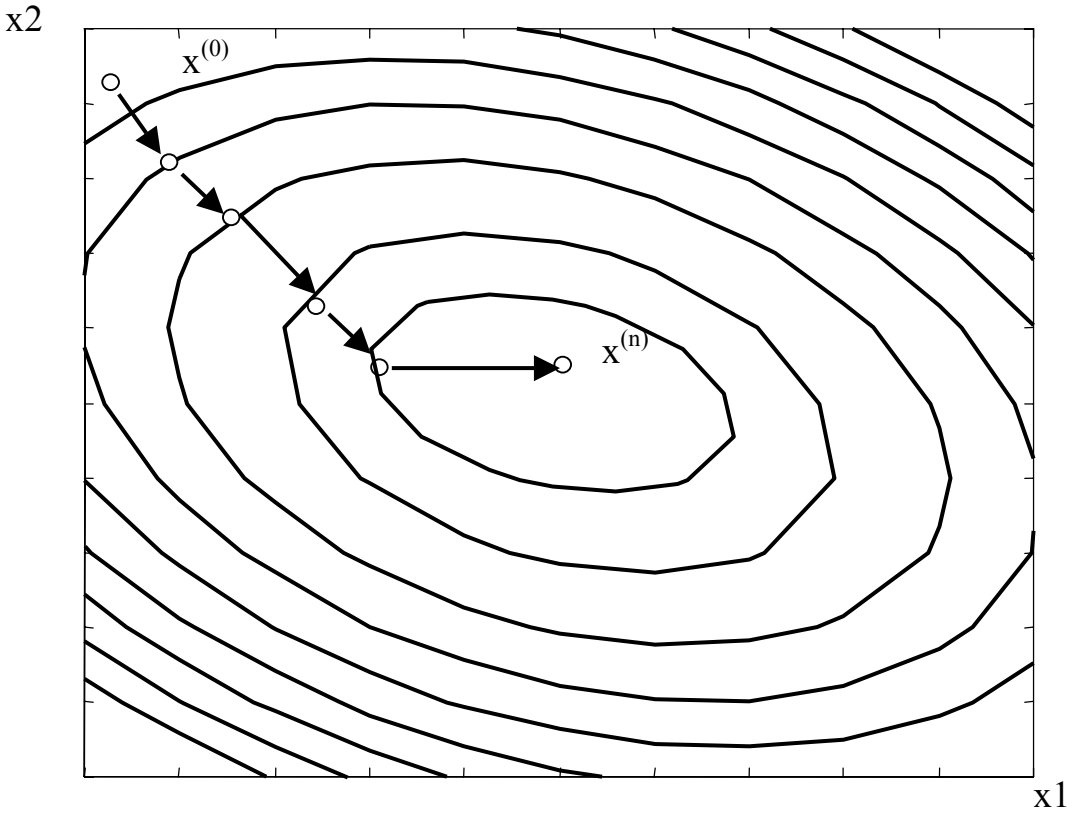


Рисунок 5 – Ілюстрація градієнтного методу

Градієнтний метод має свої недоліки. Для того, щоб рухатися завжди в напрямку антиградієнта, крок повинен бути невеликим. Тому їх буде багато. Але поскільки на кожному кроці необхідно постійно обчислювати градієнт, то наближення до оптимуму буде повільним.

Метод *найшвидшого спуску* [4] є модифікацією градієнтного методу. Він відрізняється від градієнтного тим, що градієнт обчислюється тільки в початковій точці і рух в напрямку антиградієнта продовжується до тих пір, поки зменшується значення цільової функції f(X). Вибір кроку λ при переході від точки $X^{(k)}$ в $X^{(k+1)}$ визначається з умови:

$$\lambda^{(k)} = \min_{\lambda \geq 0} f(X^{(k)} + \lambda^{(k)} \cdot \nabla f(X^{(k)})),$$

тобто на кожному кроці розв'язується одновимірна задача мінімізації.

Метод найшвидшого спуску має два недоліки: по-перше, методу властива поступова збіжність до точки мінімуму внаслідок малого $\nabla f(X)$ в околі цієї точки; по-друге, необхідно розв'язувати задачу одновимірної оптимізації – обирати на кожному кроці оптимальне значення $\lambda^{(k)}$. Головна перевага цього методу полягає в тому, що йому властива стійкість та простота обчислень. Використовують цей метод тоді, коли цільову функцію можна добре апроксимувати лінійною залежністю.

До методів **другого порядку** відносять метод Гауса-Ньютона, модифіковані ньютонівські методи, метод Левенберга-Марквардта та інші.

Якщо методи першого порядку використовують значення перших похідних цільової функції, тобто базуються на послідовній лінійній апроксимації цільової функції, то *метод Гауса-Ньютона* [4] використовує квадратичну апроксимацію з відкиданням в ряду Тейлора членів третього і більш високого порядку:

$$f(X^{(k+1)}) \approx f(X^{(k)}) + \nabla f(X^{(k)}) \cdot [\Delta X^{(k)}]^T + \frac{1}{2} \cdot \Delta X^{(k)} \cdot H(X^{(k)}) \cdot [\Delta X^{(k)}]^T, \qquad (1)$$

де T – знак транспонування;

$\nabla^2 f(X^{<k>})$ – матриця других частинних похідних, яка називається матрицею Гессе або гессіаном $H(X^{<k>})$.

$\Delta X^{(k)}$ – крок оптимізації.

Мінімізуючи вираз (1) по $\Delta X^{(k)}$, отримаємо:

$$\Delta X^{(k)} = -\nabla f(X^{(k)}) \cdot H^{-1}(X^{(k)}).$$

Тоді ітераційна формула для методу Гауса-Ньютона набуває вигляду:

$$X^{(k+1)} = X^{(k)} - \nabla f(X^{(k)}) \cdot H(X^{(k)})^{-1}.$$

Використання методу Гауса-Ньютона дозволяє знайти мінімум квадратичних функцій за одну ітерацію (при будь-якій початковій точці).

У випадку неквадратичних функцій метод Гауса-Ньютона не відрізняється надійністю. Цього недоліку позбавлені модифіковані ньютонівські методи, в яких забезпечено зменшення цільової функції від ітерації до ітерації, шляхом пошуку вздовж прямої, як в методі найшвидшого спуску. Математично це можна реалізувати за формулою

$$X^{(k+1)} = X^{(k)} - \lambda^{(k)} \cdot H(X^{(k)})^{-1} \cdot \nabla f(X^{(k)}).$$

Вибір $\lambda^{(k)}$ здійснюють таким чином, щоб $f(X^{(k+1)}) \to \min$.

Метод *Левенберга-Марквардта* [7] являє собою комбінацію методів Коші та Ньютона, в якій добре поєднується позитивні властивості обох методів. Метод реалізується за формулою:

$$X^{(k+1)} = X^{(k)} - [H(X^{(k)}) + \lambda^{(k)} \cdot I)]^{-1} \cdot \nabla f(X^{(k)}), \qquad (2)$$

де I – одинична матриця.

На початку пошуку в формулі (2) параметру $\lambda^{(0)}$ – задається велике значення, тоді

$$[H(X^{(0)}) + \lambda^{(0)} \cdot I)]^{-1} = [\lambda^{(0)} \cdot I]^{-1} = \frac{1}{\lambda^{(0)}} \cdot I,$$

і метод Левенберга-Марквардта працює як метод Коші, а при зменшенні λ до нуля метод наближається до методу Ньютона.

Головним недоліком методів другого порядку є необхідність обчислення гессіана на кожному кроці.

Найбільш поширені методи безумовної оптимізації – **квазіньютонівські методи** [4]. До квазіньютонівських методів відносяться методи Давидона-Флетчера-Пауела, Бройдена-Флетчера-Гольдарба-Шенно та інші. В основі цих методів лежить ідея апроксимації гессіана на основі градієнта.

