Міністерство освіти і науки України
Національний університет «Запорізька політехніка»

Валерій ДУБРОВІН, Лариса ДЕЙНЕГА

МЕТОДИ ОПТИМІЗАЦІЇ ТА ЇХ РЕАЛІЗАЦІЯ ЗАСОБАМИ МОВИ ПРОГРАМУВАННЯ PYTHON

Рекомендовано
Міністерством освіти і науки України
як навчальний посібник
для студентів закладів вищої освіти

2024

ББК 22.18:32.97
Д79
УДК 519.6: 004.42

Дубровін В.І., Дейнега Л.Ю.
Д79 Методи оптимізації та їх реалізація засобами мови програмування Python: навчальний посібник. – Запоріжжя: НУ «Запорізька політехніка», 2024. – с.

Рецензенти:
С.І. Гоменюк, доктор технічних наук, професор кафедри програмної інженерії, декан математичного факультету Запорізького національного університету
А.В. Переверзєв, доктор технічних наук, професор, проректор з наукової роботи Запорізького інституту економіки та інформаційних технологій

Розглянуто методи лінійного програмування, методи нелінійної одновимірної та багатовимірної оптимізації, що використовуються для розв'язання задач оптимізації технічних систем. Виклад проілюстровано прикладами розв'язання конкретних інженерних задач. Через реалізацію відповідних алгоритмів засобами мови програмування Python розв’язано задачу лінійного програмування, задачі оптимізації, в яких цільова функція задана функцією однієї змінної, задачі багатовимірної оптимізації.

Для фахівців у галузі проектування технічних систем та пристроїв, а також викладачів, аспірантів та студентів технічних закладів вищої освіти

ЗМІСТ

ВСТУП

Процес оптимізації лежить в основі діяльності випускників спеціальності „Інформаційні технології проектування” оскільки класичні функції інженера заключаються в тому, щоб, з однієї сторони, проектувати нові, більш ефективні та менш дорогі системи, а з іншої, розробляти методи збільшення якості функціонування існуючих систем.

Ефективність методів оптимізації, що дозволяють здійснити вибір найкращого варіанта без перевірки всіх можливих варіантів, тісно пов’язана з використанням тріади „модель-алгоритм-програма”.

Використання моделей зумовлено тим, що експерименти з реальними системами, як правило, вимагають дуже великих витрат засобів і часу, а також, в деяких випадках, пов’язані з ризиком. Моделі широко використовуються для проектування, оскільки це надає можливості для реалізації найбільш економічного способу дослідження впливу змін в значеннях основних незалежних змінних на показники якості функціонування системи.

Оскільки вимоги до задач оптимізації являються загальними та носять абстрактний характер, область використання методів оптимізації може бути доволі широкою. У зв’язку з цим в провідних університетах світу введені учбові дисципліни “Engineering Optimization” та “Operation Research”, які викладаються на рівні бакалаврата, а в деяких випадках – на рівні магістратури. Така тенденція спостерігається і в вузах України.

В даних методичних вказівках вирішуються задачі оптимізації, в яких цільова функція задана функцією однієї змінної. Аналіз задач такого типу займає важливе місце в оптимізаційних дослідженнях, як теоретичного, так і практичного напрямку. Це пов’язане не тільки з тим , що саме такі задачі вирішуються в інженерній практиці, але і з тим, що одновимірні методи оптимізації часто використовують для аналізу задач, які виникають при реалізації ітераційних процедур, орієнтованих на вирішення багатовимірних задач оптимізації.

1 ВИРІШЕННЯ ЗАДАЧІ ЛІНІЙНОГО ПРОГРАМУВАННЯ НА ОСНОВІ ЇЇ ГЕОМЕТРИЧНОЇ ІНТЕРПРЕТАЦІЇ

1.1 Теоретичні відомості

Застосування лінійного програмування

Приклад 1.1 Конструктор має у своєму розпорядженні три види комплектів проводів А, В, С, що мають відповідно вартість C_1, C_2, C_3. Кожен комплект містить алюмінієві і мідні дроти, причому в комплекті А міститься a_{11} - алюмінієвих, a_{21} - мідних; у комплекті В міститься a_{12} - алюмінієвих, a_{22} - мідних; у комплекті С міститься a_{13} - алюмінієвих, a_{23} - мідних проводів.

Відомо, що на монтаж установки витрачається алюмінієвих проводів не менше b_1, а мідних - не менше b_2.

Задача полягає у визначенні необхідної кількості кожного з комплектів для того, щоб змонтувати установку і витратити мінімальні витрати на придбання комплектів.

Позначимо споживання кількості комплектів через X_1, X_2 X_3. З урахуванням відомих вартостей комплектів цільова функція, що підлягає оптимізації, має вигляд:

$$F = C_1 * X_1 + C_2 * X_2 + C_3 * X_3.$$

Складемо обмежувальні нерівності:

$$a_{11} * X_1 + a_{12} * X_2 + a_{13} * X_3 \geq b_1;$$
$$a_{21} * X_1 + a_{22} * X_2 + a_{23} * X_3 \geq b_2.$$

Перше рівняння описує потребу в алюмінієвих проводах, а друге - в мідних.

Потрібно знайти невід'ємні значення елементів рішення, що мінімізують цільову функцію з урахуванням обмежень.

Таким чином, приходимо до наступної математичної задачі. Знайти мінімум функції

(1.1)

$$F = \sum_{j=1}^{n} C_j * X_j$$

за умови, що її змінні задовольняють системі рівнянь

$$\sum_{j=1}^{n} b_{ij} * X_j = B_i \qquad (i = 1, m) \tag{1.2}$$

і умові додатності $X_j \geq 0$ (j=1,n).

Сформульована задача є задачею лінійного програмування, оскільки функція (1.1) лінійна, а система (1.2) містить тільки лінійні рівняння.

Приклад 1.2 Потрібно розробити апаратуру на основі серійних модулів (мікросхем) двох типів M_1 і M_2, що випускаються.

Відомо, що без урахування резервування в схемі апаратури повинно міститися модулів M_1 не менше b_1, а модулів M_2 - не менше b_2, крім того, відомо, що випускаються монтажні плати двох типів А і В, які передбачається використовувати як конструктивні несучі елементи для апаратури, що розробляється.

Відомо також, що в монтажному просторі плати А може розміститися a_{11} модулів M_1 і a_{21} модулів M_2. На монтажній платі В може бути розміщено a_{12} модулів M_1 і a_{22} модулів M_2.

Для кожного виду плат відомий ваговий коефіцієнт вартості, встановлений експертним або яким-небудь іншим методом. Для плати А він рівний C_1; для плати В - C_2. Потрібно визначити необхідну кількість плат для проектування апаратури по типах з умовою, щоб апаратура володіла максимальною якістю при мінімальних витратах.

З урахуванням умов і вимог задачі складемо цільову функцію

$$F = C_1 * X_1 + C_2 * X_2 \quad \rightarrow$$

$\rightarrow$ min, де X_1 - кількість плат типу А, X_2 - кількість плат типу В.

Складемо обмежувальні рівняння:

$$a_{11} * X_1 + a_{12} * X_2 \geq b_1;$$
$$a_{21} * X_1 + a_{22} * X_2 \geq b_2,$$

де X_1 і X_2 - цілі числа.

Перша нерівність описує розподіл модулів M_1 серед плат А і В, а друга - розподіл серед тих же плат, але вже модулів M_2.

Приклад 1.3 Задача складання раціону.

Вміст поживних речовин в кожній складовій суміші та її вартість подані в таблиці 1.1.

Таблиця 1.1 – Складання раціону

Складова	Вміст поживних речовин в 1 кг суміші, %		Вартість 1 кг суміші, у. о.
	білку	клітковини	
Зернові	10	2	0,40
Соєві боби	50	8	0,90

Готова суміш має містити не менше як 20 % білка і не більш як 5 % клітковини. Загальна кількість суміші має становити понад 5 кг.

Визначити масу кожного з двох видів складових, що утворюють суміш мінімальної вартості, водночас задовольняючи вимогам до загальної маси суміші та її поживності.

Побудова економіко-математичної моделі. Нехай x_1 — маса зернових, а x_2 — соєвих бобів (в кг) у готовій суміші.

Загальна кількість суміші $x_1 + x_2$ має становити понад 5 кг, тобто $x_1 + x_2 \geq 5$.

Розглянемо обмеження щодо поживності суміші.

Суміш має містити не менш як 20 % білка:

$10x_1 + 50x_2 \geq 20(x_1 + x_2)$,

а також не більше як 5 % клітковини:

$2x_1 + 8x_2 \leq 5(x_1 + x_2)$.

Загалом математична модель задачі оптимізації раціону має такий вигляд:

$$\min Z = 0{,}40x_1 + 0{,}90x_2$$

за умов:

$$x_1 + x_2 \geq 5,$$
$$-10x_1 + 30x_2 \geq 0,$$
$$-3x_1 + 3x_2 \leq 0$$

$$x_1 \geq 0;\ x_2 \geq 0.$$

Загальна і основна задача лінійного програмування

У попередньому підрозділі були розглянуті приклади завдань лінійного програмування (ЛП). У цих завданнях потрібно було знайти максимум або мінімум лінійної функції за умови, що її змінні приймали невід'ємні значення і задовольняли деякій системі лінійних рівнянь або лінійних нерівностей. Кожне з цих завдань є окремим випадком загальної задачі лінійного програмування.

Визначення 1.1. Загальною задачею лінійного програмування називається задача, яка полягає у визначенні максимального (мінімального) значення функції

$$F = \sum_{j=1}^{n} c_j * X_j \tag{1.3}$$

за умов

$$\sum_{j=1}^{n} a_{ij} * X_j \leq b_i \quad (i = 1, k) \tag{1.4}$$

$$\sum_{j=1}^{n} a_{ij} * X_j = b_i \quad (i = k+1, m) \tag{1.5}$$

$$X_j \geq 0 \quad (i = 1, L; L \leq n) \tag{1.6}$$

де a_{ij}, b_i, c_j - задані постійні величини і k ≤ m.

Визначення 1.2. Функція (1.3.) називається цільовою функцією (або лінійною формою) задачі (1.3) - (1.6), а умови (1.4) - (1.6) - обмеженнями даної задачі.

Визначення 1.3. Стандартною (або симетричною) задачею лінійного програмування називається задача, яка полягає у визначенні максимального значення функції (1.3) при виконанні умов (1.4) і (1.6), де k=m і L=n.

Визначення 1.4. Основною (або канонічною) задачею лінійного програмування називається задача, яка полягає у визначенні максимального значення функції (1.3) при виконанні умов (1.5) і (1.6), де k=0 і L=n.

Визначення 1.5. Сукупність чисел $X=(X_1,X_2...X_n)$, що задовольняють обмеженням (1.5) - (1.6), називається допустимим рішенням (або планом).

Визначення 1.6. План $X^* = (X_1^*, X_2^* \dots, X_n^*)$, при якому цільова функція задачі (1.3) приймає максимальне (мінімальне) значення, називається оптимальним.

Значення цільової функції (1.3) при плані X позначатимемо через F(X). Отже, X^* - оптимальний план задачі, якщо для будь-якого X виконується нерівність

$F(X) \le F(X^*)$

/відповідно $F(X) \ge F(X^*)$/.

Вказані вище три форми задачі ЛП (загальна, стандартна і основна) еквівалентні в тому сенсі, що кожна з них за допомогою стандартних перетворень може бути переписана у формі іншої задачі. Це означає, що якщо є спосіб знаходження рішення одної з вказаних задач, то тим самим може бути визначений оптимальний план будь-якої з трьох задач.

Щоб перейти від однієї форми запису задачі ЛП до іншої, потрібно в загальному випадку вміти, по-перше, зводити задачі мінімізації функції до задачі максимізації, по-друге, переходити від обмежень-нерівностей до обмежень-рівності і навпаки, по-третє, замінювати змінні, які не підпорядковані умові додатності .

У тому випадку, коли потрібно знайти мінімум функції $F = C_1^*X_1+C_2^*X_2+\dots+C_n^*X_n$,можна перейти до знаходження максимуму функції $F_1 = -F = -C_1^*X_1 - C_2^*X_2 - \dots - C_n^*X_n$, оскільки min F = max $(-F)$.

Обмеження-нерівність початкової задачі ЛП, що має вид "≤", можна перетворити в обмеження-рівність додаванням до його лівої частини додаткової невід'ємної змінної, а обмеження-нерівність виду "≥" - в обмеження-рівність відніманням з його лівої частини додаткової невід'ємної змінної.

Таким чином, обмеження-нерівність

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n \leq b_i$$

перетвориться в обмеження-рівність

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n+X_n+\text{i} = b_i(X_n +\text{i} \geq 0) \quad (1.7)$$

а обмеження-нерівність

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n \geq b_i$$

у обмеження-рівність

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n - X_n+\text{i} = b_i(X_n + \text{i} \geq 0) \quad (1.8)$$

В той же час кожне рівняння системи обмежень

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n = b_i \quad (1.9)$$

можна записати у вигляді нерівностей:

$$a_{i1}*X_1+a_{i2}*X_2+\ldots+a_{in}*X_n \leq b_i$$
$$-a_{i1}*X_1-a_{i2}*X_2 -\ldots-a_{in}*X_n \leq - b_i$$

Число невід'ємних змінних, що вводяться, при перетворенні обмежень-нерівностей в обмеження-рівність дорівнює числу перетворюваних нерівностей.

Додаткові змінні, що вводяться, мають цілком певний сенс. Так, якщо в обмеженнях початкової задачі ЛП відображається витрата і наявність виробничих ресурсів, то числове значення додаткової змінної в плані задачі, записаного у формі основної, рівне об'єму невикористаного відповідного ресурсу.

Відзначимо, нарешті, що якщо змінна X_k не підпорядкована умові додатності, то її слід замінити двома невід'ємними змінними U_k і V_k, прийнявши

$$X_k = U_{2k} - V_k.$$

Приклад 1.4 Записати у формі основної задачі лінійного програмування наступну задачу. Знайти максимум функції

$F = 3*X_1 - 2*X_2 - 5*X_4 + X_5$

за умов

$$\begin{cases} 2*X_1+X_3 - X_4+X_5 \le 2; \\ X_1-X_3+2*X_4+X_5 \le 3; \\ 2*X_2+X_3-X_4+2*X_5 \le 6; \\ X_1 + X_4-5*X_5 \ge 8 \\ X_1,X_2,X_3,X_4,X_5 >=0 \end{cases}$$

У даному завданні потрібно знайти максимум функції, а система обмежень містить чотири нерівності. Отже, щоб записати її у формі основної задачі, потрібно перейти від обмежень-нерівностей до обмежень-рівностей. Оскільки число нерівностей, що входять в систему обмежень задачі, рівне чотирьом, то цей перехід може бути здійснений введенням чотирьох додаткових невід'ємних змінних. При цьому до лівих частин кожної з нерівностей виду "<=" відповідна додаткова змінна додається, а з лівих частин кожної з нерівностей виду ">=" віднімається. В результаті, обмеження приймають вигляд

$$\begin{cases} 2*X_1+X_3 - X_4+X_5+X_6 = 2; \\ X_1-X_3+2*X_4+X_5+X_7 = 3; \\ 2*X_2+X_3-X_4+2*X_5+X_8 = 6; \\ X_1 + X_4-5*X_5-X_9 = 8 \\ X_1, X_2..., X_9 >=0 . \end{cases}$$

Отже, дана задача може бути записана у формі основної задачі таким чином : максимізувати функцію $F = 3*X_1-2*X_2-5*X_4+X_5$ за умов

$$\begin{cases} 2*X_1+X_3 - X_4+X_5+X_6 = 2; \\ X_1-X_3+2*X_4+X_5+X_7 = 3; \\ 2*X_2+X_3-X_4+2*X_5+X_8 = 6; \\ X_1 + X_4-5*X_5-X_9 = 8 \\ X_1, X_2..., X_9 >=0 . \end{cases}$$

Приклад 1.5 Записати задачу, що полягає в мінімізації функції F = $-X_1+2*X_2-X_3+X_4$ за умов

$$\begin{cases} 2*X_1 - X_2 - X_3 + X_4 \leq 6; \\ X_1+2*X_2 + X_3 - X_4 \geq 8; \\ 3*X_1 - X_2+2*X_3+2*X_4 \leq 10; \\ -X_1+3*X_2+5*X_3-3*X_4 = 15 \\ X_1, X_2..., X_4 \geq 0 \end{cases}$$

у формі основного задачі ЛП.

У даному завданні потрібно знайти мінімум цільової функції, а система обмежень містить три нерівності.

Отже, щоб записати її у формі основної задачі, замість знаходження мінімуму функції F потрібно знайти мінімум функції F1 = -F при обмеженнях, що виходять з обмежень початкової задачі додаванням до лівих частин кожного з обмежень нерівностей виду "≤" додаткової невід'ємної змінної і відніманням з лівих частин кожного з обмежень нерівностей виду "≥".