Квазіньютонівські методи реалізуються за формулою:

$$X^{(k+1)} = X^{(k)} - \lambda^{(k)} \cdot \nabla f(X^{(k)}) \cdot A(X^{(k)}),$$

де $A(X^{<k>})$ – метрика, яка апроксимує $H^{-1}(X^{(k)})$.

В квазіньютонівських методах матриця $A(X^{(k+1)})$ обчислюється на основі метрики попереднього кроку $A(X^{(k)})$, тобто

$$A(X^{(k+1)}) = A(X^{(k)}) + \Delta A(X^{(k)}),$$

де $\Delta A(X^{(k)})$ – корегувальна матриця; на першому кроці $A(X^{(k)})=H^{-1}(X^{(k)})$.

Різні квазіньютонівські методи відрізняються тільки способом переобчислення матриці $\Delta A(X^{<k>})$.

Завдання на роботу

1. Розв'язати задачу оптимізації такими методами:
   - пошук за симплексом;
   - найшвидшого спуску;
   - Гауса-Ньютона;
   - Левенберга-Марквардта;
   - Давидона-Флетчера-Пауела;
   - Бройдена-Флетчера-Гольдарба-Шенно.

2. Провести порівняльний аналіз ефективності різних методів оптимізації.

Вимоги до виконання роботи

1. Цільову функцію та початкову точку вибрати з табл.1 згідно з варіантом.
2. Експерименти проводити для n=5.
3. Експерименти проводити в програмному середовищі MatLab з використанням бібліотеки Optimization.
4. Час розв’язання задачі оптимізації визначити за допомогою функцій tic та toc.
5. Для розв'язання задач оптимізації використовувати функціїї, що наведені в табл. 2.

Таблиця 2 – Основні функції для задач безумовної оптимізації

Метод	Функція	Options(5)	Options(6)	Options(7)
Пошук за симплексом	Fmins	0	0	0
Найшвидшого спуску	Fminu	0	2	0
Гаусса-Ньютона	Leastsq	1	0	0
Левенберга-Марквардта	Leastsq	0	0	0
Давидона-Флетчера-Пауела	Fminu	0	1	0
Бройдена-Флетчера-Гольдарба-Шенно	Fminu	0	0	0

6. Результати обробки експерименту оформити таким чином:

Метод	Координата оптимуму	Значення цільової функції в оптимумі	Кількість ітерацій	Час оптимізації	Висновок

Запитання для самоконтролю

1. Як ставиться задача безумовної оптимізації?
2. Яка особливість методів нульового порядку?
3. Чим методи нульового порядку відрізняються від методів першого та другого порядку?
4. Що таке градієнт та антиградієнт?
5. Чому квадратична функція використовується як основа для побудови алгоритмів нелінійної оптимізації?

6. Поясніть зв’язок методу Левенберга-Марквардта з методами Коші та Ньютона?
7. Чим відрізняються між собою квазіньютонівські методи?

Лабораторна робота №3

Аналіз чутливості оптимального рішення задачі лінійного програмування

Теоретичні відомості

Задачами лінійного програмування називають оптимізаційні задачи, в яких обмеження і цільова функція є лінійними.

Розглянемо задачу лінійного програмування, в якій необхідно знайти такі значення змінних x_1, x_2,…, x_n, які обертають в максимум цільову функцію $Z = \sum_{j=1}^{n} c_j \cdot x_j$ при обмеженнях:

$$\begin{array}{ccccccccc} a_{11}\cdot x_1 & + & a_{12}\cdot x_2 & + & \ldots & + & a_{1n}\cdot x_n & \le & b_1 \\ a_{21}\cdot x_1 & + & a_{22}\cdot x_2 & + & \ldots & + & a_{2n}\cdot x_n & \le & b_2 \\ \ldots. & & \ldots & & \ldots & & \ldots & & \ldots \\ a_{m1}\cdot x_1 & + & a_{m2}\cdot x_2 & + & \ldots & + & a_{mn}\cdot x_n & \le & b_m. \end{array} \qquad (3)$$

Грунтуючись на роботі [2], наведемо визначення, які необхідні для формулювання головних властивостей задач лінійного програмування.

Відрізком, який з'єднує дві довільні точки X_1 і X_2, називається множина точок з координатами $x_j = a \cdot x_{j1} + (1-a) \cdot x_{j2}$ $(0 \le a \le 1, j = \overline{1, n})$, де x_{j1}, x_{j2} – координати точок X_1 і X_2, відповідно; a – параметр, який визначає положення X.

Область називається *опукла*, якщо разом з двома довільними точками області вона містить весь відрізок, який з'єднує ці точки.

Точка X, яка належить випуклій області, називається *крайньою*, якщо в даній області немає двох таких точок $X_1 \neq X_2$, що $x_j = a \cdot x_{j1} + (1-a) \cdot x_{j2}$ $(0 \le a \le 1, j = \overline{1, n})$.

Нижче наведено дві теореми, які відображають важливі властивості лінійних задач:

теорема 1: екстремум цільової функції в задачах лінійного програмування (якщо він існує) завжди є глобальним;

теорема 2: якщо множина оптимальних розв'зків задачи лінійного програмування існує, тоді вона завжди містять хоча б одну крайню точку опуклої області.

Для розв'язання задач лінійного програмування зазвичай використовують симплекс-метод [1]. Застосування симплекс-методу вимагає представлення обмежень в стандартній формі:

$$\begin{array}{ccccccccccc} a_{11}\cdot x_1 & + & a_{12}\cdot x_2 & + & \ldots & + & a_{1n}\cdot x_n & + & x_{n+1} & = b_1 \\ a_{21}\cdot x_1 & + & a_{22}\cdot x_2 & + & \ldots & + & a_{2n}\cdot x_n & + & x_{n+2} & = b_2 \\ \ldots & & \ldots & & \ldots & & \ldots & & \ldots & \ldots \\ a_{m1}\cdot x_1 & + & a_{m2}\cdot x_2 & + & \ldots & + & a_{mn}\cdot x_n & + & x_{n+m} & = b_m \end{array}$$

$$x_i \geq 0, \quad i = \overline{1, n+m},$$

$$b_j \geq 0, \qquad j = \overline{1, m}.$$

Для розв'язування задачі за симплексом необхідно перейти від стандартної форми до канонічної [7], в якій m змінних входять з одиничними коефіцієнтами лише в одне рівняння системи, а врешту – з нулевими. Ці змінні називають *базисними*, а решта (n-m) називають *небазисними.*

Наведемо основні визначення, які використовуються в симплекс-методі:

Допустимим розв'язком називають невід'ємний вектор X, для якого виконуються обмеження AX=b,

де A – матриця коефіцієнтів a_{ij} в (3), b – вектор-стовпець вільних членів.

Допустимою областю називається множина всіх допустимих розв'язків: $S = \{X \mid A \cdot X = b\}$.

Оптимальним розв'яком називається такий допустимий вектор X°, для якого відповідні йому значення цільової функції f(X°) більші, ніж для будь-якого іншого допустимого розв'язку.

Оптимальним значенням задачі лінійного програмування називають значення цільової функції в точці X^o.

Неєдністю оптимального розв'язку називається випадок, коли задача лінійного програмування має багато оптимальних розв'язків.

Єдністю оптимуму називається випадок, коли оптимальний розв'язок задачі лінійного програмування є єдиним.

Необмеженістю оптимуму називають випадок, коли задача лінійного програмування не має кінцевого оптимуму, тобто $\max Z \to +\infty$ або $\min Z \to -\infty$.

Базисним розв'язком системи в канонічному вигляді називаються розв'язки, який отримується при нульових значеннях небазисних змінних.

Базисним розв'язком називається *допустимим*, якщо значення усіх базисних змінних є невід'ємними.

Симплекс-метод реалізується за таким алгоритмом:

Крок 1. Вибір початкового допустимого базисного розв'язку.

Крок 2. Перехід від початкового розв'язку до іншого допустимого базисного розв'язку з кращим значенням цільової функції. На цьому кроці вилучаються з розгляду всі допустимі базисні розв'язки, які гірші поточного.

Крок 3. Продовження пошуку допустимих базисних розв'язків, які покращують значення цільової функції. Якщо деякий допустимий базисний розв'язок неможливо покращити, то він є оптимальним і алгоритм симплекс-методу завершується.

Більш детально з симплекс-методом можна ознайомитися в [7].

Аналіз чутливості це – процедура, яка дозволяє встановити залежність оптимального розв'язку до варіації початкових даних. Аналіз

чутливості відіграє велику роль в задачах оптимізації. Аналіз чутливості потрібно проводити з двох причин:

1. Деякі параметри задач лінійного програмування, такі як фінансові кошти, запаси ресурсів можно регулювати. Аналіз чутливості дозволяє оцінити вплив зміни цих параметрів на оптимальний розв'язок. Якщо виявиться, що оптимальний розв'язок (наприклад, відношення прибутку до витрат) можна значно покращити за рахунок невеликих змін параметрів, то необхідно провести ці зміни.

2. В багатьох випадках оцінки параметрів отримуються шляхом статистичної обробки екпериментальних даних. Тому такі оцінки не можуть бути точними. Якщо вдається визначити, які параметри більшою мірою впливають на значення цільової функції в точці оптимуму, то необхідно збільшити точність їх оцінок.

Важливу роль при аналізі чутливості виробничих задач відіграють тіньові ціни та маргінальні оцінки. Тіньові ціни визначають приріст максимального прибутку при використанні додаткової одиниці деякого ресурсу. Значення маргінальної оцінки показує, наскільки знижується максимальний прибуток при випуску одиниці цієї продукції. Більш детальна інформація про аналіз чутливості в задачах лінійного програмування наведена в [7].

Завдання на роботу

1. Самостійно придумати та розв'язати задачу лінійного програмування.

2. Провести аналіз чутливості оптимального розв'язку задачі лінійного програмування до варіації початкових даних.

Вимоги до виконання роботи

1. Задача оптимізація повинна грунтуватися на реальних експериментальних даних.

2. Кількість керованих змінних в задачі лінійного програмування повинна бути не меншою за 5.

3. Кількість обмежень на значення керованих змінних повинна бути не меншою за 4.

4. Експерименти проводити в програмному середовищі MatLab з використанням бібліотеки Optimization.

5. Задачу оптимізації розв'язати, застосовуючи функцію lp.

6. Змінну для аналізу чутливості визначає викладач.

7. При побудові графічних зображень використовувати функцію plot.

8. В звіті необхідно навести змістовну та формалізовану постановки задачі.

Рекомендації до виконання роботи та приклади

1. Як приклад задачі лінійного програмування наведемо задачу про розподілення ресурсів.

Постановка задачі: для здійснення n різних технологічних процесів $T_1,\ldots,T_n$ заводу необхідно m видів ресурсів $S_1,..,S_m$ (сировина, паливо, матеріали, інструменти і т.д.). Запаси ресурсів кожного типу обмежені і дорівнюють $b_1,\ldots,b_m$, відповідно. Відомі витрати ресурсів на одиницю продукції по кожному технологічному процесу. Необхідно визначити такий обсяг продукції кожного типу, що забезпечує максимальний прибуток.

Введемо позначення, що необхідні для формалізованої постановки задачі:

a_{ij} – витрати ресурсів типу S_i на одиницю продукції типу T_j;

c_j – прибуток від реалізації одиниці продукції типу T_j;

x_j – кількість одиниць продукції типу T_j, яку треба випустити для отримання максимального прибутку.

Обмеженнями в цій задачі є вимоги, щоб витрати ресурсів типу S_i на випуск продукції не перевищували запаси:

$$\sum_{j=1}^{n} a_{ij} \cdot x_j \le b_i, \; i = \overline{1, m}. \tag{4}$$

Прибуток від реалізації випущеної продукції визначається за формулою:

$$Z = \sum_{j=1}^{n} c_j \cdot x_j. \tag{5}$$

Задача оптимального розподілення ресурсів формально ставиться таким чином: знайти такі невід'ємні значення (x_1, x_2,…, x_n), які задовольняють систему обмежень (4) та обертають лінійну функцію (5) в максимум.

2. Приклад аналізу чутливості

Проведемо аналіз чутливості оптимального розв'язку до варіації коефіцієнта при x_2. Цільова функція:

$$Z = -x_1 + x_2 \to \max;$$

обмеження:

$$x_1 + 2 \cdot x_2 \le 10,$$

$$2 \cdot x_1 + x_2 \le 10,$$

$$x_1 \ge 0, x_2 \ge 0.$$

Рішення цієї задачі графічним методом [7] показано на рис. 6.

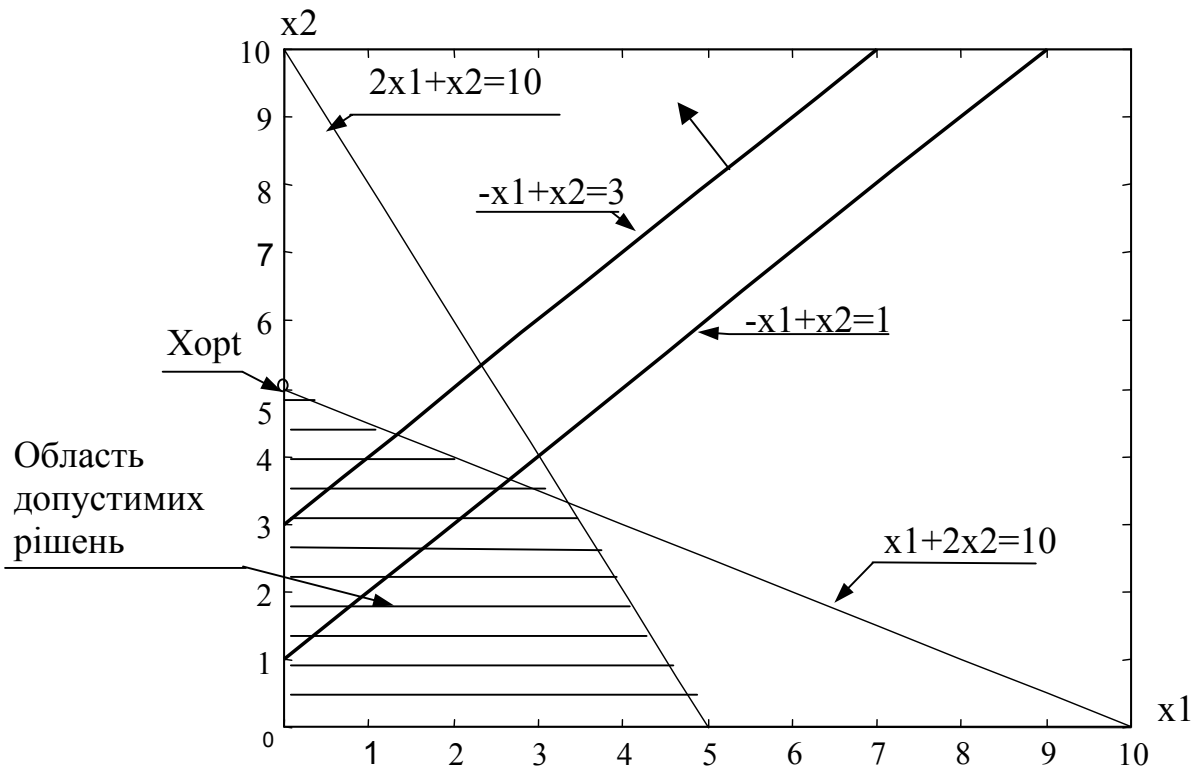


Рисунок 6 – графічне розв’язування задачі лінійного програмування

Проведемо аналіз чутливості оптимального розв’язку до варіації другого коефіцієнта c_2 цільової функції $Z = -x_1 + c_2 \cdot x_2$. При зміні коефіцієнта c_2 змінюється кут нахилу сліду цільової функції до координатних осей Ox_1 і Ox_2. Тому оптимальний розв’язок зміниться тоді, коли слід цільової функції буде паралельний прямій x1=0 або прямій $x_1 + 2 \cdot x_2 = 10$.