Отже, початкова задача може бути записана у формі основної задачі ЛП так: знайти максимум функції
$F_1 = X_1-2*X_2+X_3-X_4$ за умов

$$\begin{cases} 2*X_1 - X_2 - X_3+ X_4+X_5 = 6; \\ X_1+2*X_2 + X_3 - X_4-X_6 = 8; \\ 3*X_1 - X_2+2*X_3+2*X_4+X_7 = 10; \\ -X_1+3*X_2+5*X_3-3*X_4 = 15 \\ X_1, X_2..., X_7 \geq 0 . \end{cases}$$

Властивості основного задачі лінійного програмування

Розглянемо основну задачу ЛП. Як було відмічено вище, вона полягає у визначенні максимального значення функції

$$F = \sum_{j=1}^{n} C_j * X_j$$

за умов

$$\sum_{j=1}^{n} A_{ij} * X_j = B_i,$$

(i=1,m), $X_j \geq 0$ (j=1,n).

Перепишемо цю задачу у векторній формі. Знайти максимум функції

$$F = C*X \tag{1.10}$$

за умов

$$X_1 *P_1 + X_2 *P_2 + ... + X_n *P_n = P_0; \tag{1.11}$$

$$X \geq 0 \tag{1.12}$$

де $C = (C_1, C_2 ..., C_n)$; $X = (X_1, X_2 ..., X_n)$; C*X - скалярний добуток; $P_1, ..., P_n$ і P_0 - m-мірні вектори-стовпці, складені з коефіцієнтів при невідомих і вільних членах системи рівнянь задачі:

$$P_0 = \begin{bmatrix} b_1 \\ b_2 \\ . \\ . \\ . \\ b_n \end{bmatrix} ; \quad P_1 = \begin{bmatrix} a_{11} \\ a_{21} \\ . \\ . \\ . \\ a_{m1} \end{bmatrix} ; \quad P_2 = \begin{bmatrix} a_{12} \\ a_{22} \\ . \\ . \\ . \\ a_{m2} \end{bmatrix} ; \ldots ; \quad P_n = \begin{bmatrix} a_{1n} \\ a_{2n} \\ . \\ . \\ . \\ a_{mn} \end{bmatrix}$$

Визначення 1.6. План $X = (X_1, X_2 ..., X_n)$ називається опорним планом основної задачі лінійного програмування, якщо система векторів P_j, що входять в розкладання (1.11) з додатними коефіцієнтами X_j, лінійно незалежна.

Оскільки вектори P_j є m-мірними, то з визначення опорного плану виходить, що число його додатних компонент не може бути більше, ніж m.

Визначення 1.7. Опорний план називається невиродженим, якщо він містить рівно m додатних компонент, інакше він називається виродженим.

Властивості основної задачі ЛП (1.10) - (1.12) тісним чином пов'язані з властивостями опуклих множин.

Визначення 1.8. Непорожня безліч планів основної задачі лінійного програмування називається багатогранником рішень, а будь-яка кутова точка багатогранника рішень - вершиною.

Теорема. Якщо основна задача лінійного програмування має оптимальний план, то максимальне значення цільова функція задачі приймає в одній з вершин багатогранника рішень. Якщо максимальне значення цільова функція приймає більш ніж в одній вершині, то вона приймає його у всякій точці, яка є опуклою лінійною комбінацією цих вершин.

Теорема. Якщо система векторів P_1, P_2 ..., P_k (k ≤ n) у розкладанні (1.11) лінійно-незалежна і така, що

$$X_1*P_1+X_2*P_2+...+X_k*P_k = P_0 \qquad (1.13)$$

де все X_j>=0, то точкка $X = (X_1, X_2..., X_k,0,...,0)$ є вершиною багатогранника рішень.

Геометрична інтерпретація задачі лінійного програмування

Сформульовані теореми дозволяють зробити наступні висновки. Непорожню безліч планів основної задачі ЛП утворює опуклий багатогранник. Кожна вершина цього багатогранника визначає опорний план. В одній з вершин багатогранника рішень (тобто для одного з опорних планів) значення цільової функції є максимальним (за умови, що функція обмежена зверху на безлічі планів). Якщо максимальне значення функція приймає більш ніж в одній вершині, то це ж значення вона приймає в будь-якій точці, яка є опуклою лінійною комбінацією даних вершин.

Вершину багатогранника рішень, в якій цільова функція приймає максимальне значення, знайти порівняно просто, якщо задача, записана в стандартній формі, містить не більше двох змінних, тобто n-r ≤ 2, де n - число змінних; r - ранг матриці, складеної з коефіцієнтів в системі обмежень задачі.

Знайдемо рішення задачі, що полягає у визначенні максимального значення функції

$$F = C_1 * X_1 + C_2 * X_2 \qquad (1.14)$$

за умов

$$a_{i1} * X_1 + a_{i2} * X_2 \leq b_i \qquad (i=1,k) \qquad (1.15)$$

$$X_j \geq 0 \qquad (j=1,2) \qquad (1.16)$$

Кожна з нерівностей (1.15), (1.16) системи обмежень визначає напівплощина відповідно з граничними прямими

$a_{i1} * X_1 + a_{i2} * X_2 = b_i$; (i=1,k);

$X_1=0$ і $X_2=0$.

В тому випадку, якщо система нерівностей сумісна, область її рішень є безліч точок, що належать всім вказаним напівплощинам. Оскільки безліч точок перетину даних напівплощин - опукла, то областю допустимих рішень задачі (1.14) - (1.16) є опукла безліч точок, яка називається багатокутником рішень (введений раніше термін "багатогранник рішень" зазвичай уживається, якщо n≥3). Сторони цього багатокутника лежать на прямих, рівняння яких виходять з початкової системи обмежень заміною знаків нерівностей на знаки точної рівності.

Таким чином, початкова задача ЛП полягає в знаходженні такої точки багатокутника рішень, в якій цільова функція F приймає максимальне значення. Ця точка існує тоді, коли багатокутник рішень не порожній і на ньому цільова функція обмежена зверху.

За вказаних умов в одній з вершин багатокутника рішень цільова функція приймає максимальне значення. Для визначення даної вершини побудуємо лінію рівня $C_{11} * X_1 + C_2 * X_2 = h$

(де h - деяка постійна), що проходить через багатокутник рішень і пересуватимемо її у напрямі вектора $C=(C_1, C_2)$ до тих пір, поки вона не пройде через останню її загальну точку з багатокутником рішень. Координати вказаної точки і визначають оптимальний план даної задачі.

Закінчуючи розгляд геометричної інтерпретації задачі (1.14) - (1.16), відзначимо, що при знаходженні її рішення можуть зустрітися випадки, зображені на рис.1.1-1.4. Рис.1.1. характеризує такий випадок,

коли цільова функція приймає максимальне значення в єдиній точці А. З рис.1.2. видно, що максимальне значення цільова функція приймає в будь-якій точці відрізка АВ. На рис.1.3. зображений випадок, коли цільова функція не обмежена зверху на безлічі допустимих рішень, а на рис.1.4. - випадок, коли система обмежень задачі несумісна.

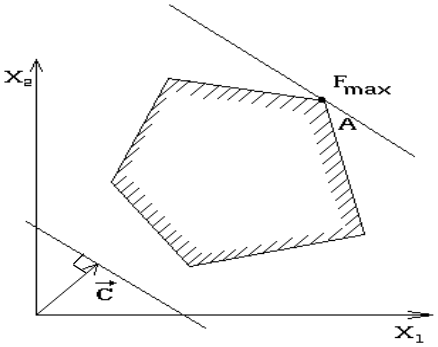


Рисунок 1.1

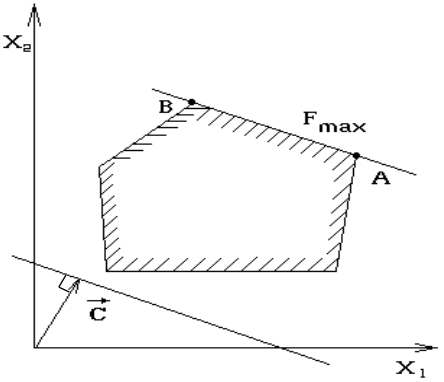


Рисунок 1.2

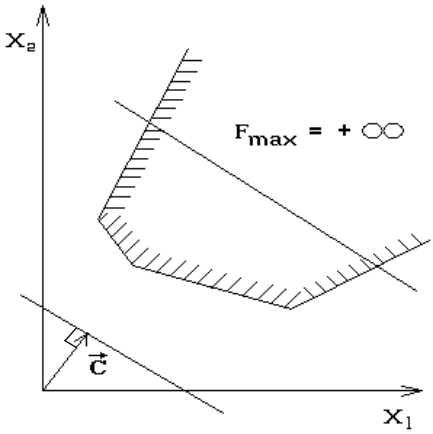


Рисунок 1.3

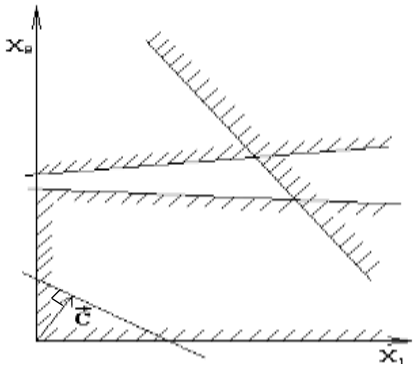


Рисунок 1.4

Відзначимо, що знаходження мінімального значення лінійної функції при даній системі обмежень відрізняється від знаходження її мінімального значення при тих же обмеженнях лише тим, що лінія рівня $C_1*X_1+C_2*X_2$ = h пересувається не у напрямі вектора C= (C_1, C_2), а в протилежному напрямі. Таким чином, відмічені вище випадки, що зустрічаються при знаходженні максимального значення цільової функції, мають місце і при знаходженні її мінімального значення.

Отже, знаходження рішення задачі ЛП (1.14) - (1.16) на основі її геометричної інтерпретації включає наступні етапи.

1. Будують прямі, рівняння яких виходять в результаті заміни в обмеженнях (1.15) і (1.16) знаків нерівностей на знаки точної рівності.

2. Знаходять напівплощини, які визначаються кожним з обмежень задачі.

3. Знаходять багатокутник рішень.

4. Будують вектор C= (C_1, C_2).

5. Будують пряму $C_1*X_1+C_2*X_2$ = h, що проходить через багатокутник рішень.

6. Пересувають пряму $C_1*X_1+C_2*X_2$ = h у напрямі вектора C, внаслідок чого знаходять точку (точки), в яких цільова функція приймає максимальне значення, або встановлюють необмеженість зверху функції на безлічі планів.

7.Визначають координати точки максимуму функції і обчислюється значення цільової функції в цій точці.

Приклад 1.6 Для виробництва двох видів виробів А і В підприємство використовує три види комплектуючих. Норми витрат комлектуючих кожного виду на виготовлення одиниці продукції даного виду приведені в табл.1.2.

Таблиця 1.2

Види комплектуючих	Витрата комплектуючих, шт, на один виріб		Загальна кількість комплектуючих, шт
	А	В	
I	12	4	300
II	4	4	120
III	3	12	252
Прибуток від реалізації одного виробу, грн.	30	40	

У табл.1.2. вказані також прибуток від реалізації одного виробу кожного виду і загальна кількість комплектуючих даного виду, яка може бути використана підприємством.

Враховуючи, що вироби А і В можуть вироблятися в будь-яких країнах, потрібно скласти такий план їх випуску, при якому прибуток підприємства від реалізації всіх виробів виявиться максимальним.

Припустимо, що підприємство виготовить X_1 виробів виду А і X_2 виробів виду В. Оскільки виробництво продукції обмежене комплектуючими кожного виду, що є у розпорядженні підприємства і кількість виробів, що виготовляються, не може бути від'ємною, повинні виконуватися нерівності

$$\begin{cases} 12*X_1 + 4*X_2 \le 300; \\ 4*X_1 + 4*X_2 \le 120; \\ 3*X_1 + 12*X_2 \le 252 \\ X_1 \ , X_2 \ge 0 \ . \end{cases}$$

Загальний прибуток від реалізації Х1 виробів виду А і Х2 виробів виду В складе F = 30*X1 + 40*X2.

Таким чином, ми приходимо до наступної математичної задачі. Серед всіх невід'ємних рішень даної системи лінійних нерівностей потрібно знайти таке, при якому функція F приймає максимальне значення.

Знайдемо рішення сформульованої задачі, використовуючи її геометричну інтерпретацію. Спочатку визначимо багатокутник рішень. Для цього в нерівностях системи обмежень і умовах додатності змінних знаки нерівностей замінимо на знаки точної рівності і знайдемо відповідні прямі:

$12*X_1 + 4*X_2 = 300$ (I)
$4*X_1 + 4*X_2 = 120$ (II)
$3*X_1 + 12*X_2 = 252$ (III)
$X_1 = 0$ (IV)
$X_2 = 0.$ (V).

Ці прямі зображені на рис.1.5. Кожна з побудованих прямих ділить площину на дві напівплощини. Координати точок однієї напівплощини задовольняють початковій нерівності, а інший - ні. Щоб визначити шукану напівплощину, потрібно узяти яку-небудь точку що належить одній з напівплощин, і перевірити, чи задовольняють її координати даній нерівності.

Якщо координати узятої точки задовольняють даній нерівності, то шуканою є та напівплощина, якій належить ця точка, інакше - інша напівплощина.

Знайдемо, наприклад, напівплощину , яка визначена нерівністю $12*X_1 + 4*X_2 < 300$ (на рис.1.5. це пряма I), візьмемо яку-небудь точку, що належить одній з напівплощин, наприклад, початок координат - точку (0;0). Координати цієї точки задовольняють нерівності $12*0 + 4*0 < 300$; таким чином, напівплощина якій належить точка (0;0), визначається нерівністю

$12*X_1+4*X_2 \leq 300$. Це і показано стрілками на рис.1.5.

Перетин отриманих напівплощин і визначає багатокутник рішень даної задачі.

Як видно з рис.1.5, багатокутником рішень є п'ятикутник OABCD. Координати будь-якої точки, що належить цьому п'ятикутнику, задовольняють даній системі нерівностей і умові додатності змінних. Тому, сформульована задача буде вирішена, якщо ми зможемо знайти точку, що належить п'ятикутнику OABCD, в якій функція F приймає максимальне значення.

Щоб знайти вказану точку, побудуємо вектор C = (30;40) і пряму $30*X_1 + 40*X_2$= h, де h - деяка постійна, при якій пряма $30*X_1 + 40*X_2$= h має загальні точки з багатокутником рішень. Покладемо, наприклад, h = 480 і побудуємо пряму $30*X_1 + 40*X_2 =$ 480 (рис.1.5).

Якщо тепер узяти яку-небудь точку, що належить побудованій прямій і багатокутнику рішень, то її координати визначають такий план виробництва виробів A і B, при якому прибуток від їх реалізації рівний 480 грн. Далі, вважаючи h рівним деякому числу, більшому ніж 480, ми будемо отримувати різні паралельні прямі. Якщо вони мають загальні точки з багатокутником рішень, то ці точки визначають плани виробництва деталей A і B, при яких прибуток від їх реалізації перевершить 480 грн.

Переміщаючи побудовану пряму $30*X_1 + 40*X_2 = 480$ у напрямі вектора C, бачимо, що останньою загальною точкою її з

багатокутником рішень задачі служить точка В. Координати цієї точки і визначають план випуску виробів А і В, при якому прибуток від їх реалізації є максимальним.

Знайдемо координати точки В як точки перетину прямих II і III. Отже, її координати задовольняють рівнянням цих прямих

$$\begin{cases} 4*X_1 + 4*X_2 = 120 \\ 3*X_1 + 12*X_2 = 252. \end{cases}$$

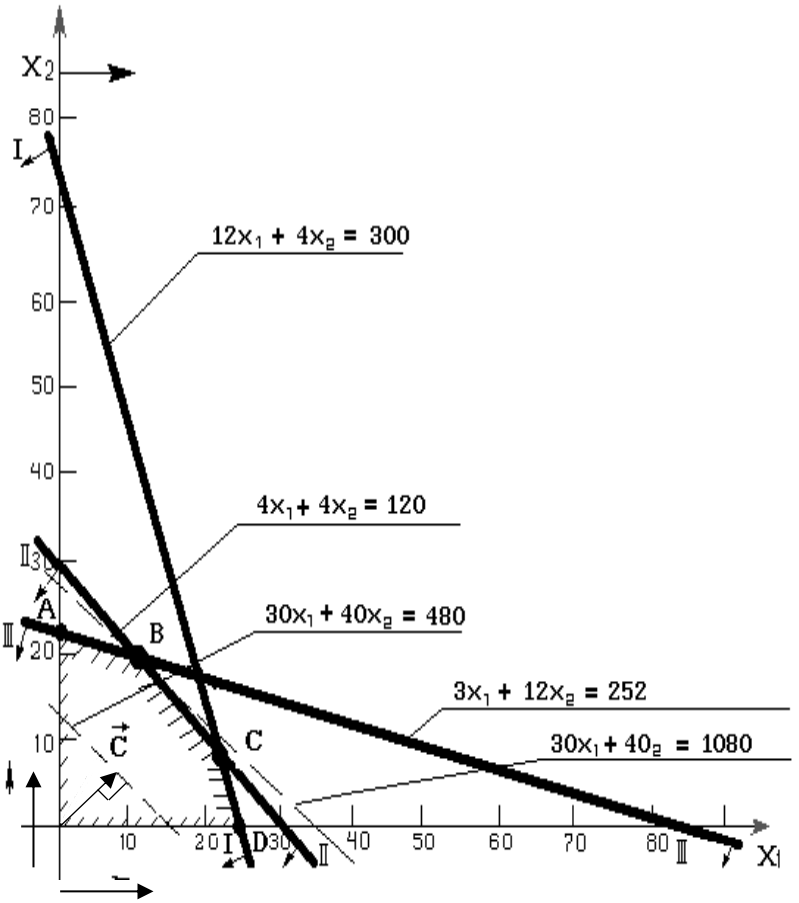


Рисунок 1.5

Вирішивши цю систему рівнянь, отримаємо X_1'=12, X_2'=18. Отже, якщо підприємство виготовить 12 виробів виду А і 18 виробів виду В, то воно отримає максимальний прибуток, F_{max}= 30*12 + 40*18 = 1080 грн.