Пряма $x_1 + 2 \cdot x_2 = 10$ буде паралельна прямій $Z = -x_1 + c_2 \cdot x_2$ тоді, коли $\frac{1}{1} = \frac{2}{c_2}$, звідки c_2=2.

Пряма x1=0 буде паралельна прямій $Z = -x_1 + c_2 \cdot x_2$, коли $\frac{1}{1} = \frac{0}{c_2}$, звідки c_2=0.

Отже, оптимальний розв’язок не зміниться, якщо коефіцієнт $c_2 \in [0, 2]$.

3. Для розв’язання задачі лінійного програмування необхідно використовувати функцію lp в такому форматі:

Xopt=lp(C,A,B),

де Xopt – координата оптимум.

Функція lp дозволяє знаходити мінімум функції $y = C' \cdot X$ при обмеженнях $A \cdot X \le B$.

Якщо керовані змінні обмежені інтервалом $VLB \le X \le VUB$, тоді необхідно викликати функцію lp в такому форматі:

X=LP(f,A,b,VLB,VUB).

Для отримання додаткової інформації по застосуванню функції lp використовуйте команду help lp.

Запитання для самоконтролю

1. Як ставиться задача лінійного програмування?
2. Чи є задача лінійного програмування задачею безумовної оптимізації?
3. Наведіть основні етапи симплекс-методу розв’язання задачі лінійного програмування.
4. Навіщо проводити аналіз чутливості оптимального розв’язку задачі оптимізації?

Лабораторна робота №4

Дослідження залежності якості розв'язання задачі нелінійного програмування від кількості початкових точок

Теоретичні відомості

Задачами нелінійного програмування називають такі задачі оптимізації, в яких цільова функція або хоча б одне обмеження є нелінійною функцією.

Розглянемо задачу мінімізації функції:

$$f(X) \to \min$$

при обмеженнях

$$h_k(X) = 0, k = \overline{1, m}. \qquad (7)$$

Цю задачу можна розв'язати як задачу безумовної оптимізації шляхом виключення з цільової функції m незалежних змінних за допомогою заданих рівностей, тобто ми зменшуємо розмірність задачи з n до n-m. Але може бути така ситуація, коли обмеження будуть такими, що виключення якоїсь змінної неможливе. Наприклад, в обмеженні вигляду $h_1(x) = x_1^{\,2} \cdot x_3 + x_2 \cdot x_3^{\,2} + x_2^{\,-1} \cdot x_1 = 0$ неможливо аналітично виразити будь-яку змінну через інші. В подібних випадках можна зостосувати метод Лагранжа [1], ідея якого полягає в тому, що задача оптимізації з обмеженнями у вигляді рівностей перетворюється в еквівалентну задачу безумовної оптимізації.

Згідно з методом Лагранжа [3] задачу (7) можна перетворити в таку задачу безумовної оптимізації:

$$L(X;V) = f(X) - \sum_{k=1}^{m} v_k \cdot h_k \to \min,$$

де $L(X,V)$ – функція Лагранжа;

$v_к$ – невідома постійна, яка має назву множника Лагранжа.

Для розв'язування цієї задачи необхідно скористатися одним з відомих методів безумовної оптимізації, які було розглянуто вище.

Метод множників Лагранжа можна використовувати в задачах оптимізації, коли обмеження представлені у вигляді рівностей. Кун і Такер [7] узагальнили цей підхід на випадок, коли обмеження представлені як у вигляді рівностей, так і нерівностей.

Розглянемо таку задачу:

$$f(X) \to \min \tag{8}$$

при обмеженнях

$$g_j(X) \geq 0,\ j = \overline{1,r}, \tag{9}$$

$$h_k(X) = 0,\ k = \overline{1,m}. \tag{10}$$

Кун і Такер побудували необхідні та достатні умови оптимальності для задач нелінійного програмування, припускаючи, що функції f, g_j і h_k диференційовані.

Умова Куна-Такера та задача Куна-Такера [7] формулюється таким чином:

Знайти вектори $x_{(n+1)}$, $u_{(1\times r)}$ і $v_{(1\times m)}$, які задовольняють такі умови:

$$\nabla f(X) - \sum_{j=1}^{r} u_j \cdot \nabla g_j(X) - \sum_{k=1}^{m} v_k \cdot \nabla h_k(X) = 0\ , \tag{11}$$

$$g_j(x) \geq 0,\ \ j = \overline{1,r}, \tag{12}$$

$$h_k(x) = 0,\ \ k = \overline{1,m}, \tag{13}$$

$$u_j \cdot g_j(x) = 0,\ \ j = \overline{1,r}, \tag{14}$$

$$u_j \geq 0,\ \ \ j = \overline{1,r}. \tag{15}$$

Теорема про необхідність умов Куна-Такера [7]:

Розглянемо задачу нелінійного програмування (8) – (10). Нехай f,g та h диференційовні функції, а X^* – допустимий розв'язок даної задачі. Припустимо, що $I = \{j \mid g_j(X^*) = 0\}$. Нехай $\nabla g_j(X^*)$ і $\nabla h_k(X^*)$ лінійно незалежні при $j \in I$ і $k = \overline{1, m}$. Якщо X^* – оптимальний розв'язок задачі, то існує така пара векторів (U^*, V^*), що (X^*, U^*, V^*) є розв'язком задачі Куна-Такера (11) – (15).

Цю теорему можна використовувати для доведення того, що задана допустима точка, яка задовольняє умову лінійної незалежності, не є оптимальною, якщо вона не задовольняє умови Куна-Такера. З іншої сторони, якщо в цій точці умова Куна-Такера виконується, то нема гарантії того, що ця точка є оптимальним розв'язком задачі. Наступна теорема встановлює умову, за якою точка, яка задовольняє умови Куна-Такера, автоматично відповідає оптимальному розв'язку задачі:

Теорема про достатню умову Куна-Такера [7]

Розглянемо задачу (8) – (10). Нехай цільова функція f(X) опукла, всі обмеження у вигляді нерівностей є вгнуті функції $g_j(X)$, $j = \overline{1, r}$, а обмеження у вигляді рівностей є лінійні функції $h_k(X)$, $k = \overline{1, m}$. Тоді, якщо існує розв'язок (X^*, U^*, V^*), який задовольняє умови Куна-Такера (11) – (15), тоді X^* – оптимальний розв'язок задачі.

Тепер розглянемо таку задачу:

$$f(X) \to \min, \tag{16}$$

при обмеженнях

$$g_j(X) \geq 0,\ j = \overline{1, r}, \tag{17}$$

$$h_k(X) = 0,\ k = \overline{1, m}, \tag{18}$$

$$x_i^{(l)} \leq x_i \leq x_i^{(u)},\ i=1,2,\ldots,n. \tag{19}$$

Нехай вектор X^* – розв'язок нашої задачи. Припустимо, що відомо деяке початкове наближення $X^{(0)}$, яке може бути недопустимим, тобто таке, що не задовольняє умови (17)-(19). За допомогою алгоритмів, які будуть розглянуті нижче, в просторі будується кінцева послідовність точок $X^{(t)}$, t=1,2,…,T, яка починається з заданої точки $X^{(0)}$ і закінчується точкою $X^{(T)}$, яке дає найкраще наближення до X^* серед усіх точок побудованої послідовності. Ця послідовність будується за допомогою штрафної функції. Штрафна функція – це така функція, яка допомогає перейти від задачи умовної оптимізації до задачи безумовної оптимізації. Вона визначається таким виразом:

$$P(X,R) = f(X) + \Omega(R, g(X), h(X)),$$

де R – набір штрафних параметрів;

Ω – це функція штрафу, яка залежить від R та функцій, які задають обмеження.

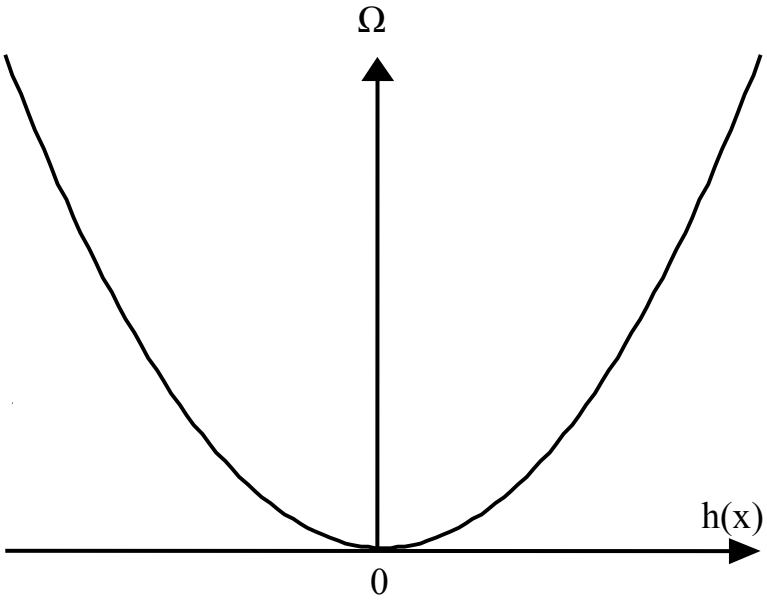


Рисунок 7 – Квадратичний штраф

Найбільш широкого розповсюдження набули квадратичний штраф (рис. 7) та штраф типу квадрата зрізки (рис. 8).

Квадратичний штраф використовують для обмежень у вигляді рівностей: $\Omega = R \cdot \{h(X)\}^2$. При мінімізації цей штраф перешкоджає відхиленню величини h(X) від нуля (як в сторону додатних, так і в сторону від'ємних значень). Зрозуміло, що при збільшенні R стаціонарна точка відповідної штрафної функції P(X, R) наближається до точки X^*, оскільки $\lim h(X^T) = 0$

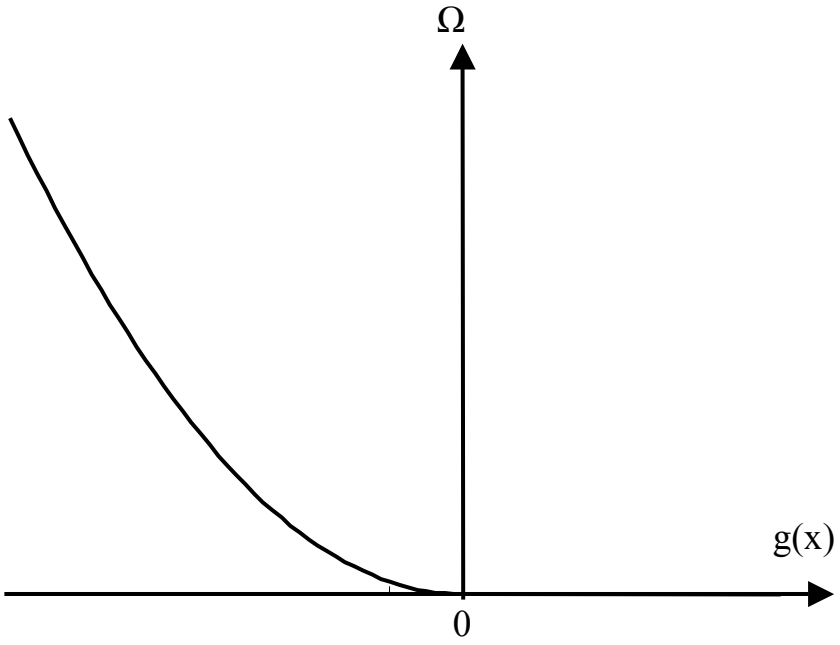


Рисунок 8 – Штраф типу квадрата зрізки

Для обмежень у вигляді нерівностей використовують штраф типу квадрата зрізки (рис. 8).

$$\Omega = R \cdot < h(X) >^2,$$

де

$$\langle h(X) \rangle = \begin{matrix} h(X)^2, \text{ при } h(X) \le 0 \\ 0, \text{ при } h(X) \ge 0 \end{matrix}$$

Штраф типу квадрата зрізки зручний у використанні тим, що функція P(X, R) визначена і неперервна всюди. Обчислення проводять з додатним R; після кожної ітерації значення R збільшують.

Завдання на роботу

Експериментально дослідити залежність якості розв'язання задачі багатоекстремальної задачі умовної оптимізації від кількості початкових точок.

Таблиця 3 Варіанти завдань

Варіант	Діапазон зміни аргументів	Обмеження	Цільова функція	Постановка задачі
1	$1\le x\le 40$ $1\le y\le 40$	$y-5x\le 0$ $y+0.01(x-20)^2\le 37$	$Z=\cos(0.3x)\cos(0.25y-7)-\frac{1}{x}$	min
2	$1\le x\le 70$ $1\le y\le 70$	$y-0.3x\ge 0$ $y+0.5x^2-70\ge 0$	$Z=\cos(0.3y)x+\cos(0.25x)y$	max
3	$1\le x\le 90$ $1\le y\le 90$	$y-0.7\sqrt{x}\ge 0$ $y-7x\le 0$	$Z=\sin(0.17y)(x+10)-$ $-\sin(0.17x)\cdot(y-5)$	min
4	$1\le x\le 30$ $1\le y\le 30$	$y-\frac{x^3}{100}\le 0$ $y-25+1.25x\ge 0$	$Z=\cos(0.8y)e^{\frac{x}{8}}+\sin(0.7x)e^{\frac{y}{8}}$	min
5	$1\le x\le 50$ $1\le y\le 50$	$y-\frac{50}{x}\ge 0$ $y-30-\frac{4}{3}x\le 0$	$Z=\log(y)\cos(0.3x)+e^{\sin(0.35y)}$	max
6	$1\le x\le 70$ $1\le y\le 70$	$y-4\sqrt{x}\ge 0$ $x-23\ge 0$	$Z=y\cdot e^{\sin(0.3x)}+x\cdot e^{\cos(0.35y)}$	min
7	$1\le x\le 40$ $1\le y\le 40$	$y-0.1e^x\le 0$ $y-3\ge 0$	$Z=\sin(0.4x+3)\sin(0.2y-2)-\frac{1}{x}$	min