1.2 Контрольні запитання

1.2.1 Приведіть приклад використання ЛП . Складіть математичну модель задачі.

1.2.2 Сформулюйте загальну задачі ЛП.

1.2.3 Дайте визначення стандартної (симетричної) і основної (канонічної) задачі ЛП.

1.2.4 Що називається допустимими і оптимальним рішеннями задачі ЛП? Дайте геометричну інтерпретацію.

1.2.5 Як визначити кількість додаткових невід'ємних змінних, які слід вводити при переході від обмежень-нерівностей задачі ЛП до обмежень-рівності. Який їх сенс?

1.2.6 Як можна звести задачі мінімізації цільової функції до задачі максимізації?

1.2.7 Яким чином замінюються змінні, що не підкоряються умові додатності?

1.2.8 Як підготувати і вирішити задачу ЛП на основі її геометричної інтерпретації?

1.2.9 Дайте геометричну інтерпретацію можливих при рішенні задачі ЛП випадків: коли цільова функція приймає максимальне значення в єдиній точці або на відрізку; коли цільова функція не обмежена зверху на безлічі рішень; коли система обмежень задачі несумісна.

2 ВИРІШЕННЯ ЗАДАЧІ ЛІНІЙНОГО ПРОГРАМУВАННЯ СИМПЛЕКСНИМ МЕТОДОМ

2.1 Теоретичні відомості

Знаходження оптимального плану задачі лінійного програмування симплексним методом

Вирішення задачі лінійного програмування можна знайти симплексним методом. Перш ніж застосовувати цей метод, слід записати початкову задачу у формі основної задачі ЛП, якщо вона не має такої форми запису.

Симплексний метод рішення задачі ЛП заснований на переході від одного опорного плану до іншого, при якому значення цільової функції зростає (за умови, що дана задача має оптимальний план і кожен її опорний план є невиродженим).

Вказаний перехід можливий, якщо відомий який-небудь початковий опорний план. Розглянемо задачу, для якої цей план можна безпосередньо записати.

Нехай потрібно знайти максимальне значення функції

$$F = C_1 * X_1 + C_2 * X_2 + \ldots + C_n * X_n$$

за умов

$$\begin{cases} X_1 + A_{1m} + i * X_m + i + \ldots + A_{1n} * X_n = B_1 \\ X_2 + A_{1m} + i * X_m + i + \ldots + A_{1n} * X_n = B_2 \end{cases}$$

представлених у вигляді лінійної комбінації векторів даного базису.

Нехай

$$P_j = \sum_{i=1}^{m} X_{ij} * P_i \quad (j=0,n).$$

Покладемо

$$Z_j = \sum_{i=1}^{m} C_i * X_{ij} \qquad (j=1,n); \qquad j = Z_j - C_j \quad (j=1,n).$$

Оскільки вектори P_1, P_2..., P_m - одиничні, то

$$X_{ij} = A_{ij} \quad \text{і} \quad Z_j = \sum_{i=1}^{m} C_i * A_{ij} \text{ , а}$$

$$j = \sum_{i=1}^{m} C_i * A_{ij} - C_j$$

Теорема 2.1. (ознака оптимальності опорного плану).

Опорний план $X'=(X_1'; X_2';...; X_m';0;0;...;0)$ задачі є оптимальним, якщо $j \geq 0$ для будь-якого j (j=1,n).

Теорема 2.2. Якщо к<0 для деякого j=к і серед чисел $A_{ік}$ (i=1,m) немає додатних ($A_{ik} \leq 0$),то цільова функція задачі не обмежена на безлічі її планів.

Теорема 2.3. Якщо опорний план X задачі невироджений і k<0, але серед чисел $A_{ік}$ є додатні (не всі A_{ik}<=0), то існує опорний план X' такий, що $F(X')> F(X)$.

Сформульовані теореми дозволяють перевірити, чи є знайдений опорний план оптимальним і виявити доцільність переходу до нового опорного плану.

Дослідження опорного плану на оптимальність, а також подальший обчислювальний процес зручніше вести, якщо умови задачі і первинні дані, отримані після визначення початкового опорного плану, записати, як показано в табл.2.1.

Таблиця 2.1

i	Базис	$C_б$	P_0	C_1	C_2	…	C_r	…	C_m	C_{m+1}	…	C_k	…	C_n
				P_1	P_2	…	P_r	…	P_m	P_{m+1}	…	P_k	…	P_n
1	P_1	C_1	B_1	1	0	…	0	…	0	A_{1m+1}	…	A_{1k}	…	A_{1n}
2	P_2	C_2	B_2	0	1	…	0	…	0	A_{2m+1}	…	A_{2k}	…	A_{2n}
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
r	P_r	C_r	B_r	0	1	…	0	…	0	A_{rm+1}	…	A_{rk}	…	A_m
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
m	P_m	C_m	B_m	0	0	…	1	…	1	A_{mm+1}	…	A_{mk}	…	A_{mn}
m+1			F_0	0	0	…	0	…	0	Δ_{1m+1}	…	Δ_k	…	Δ_n

У стовпці $C_б$ цієї таблиці записують коефіцієнти при невідомих цільової функції, що мають ті ж індекси, що і вектори даного базису.

У стовпці P_0 записують додатні компоненти початкового опорного плану, у ньому ж в результаті обчислень отримують додатні компоненти опорного плану. Стовпці векторів P_j являють собою коефіцієнти розкладання цих векторів за векторами даного базису.

У табл.2.1. перші m рядків визначаються початковими даними задачі, а показники (m+1) -го рядка обчислюють.

У цьому рядку в стовпці вектора P_0 записують значення цільової функції, яке вона приймає при даному опорному плані, а в стовпці вектора P_j - значення $j = Z_j - C_j$.

Значення Z_j знаходиться як скалярний добуток вектора P_j (j=1,n) на вектор $C_б = (C_1, C_2..., C_m)$:

$$Z_j = \sum_{i=1}^{m} C_i * A_{ij}\,, \qquad (\text{j=1,n}).$$

Значення F_0 рівне скалярному добутку вектора P_0 на вектор $C_б$:

$$F_0 = \sum_{i=1}^{m} C_i * B_i$$

Після заповнення табл.2.1. початковий опорний план перевіряють на оптимальність. Для цього проглядають елементи (m+1) -го рядка таблиці. В результаті може мати місце один з наступних трьох випадків:

а) $\Delta_j \geq 0$ для j=m+1,m+2...,n (при $\overline{j=1,m}$ $Z_j=C_j$ i, відповідно, j=0). Тому в даному випадку числа $j \geq 0$ для всіх j від 1 до n;

б) $\Delta_j < 0$ для деякого j та всі відповідні цьому індексу величини $A_{ij} \leq 0$ (i=1,m);

в) $\Delta_j < 0$ для деяких індексів j та для кожного такого j хоча б одне з чисел A_{ij} додатне.

У першому випадку на підставі ознаки оптимальності початковий опорний план є оптимальним. У другому випадку цільова функція не обмежена зверху на безлічі планів, а в третьому випадку можна перейти від початкового плану до нового опорного плану, при якому значення цільової функції збільшиться. Цей перехід від одного опорного плану до іншого здійснюється виключенням з початкового базису якого-небудь з векторів і введенням в нього нового вектору. Як вектор, що вводиться в базис, можна узяти будь-який з векторів P_j , що має індекс j, для якого $\Delta_j < 0$. Нехай, наприклад, $\Delta_k < 0$ і вирішено ввести в базис вектор P_k. Для визначення вектора, який підлягає виключенню з базису, знаходять min (B_i/A_{ik}) для всіх $A_{ik} > 0$. Хай цей мінімум досягається при i = r. Тоді з базису виключають вектор P_r, а число A_{rk} називають розв'язуючим елементом. Стовпець і рядок, на перетині яких знаходиться розв'язуючий елемент, називають направляючим.

Після виділення розв'язуючого рядка і розв'язуючого стовпця знаходять новий опорний план X і коефіцієнти розкладання векторів Pj через вектори нового базису, відповідні новому опорному плану. Додатні компоненти нового опорного плану обчислюють по формулам

$$B_i' = \begin{cases} B_i - (B_r/A_{rk}) * A_{ik} & \text{при } i \neq r, \\ B_r/A_{rk} & \text{при } i=r, \end{cases} \qquad (2.1)$$

Коефіцієнти розкладання векторів P_j через вектори нового базису, що відповідають новому опорному плану, по формулам

$$A_{ij}' = \begin{cases} A_{ij}-(A_{rj}/A_{rk})* A_{ik} & \text{при } i<>r; \\ A_{rj}/A_{rk} & \text{при } i=r. \end{cases} \qquad (2.2)$$

Після обчислення B_i' і A_{ij}' згідно (2.1) і (2.2) їх значення заносять в табл.2.2.

Таблиця 2.2

i	Базис	$C_б$	P_0	C_1	C_2	...	C_r	...	C_m	C_{m+1}	...	C_k	...	C_n
				P_1	P_2	...	P_r	...	P_m	P_{m+1}	...	P_k	...	P_n
1	P_1	C_1	B_1'	1	0	...	A_{1r}'	...	0	A_{1r+1}'	...	0	...	A_{1n}'
2	P_2	C_2	B_2'	0	1	...	A_{2r}'	...	0	A_{2r+1}'	...	0	...	A_{2n}'
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
r	P_r	C_r	B_r'	0	1	...	A_{rr}'	...	0	A_{rm+1}'	...	1	...	A_{rn}'
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
m	P_m	C_m	B_m'	0	0	...	A_{mr}'	...	1	A_{mn+1}'	...	0	...	A_{mn}'
m+1			F_0'	0	0	...	X_r'-C_r	...	0	Z_{m+1}'-C_{m+1}	...	0	...	Z_n'-C_n

Елементи (m+1) -го рядка цієї таблиці можуть бути обчислені по формулам

$$F_0' = F_0 - (B_r/A_{rk})*\Delta_k; \qquad (2.3)$$
$$\Delta_j' = \Delta_j - (A_{rj}/A_{rk})* \Delta_k \qquad (2.4)$$

або на підставі їх визначення.

Наявність двох способів знаходження елементів (m+1) -го рядка дозволяє здійснювати контроль правильності обчислень, що проводяться.

З (2.3) витікає, що при переході від одного опорного плану до іншого найдоцільніше ввести в базис вектор P_j, що має індекс j, при якому максимальним по абсолютній величині є число $(B_r/A_{rj})^*$ j (j > 0, $A_{rj} > 0$). Проте в цілях спрощення обчислювального процесу надалі будемо вектор, що вводиться в базис, визначати, виходячи з максимальної абсолютної величини від'ємних чисел j. Якщо ж таких чисел декілька, то в базис вводитимемо вектор, що має такий же індекс, як і максимальне з чисел Cj, що визначаються даними числами j (j<0).

Отже, перехід від одного опорного плану до іншого зводиться до переходу від однієї симплекс-таблиці до іншої. Елементи нової симплекс-таблиці можна обчислити як за допомогою формул (2.1) - (2.4), так і по правилам, що безпосередньо витікають з них. Ці правила полягають в наступному.

У стовпцях векторів, що входять в базис, на перетині рядків і стовпців однойменних векторів проставляють одиниці, а решта всіх елементів даних стовпців покладають рівними нулю.

Елементи векторів P_0 і P_j в рядку нової симплекс-таблиці, в якій записаний вектор, що вводиться в базис, отримують з елементів цього ж рядка початкової таблиці діленням їх на величину розв'язуючого елементу. У стовпці $C_б$ в рядку вектора, що вводиться, проставляють величину C_k, де k - індекс вектора, що вводиться.

Решта елементів стовпців векторів P_0 і P_j новою симплекс-таблиці, обчислюють за правилом трикутника. Для обчислення кожного з цих елементів знаходять три числа.

1. Число, що стоїть в початковій симплекс-таблиці на місці шуканого елементу нової симплекс-таблиці.

2. Число, що стоїть в початковій симплекс-таблиці на перетині рядка, в якому знаходиться шуканий елемент нової

симплекс-таблиці, і стовпця, відповідного вектору, що вводиться в базис.

3. Число, що стоїть в новій симплекс-таблиці на перетині

стовпця, в якому стоїть шуканий елемент, і рядка вектора, що знов вводиться в базис. Як відзначено раніше, цей рядок виходить з

рядка початкової симплекс-таблиці діленням її елементу на розв'язуючий елемент.

Ці три числа утворюють своєрідний трикутник, дві вершини якого відповідають числам, що знаходяться в початковій симплекс-таблиці, а третя - числу, що знаходиться в новій симплекс-таблиці. Для визначення шуканого елементу нової симплекс-таблиці з першого числа віднімають добуток другого і третього.

Після заповнення нової симплекс-таблиці переглядають елементи (m+1) -го рядка. Якщо всі Z_j'-$C_j \geq 0$, то новий опорний план є оптимальним. Якщо ж серед вказаних чисел є від'ємні, то, використовуючи описану вище послідовність дій, знаходять новий опорний план. Цей процес продовжують до тих пір, поки або не отримують оптимальний план задачі, або не встановлюють її нерозв'язність.

При знаходженні рішення задачі ЛП припускали, що ця задача має опорні плани і кожен такий план є невиродженим. Якщо ж задача має вироджені опорні плани, то на одній з ітерацій одна або декілька змінних опорного плану можуть виявитися рівними нулю.

Таким чином, при переході від одного опорного плану до іншого значення функції може залишитися тим самим. Більш того, можливий випадок, коли функція зберігає своє значення протягом декількох ітерацій, а також можливе повернення до первинного базису.

В останньому випадку зазвичай говорять, що відбулося зациклення. Проте при рішенні задач цей випадок зустрічається дуже рідко, тому на ньому зупинятися не будемо.

2.1.2 Приклад вирішення задачі лінійного програмування симплексним методом

Для виготовлення різних виробів А, В і С підприємство використовує три різні види сировини. Норми витрати сировини на виробництво виробів кожного виду, ціна одного виробу А, В і С, а також загальна кількість сировини кожного виду, яка може бути використана підприємством, наведені в табл.2.3.

Таблиця 2.3

Тип сировини	Норми витрат сировини, кг на один виріб			Загальний фонд сировини, кг
I	8	5	2	360
II				192
III				240
Ціна одного виробу, грн.		0	6	-

Вироби A, B і C можуть проводитися в будь-яких співвідношеннях (збут забезпечений), але виробництво обмежене виділеною підприємству сировиною кожного виду.

Скласти план виробництва виробів, при якому загальна вартість всієї проведеної підприємством продукції є максимальною.

Складемо математичну модель задачі. Шуканий випуск виробів A позначимо через X_1, виробів B - через X_2, виробів C - через X_3. Оскільки є обмеження на виділений підприємству фонд сировини сировини сировини кожного виду, змінні X_1, X_2, X_3 повинні задовольняти наступній системі нерівностей:

$$\begin{cases} 18*X_1+15*X_2+12*X_3 <= 360; \\ 6*X_1+4*X_2+8*X_3 <= 192; \\ 5*X_1+3*X_2+3*X_3 <= 180. \end{cases} \quad (2.5)$$

Загальна вартість проведеної підприємством продукції за умови випуску X_1 виробів A, X_2 виробів B і X_3 виробів C складає

$$F = 9*X_1 + 10*X_2 + 16*X_3. \quad (2.6)$$

За своїм змістом змінні X_1, X_2, X_3 можуть приймати тільки невід'ємні значення

$$X_1, X_2, X_3 \geq 0. \qquad (2.7)$$

Таким чином, приходимо до наступного математичного задачі: серед всіх невід'ємних рішень системи нерівностей (2.5) потрібно знайти таке, при якому функція (2.6) приймає максимальне значення.

Запишемо цю задачу у формі основної задачі ЛП. Для цього перейдемо від обмежень-нерівностей до обмежень-рівностей.

Введемо три додаткові змінні, внаслідок чого обмеження запишуться у вигляді системи рівнянь

$$\begin{cases} 18*X_1+15*X_2+12*X_3+X_4 = 360; \\ 6*X_1+4*X_2+8*X_3+X_5 = 192; \\ 5*X_1+3*X_2+3*X_3+X_6 = 180. \end{cases}$$

Введені додаткові змінні означають не використовувану при даному плані виробництва кількість сировини того або іншого виду. Наприклад, X4 - це невживана кількість сировини I виду.

Перетворену систему рівнянь запишемо у векторній формі:

$X_1*P_1 + X_2*P_2 + X_3*P_3 + X_4*P_4 + X_5*P_5 + X_6*P_6 = P_0$,
де

$$P_1 = \begin{bmatrix} 18 \\ 6 \\ 5 \end{bmatrix}; P_2 = \begin{bmatrix} 15 \\ 4 \\ 3 \end{bmatrix}; \quad P_3 = \begin{bmatrix} 12 \\ 8 \\ 3 \end{bmatrix}; \quad P_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix};$$

$$P_5 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}; \quad P_6 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; \quad P_0 = \begin{bmatrix} 360 \\ 192 \\ 180 \end{bmatrix}.$$

Оскільки серед векторів P_1, P_2, P_3 є три одиничні вектори (нагадаємо, що одиничним називається вектор, модуль якого рівний одиниці), для даного задачі можна безпосередньо записати опорний план. Таким є план X = (0;0;0;360;192;180), що визначений системою

тривимірних одиничних векторів P_4, P_5, P_6, які утворюють базис тривимірного векторного простору.

Складаємо симплексну таблицю для I ітерації (табл.2.4), підраховуємо значення F_0, Z_j-C_j і перевіряємо початковий опорний план на оптимальність:

F_0= (C, P_0) =0; Z_1= (C_1, P_1) = 0; Z_2 = (C_2, P_2) = 0;
Z_3 = (C_3, P_3) = 0; Z_1-C_1 = 0-9 = -9; Z_2-C_2 = 0-10 = -10;
Z_3-C_3 = -16.

Для векторів базису, тобто для векторів P_4, P_5, P_6

$$Z_j - C_j = 0.$$

Таблиця 2.4

i	Базис	СБ	P_0	9	10	16	0	0	0
				P_1	P_2	P_3	P_4	P_5	P_6
1	P_4	0	360	18	15	12	1	0	0
2	P_5	0	192	6	4	8	0	1	0
3	P_6	0	180	5	3	3	0	0	1
4			0	-9	-10	-16	0	0	0

Як видно з табл.2.4, значення всіх основних змінних X_1, X_2, X_3 рівні нулю, а додаткові змінні X_4, X_5, X_6 приймають свої значення відповідно до обмежень задачі.

Ці значення змінних відповідають такому "плану", при якому нічого не проводиться, сировина не використовується і значення цільової функції рівне нулю (тобто вартість проведеної продукції відсутня). Цей план, звичайно, не є оптимальним.

Це видно і з 4-го рядка табл.2.4, оскільки в ній є три від'ємні числа: Z_1-C_1 = -9; Z_2-C_2 = -10; Z_3-C_3 = -16.

Від'ємні числа свідчать не тільки про можливість збільшення загальної вартості виготовляємої продукції, але і показують, на скільки збільшиться ця сума при введенні в план одиниці того або іншого виду продукції.

Так, число -9 означає, що при включенні в план виробництва одного виробу А забезпечується збільшення випуску продукції на 9 грн. Якщо включити в план виробництва по одному виробу В і С, то

загальна вартість продукції, що виготовляється, зростає відповідно на 10 і 16 грн. Тому найбільш доцільним є включення в план виробництва виробів С. Те ж необхідно зробити і на підставі формальної ознаки симплексного методу, оскільки максимальне по абсолютній величині від'ємне число стоїть в 4-му рядку стовпця вектора P_3. Отже, в базис введемо вектор P_3.

Визначимо вектор, що підлягає виключенню з базису. Для цього знаходимо

θ_0= min(B_i/A_{i3}) для $A_{i3} > 0$, тобто

θ_0 = min(360/12; 192/8; 180/3)= 192/8.

Знайшовши число 192/8 = 24, ми тим самим визначили, яка кількість виробів С підприємство може виготовляти з урахуванням норм витрати і наявних об'ємів сировини кожного виду. Оскільки сировини даного виду відповідно є 360, 192 і 180 кг, а на один виріб С потрібно витратити сировини кожного виду відповідно 12, 8 і 3 кг, то максимальне число виробів С, яке може бути виготовлене підприємством рівне min (360/12; 192/8; 180/3 = 192/8 = 24), тобто обмежуючим чинником для виробництва виробів С є наявний об'єм сировини II виду. З урахуванням його наявності підприємство може виготовити 24 вироби С. При цій сировині II виду буде повністю реалізовано.

Отже, вектор P_5 підлягає виключенню з базису. Стовпець вектора P_3 і 2-й рядок є направляючим. Складемо таблицю для II ітерації (табл.2.5).

Таблиця 2.5

i	Базис	СБ	P_0	9	10	16	0	0	0
				P_1	P_2	P_3	P_4	P_5	P_6
1	P_4	0	72	9	9	0	1	-3/2	0
2	P_3	16	24	3/4	1/2	1	0	1/8	0
3	P_6	0	108	11/4	3/2	0	0	-3/8	1
4			384	3	-2	0	0	2	0

Спочатку заповнюємо рядок вектора, знов введеного в базис, тобто рядок, номер якого співпадає з номером направляючого рядка.

Тут направляючим є другий рядок. Елементи цього рядка табл.2.5. отримуються з відповідних елементів табл. 2.4. діленням їх на розв'язувальний елемент (тобто на 8). При цьому в стовпці C_6 записується коефіцієнт $C_3 = 16$, що стоїть в стовпці вектора Р2, що вводиться в базис.

Потім заповнюємо елементи стовпців для векторів P_4, P_3, P_6, що входять в новий базис. У цих стовпцях на перетині рядків і стовпців однойменних векторів проставляємо одиниці, а решта всіх елементів вважаємо рівними нулю.

Для визначення решти елементів табл.2.5 застосовуємо правило трикутника.

Обчислимо елементи табл.2.5, що стоять в стовпці вектора P_0.

Перший з них знаходиться в 1-му рядку цього стовпця. Для його обчислення знаходимо три числа:

- число, що стоїть в табл.2.4 на перетині стовпця вектора P_0 і 1-го рядка (360);

- число, що стоїть в табл.2.4 на перетині стовпця вектора P_3 і 1-го рядка (12);

- число, що стоїть в табл.2.5 на перетині стовпця вектора P_0 і 2-го рядка (24).

Віднімаючи з першого числа добуток двох інших, знаходимо шуканий елемент: 360-12*24=72; записуємо його в 1-му рядку стовпця P_0 табл.2.5.

Другий елемент вектора P_0 табл.2.5 у нас обчислений раніше.

Для обчислення третього елементу стовпця вектора P_0 також знаходимо три числа. Перше з них (180) знаходиться на перетині 3-го рядка і стовпця вектора P_0 табл.2.4, друге (3) - на перетині 3-го рядка і стовпця вектора P_3 табл.2.4, третє (24) - на перетині 2-го рядка і стовпця вектора P_0 табл.2.5. Отже, вказаний елемент є 180-24*3=108. Число 108 записуємо в третьому рядку стовпця вектора P_0 табл.2.5. Значення F_0 в 4-му рядку табл.2.5 вектора P_0 можна знайти двома способами:

- за формулою F_0 = (C, P_0), тобто F_0 = 0*72+16*24+0*108 = 384;

- за правилом трикутника; в даному випадку трикутник утворений числами 0, -16, 24. Цей спосіб приводить до того ж результату: 0-(-16)*24 = 384.

При визначенні за правилом трикутника елементів стовпця вектора P_0 третє число (24), що стоїть в нижній вершині трикутника, весь час залишалося незмінним і мінялися лише перші два числа. Врахуємо це при знаходженні елементів стовпця вектора P_1 табл.2.5. Для обчислення вказаних елементів перші два числа беремо із стовпців векторів P_1 і P_3 табл.2.4, а третє число - з табл.2.5. Це число (3/4) стоїть на перетині 2-го рядка і стовпця вектора P_1 останньої таблиці. В результаті отримуємо значення шуканих елементів: 18 - 12 * (3/4) = 9, 5 - 3 * (3/4) = 11/4.

Число Z_1-C_1 в 4-му рядку стовпця вектора P_1 табл.2.5 можна знайти двома способами:

- за формулою Z_1-C_1=(3, P_1)- C_1 маємо 0*9+16*3/4+0*11/4-9=3;
- за правилом трикутника отримаємо -9 - (-16) * (3/4) = 3.

Аналогічно знаходимо елементи вектора P_2. Елементи стовпця вектора P_5 обчислюємо за правилом трикутника. Проте, побудовані для визначення цих елементів трикутники виглядають інакше.

При обчисленні елементу 1-го рядка вказаного стовпця виходить трикутник, утворений числами 0, 12 і 1/8. Отже, шуканий елемент рівний 0 - 12 * (1/8) = -3/2. Елемент, що стоїть в 3-му рядку даного стовпця, дорівнює 0 -3 * (1/8) = -3/8.

Після закінчення розрахунку всіх елементів табл.2.5 в ній отримані новий опорний план і коефіцієнти розкладання векторів P_j (j=1,6) через базисні вектори P_3, P_4, P_6 і значення ΔJ і F_o. Як видно з цієї таблиці, новим опорним планом задачі є план X = (0; 0; 24; 72; 0; 108). При даному плані виробництва виготовляються 24 вироби С і залишається невикористаним 72 кг сировини I виду і 106 кг сировини III виду. Вартість всієї вироблюваної при цьому продукції дорівнює 384 грн. Вказані числа записані в стовпці вектора P_0 табл.2.5. Як видно, дані цього стовпця є параметрами даної задачі, хоча вони зазнали значні зміни.

Змінилися дані і інших стовпців, а їх смисловий зміст став складнішим. Візьмемо, наприклад, дані стовпця вектора P_2. Число 1/2 в 2-му рядку цього стовпця показує, на скільки слід зменшити виготовлення виробів, якщо запланувати випуск одного виробу.

Числа 9 і 3/2 в 1-му і 3-му рядках вектора P_2 показують відповідно, скільки буде потрібно сировини I і III виду при включенні в план виробництва одного виробу В, а число -2 у 4-му рядку показує,

що якщо буде запланований випуск одного виробу В, то це забезпечить збільшення випуску продукції у вартістю на 2 грн. Іншими словами, якщо включити в план виробництва продукції один виріб, то це спричинить зменшення випуску виробу С на 1/2 од. і зажадає додаткових витрат 9 кг сировини I виду і 3/2 кг III виду, а загальна вартість продукції, що виготовляється, відповідно до нового опорного плану зростає на 2 грн.

Таким чином числа 9 і 3/2 виступають як би новими "нормалями" витрат сировини I і III виду на виготовлення одного виробу В (як видно з табл.2.4, раніше вони були рівні 15 і 3), що пояснюється зменшенням випуску виробів С.

Такий же зміст мають і дані стовпця вектора P_1 табл.2.5.

Декілька інший зміст мають числа, записані в стовпці вектора P_5. Число 1/8 в 2-му рядку цього стовпця показує, що збільшення об'ємів сировини II виду на 1 кг дало б можливість збільшити випуск виробів на 1/8 од. Одночасно було б потрібно додатково 3/2 кг сировини I виду і 3/8 кг сировини III виду. Збільшення випуску виробів С на 1/8 од. приведе до зростання випуску продукції на 2 грн.

З викладеного змісту даних табл.2.5 виходить, що знайдений на II ітерації план задачі не є оптимальним. Це видно і з 4-го рядка табл.2.5, оскільки в стовпці вектора P_2 цього рядка стоїть від'ємне число -2. Значить, в базис слід ввести вектор P_2, тобто в новому опорному плані слід передбачити випуск виробів В. При визначенні можливого числа виготовлених виробів В слід враховувати наявну кількість сировини кожного виду, а саме: можливий випуск виробів В визначається $\min(B'_i/A'_{i2})$ для A'_{i2}>0, тобто знаходимо

Q_0 = min(72/9; 24*2/1; 108*2/3)= 72/9 = 8.

Отже, виключенню з базису підлягає вектор P_4, іншими словами, випуск виробів В обмежений сировиною I виду, що є у розпорядженні підприємства. З урахуванням наявних об'ємів цієї сировини підприємству слід виготовити 8 виробів. Число 9 - розв'язувальний елемент, а стовпець вектора P_2 і 1-й рядок табл.2.5 є направляючим. Складаємо таблицю для III ітерації (табл.2.6).

Таблиця 2.6

і	Базис	Сб	P_0	9	10	16	0	0	0
				P_1	P_2	P_3	P_4	P_5	P_6
1	P_2	10	8	1	1	0	1	-1/6	0
2	P_3	16	20	1/4	0	1	-1/18	5/24	0
3	P_5	0	96	5/4	0	0	-1/6	-1/8	1
4			400	5	0	0	2/9	5/3	0

В табл.2.6 спочатку заповнюємо елементи 1-го рядка, який є рядком що знов вводиться в базис вектора P_2. Елементи цього рядка отримуємо з елементів 1-го рядка табл.2.5 діленням останніх на розв'язувальний елемент (тобто на 9). При цьому в стовпці C_6 даного рядка записуємо C_2 = 10.

Потім заповнюємо елементи стовпців векторів базису і за правилом трикутника обчислюємо елементи решти стовпців. В результаті в табл.2.6 отримуємо новий опорний план

X = (0; 8; 20;0; 0; 96) і коефіцієнти розкладання векторів P_j (j=1,6) через базисні вектори P_2, P_3, P_6 і відповідні значення Δj і F_o.

Перевіряємо, чи є даний опорний план оптимальним. Для цього розглянемо 4-й рядок табл.2.6. У цьому рядку серед чисел Δj немає від'ємних. Це означає, що знайдений опорний план є оптимальним і Fmax = 400.

Отже, план випуску продукції, що включає виготовлення 8 виробів В і 20 виробів С, є оптимальним. При даному плані випуску виробів повністю використовується сировина І і ІІ видів і залишається невикористаними 96 кг сировини ІІІ виду, а вартість продукції дорівнює 400 грн.

Оптимальним планом виробництва продукції не передбачається виготовлення виробів А. Додавання до плану випуску виробів виду А призвело б до зменшення вказаної загальної вартості. Це видно з 4-го рядка стовпця вектора P_1, де число 5 показує що при даному плані включення в його випуску одиниці виробу А приводить лише до зменшення загальної вартості на 5 грн.

Рішення даного прикладу симплексним методом можна було б проводити, використовуючи лише одну таблицю (табл.2.7). У цій

таблиці послідовно записано одна за одною всі три ітерації обчислювального процесу.

Таблиця 2.7

i	Базис	$C_б$	P_0	9	10	16	0	0	0
				P_1	P_2	P_3	P_4	P_5	P_6
1	P_4	0	360	18	15	12	1	0	0
2	P_5	0	192	6	4	8	0	1	0
3	P_6	0	180	5	3	3	0	0	1
4			0	-9	-10	-16	0	0	0
1	P_4	0	72	9	9	0	1	-3/2	0
2	P_5	16	24	3/4	½	1	0	1/8	0
3	P_6	0	108	11/4	3/2	0	0	-3/8	1
4			384	3	-2	0	0	2	0
1	P_2	10	8	1	1	0	1/9	-1/6	0
2	P_3	16	20	1/4	0	1	-1/8	5/24	0
3	P_6	0	96	5/4	0	0	-1/6	-1/8	1
4			400	5	0	0	2/9	5/3	0

2.2 Контрольні запитання

2.2.1 Яким чином за умовами задачі ЛП заповнюється результатна симплекс-таблиця?

2.2.2 Як можна отримати початковий опорний план?

2.2.3 Як перевірити опорний план на оптимальність? Назвіть умови отримання оптимального плану.

2.2.4 Як за допомогою симплексного методу визначити, що задача ЛП має необмежений оптимум (тобто цільова функція необмежена зверху на безлічі планів і рішення задачі відсутнє)?

2.2.5 Як вибрати направляючий стовпець і направляючий рядок?

2.2.6 Як визначити розв’язувальний елемент?

2.2.7 Як визначити елементи нової симплекс-таблиці?

2.2.8 Який фізичний зміст правила мінімального відношення, використовуваного для визначення вектора, що підлягає виключенню з базису?

2.2.9 Назвіть етапи рішення задачі ЛП симплексним методом.

3 ТРАНСПОРТНА ЗАДАЧА

3.1 Теоретичні відомості

Транспортна задача є типовою задачею лінійного програмування, отже, її розв'язок можна отримати звичайним симплексним методом. Однак, у деяких випадках застосування універсальних алгоритмів є нераціональним. Специфічна структура транспортної задачі дає змогу отримати альтернативний метод відшукання оптимального плану у вигляді простішої у порівнянні з симплексним методом обчислювальної процедури.

Класична транспортна задача лінійного програмування формулюється так: деякий однорідний продукт, що знаходиться у m постачальників A_i в обсягах $a_1, a_2, \dots, a_m$ одиниць відповідно необхідно перевезти n споживачам B_j в обсягах $b_1, b_2, \dots, b_n$ одиниць. При цьому виконується умова, що загальний наявний обсяг продукції у постачальників дорівнює загальному попиту всіх споживачів. Відомі вартості c_{ij} перевезень одиниці продукції від кожного A_i-го постачальника до кожного B_j-го споживача, що подані як елементи матриці виду:

$$c_{ij} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{pmatrix}$$

Необхідно визначити план перевезень, за якого вся продукція була б вивезена від постачальників, повністю задоволені потреби споживачів і загальна вартість всіх перевезень була б мінімальною.

У такій постановці задачі ефективність плану перевезень визначається його вартістю і така задача має назву ***транспортної задачі за критерієм вартості перевезень***.