Продовження табл. 3

Варіант	Діапазон зміни аргументів	Обмеження	Цільова функція	Поста-новка задачі
8	$1\le x\le 80$ $1\le y\le 80$	$y-\frac{250}{x}+20\ge 0$ $y-\frac{7}{8}(x-10)\ge 0$	$Z=x(\sin(0.3y-5)-\cos(0.3x+5))$	max
9	$-1\le x\le 1$ $-1\le y\le 1$	$y-3x+2.3\ge 0$ $y+0.2(x+1)^2+0.5\le 0$	$Z=y^2+x^2-\cos(9x)-\cos(9y)$	max
10	$25\le x\le 25$ $25\le y\le 25$	$y-2x-25\le 0$ $y-0.7\sqrt{\lvert x\rvert}+25\ge 0$	$Z=y\sin(0.3y-2)+x\cos(0.4x)$	min
11	$1\le x\le 70$ $1\le y\le 70$	$y-0.5e^x\le 0$ $y-\frac{1}{3}x-35\le 0$	$Z=\cos(0.3x+4)\cos(e^{0.03y})+\frac{1}{y}$	max
12	$1\le x\le 100$ $1\le y\le 100$	$y-12x+15\le 0$ $y-10\log(x)\ge 0$	$Z=\cos(0.2y)+\log_2(\frac{y}{10})\sin(0.2x)$	min
13	$1\le x\le 60$ $1\le y\le 60$	$y-4.5\sqrt{x}-25\le 0$ $x-55\le 0$	$Z=x(\cos(0.3y)-\sin(0.3x))+\frac{1}{y}$	min
14	$1\le x\le 50$ $1\le y\le 50$	$y-\frac{50}{x}\ge 0$ $y-x\ge 0$	$Z=e^{0.1x}(\sin(0.6y)+\sin(0.4x-9))$	max
15	$0\le x\le 100$ $0\le y\le 100$	$\sqrt{y}-x\le 0$ $y-x\ge 0$	$Z=\sin(0.34y+2)(x-35)^3+\frac{123}{x}$	max
16	$1\le x\le 100$ $1\le y\le 100$	$3y-x-20\ge 0$ $y-\frac{100}{x}\ge 0$	$Z=\sqrt{x^7-1}\cdot\cos(0.32y-5)$	min

Вимоги до виконання роботи

1. Цільову функцію та обмеження вибрати з табл. 3 згідно з варіантом.

2. Початкові точки генерувати випадково.

3. Кількість експериментів повинна бути не меншою за 10.

4. Кількість обмежень на значення керованих змінних повинна бути не меншою за 4.

5. Експерименти проводити в програмному середовищі MatLab з використанням бібліотеки Optimization.

6. При побудові графічних зображень використовувати функції plot та contour.

Рекомендації до виконання роботи та приклади

1. В даній лабораторній роботі необхідно розв’язати багатоекстремальну нелінійну задачу з обмеженнями і дослідити залежність якості розв’язання задачі нелінійного програмування від кількості початкових точок. При однократному розв’язанні задач замість глобального оптимуму можна отримати локальний. Приклад такої ситуації при мінімізації функції однієї змінної з обмеженням $a \le x \le b$ наведено на рис. 9. Якщо початкова точка взята праворуч точки c, то використання довільного методу з лабораторної роботи №2 приведе до значення xopt=b.

Глобальний мінімум отримається при x=a; його можна отримати тоді, коли початкова точка буде знаходитися ліворуч точки c. Такі випадки присутні і в багатовимірних задачах. Отже, визначення глобального оптимуму залежить від успішного вибору початкової точки. Таким чином, пошук екстремуму необхідно здійснювати при різних початкових точках. Для генерування випадкових початкових точок необхідно використовувати функцію rand.

2. Для дослідження залежності якості розв’язання задачі від кількості початкових точок необхідно побудувати графік, приклад якого наведено на рис. 10.

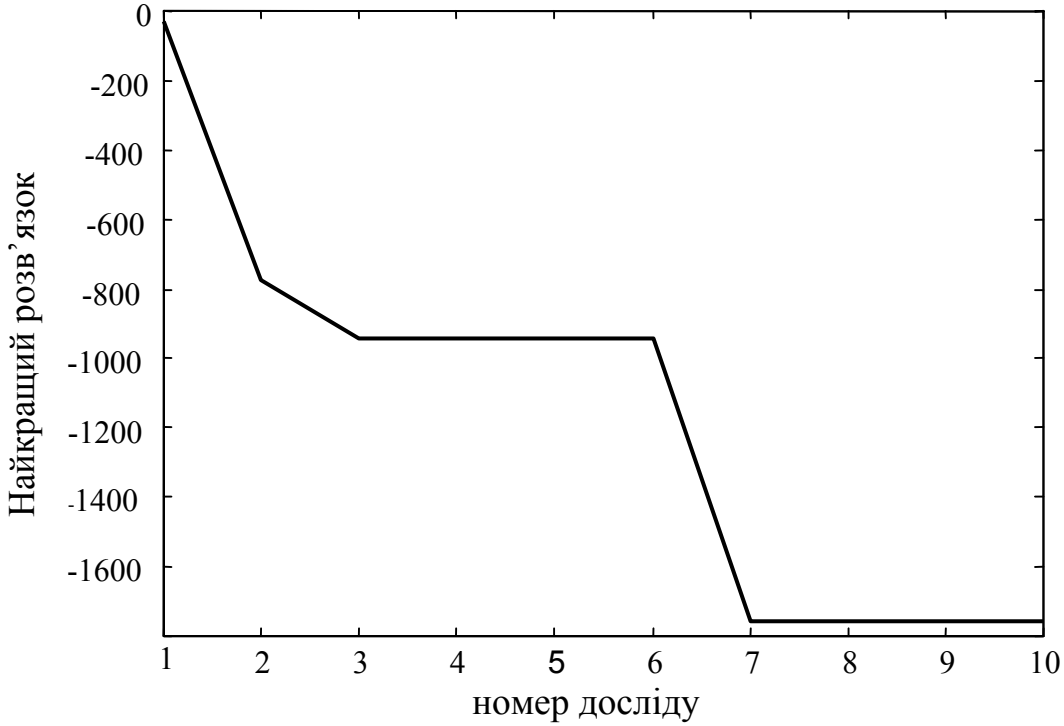


Рисунок 10 – Графік залежності якості рішення задачі від кількості початкових точок

3. Для розв'язання задачі нелінійного програмування з обмеженнями необхідно використовувати функцію constr в такому форматі:

[Xopt options]=constr(fun,x0,options,vlb,vub),

де Xopt – координата оптимум;

options – параметри оптимізації;

fun – ідентифікатор цільової функції та обмежень;

x0 – початкова точка;

vlb (vub) – нижня (верхня) межа можливих значень керованих змінних.

З метою контролю за процесом оптимізації необхідно виводити проміжні результати на екран. Для цього потрібно ввести команду options(1)=1. Тоді на екрані з'являться результати такого формату

f-COUNT	**FUNCTION**	**MAX{g}**	**STEP**	**Procedures**

В перший стовпець виводиться номер ітерації, в другому - значення цільової функції, в третьому максимальне значення обмеження, в четвертому значення кроку оптимізації.

Цільова функція та обмеження задаються за допомогою однієї змінної fun наступним чином:

fun = [funf fung],

де funf – ідентифікатор цільової функції, наприклад, funf='goal_function(x)';

fung– ідентифікатор обмежень.

Обмеження задаються як нерівності типу $g(x) \le 0$. Наприклад, при наявності обмежень $x_1 + 4 \cdot x_2 \le 0$ та $x_1 \cdot \sin(x_2) \le 0$ необхідно зміну fung задати таким чином: fung='g=[x(1)+4*x(2); x(1)*sin(x(2))];';

Якщо в задачі оптимізації є обмеження в вигляді рівностей, тоді їх необхідно задати спочатку в змінній fung, а їх кількість вказати в options(13). Додаткову інформацію про функцію constr можно отримати за допомогою команди help constr.

Приклад графічного представлення результатів пошуку глобального екстремуму наведено на рис. 11.