Запишемо її математичну модель. Позначимо через X_{ij} обсяг продукції, що перевозиться від A_i постачальника до B_j споживача . (i = $\overline{1,m}$; j=$\overline{1,n}$)Тоді умови задачі зручно подати у вигляді такої таблиці:

Таблиця 3.1

Споживачі		B_1	B_2	...	$\boldsymbol{B_n}$
Постачальники		b_1	b_2	...	b_n
A_1	a_1	c_{11} x_{11}	c_{12} x_{12}	...	c_{1n} x_{1n}
A_2	a_2	c_{21} x_{21}	c_{22} x_{22}	…	c_{2n} x_{2n}
…	…	…	…	…	…
A_m	a_m	c_{m1} x_{m1}	c_{m2} x_{m2}	…	c_{mn} x_{mn}

Мають виконуватися такі умови:

- сумарний обсяг продукції, що вивозиться з кожного і-го пункту, має дорівнювати запасу продукції в даному пункті:

- сумарний обсяг продукції, що ввезений кожному j-му споживачеві, має дорівнювати його потребам:

- сумарна вартість всіх перевезень повинна бути мінімальною:

Очевидно, що $x_{ij} \geq 0$, i = $\overline{1,m}$; j=$\overline{1,n}$.

У скороченій формі запису математична модель транспортної задачі за критерієм вартості перевезень має такий вигляд:

$$\min F = \sum_{i=1}^{m}\sum_{j=1}^{m} C_{ij}X_{ij} \tag{3.1}$$

за обмежень:

$$\sum_{j=1}^{n} X_{ij} = a_i \quad (i = \overline{1,m}) \tag{3.2}$$

$$\sum_{i=1}^{m} x_{ij} = b_j \quad (j = \overline{1,n}) \tag{3.3}$$

$$x_{ij} \geq 0 \ (i = \overline{1,m};\ j = \overline{1,n}) \tag{3.4}$$

У розглянутій задачі має виконуватися умова:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \tag{3.5}$$

Транспортну задачу називають ***збалансованою***, або ***закритою,*** якщо виконується умова (3.5). Якщо ж така умова не виконується, то транспортну задачу називають ***незбалансованою***, або ***відкритою***.

Планом транспортної задачі називають будь-який невід'ємний розв'язок системи обмежень (3.2)—(3.4), який позначають матрицею $X = x_{ij}\ (i = \overline{1,m};\ \ j = \overline{1,n})$ Значення невідомих величин x_{ij}— обсяги продукції, що мають бути перевезені від i-х постачальників до j-х споживачів, називатимемо ***перевезеннями***.

Оптимальним планом транспортної задачі називають матрицю $X^* = x_{ij}^*\ (i = \overline{1,m};\ j = \overline{1,n})$ яка задовольняє умовам задачі, і для якої цільова функція (3.1) набирає найменшого значення.

Теорема *(умова існування розв'язку транспортної задачі):* необхідною і достатньою умовою існування розв'язку транспортної задачі є її збалансованість:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

Якщо при перевірці збалансованості виявилося, що транспортна задача є відкритою, то її необхідно звести до закритого типу. Це здійснюється введенням фіктивного (умовного) постачальника A_{m+1}у разі перевищення загального попиту над запасами

$$(\sum_{j=1}^{n} b_j > \sum_{i=1}^{m} a_i)$$

із ресурсом обсягом

$$a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$$

Якщо ж загальні запаси постачальників перевищують попит споживачів

$$\left(\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j \right),$$

то до закритого типу задача зводиться введенням фіктивного (умовного) споживача B_{n+1} з потребою

$$b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$$

Вартість перевезення одиниці продукції від фіктивного постачальника A_{m+1} (або фіктивного споживача B_{n+1}) до кожного зі споживачів (виробників) має дорівнювати нулю або бути набагато більшою за реальні витрати $c_{ij} (i = \overline{1,m};\ j = \overline{1,n})$ Як правило, у такому разі використовують нульові значення вартостей перевезень, що дає змогу спростити обчислення.

Назвемо ***опорним планом*** транспортної задачі такий допустимий її план, що містить не більш ніж $m + n - 1$ додатних компонент, а всі інші його компоненти дорівнюють нулю. Такий план є невиродженим. Якщо ж кількість базисних змінних менша ніж $m + n - 1$, то маємо вироджений опорний план.

Якщо **умови транспортної задачі** і її **опорний план** записані у вигляді табл. 1.1, то клітини, в яких $x_{ij} > 0$ (ненульові значення поставок), називаються заповненими, всі інші — пустими. Заповнені клітини відповідають базисним змінним і для невиродженого плану їх кількість дорівнює $m + n - 1$.

Назвемо ***циклом*** таку послідовність заповнених клітин таблиці 1.1, яка задовольняє умову, що лише дві сусідні клітини містяться або в одному рядку, або в одному стовпці таблиці, причому перша клітина

циклу є і його останньою клітиною. Якщо для певного набору заповнених клітин неможливо побудувати цикл, то така послідовність клітин є ациклічною.

Лема. Кількість клітин, які утворюють будь-який цикл транспортної задачі, завжди парна.

Теорема. Щоб деякий план транспортної задачі був опорним, необхідно і достатньо його ациклічності.

Отже, базисні клітини опорного плану завжди утворюють ациклічну послідовність клітин.

Наслідок теореми. Будь-яка сукупність з $m+n$ клітин матриці транспортної задачі утворює цикл.

Теорема. Якщо всі запаси a_i $(i=\overline{1,m})$і всі потреби $b_j, (j=\overline{1,n})$ є невід'ємними цілими числами, то будь-який опорний план складається із значень, що є цілими числами.

Методи побудови опорного плану транспортної задачі

Як і в звичайному симплексному методі, **розв'язування транспортної задачі** полягає в цілеспрямованому переборі та перевірці на оптимальність опорних планів. Початком такого ітераційного процесу є побудова першого опорного плану.

Існують кілька простих методів побудови опорного плану. Розглянемо методи північно-західного кута, мінімальної вартості, подвійної переваги та метод апроксимації Фогеля. Побудову опорного плану зручно подавати у вигляді таблиці, в якій постачальники продукції відповідають рядкам, а споживачі — стовпчикам.

Нехай умови конкретної транспортної задачі подані в табл. 3.2.

Ідея ***методу північно-західного кута*** полягає в тому, що заповнення таблиці починають, не враховуючи вартостей перевезень, з лівого верхнього (північно-західного) кута. У клітину записують менше з двох чисел a_1та b_1. Далі переходять до наступної клітини в цьому ж рядку або у стовпчику і заповнюють її, і т. д. Закінчують заповнення таблиці у правій нижній клітинці. У такий спосіб значення поставок будуть розташовані по діагоналі таблиці.

Розглянемо цей процес детальніше на прикладі.

Спочатку, не враховуючи вартості перевезень, завжди задовольняють потреби першого споживача B_1, використовуючи запаси першого постачальника A_1. У нашому прикладі (табл. 3.2) потреби споживача B_1 становлять b_1= 110, а запаси постачальника — a_1 = 150 одиниць (тобто із запасів першого постачальника можна повністю задовольнити потреби першого споживача), тому в клітинку A_1B_1 записуємо менше із значень a_1, b_1, тобто 110. Тепер потреби першого споживача повністю задоволені, і переходимо до задоволення потреб наступного (другого) споживача B_2. Обсяг його потреб b_2= 50. Після задоволення потреб першого споживача залишок запасів першого постачальника становить 150 – – 110 = 40. Отже, від першого виробника другому споживачеві можна перевезти лише 40 одиниць продукції, тому в клітину A_1B_2 записуємо число 40. Після цього, оскільки запаси першого постачальника повністю вичерпані, переходимо до використання запасів наступного постачальника A_2. Його запаси a_2 = 60, а незадоволені потреби другого споживача 50 – 40 = 10, тому в клітинку A_2B_2 записуємо число 10, і другий споживач у такий спосіб також повністю отримав необхідну кількість продукції. Переходимо до задоволення потреб наступного споживача B_3. У результаті часткового використання запасів другого постачальника його залишок продукції становить 60 – 10 = 50. Отже, від другого виробника до третього споживача можна перевезти 50 одиниць продукції. Клітинка A_2B_3 міститиме зазначене число 50, і цим запаси постачальника A_2 будуть повністю вичерпані. Переходимо до розподілу запасів останнього (третього) постачальника A_3. Залишились незадоволеними потреби третього споживача в обсязі 60 –50 = 10. Для їх задоволення скористаємося запасами постачальника A_3. У клітинку A_3B_3 записуємо число 10, і потреби споживача B_3 також повністю задоволені. Переходимо до останнього споживача B_4 з потребами b_4 = 80, які повністю задовольняються за рахунок залишку запасів третього постачальника: 90 – 10 = 80.

Отже, в таблиці 3.2 у заповнених клітинках знаходяться числа, що означають можливий план перевезень продукції. Сума чисел (перевезень) по рядках дорівнює обсягам запасів постачальників, а сума чисел по стовпцях — обсягам потреб відповідних споживачів.

Аналогічний результат можна отримати, якщо почати з правого нижнього кута таблиці, рухаючись до лівого верхнього. Процедуру

методу можна застосовувати також, починаючи розподіл поставок з лівого нижнього кута і рухаючись до правого верхнього по діагоналі. В такому разі спосіб розподілу перевезень можна було б назвати методом південно-західного кута, тому цей метод ще називають діагональним. Метод північно-західного кута є найпростішим, однак і найменш ефективним.

Таблиця 3.2

Постачальник	Запаси	Споживачі			
		B_1	B_2	B_3	B_4
		Потреби			
		b_1=110	b_2=50	b_3=60	b_4=80
A_1	a_1= 150	4 110	4 40	2	
A_2	a_2= 60	5	3 10	1 50	
A_3	a_3= 90	2	1	4 10	 80

Визначимо загальну вартість перевезень згідно з початковим опорним планом. Від першого постачальника до першого споживача необхідно перевезти 110 одиниць продукції за ціною 4 ум. од. (ціна записана в правому верхньому куті кожної клітини), отже, це коштуватиме 110 * 4 =440 ум. од. Крім того, необхідно перевезти від першого постачальника 40 одиниць продукції до другого споживача за ціною 4 ум. од. і т. д. У такий спосіб визначимо загальну вартість усіх перевезень:

*F= 110*4+40*4+10*3+50*1+10*4+80*2=880* (ум. од.).

Теорема. Опорний план транспортної задачі, знайдений методом північно-західного кута, завжди ациклічний.

Очевидно, якщо за побудови опорного плану враховувати вартості перевезень, то сумарна вартість всіх постачань може бути зменшена, і отриманий опорний план буде ближчим до оптимального.

Ідея ***методу мінімальної вартості*** полягає в тому, що на кожному кроці заповнюють клітинку таблиці, яка має найменшу вартість перевезення одиниці продукції. Такі дії повторюють доти, доки не буде розподілено всю продукцію між постачальниками та споживачами.

Складемо за допомогою цього методу план розглянутої задачі (табл. 3.3).

Найменшу вартість мають перевезення, які здійснюються від A_2 до B_3та від A_3 до B_2 (ціна перевезення одиниці продукції — 1 ум. од.). Заповнимо будь-яку з них, наприклад, A_2B_3. Оскільки постачальник має 60 одиниць продукції, а споживач потребує саме такої її кількості, то в клітину A_2B_3 ставимо значення 60. У такий спосіб запаси другого постачальника повністю вичерпані, а потреби третього споживача повністю задоволені. Також мінімальною є вартість перевезень від третього постачальника до другого споживача, тому заповнимо також клітину A_3B_2.

З клітинок таблиці, що залишились незаповненими, вибираємо наступне мінімальне значення вартості перевезень, яке дорівнює 2 ум. од. — для клітин A_1B_3, A_2B_4, A_3B_1 та A_3B_4. Заповнення клітин A_2B_4 та A_1B_3 неможливе, оскільки постачальник A_2 вже повністю вичерпав власний обсяг запасів, задовольняючи потреби споживача B_3, а споживач B_3 повністю задовольнив свої потреби. Отже, можна заповнити тільки клітину A_3B_1 чи A_3B_4. Заповнимо A_3B_1. Обсяг запасів a_3 = 90, причому 50 одиниць продукції вже надано другому споживачеві. Отже, маємо залишок 90 – 50 = 40, а потреби b_1 = 110, тому від третього постачальника до першого споживача плануємо перевезти 40 одиниць продукції. Тепер у клітину A_3B_4 не можна записати будь-який обсяг постачання, оскільки запаси третього постачальника вже повністю вичерпані.

Знову вибираємо найменшу вартість для клітин таблиці, що залишилися пустими, і продовжуємо процес доти, поки всі запаси не будуть розподілені, а потреби — задоволені.

Таблиця 3.3

b_j / a_i	b_1= 110	b_2= 50	b_3= 60	b_4= 80
a_1= 150	4 70	4	2	5 80
a_2= 60	5	3	1 60	2
a_3= 90	2 40	1 50	4	2

В результаті таких міркувань отримали початковий опорний план, загальна вартість перевезень для якого становить:

F= 70*4+80*5+60*1+50*1+40*2=870 (ум. од.).

Значення цільової функції менше за попередній варіант, значить цей план ближчий до оптимального.

Метод потенціалів розв'язування транспортної задачі

Для того, щоб плани відповідних спряжених задач були оптимальними, необхідно і достатньо, щоб виконувалися умови доповнюючої нежорсткості:

$$x_{ij}^{*}\left(u_i^{*}+v_j^{*}-c_{ij}\right)=0, i=\overline{1,m}; j=\overline{1,n}; \qquad (3.6)$$

$$\begin{cases} u_i^{*}\left(\sum_{j=1}^{n} x_{ij}^{*}-a_i\right)=0, i=\overline{1,m}; \\ v_j^{*}\left(\sum_{i=1}^{m} x_{ij}^{*}-b_j\right)=0, j=\overline{1,n}. \end{cases} \qquad (3.7)$$

Зауважимо, що друга група умов для транспортної задачі виконується автоматично, оскільки всі обмеження задачі є рівняннями.

Перша умова виконується у двох випадках:

а) якщо $x_{ij}^{*}=0$. Другий співмножник $(u_i^{*}+v_j^{*}-c_{ij})\le 0$, бо за умовою $u_i+v_j\le c_{ij}$ $(i=1,2,\dots,m; j=1,2,\dots,n)$;

б) якщо $x_{ij}^{*}\neq 0$, то за умовою транспортної задачі $x_{ij}^{*}>0$, тоді $(u_i^{*}+v_j^{*}-c_{ij})=0 \rightarrow u_i^{*}+v_j^{*}=c_{ij}(i=1,2,\dots,m; j=1,2,\dots,n)$.

Теорема (*умова оптимальності* опорного плану транспортної задачі). Якщо для деякого опорного плану *X** = (*xij**) існують числа *ui* та *vj*, для яких виконуються умови:

$u_i+v_j=c_{ij}, x_{ij}>0,$

$u_i+v_j\le c_{ij}, x_{ij}=0$

для всіх i=$\overline{1,m}$ та j=$\overline{1,n}$, то він є оптимальним планом транспортної задачі.

Використовуючи наведені умови існування розв'язку транспортної задачі, методи побудови опорних планів та умову оптимальності опорного плану транспортної задачі, сформулюємо алгоритм методу потенціалів, який по суті повторює кроки алгоритму симплексного методу.

Алгоритм методу потенціалів складається з таких етапів:

1. Визначення типу транспортної задачі (відкрита чи закрита). За необхідності слід звести задачу до закритого типу.

2. Побудова першого опорного плану транспортної задачі одним з відомих методів.

3. Перевірка опорного плану задачі на виродженість. За необхідності вводять нульові постачання.

4. Перевірка плану транспортної задачі на оптимальність.

Визначення потенціалів для кожного рядка і стовпчика таблиці транспортної задачі. Потенціали опорного плану визначають із системи рівнянь $u_i + v_j = c_{ij}$, які записують для всіх заповнених клітинок транспортної таблиці, кількість яких дорівнює m+n-1, а кількість невідомих — (m+n). Кількість рівнянь на одне менша, ніж невідомих, тому система є невизначеною, і одному з потенціалів надають нульове значення. Після цього всі інші потенціали розраховують однозначно.

Перевірка виконання умови оптимальності для пустих клітин. За допомогою розрахованих потенціалів перевіряють умову оптимальності $u_i + v_j \leq c_{ij}$ для незаповнених клітинок таблиці. Якщо хоча б для однієї клітини ця умова не виконується, тобто $u_i + v_j > c_{ij}$, то поточний план є неоптимальним, і від нього необхідно перейти до нового опорного плану.

Вибір змінної для введення в базис на наступному кроці. Загальне правило переходу від одного опорного плану до іншого полягає в тому, що з попереднього базису виводять певну змінну (вектор), а на її місце вводять іншу змінну (вектор), яка має покращити значення цільової функції. Аналогічна операція здійснюється і в алгоритмі методу потенціалів.

Перехід від одного опорного плану до іншого виконують заповненням клітинки, для якої порушено умову оптимальності. Якщо

таких клітинок кілька, то для заповнення вибирають таку, що має найбільше порушення, тобто max ($\Delta_{ij}= \left(u_i + v_j \right) - c_{ij}$)

Побудова циклу і перехід до наступного опорного плану. Вибрана порожня клітина разом з іншими заповненими становить m+n, отже, з цих клітин обов'язково утвориться цикл (теореми та наслідок § 5.2). У межах даного циклу здійснюють перерахування, які приводять до перерозподілу постачань продукції. Кожній вершині циклу приписують певний знак, причому вільній клітинці — знак «+», а всім іншим — за черговістю знаки «–» та «+». У клітинках зі знаком «–» вибирають значення q=minx_{ij} і переносять його у порожню клітинку. Одночасно це число додають до відповідних чисел, які містяться в клітинках зі знаком «+», та віднімають від чисел, що позначені знаком «–». Якщо значенню q відповідає кілька однакових перевезень, то при відніманні залишаємо у відповідних клітинках нульові величини перевезень у такій кількості, що дає змогу зберегти невиродженість опорного плану.

Внаслідок наведеного правила вибору q дістаємо новий опорний план, який не містить від'ємних перевезень і задовольняє умови транспортної задачі. Оскільки кількість всіх клітин таблиці, що входять у цикл, є парною і до половини з них те саме число q додається, а від половини віднімається, то загальна сума перевезень по всіх колонках і рядках залишається незмінною.

Клітинка, що була вільною, стає заповненою, а відповідна клітинка з мінімальною величиною x_{ij}вважається порожньою. У результаті такого перерозподілу перевезень продукції дістанемо новий опорний план транспортної задачі.

Перевірка умови оптимальності наступного опорного плану. Якщо умова оптимальності виконується — маємо оптимальний план транспортної задачі, інакше необхідно перейти до наступного опорного плану.

Зауважимо, що аналогічно з розв'язуванням загальної задачі лінійного програмування симплексним методом, якщо за перевірки оптимального плану транспортної задачі для деяких клітин виконується рівність $u_i + v_j = c_{ij}$, то це означає, що задача має альтернативні оптимальні плани. Отримати їх можна, якщо побудувати цикли перерозподілу обсягів перевезень для відповідних клітин.

Приклад розв'язування транспортної задачі методом потенціалів

Компанія контролює три фабрики A_1, A_2, A_3, здатні виготовляти відповідно 150, 60 та 80 тис. од. продукції щотижня. Вона уклала договір із чотирма замовниками B_1, B_2, B_3, B_4, яким потрібно щотижня доставляти відповідно 110, 40, 60 та 80 тис. од. продукції. Вартість транспортування 1000 од. продукції замовникам з кожної фабрики наведена в табл. 3.4.

Таблиця 3.4 - Вартість транспортування продукції

Фабрика	Вартість транспортування 1000 од. продукції замовнику			
	B_1	B_2	B_3	B_4
A_1	4	4	2	5
A_2	5	3	1	2
A_3	2	1	4	2

Визначити оптимальний план перевезень продукції від кожної фабрики до замовників, що мінімізує загальну вартість транспортних послуг.

Побудова математичної моделі. Нехай x_{ij}— кількість продукції, що перевозиться з *i*-ї фабрики до *j*-го замовника (i=$\overline{1,\ 3}$; j=$\overline{1,\ 4}$). Оскільки транспортна задача за умовою є збалансованою, закритою ($\sum_{i=1}^{3} a_i = \sum_{j=1}^{4} b_j = 290$), то математична модель задачі матиме вигляд:

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 150; \\ x_{21} + x_{22} + x_{23} + x_{24} = 60; \\ x_{31} + x_{32} + x_{33} + x_{34} = 80; \end{cases}$$

Економічний зміст записаних обмежень полягає в тому, що вся вироблена на фабриках продукція має вивозитися до замовників повністю.

Аналогічні обмеження можна записати відносно замовників: продукція, що може надходити до споживача від трьох фабрик, має повністю задовольняти його попит. Математично це записується так:

$$\begin{cases} x_{11} + x_{21} + x_{31} = 110; \\ x_{12} + x_{22} + x_{32} = 40 \\ x_{13} + x_{23} + x_{33} = 60 \\ x_{14} + x_{24} + x_{34} = 80 \end{cases}$$

Загальні витрати, пов'язані з транспортуванням продукції, визначаються як сума добутків обсягів перевезеної продукції на вартості транспортування 1000 од. продукції до відповідного замовника і за умовою задачі мають бути мінімальними. Тому формально це можна записати так:

$\min Z = 4x_{11} + xx_{12} + 2x_{13} + 5x_{14} + 5x_{21} + x_{22} + x_{23} + 2x_{24} + 2x_{31} + + x_{32} + 4x_{33} + 2x_{34}$.

Загалом математична модель сформульованої задачі має вигляд:

$\min Z = 4x_{11} + 4x_{12} + 2x_{13} + 5x_{14} + 5x_{21} + 3x_{22} + x_{23} + 2x_{24} + 2x_{31} + x_{32} + + 4x_{33} + 2x_{34}$

за умов:

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 150; \\ x_{21} + x_{22} + x_{23} + x_{24} = 60; \\ x_{31} + x_{32} + x_{33} + x_{34} = 80; \\ x_{12} + x_{22} + x_{32} = 40; \\ x_{13} + x_{23} + x_{33} = 60; \\ x_{14} + x_{24} + x_{34} = 80; \end{cases}$$

$$x_{ij} \geq 0, i = \overline{1,3};\ j = \overline{1,4}.$$

Розв'язання. Запишемо умови задачі у вигляді транспортної таблиці (табл. 3.5) та складемо її перший опорний план у цій таблиці методом мінімальної вартості.

Таблиця 3.5

A_i	B_j				u_i
	$b_1 = 110$	$b_2 = 40$	$b_3 = 60$	$b_4 = 80$	
$a_1 = 150$	110 4		42 + 2 -	5	$u_1 = 5$
$a_2 = 60$	5	3	-1 60	+ 2 0	$u_2 = 2$
$a_3 = 80$	2	1 40	4 40	2	$u_3 = 2$
v_j	$v_1 = -1$	$v_2 = -1$	$v_3 = -1$	$v_4 = 0$	

Загальна вартість перевезень продукції згідно з першим опорним планом визначається у такий спосіб:

Z_1= 4 * 110 + 5 * 40 + 1 * 60 + 1 * 40 + 2 * 40 = 820 (ум. од.).

Перший опорний план транспортної задачі вироджений, оскільки кількість заповнених клітинок у таблиці дорівнює п'яти, а $(m + n - 1) = 3 + 4 - 1 = 6$.

Для дальшого розв'язування задачі необхідно в одну з порожніх клітинок записати «нульове перевезення» так, щоб не порушити опорності плану, тобто можна зайняти будь-яку пусту клітинку, яка не утворює замкненого циклу із заповненими клітинами. Наприклад, заповнимо нулем клітинку A_2B_4 . Тепер перший план транспортної задачі є невиродженим, і його можна перевірити на оптимальність методом потенціалів.

На основі першої умови оптимальності $u_i + v_j = c_{ij}$ складемо систему рівнянь (для заповнених клітин таблиці) для визначення потенціалів першого опорного плану:

$$\begin{cases} u_1 + v_1 = 4; \\ u_1 + v_4 = 5; \\ u_2 + v_3 = 1; \\ u_2 + v_4 = 2; \\ u_3 + v_2 = 1; \\ u_3 + v_4 = 2; \end{cases}$$

Записана система рівнянь є невизначеною, і один з її розв'язків дістанемо, узявши, наприклад, v_4= 0. Тоді всі інші потенціали однозначно визначаються з цієї системи рівнянь:u_1 = 5, u_2 = 2, u_3 = 2, v_1 = – 1, v_2 = – 1, v_3 = – 1. Ці значення потенціалів першого опорного плану записуємо у транспортну таблицю.

Потім згідно з алгоритмом методу потенціалів перевіряємо виконання другої умови оптимальності $u_i + v_j \leq c_{ij}$ (для порожніх клітинок таблиці):

A_1B_2: $u_1 + v_2$ = 5 + (–1) = 4 = 4;
A_1B_3 : $u_1 + v_3$ = 5 + (–1) = 4>2;
A_2B_1 : $u_2 + v_1$ = 2 + (–1) = 1<5;
A_2B_2 : $u_2 + v_2$ = 2 + (–1) = 1<3;

$$A_3B_1 : u_3 + v_2 = 2 + (-1) = 1<2;$$
$$A_3B_3 : u_3 + v_3 = 2 + (-1) = 1<4.$$

Умова оптимальності не виконується для клітинки A_1B_3. Порушення $D_{13} = (u_1 + v_3) - c_{13} = 4 - 2 = 2$ записуємо в лівому нижньому кутку відповідної клітинки.

Отже, перший опорний план транспортної задачі неоптимальний. Тому від нього необхідно перейти до другого плану, змінивши співвідношення заповнених і порожніх клітинок таблиці.

Потрібно заповнити клітинку A_1B_3, в якій є єдине порушення умови оптимальності. Ставимо в ній знак «+». Для визначення клітинки, що звільняється, будуємо цикл, починаючи з клітинки A_1B_3, та позначаємо вершини циклу почергово знаками «–» і «+». Тепер необхідно перемістити продукцію в межах побудованого циклу. Для цього у порожню клітинку A_1B_3 переносимо менше з чисел x_{ij}, які розміщені в клітинках зі знаком «–». Одночасно це саме число x_{ij}додаємо до відповідних чисел, що розміщені в клітинках зі знаком «+», та віднімаємо від чисел, що розміщені в клітинках, позначених знаком «–».

У даному разі min (60; 40) = 40, тобто $\min x_{ij} = 40$. Виконавши перерозподіл перевезень продукції згідно із записаними правилами, дістанемо такі нові значення: для клітинки A_1B_3 — 40 од. продукції, а для A_2B_3 – (60 – 40) = 20 од., а для A_2B_4 – (0 + 40) = 40 од. Клітинка A_1B_4 звільняється і в новій таблиці буде порожньою. Усі інші заповнені клітинки першої таблиці, які не входили до циклу, переписуємо у другу таблицю без змін. Кількість заповнених клітинок у новій таблиці також має відповідати умові невиродженості плану, тобто дорівнювати ($n + m - 1$).

Отже, другий опорний план транспортної задачі матиме такий вигляд (табл. 3.6):

Таблиця 3.6

A_i	B_j				u_i
	$b_1 = 110$	$b_2 = 40$	$b_3 = 60$	$b_4 = 80$	
$a_1 = 150$	4 110 -	4	2 + 40	5	$u_1 = 0$
$a_2 = 60$	5	3	1 - 20	2 + 40	$u_2 = -1$
$a_3 = 80$	+ 2 1	1 40	4	2 - 40	$u_3 = -1$
v_j	$v_1 = 4$	$v_2 = -2$	$v_3 = 2$	$v_4 = 3$	

Розрахуємо значення цільової функції відповідно до другого опорного плану задачі:

Z_2= 4 * 110 + 2 * 40 + 1 * 20 + 2 * 40 + 1 * 40 + 2 * 40 = 740 (ум. од.).

Новий план знову перевіряємо на оптимальність, тобто повторюємо описані раніше дії. Другий опорний план транспортної задачі також неоптимальний (має місце порушення для клітинкиA_3B_1). За допомогою побудованого циклу, виконавши перехід до третього опорного плану транспортної задачі, отримуємо табл. 3.7:

Таблиця 3.7

Ai	*Bj*				u_i
	*b*1 = 110	*b*2 = 40	*b*3 = 60	*b*4 = 80	
*a*1 = 150	4 90	4	2 60	5	$u_1 = 2$
*a*2 = 60	5	3	1	2 60	$u_2 = 0$
*a*3 = 80	2 20	1 40	4	2 20	$u_3 = 0$
v_j	$v_1 = 2$	$v_2 = 1$	$v_3 = 0$	$v_4 = 2$	

Визначимо загальну вартість витрат на транспортування продукції згідно з третім опорним планом:

Z_3 = 4 * 90 + 2 * 60 + 2 * 60 + 2 * 20 + 1 * 40 + 2 * 20 = 720 (ум. од.).

Перевірка останнього плану на оптимальність за допомогою методу потенціалів показує, що він оптимальний. Тому:

$$X^* = \begin{pmatrix} 90 & 0 & 60 & 0 \\ 0 & 0 & 0 & 60 \\ 20 & 40 & 0 & 20 \end{pmatrix}$$

За оптимальним планом перевезень перший замовник отримує 90 тис. од. продукції з першої фабрики та 20 тис. од. — з третьої. Другий споживач задовольняє свій попит за рахунок виробництва та перевезення 40 тис. од. продукції з третьої фабрики і т. д. При цьому загальна вартість перевезень всієї продукції є найменшою і становить 720 ум. од.

3.2 Контрольні запитання

3.2.1 Наведіть математичну модель транспортної задачі.

3.2.2 Яка транспортна задача є відкритою, яка є закритою?

3.2.3 Як перетворити відкриту транспортну задачу до закритої?

3.2.4 Яка умова є необхідною і достатньою умовою існування розв’язку транспортної задачі?

3.2.5 Перелічіть методи побудови початкового опорного плану задачі. Які з них є більш ефективними?

3.2.6 Наведіть алгоритм методу потенціалів.

3.2.7 Як здійснюється перехід до наступного опорного плану?

3.2.8 Наведіть умову оптимальності опорного плану транспортної задачі.

4 ОДНОВИМІРНИЙ ПОШУК ОПТИМУМУ. МЕТОДИ ОПТИМІЗАЦІЇ З ВИКЛЮЧЕННЯМ ІНТЕРВАЛІВ

4.1 Теоретичні відомості

Аналіз завдань одновимірного пошуку оптимуму займає центральне місце в оптимізаційних дослідженнях. Це пов'язано не тільки з тим, що саме такі завдання зазвичай вирішуються на практиці, але й з тим, що одновимірні методи оптимізації часто використовуються для аналізу завдань, які виникають при реалізації процедур, що орієнтовані на рішення багатомірних завдань оптимізації.

Важливість оптимізаційних завдань із однією керованою змінною обумовила розробку великої кількості алгоритмів їх вирішення. У даному розділі досліджуються одновимірні методи пошуку, орієнтовані на знаходження точки оптимуму всередині заданого інтервалу. Методи пошуку, що дозволяють визначити оптимум функції однієї змінної шляхом послідовного виключення підінтервалів і, отже, шляхом зменшення інтервалу пошуку, називаються методами виключення інтервалів.

Методи пошуку, що розглядаються, засновані на припущенні, що досліджувана функція в припустимій області має властивість унімодальності. Користь цієї властивості визначається тим фактором, що для унімодальної функції порівняння значень функції у двох різних точках інтервалу пошуку дозволяє визначити, у якому із заданих двома зазначеними точками підінтервалів точка оптимуму відсутня.

Визначення границь інтервалу

На цьому етапі спочатку вибирається вихідна точка, а потім на основі правила виключення будується відносно широкий інтервал, що містить точку екстремуму. Звичайно пошук граничних точок такого інтервалу проводиться за допомогою евристичних методів пошуку, хоча в ряді випадків можна також використовувати методи екстраполяції. Відповідно до одного з евристичних методів, що був запропонований Свенном, $(k+1)-a$ пробна точка визначається за

рекурентною формулою

$$x_{k+1} = x_k + 2^k \Delta,\ k = 0, 1, 2, \ldots ,$$

де x_k – довільно вибрана початкова точка, Δ – величина кроку, що підбирається деяким способом. Знак Δ визначається шляхом порівняння значень

$$f(x_0), f(x_0) + |\Delta| \ i \ f(x_0 - |\Delta|).$$

Якщо

$$f(x_0 - |\Delta|) \geq f(x_0) \geq f(x_0 + |\Delta|),$$

то, відповідно до припущення про унімодальність, точка мінімуму повинна розміщуватися правіше точки x_0 і величина Δ вибирається додатною (рис.1). Якщо змінити знаки нерівностей на протилежні, то варто вибирати від'ємною. Якщо виконується

$$f(x_0 - |\Delta|) \geq f(x_0) \leq f(x_0 + |\Delta|),$$

то точка мінімуму лежить між $(x_0 - |\Delta|)$ та $(x_0 + |\Delta|)$ і пошук граничних точок завершений.

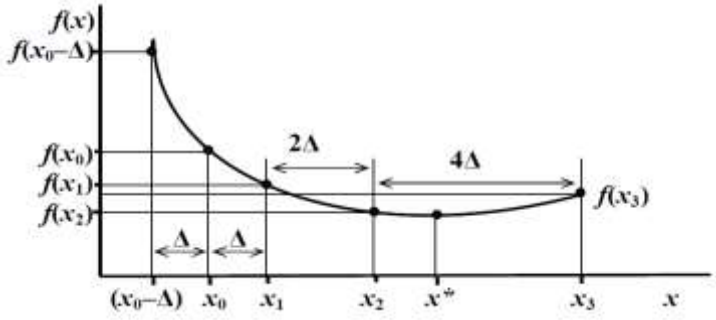


Рисунок 4.1 – Графічні ілюстрації до процедури пошуку граничних точок

Випадок, коли

$$f(x_0 - |\Delta|) \leq f(x_0) \geq f(x_0 + |\Delta|),$$

суперечить припущенню про унімодальність при пошуку точки мінімуму. Виконання цієї умови свідчить про те, що функція не є унімодальною.

Відзначимо, що ефективність пошуку граничних точок безпосередньо залежить від величини кроку Δ. Якщо Δ велике, то отримуємо грубі оцінки координат граничних точок, і побудований інтервал виявляється досить широким. З іншого боку, якщо Δ мале, для визначення граничних точок може знадобитися досить великий обсяг обчислень.