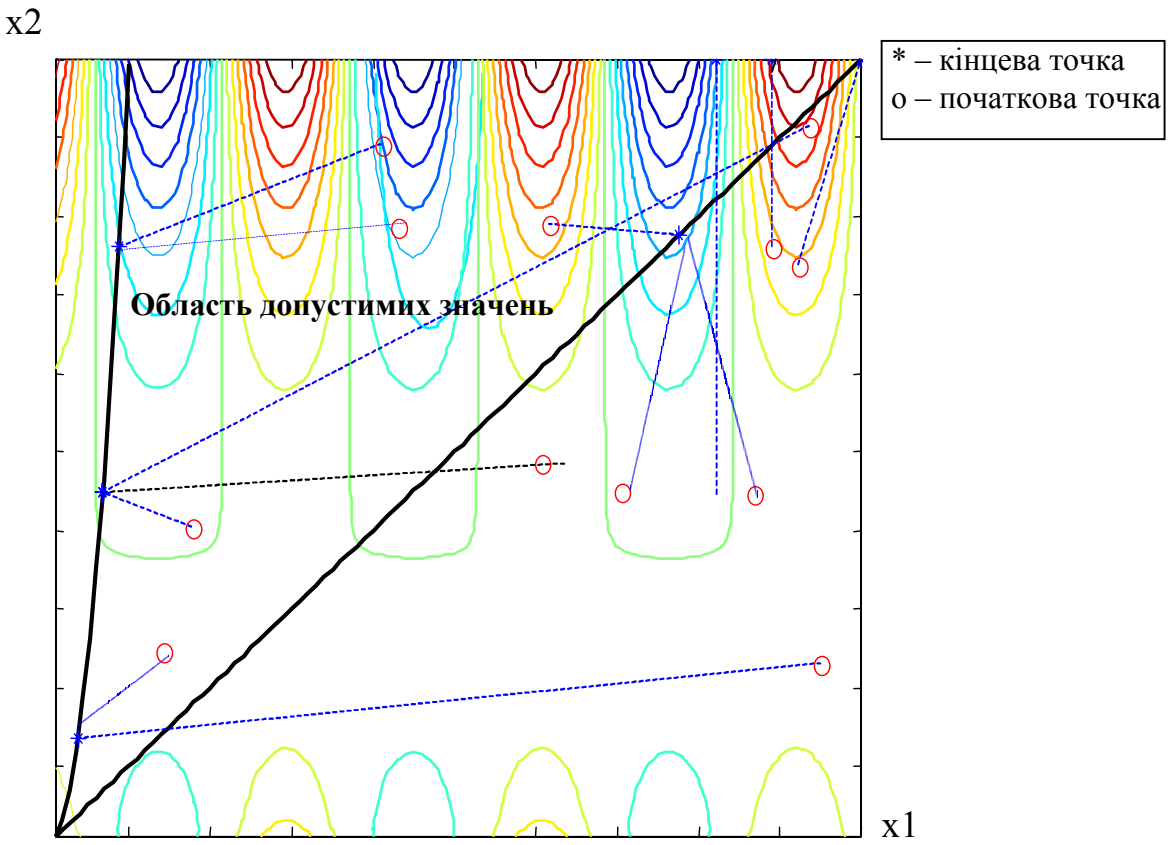


Рисунок 11 – Оптимізація багатоекстремальної функції

4. Приклад програми для знаходження мінімуму функції $Z = x^2 \cos(0.2y)\cos(e^{\frac{x}{250}})$ при обмеженнях y-12x+15≤0, y-10log(x)≥0, 1≤x≤100, 1≤y≤100.

```
funf='f=  x(1)^2*cos(0.2*x(2))*cos(exp(x(1)/250));';  % індефікатор функції
fung='g=[x(2)-12*x(1)+15;  10*log(x(1))-x(2)];';       % індефікатор обмежень
fun = [funf fung];
vlb=[1 1];                          % обмеження знизу на керовані змінні
vub=[100 100];                      % обмеження зверху на керовані змінні
x0=[100*rand(1)  100*rand(1)]       % випадкова генерація початкової точки
options(1)=1;                       % виведення проміжних результатів на  екран
[Xopt options]=constr(fun,x0,options,vlb,vub);  % розв'язок задачі
Xopt                                % координата оптимуму
options(8)                          % значення цільової функції в точці Xopt
```

Запитання для самоконтролю

1. Як ставиться задача нелінійного програмування?
2. Поясніть яким чином можливо використовувати методи безумовної оптимізації до розв'язання задач оптимізації з обмеженнями.
3. Наведіть необхідні та достатні умови глобального екстремуму.
4. Як використовувати методи локальної оптимізації для пошуку глобального оптимуму?
5. Як визначити необхідну кількість початкових точок?

Література

1. Вентцель Е.С. Исследование операций: Задачи, принципы, методология.-М.:Наука, 1988. – 75с.
2. Дехтярев Ю. И. Методы оптимизации.- М.:Советское радио.-1980. – 270с.
3. Глушков В.М. Введение в АСУ.-К.:Техніка.-1974.-317с.
4. Жилинскас А., Шалтянис В. Поиск оптимума. – М.:Наука.-1989. – 45с.
5. Коршунов Ю.М. Математические основы кибернетики. – М.:Энергия.-1980.-421с.
6. Потемкин В.Г. Система MatLab 5 для студентов. Справочное пособие.-М.: Диалог-МИФИ, 1998.-314с.
7. Реклейтис Г., Рейвиндран А., Рэгсдел К. Оптимизация в технике. Кн.1.-М.: Мир.- 1986.- 347с.
8. Компьютер и задачи выбора.-М.:Наука.-1989.-204с.
9. Optimization Toolbox. User’s Guide. Version 2. The MathWorks Inc.- 1999

Додаток А

Основні функції Optimization Toolbox

Функції матричних задач

1. **lp** – дозволяє знайти мінімум цільової функції $y = C' \cdot X$ при обмеженнях $A \cdot X \le B$ в лінійному програмуванні. Найчастіше ця функція використовується в такому форматі:

Xopt=lp(C,A,B),

де Xopt – координата оптимуму.

Для отримання додаткової інформації з застосування функції lp використовуйте команду help lp.

2. **qp** – дозволяє знайти мінімум цільової функції $y=x' \cdot H \cdot x+f' \cdot x$ при обмеженнях $A \cdot X \le B$ в квадратичному програмуванні. Найчастіше ця функція використовується в такому форматі:

Xopt=QP(H,f,A,B),

де Xopt – координата оптимуму.

Для отримання додаткової інформації з застосування функції QP використовуйте команду help QP.

Функції задач нелінійного програмування

1. **constr** – дозволяє знайти мінімум цільової функції y=f(X) при наявності обмежень в нелінійному програмуванні. Ця функція найчастіше використовується в форматі:

[Xopt options]=constr(fun,x0,options,vlb,vub),

де Xopt – координата оптимуму;

options – параметри оптимізації;

fun – ідентифікатор цільової функції та обмежень;

x0 – початкова точка;

vlb (vub) – нижня (верхня) межа можливих значень керованих змінних.

Для отримання додаткової інформації з застосування функції constr використовуйте команду help constr;

2. **fmin** – дозволяє знайти мінімум цільової функції y=f(x) одного аргументу на інтервалі [x1, x2]. Найчастіше ця функція використовується в такому форматі:

Xopt=fmin('f',x1,x2),

де Xopt – координата оптимуму.

Для отримання додаткової інформації по застосуванню функції fmin використовуйте команду help fmin;

3. **fminu** – дозволяє знайти мінімум цільової функції y=f(X) багатьох аргументів. Ця функція реалізує градієнтні методи і найчастіше використовується в такому форматі:

Xopt=fmin('f',x0,options),

де x0 – початкова точка, з якої починається розв'язання задачі;

options – параметри оптимізації;

Xopt – координата оптимуму.

Для отримання додаткової інформації з застосування функції fminu використовуйте команду help fminu;

4. **fmins** – дозволяє знайти мінімум цільової функції y=f(X) багатьох аргументів. Ця функція реалізує пошук за симплексом і найчастіше використовується в такому форматі:

Xopt=fmins('f',x0,options).

Для отримання додаткової інформації з застосування функції fmins використовуйте команду help fmins;

5. **fzero** – дозволяє розв'язати нелінійне рівняння f(x)=0 з однією змінною. Найчастіше використовується в такому форматі

Xopt=fzero('f',x0),

де x0 – початкова точка, з якої починається розв'язання задачі;

Xopt – координата оптимуму;

6. **fsolve** – дозволяє розв'язати систему нелінійних рівнянь f(X)=0. Найчастіше використовується в такому форматі

Xopt=fmin('f',x0,options).