Етап зменшення інтервалу

Після того як установлені границі інтервалу, що містить точку екстремуму, можна застосувати більш складну процедуру зменшення інтервалу пошуку з метою отримання уточнених оцінок координат екстремуму. Величина підінтервалу, що виключається на кожному кроці, залежить від розміщення пробних точок x_1 і x_2 усередині інтервалу пошуку. Оскільки місцезнаходження точки екстремуму апріорі невідомо, доцільно припустити, що розміщення пробних точок повинне забезпечувати зменшення інтервалу в тому самому відношенні. Крім того, з метою підвищення ефективності алгоритму бажано, щоб зазначене відношення було максимальним. Подібну стратегію називають мінімаксною стратегією пошуку.

Метод виключення інтервалів

Теорема

Нехай функція $f(x)$ унімодальна на замкненому інтервалі $a \leq x \leq b$, а її мінімум досягається в точці x^*. Розглянемо точки x_1 і x_2, які розміщені в інтервалі в такий спосіб, що $a < x_1 < x_2 < b$. Порівнюючи значення функції в точках x_1 і x_2 , можна зробити наступні висновки.

1. Якщо $f(x_1) > f(x_2)$, то точка мінімуму $f(x)$ не лежить в інтервалі (a, x_1), тобто $x^* \in (x_1, b)$ (рис. 2, а).

2. Якщо $f(x_1) > f(x_2)$, то точка мінімуму $f(x)$ не лежить в інтервалі (x_2, b), тобто $x^* \in (a, x_2)$ (рис. 2, б).

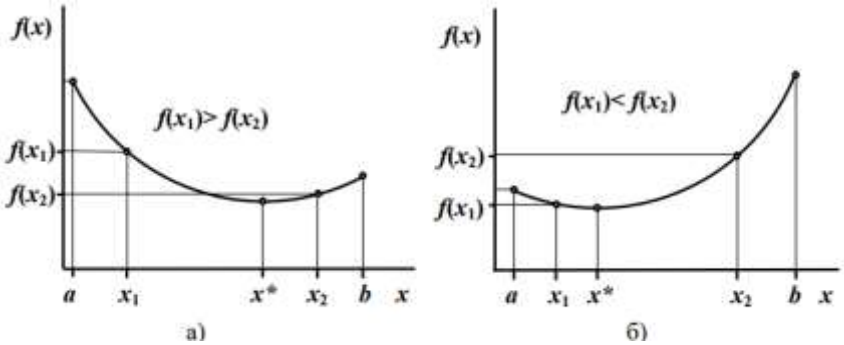


Рисунок 4.2 – Графічні ілюстрації до теореми

Примітка. Якщо $f(x_1) = f(x_2)$, то можна виключити обидва крайні інтервали (a, x_1) і (x_2, b), при цьому точка мінімуму повинна розміщуватися в інтервалі (x_1, x_2).

Метод розподілу інтервалу навпіл

Розглянутий метод дозволяє виключати точно половину інтервалу на кожній ітерації. Іноді цей метод називають триточковим пошуком на рівних інтервалах, оскільки його реалізація заснована на виборі трьох пробних точок, які рівномірно розподілені в інтервалі пошуку. Нижче приводиться опис основних кроків пошукової процедури, орієнтованих на знаходження точки мінімуму функції $f(x)$ в інтервалі (a, b).

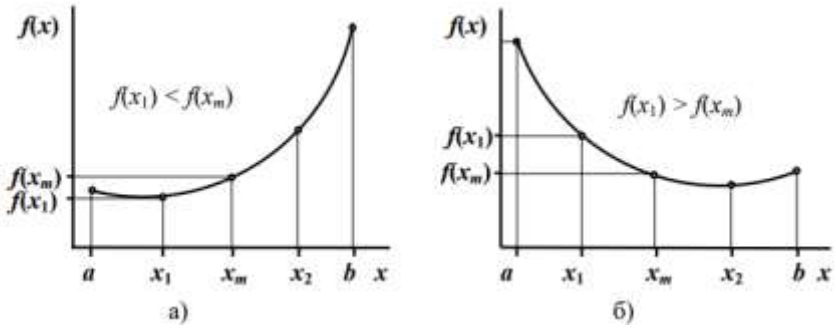


Рисунок 4.3 – Графічні ілюстрації до методу розподілу інтервалу навпіл

Крок 1. Покласти $x_m = (a + b)/\,2\; i\; L = -a$. Обчислити значення $f(x_m)$.

Крок 2. Покласти $x_1 = a + L/4\; i\; x_2 = b - L/4$. Помітимо, що точки $x_1, x_m,$ і x_2 поділяють інтервал (a, b) на чотири рівні частини. Обчислити значення $f(x_1)\; i\; f(x_2)$.

К р о к 3. Порівняти $f(x_1)\; i\; f(x_m)$.

(1) Якщо $f(x_1) < f(x_m)$ (рис. 3, а), виключити інтервал (x_m, b), поклавши $b = x_m$. Середньою точкою нового інтервалу пошуку стає точка x_1. Отже, необхідно покласти $x_m = x_1$. Перейти до кроку 5.

(2) Якщо $f(x_1) \geq f(x_m)$ (рис. 3, б), перейти до кроку 4.

К р о к 4. Порівняти $f(x_2)$ *і* $f(x_m)$.

(1) Якщо $f(x_2) < f(x_m)$, виключити інтервал (a, x_m), поклавши $a = x_m$. Тому що середньою точкою нового інтервалу стає точка x_2 покласти $x_m = x_2$. Перейти до кроку 5.

(2) Якщо $f(x_2) \geq f(x_m)$, виключити інтервали (a, x_1) і (x_2, b). Покласти $a = x_1$ і $b = x_2$. Відзначимо, що x_m продовжує залишатися середньою точкою нового інтервалу. Перейти до кроку 5.

К р о к 5. Обчислити $L = b - a$. Якщо величина $|L|$ мала, закінчити пошук. У противному разі повернутися до кроку 2.

Зауваження

1. На кожній ітерації алгоритму виключається точно половина інтервалу пошуку.

2. Середня точка послідовно отриманих інтервалів завжди збігається з однією з пробних точок x_1, x_2 або x_m, знайдених на попередній ітерації. Отже, на кожній ітерації потрібно не більш двох обчислень значення функції;

3. Якщо проведено n обчислень значення функції, то довжина отриманого інтервалу складає $(1/2)^{n/2}$ величини вихідного інтервалу.

4. У роботі [3] показано, що з усіх методів пошуку на рівних інтервалах (двоточковий, триточковий, чотириточковий і т.д.) триточковий пошук, чи метод розподілу інтервалу навпіл, відрізняється найбільшою ефективністю.

Пошук за допомогою методу золотого перетину

З проведеного вище розгляду методів виключення інтервалів і мінімаксних стратегій пошуку можна зробити наступні висновки.

1. Якщо кількість пробних точок приймається рівною двом, то їх варто розміщати на однакових відстанях від середини інтервалу.

2. Відповідно до загальної мінімаксної стратегії пробні точки повинні розміщатися в інтервалі за симетричною схемою таким чином, щоб відношення довжини підінтервалу, що вилучається, до величини інтервалу пошуку залишається постійним.

3. На кожній ітерації процедури пошуку повинне обчислюватися тільки одне значення функції в отримуваній точці.

Керуючись цими висновками, розглянемо симетричне

розміщення двох пробних точок на початковому інтервалі одиничної довжини, яке показане на рис. 4. (Вибір одиничного інтервалу зумовлений розуміннями зручності.) Пробні точки віддалені від граничних точок інтервалу на відстані τ. При такому симетричному розміщенні точок довжина інтервалу, що залишається після ви ключання, завжди дорівнює незалежно від того, яке зі значень функції в пробних точках виявляється меншим.

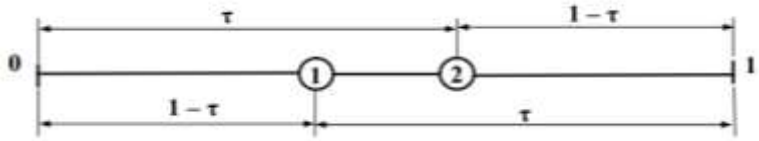


Рисунок 4.4 – Пошук за допомогою методу золотого перетину

Припустимо, що вилучається правий підінтервал. На рис. 5 показано, що підінтервал довжини τ, що залишився, містить одну пробну точку, яка розміщена на відстані $(1-\tau)$ від лівої граничної точки.

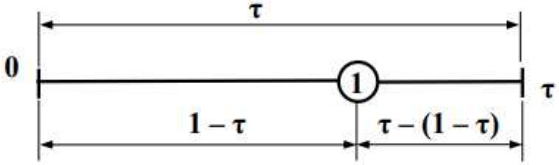


Рисунок 4.5 – Інтервали, отримані методом золотого перетину

Для того щоб симетрія пошукового зразка зберігалася, відстань $(1-\tau)$ повинна складати $\tau - y$ частину довжини інтервалу (яка дорівнює τ). При такому виборі τ наступна пробна точка розміщується на відстані, рівній τ-й частині довжини інтервалу, від правої граничної точки інтервалу (рис. 6).

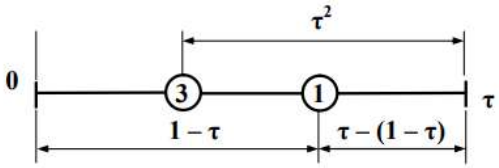


Рисунок 4.6 – Симетрія золотого перетину інтервалу

Отже, на першому кроці пошуку (рис.4) для точки 1 виконується співвідношення $\tau/1 = (1-\tau)/\tau$. Звідси випливає: $1-\tau = \tau^2$.

Розв'язуючи це квадратне рівняння, отримуємо

$$\tau = (-1 \pm \sqrt{5})/2,$$

отже, додатне рішення $\tau = 0{,}61803\ldots$.

Перевіримо співвідношення отриманих відрізків (рис. 4) на наступному кроці пошуку, коли довжина інтервалу $(1, 2)$ дорівнює $\tau - (1-\tau)$:

$$\frac{\tau-(1-\tau)}{1-\tau} = \frac{1-\tau}{\tau},$$
$$\tau^2 - \tau(1-\tau) = (1-\tau)^2.$$

Поділимо обидві частин рівняння на $(1-\tau)$.

Після перетворень остаточно отримуємо раніше записану умову симетрії пошукового зразка $\tau^2 = 1-\tau$.

Таким чином, при виборі відповідно до умови $1-\tau = \tau^2$ симетрія пошукового зразка, що показаний на рис. 4, зберігається при переході до зменшеного інтервалу, зображеного на рис. 6.

Схема пошуку, за якою пробні точки ділять інтервал у цьому відношенні, відома за назвою пошуку за допомогою методу золотого перетину. Зауважимо, що після перших двох обчислень значень функції кожне наступне обчислення дозволяє вилучити підінтервал, величина якого складає $(1-\tau) - y$ частку від до вжини інтервалу пошуку.

Отже, якщо початковий інтервал має одиничну довжину, то величина інтервалу, який отриманий в результаті n обчислень значень функції, дорівнює τ^{n-1}. Можна показати, що пошук за допомогою методу золотого перетину є аси мптотично найбільш ефективним способом реалізації мінімаксної стратегії пошуку.

У загальному випадку якщо праві і ліва граничні точки інтервалу невизначеності (позначимо їх через XR і XL) відомі, то координати всіх наступних пробних точок, отримуваних згідно з методом золотого перетину, можна розрахувати за формулою:

$$w = XR - \tau^n \text{ або } w = XL - \tau^n,$$

у залежності від того, який підінтервал був вилучений на попередній ітерації – лівий чи правий. У наведених вище формулах через τ^n позначена n-на ступінь τ, де n – кількість обчислень значень функції.

Пошук за допомогою методу золотого перетину може бути

закінчений виходячи з заданої кількості обчислень значень функції (і, отже, величини інтервалу невизначеності), або по досягненні відносної точності шуканого значення функції. Найкращим є використання обох критеріїв одночасно.

Порівняння методів виключення інтервалів

Нижче проводиться порівняння відносних ефективностей розглянутих методів виключення інтервалів. Позначимо довжину вихідного інтервалу невизначеності через L, а довжину інтервалу, отримуваного в результаті n обчислень значень функції, – через L_n. Як показник ефективності того чи іншого методу виключення інтервалів уведемо в розгляд характеристику відносного зменшення початкового інтервалу $K_L(n) = L_n/L$.

Нагадаємо, що при використанні методу розподілу інтервалу навпіл і методу золотого перетину довжина отримуваного інтервалу складає $L(0{,}5)^{n/2}$ і$L(0{,}618)^{n-1}$, відповідно. Отже, відносне зменшення інтервалу після n обчислень значень функції дорівнює

$K_L(n) = 0{,}5^{n/2}$– для методу розподілу інтервалу навпіл;

$K_L(n) = 0{,}618^{n-1}$ – для методу золотого перетину.

Для порівняння розглянемо також метод рівномірного пошуку (рис. 7), відповідно до якого оцінювання функції проводиться в n рівновіддалених одна від одної точках, при цьому інтервал L ділиться на $(n+1)$ рівних інтервалів довжини $L/(n+1)$. Нехай x^* – точка, у якій спостерігається мінімум функції $f(x)$. Тоді точка істинного мінімуму $f(x)$, виявляється, міститься в інтервалі

$$\{[x^* - L/(n+1)], \qquad [x^* + L/(n+1)]\},$$

отже, $L_n = 2L/(n+1)$.

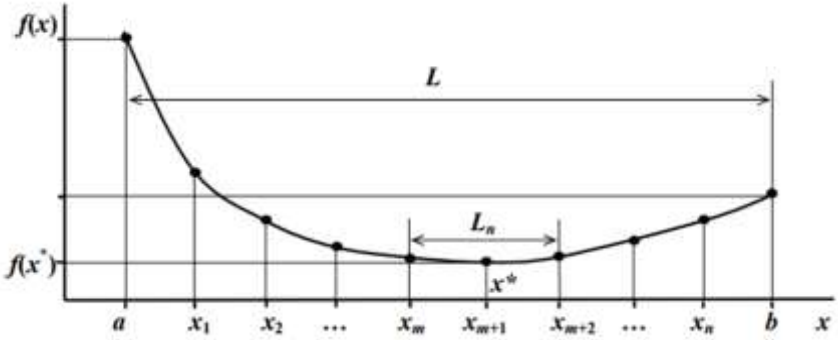


Рисунок 4.7 – Графічні ілюстрації до методу рівномірного пошуку

Відносне зменшення інтервалу для методу рівномірного пошуку

$$K_L(n) = 2/(n+1)$$

У табл. 4.1 подані значення K_L (n), що відповідають обраним n, для трьох методів пошуку. З таблиці випливає, що пошук за допомогою методу золотого перетину забезпечує найбільше відносне зменшення вихідного інтервалу при однаковій кількості обчислень значень функції.

Таблиця 4.1 – Величини відносного зменшення інтервалу

Метод пошуку	Кількість обчислень значень функцій				
	n=2	n=5	n=10	n=15	n=20
Метод розподілу інтервалу навпіл	0,5	0,177	0,031	0,006	0,0009
Метод золотого перетину	0,618	0,146	0,013	0,001	0,0001
Метод рівномірного пошуку	0,667	0,333	0,182	0,125	0,095

З іншого боку, можна також порівняти кількості обчислень значення функції, що необхідні для досягнення заданої величини

відносного зменшення інтервалу або заданого ступеня точності. Якщо величина $K_L(n) = E$ задана, то значення n обчислюється за наступними формулами:

– для методу розподілу інтервалу навпіл

$$n = 2\,ln(E)/\,ln(0{,}5);$$

– для методу золотого перетину

$$n = l + [ln(E)/\,ln(0{,}618)];$$

– для методу рівномірного пошуку

$$n = (2/E) - 1.$$

У табл. 4.2 подано дані про кількість обчислень значень функції, яка необхідна для визначення координати точки мінімуму з заданою точністю.

Таблиця 4.2 – Необхідна кількість обчислень значень функції

Метод пошуку	Задана точність			
	E=0,1	*E*=0,05	*E*=0,01	*E*=0,001
Метод розподілу інтервалу навпіл	7	9	14	20
Метод золотого перетину	5	8	11	16
Метод рівномірного пошуку	19	36	199	1999

Слід зазначити, що метод золотого перетину виявляється більш ефективним порівняно з іншими двома методами, оскільки він вимагає найменшого числа оцінювань значення функції для досягнення однієї і тієї самої заданої точності.

4.2 Приклади розв'язання задач

Приклад 1. (Визначення границь інтервалу)

Розглянемо задачу мінімізації функції $f(x) = (100 - x)^2$ при заданій початковій точці $x_0 = 30$ і величині кроку $|\Delta| = 5$. Знак Δ визначається на основі порівняння значень

$f(x_0) = f(30) = 4900,$

$f(x_0 + |\Delta|) = f(35) = 4225,$

$f(x_0 - |\Delta|) = f(25) = 5625.$

Тому що

$f(x_0 - |\Delta|) \geq f(x_0) \geq f(x_0 + |\Delta|),$

то величина повинна бути додатною, а координата точки мінімуму x^* повинна бути більше 30.