7. **leastsq** – дозволяє знайти мінімум цільової функцій y=f(X) багатьох аргументів. Ця функція реалізує методи другого порядку і найчастіше використовується в такому форматі

Xopt=leastsq('f',x0,options),

де x0 – початкова точка, з якої починається розв'язання задачі;

options – параметри оптимізації;

Xopt – координата оптимуму.

Для отримання додаткової інформації з застосування функції leastsq використовуйте команду help leastsq;

Параметри оптимізації

options(1) – параметр, який дозволяє виводити на екран проміжні результати. Даний параметр приймає два значення 0 або 1. Якщо параметр приймає значення 1, то результат виводиться на екран, в іншому випадку – не виводиться;

options(2) – параметр, в якому визначається міра точності за X;

options(3) – параметр, в якому визначається міра точності за функцією f;

options(4) – параметр, в якому визначається міра точності за обмеженнями;

options(5), options(6), options(7) – параметри, які визначають стратегію і алгоритм оптимізації;

options(8) – значення цільової функції в точці оптимального розв’язку;

options(9) – даний параметр приймає два значення 1 або 0. Якщо параметр приймає значення 1, то градієнт обчислюється за аналітичним виразом, в іншому випадку градієнт чисельно апроксимується.

options(10) – кількість ітерацій, які були використані для розв’язання задачі;

options(11) – кількість ітерацій, які були використані для обчислення градієнта;

options(12) – параметр, яким визначається кількість обмежень;

options(13) – параметр, яким визначається кількість обмежень у вигляді рівностей;

options(14) – параметр, яким визначається максимальна кількість ітерацій, які необхідні для розв’язання задачі;

options(15) – параметр, яким визначається кількість критеріїв, за якими проводять оптимізацію;

options(16) – параметр, яким визначається мінімальний приріст аргументів для обчислення градієнта;

options(17) – параметр, яким визначається максимальний приріст аргументів для обчислення градієнта;

options(18) – параметр, яким визначається крок оптимізації (значення кроку повинно бути не більше 1);

Додаток В

Основні функції мови программування Matlab

Спеціальні символи

: – перетин масиву;

() – вказівка на виконання послідовності операцій;

[] – формування масиву;

; – заборона виведення на екран;

% – коментарій;

= – привласнення;

‘ – транспонування матриці.

Приклад 1. Сформувати масив $A = (1 \;\; 3 \;\; 17 \;\; 9 \;\; 12 \;\; 8 \;\; 11)$ та матрицю $B = \begin{pmatrix} 1 & 3 \\ 5 & 7 \\ 11 & 13 \end{pmatrix}$. Знайти перетин з другого по п’ятий елементи масиву А та транспанувати матрицю В.

Формування масиву: A=[1 3 17 9 12 8 11]; матриці: B=[1 3; 5 7; 11 13].

Перетин масиву: A=(2:5) 3 17 9 12

Транспонування матриці C=B’ $C = \begin{pmatrix} 1 & 5 & 11 \\ 3 & 7 & 13 \end{pmatrix}$

Арифметичні оператори

+ – додавання;

- – віднімання;

***** – множення матриць;

.* – поелементне множення для масивів ;

^ – піднесення матриці до степеня;

.^ – піднесення до степеня для масивів;

\ – ліве ділення матриць;

/ – праве ділення матриць;

. – ліве ділення для масивів;

./ – праве ділення для масивів.

Приклад 2. Задані два вектори A=[3 9 5] і B=[2 1 5]. Виконати дії: A+3B, A./ B.^2, (A./B).^2.

Результати виконання:

C= A+3B C = 9.0000 12.0000 20.0000

C= A./ B.^2 C = 0.7500 9.0000 0.2000

C=(A./B).^2 C = 2.2500 81.0000 1.0000

Приклад 3. Задані дві матриці $A=\begin{pmatrix}8 & 1 & 6\\ 3 & 5 & 7\\ 4 & 9 & 2\end{pmatrix}$ та $B=\begin{pmatrix}2 & 2 & 2\\ 2 & 2 & 2\\ 2 & 2 & 2\end{pmatrix}$.

Виконати дії A+2, A*B та A.*B.

Результати виконання:

C=A+2 $C=\begin{pmatrix}10 & 3 & 8\\ 5 & 7 & 9\\ 6 & 11 & 4\end{pmatrix}$

C=A*B $C=\begin{pmatrix}30 & 30 & 30\\ 30 & 30 & 30\\ 30 & 30 & 30\end{pmatrix}$

C=A.*B $C=\begin{pmatrix}16 & 2 & 12\\ 6 & 10 & 14\\ 8 & 18 & 4\end{pmatrix}$

Спеціальні змінні

ans – результат виконання останньої дії;

eps – машинна точність;

inf – нескінченне значення;

NaN – нечислове значення.

Масиви та їх характеристики

zeros – формування масиву нулів;

ones – формування масиву одиниць;

eye – формування одиничної матриці;

rand – формування масиву елементів, розподілених за рівномірним законом;

randn – формування масиву елементів, розподілених за нормальним законом;

size – розмір масиву;

lenght – довжина вектора.

Базові операції

max – максимальний елемент масиву;

min – мінімальний елемент масиву;

mean – елемент середніх значень масиву;

sort – сортування за зростанням;

sum – підсумовування елементів масиву;

prod – добуток елементів масиву;

abs – модуль;

sqrt – квадратний корінь;

sign – визначення знака числа.

Трансцендентні функції

exp – експонтенціальна функція;

log – функція натурального логарифма;

log10 – логарифм за основою 10;

log2 – логарифм за основою 2;

sin – синус;

cos – косинус;

tan – тангес;

cot – котангес.

Робота з таймером

tic – вмикання таймера;

toc –вимикання таймера;

profile – профілювання програми.

Робота з графікою

plot – побудова ліній та точок на площині;

contour – побудова ліній рівня для тривимірної поверхні;

xlabel – позначення на осі Ox

ylabel – позначення на осі Oy

zlabel – позначення на осі Oz;

clabel – маркування ліній постійного рівня;

title – заголовок графіка;

text – добавлення тексту в графічне вікно;

legend – пояснення до графіка;

xlim – обмеження графіка по осі Ox;

ylim – обмеження графіка по осі Oy;

figure(s) – графічне вікно з номером s;

hold on – продовження виведення інформації в те ж саме графічне вікно.

Приклад 4. Побудувати графік функції $y = x^2$ на інтервалі [0 50] та побудувати лінію, яка з'єднує дві точки з координатами (10 12) та (34 23). Графічну інформацію зобразити на одному графіку.

Для побудови графіка та лінії необхідно виконати таку послідовність дій:

а) формування масиву x: `x=0:50;`

б) формування масиву y: `y=x.^2;`

в) побудова графіка: `plot(x,y) ;`

г) продовження виведення графічної інформації: `hold on;`

д) побудова лінії: `plot([10 34] , [12 23]).`

Навчальне видання

Штовба Сергій Дмитрович

МЕТОДИ ОПТИМІЗАЦІЇ В СЕРЕДОВИЩІ MATLAB
Лабораторний практикум

Навчальний посібник

Оригінал макет підготовлено автором

Редактор В. О. Дружиніна

Підписано до друку

Формат 29,7х42? Гарнітура Times New Roman

Друк різографічний Ум. друк. арк.
Тираж 75 прим.
Зам. №

Віддруковано в комп'ютерному інформаційно-видавничому центрі
Вінницького державного технічного університету
21021, м. Вінниця, Хмельницьке шосе, 95, ВДТУ, ГНК, 9-й поверх
Тел. (0432)-440-159