Маємо $x_1 = x_0 + \Delta = 35$. Далі $x_2 = x_1 + \Delta = 45,$

$f(45) = 3025 < f(x_1),$

отже $x^* > 35.$

$x_3 = x_2 + 2^2\Delta = 65, f(65) = 1225 < f(x_2),$

отже, $x^* > 45.$

$x_4 = x_3 + 2^3\Delta = 105, f(105) = 25 < f(x_3),$

отже, $x^* > 65.$

$x_5 = x_4 + 2^4\Delta = 185, f(185) = 7225 < f(x_4),$

остаточно , $x^* < 185$. Таким чином, шість кроків обчислень x^* дозволили виявити інтервал $65 \leq x^* \leq 185$, у якому розміщено точку x^*.

Приклад 2. (Метод розподілу інтервалу навпіл)

Мінімізувати $f(x) = (100 - x)^2$ в інтервалі $60 \leq x \leq 150$. Тут $a = 60, b = 150$ *і* $L = 150 - 60 = 90.$

$x^m = (60 + 150)/2 = 105.$

Ітерація 1.

$x_1 = a + (L/4) = 60 + (90/4) = 82{,}5,$

$x_2 = b - (L/4) = 150 - (90/4) = 127{,}5,$

$f(82{,}5) = 306{,}25 > f(105) = 25,$

$f(127{,}5) = 756{,}25 > f(105).$

Таким чином, вилучаються інтервали (60, 82,5) і (127,5, 150). Довжина інтервалу пошуку зменшується з 90 до 45.

Ітерація 2.

$a = 82{,}5, b = 127{,}5, x_m = 105,$

$L = 127{,}5 - 82{,}5 = 45,$

$x_1 = 82{,}5 + (45/4) = 93{,}75,$

$x_2 = 127{,}5 - (45/4) = 116{,}25,$

$f(93{,}75) = 39{,}06 > f(105) = 25,$

$f(116{,}25) = 264{,}06 > f(105).$

Таким чином, інтервал невизначеності дорівнює (93,75, 116,25).

Ітерація 3.

$A = 93{,}75,\ b = 116{,}25,\ x_m = 105,$

$L = 116{,}25 - 93{,}75 = 22{,}5,$

$x_1 = 99{,}375,\ x_2 = 110{,}625,$

$f(x_1) = 0{,}39 < f(105) = 25.$

Таким чином, вилучається інтервал (105, 116,25). Новий інтервал невизначеності дорівнює (93,75, 105), його середня точка є 99,375 (точка x_1 на ітерації 3). Відзначимо, що за три ітерації (шість обчислень значення функції) вихідний інтервал пошуку довжини 90 зменшився до величини $90 \times (1/2)^3 = 11{,}25$.

Приклад 3. (Метод золотого перетину)

Розглянемо задачу за прикладом 1, у якому потрібно мінімізувати $f(x) = (100 - x^2)$ в інтервалі $60 < x < 150$.

Для того щоб перейти до інтервалу одиничної довжини, проведемо заміну перемінної, поклавши $w = (x - 60)/90$. Таким чином, задача набуває такого виду:

мінімізувати $f(w) = (40 - 90w)^2$

при обмеженні $0 \le w \le 1$.

Ітерація 1. $I_1 = (0{,}1);\ L_1 = 1$. Проведемо два перших обчислення значень функції:

$w_1 = \tau = 0{,}618, f(w_1) = 244{,}0,$

$w_2 = l - \tau = \tau^2 = 0{,}382, f(w_2) = 31{,}6.$

Тому що $f(w_2) < f(w_1)\ i\ w_2 < w_1$, інтервал $w \ge w_1$ вилучається.

Ітерація 2. $I_2 = (0{,}618);\ L_2 = 0{,}618 = \tau$. Наступне обчислення значення функції проводиться у точці

$w_3 = \tau - \tau^2 = \tau(1 - \tau) = \tau^3 = 0{,}236, f(w_3) = 352.$

Тому що $f(w_3) < f(w_2)\ i\ w_3 < w_2$ інтервал $w \ge w_3$ вилучається.

Ітерація 3 . $I_3 = (0{,}236, 0{,}618);\ L_3 = 0{,}382 = \tau^2$. Наступне обчислення значення функції проводиться у точці, розміщеній на відстані $\tau \times$ (довжина отриманого інтервалу) від лівої граничної точки інтервалу, або на відстані $(1 - \tau) \times$ (довжина інтервалу) від правої граничної точки. Таким чином,

$$w_4 = 0{,}618 - (1-\tau)L_3 = 0{,}618 - \tau^2 L_3 = 0{,}618 - \tau^2(\tau^2) = 0{,}618 - \tau^4 = 0{,}472,$$

$f(w_4) = 6{,}15$

Тому що, $f(w_4) > f(w_2)$ *і* $w_4 < w_2$ інтервал $w \le w_2$ вилучається. У результаті отримано наступний інтервал невизначеності:

$0{,}382 \le w \le 0{,}618$ для перемінної *w*, або $94{,}4 \le w \le 115{,}6$ для перемінної *x.* Якщо в процесі пошуку проведено шість обчислень значень функції, то довжина результуючого інтервалу для перемінної *w* дорівнює

$$\tau^{N-1} = \tau^5 = 0{,}09,$$

що відповідає інтервалу довжини 8,1 для перемінної *x.* Для порівняння нагадаємо, що в аналогічній ситуації метод розподілу інтервалу навпіл призвів до отримання інтервалу довжини 11,25.

4.3 Контрольні запитання

4.3.1 У чому полягають питання аналізу "у статиці" і "в динаміці", що виникають при аналізі оптимізаційних завдань?

4.3.2 У чому полягають необхідні умови того, що дана точка є точкою локального мінімуму (максимуму)?

4.3.3 Сформулюйте достатні умови оптимальності.

4.3.4 Що таке стаціонарна точка?

4.3.5 Що таке точка перегину (сідлова точка) і як її ідентифікувати?

4.3.6 Як перевірити, чи є функція випуклою або ввігнутою?

4.3.7 У чому полягає властивість унімодальності функцій і в чому полягає важливе значення цієї властивості при рішенні завдань оптимізації з однією змінною?

4.3.8 Нехай дана точка задовольняє достатнім умовам існування локального мінімуму. Як встановити, чи є цей мінімум глобальним?

4.3.9 Як вирішується завдання керування запасами?

4.3.10 Сформулюйте правило виключення інтервалів.

4.3.11 Яким чином визначаються пробні точки при встановленні границь інтервалу, що містить точку оптимуму?

4.3.12 Яким чином визначається знак кроку при становленні границь інтервалу і як його величина впливає на ефективність пошуку граничних точок?

4.3.13 У чому полягає мінімаксна стратегія пошуку?

4.3.14 Опишіть пошукову процедуру, що реалізує метод розподілу інтервалу навпіл.

4.3.15 Опишіть схему пошуку за допомогою методу золотого перетину.

4.3.16 Порівняєте методи розподілу інтервалу навпіл і золотого перетину, використовуючи як показники ефективності характеристику відносного зменшення вихідного інтервалу й кількість обчислень значення функції, необхідних для досягнення заданого ступеня точності (заданої величини відносного зменшення інтервалу).

5 ПОЛІНОМІАЛЬНА АПРОКСИМАЦІЯ ТА МЕТОДИ ТОЧКОВОГО ОЦІНЮВАННЯ

5.1 Теоретичні відомості

У даному розділі розглядаються методи пошуку, які дозволяють врахувати відносні зміни значень функції та, як наслідок, у ряді випадків виявляються більш ефективними, ніж методи виключення інтервалів. Однак виграш в ефективності досягається ціною введення додаткової вимоги, відповідно до якої досліджувані функції повинні бути досить гладкими (гладка функція - це функція, що має безперервну похідну).

Основна ідея цих методів зв'язана з можливістю апроксимації гладкої функції поліномом і наступного використання полінома, що апроксимує, для оцінювання координати точки оптимуму. Відповідно до теореми Вейсрштраса про апроксимацію, якщо функція безперервна в деякому інтервалі, то її з будь-яким ступенем точності можна апроксимувати поліномом досить високого порядку. Отже, якщо функція унімодальна і знайдений поліном, що досить точно її апроксимує, то координату точки оптимуму функції можна оцінити шляхом обчислення координати точки оптимуму полінома. Відповідно до теореми Вейєрштраса, (якість оцінок координати (точки 1 оптимуму. олержуваних за допомогою полінома, що апроксимує, можна підвищити двома способами: використанням полінома більш високого порядку і зменшенням інтервалу апроксимації). Другий спосіб є кращим, оскільки побудова полінома апроксимації порядку вище третього стає дуже складною процедурою, тоді як зменшення інтервалу в умовах, коли виконується припущення про унімодальність функції, особливої складності не представляє.

Метод поліноміальної апроксимації

Ідея цих методів полягає в апроксимації $f_0(u)$ деякою модельною функцією $f_0(u)$, точку мінімуму якої знайти легко, і ця точка береться як поточне наближення шуканої.

Найпростішою модельною функцією буде квадратична:

$$f_0^*(u) = a_0 + a_1(u - u_1) + a_2(u - u_1)(u - u_2). \qquad (5.14)$$

Якщо задана послідовність точок u_1, u_2, u_3 і відомі $f_0(u_1), f_0(u_2), f_0(u_3)$, то можна визначити постійні величини a_0, a_1, a_2 такі, що значення $f_0(u)$ й $f_0^*(u)$ співпадуть принаймні в 3-х точках u_1, u_2, u_3.

При цьому:

$$a_0 = f_0(u_1), a_1 = \frac{f_0(u_2) - f_0(u_1)}{u_2 - u_1},$$
$$a_2 = \frac{1}{u_3 - u_2}\left(\frac{f_0(u_3) - f_0(u_1)}{u_3 - u_1} - \frac{f_0(u_2) - f_0(u_1)}{u_2 - u_1}\right).$$

Визначивши точку екстремуму u_4 модельної функції $f\_0^*(u)$, необхідно обчислити значення $f_0^*(u_4)$ й $f_0(u_4)$. У тому випадку, якщо ці значення будуть досить близькі, можна вважати, що знайдено екстремум $f_0(u)$. Якщо значення $f_0(u_4)$ й $f_0^*(u_4)$ будуть відрізнятися, то необхідно повторити процедуру квадратичної апроксимації відкинувши величини u_1 й $f_0(u_1)$ і прийнявши, що

$$u_1 = u_2, f_0(u_1) = f_0(u_2), u_2 = u_3, f_0(u_2) = f_0(u_3), u_3 = u_4, f_0(u_3) = f_0(u_4).$$

Аналогічні операції необхідно проводити доти, поки не буде досягнутий бажаний збіг $f_0(u)$ й $f_0^*(u)$ у допустимій точці екстремуму.

Необхідно відзначити, що якщо є чотири числові характеристики $u_1 \div u_4$ і $f_0(u_1) \div f_0(u_4)$, то можна здійснювати кубічну апроксимацію функції $f_0(u)$.

Метод послідовного оцінювання із використанням квадратичної апроксимації

Цей метод, розроблений Пауелом, заснований на послідовному застосуванні процедури оцінювання з використанням квадратичної апроксимації. Схему алгоритму можна описати в такий спосіб. Нехай x_1 – початкова точка, Δx - обрана величина кроку по осі *x*.

Крок 1. Обчислити $x_2 = x_1 + \Delta x$.

Крок 2. Обчислити $f(x_l)$ і $f(x_2)$.

Крок 3. Якщо $f(x_l) > f(x_2)$, припускаємо, що $x_3 = x_1 + 2\Delta x$. Якщо $f(x_l) \leq f(x_2)$, припускаємо $x_{32} = x_1 - \Delta x$.

Крок 4. Обчислити $f(x_3)$ і знайти:

$F_{\text{мін}} min = \{f_1, f_2, f_3\}, X_{\text{мін}} =$ точка x_i відповідна $F_{\text{мін}}$

Крок 5. По трьох точках x_1, x_2, x_3 обчислити $\bar{x}$, використовуючи формулу для оцінювання за допомогою квадратичної апроксимації. *Крок 6*. Перевірка на закінчення пошуку.

а) чи є різниця $F_{\text{мин}} - f(\overline{x})$ досить малою?

б) чи є різниця $X_{\text{мин}} - \overline{x}$ досить малою?

Якщо обидві умови виконуються, пошук закінчується. В іншому випадку перейти до кроку 7.

Крок 7. Вибрати «найкращу» точку ($X_{\text{мін}}$ або *x*) і дві точки з обох сторін від неї. Позначити ці точки в природному порядку і перейти до кроку 4.

При першій реалізації кроку 5 границі інтервалу, що містить точку мінімуму, необов'язково виявляються встановленими. При цьому отримана точка $\overline{x}$ може перебувати за точкою x_3. Для того, щоб виключити можливість занадто великого екстраполяційного переміщення, варто провести після кроку 5 додаткову перевірку і у випадку, коли точка $\overline{x}$ знаходиться занадто далеко від x_3, замінити $\overline{x}$

точкою, координата якої обчислюється із врахуванням заздалегідь установленої довжини кроку.

5.2.Приклад застосування методу Пауела

Мінімізувати $f(x) = 2x^2+(16/x)$. Нехай початкова точка $x_1=1$ і довжина кроку $\Delta x = 1$. Для перевірки на закінчення пошуку використовуються наступні параметри збіжності:

$$\left|\frac{\text{Різниця значень } x}{x}\right| \le 3 \times 10^{-2} \quad \left|\frac{\text{Різниця значень } f}{f}\right| \le 3 \times 10^{-3}.$$

Ітерація 1

Крок 1. $x_2 = x_1 + \Delta x = 2.$

Крок 2. $f(x_1) = 18, f(x_2) = 16.$

Крок 3. $f(x_1) > f(x_2)$, отже приймаємо $x_3=1+2=3$.

Крок 4. $f(x_3) = 23.33, F_{\text{мин}} = 16, X_{\text{мин}} = x_2..$

Крок 5. $a_1 = \frac{16-18}{2-1} = -2,$

$$a_2 = \frac{1}{3-2}\left\{\frac{23{,}33-18}{3-1} - a_1\right\} = \frac{5{,}33}{2} + 2 = 4{,}665,$$

$$\overline{x} = \frac{1+2}{2} - \frac{(-2)}{2(4{,}665)} = 1{,}5 + \frac{1}{4{,}665} = 1{,}714,$$

$f(x) = 15{,}210.$

Крок 6. Перевірка на закінчення пошуку:

$$\left|\frac{16-15{,}210}{15{,}210}\right| = 0{,}0519 > 0{,}003$$

Отже, продовжуємо пошук.

Крок 7. Вибираємо x як «найкращу» точку, а $x_1=1$ і $x_2=2$ – як точки, які її оточують. Позначаємо ці точки в природному порядку і переходимо до ітерації 2, що починається із кроку 4.

Ітерація 2

Крок 4. $x_1=1, f_1=18, x_2=1{,}714, f_2=15{,}210=F_{мін}, X_{мін}=x_2, x_3=2, f_3=16.$

Крок 5.

$$a_1 = \frac{15{,}210 - 18}{1{,}714 - 1} = -3{,}908$$

$$a_2 = \frac{1}{2 - 1{,}714}\left\{\frac{16 - 18}{2 - 1} - (-3{,}908)\right\} = \frac{1{,}908}{0{,}286} = 6{,}671,$$

$$\overline{x} = \frac{2{,}714}{2} - \frac{(-3{,}908)}{2(6{,}671)} = 1{,}6125,$$

$$f(\overline{x}) = 15{,}123.$$

Крок 6. Перевірка на закінчення пошуку:

$$\left|\frac{15{,}210 - 15{,}142}{15{,}142}\right| = 0{,}0045 > 0{,}003$$

(умова не виконується)

Крок 7. Вибираємо *x* як «найкращу» точку, а x_1=1 й x_2= 1,714 – як точки, які її оточують.

Ітерація 3

Крок 4. $x_1=1$; $f_1=18$; $x_2=1{,}65$; $f_2=15{,}142=F_{мін}$; $X_{мін}=x_2$; $x_3=1{,}714$; $f_3=15{,}210$.

Крок 5.

$$a_1 = \frac{15{,}142 - 18}{1{,}65 - 1} = 4{,}397,$$

$$a_2 = \frac{1}{1{,}714 - 1{,}650}\left\{\frac{15{,}210 - 18}{1{,}714} - (-4{,}397)\right\} = 7{,}647,$$

$$\overline{x} = \frac{2{,}65}{2} - \frac{(-4{,}397)}{2(7{,}647)} = 1{,}6125,$$

$$f(\overline{x}) = 15{,}123$$

Крок 6. Перевірка на закінчення пошуку:

$$\left|\frac{15{,}142 - 15{,}123}{15{,}123}\right| = 0{,}0013 < 0{,}003,$$

$$\left|\frac{1{,}65 - 1{,}6125}{1{,}6125}\right| = 0{,}023 < 0{,}03.$$

Отже, пошук закінчено.

5.3 Контрольні запитання

5.3.1 У чому полягає основна ідея методів поліноміальної апроксимації й точкового оцінювання?

5.3.2 Сформулюйте необхідні умови ефективної реалізації методу пошуку, заснованого на поліноміальній апроксимації.

5.3.3 По заданих точках і відповідних значеннях функції виведіть формули для оцінки параметрів апроксимуючого квадратичного полінома й оцінки координати точки оптимуму.

5.3.4 Наведіть схему алгоритму методу послідовного оцінювання з використанням квадратичної апроксимації (метод Пауелла).

5.3.5 Чи є методи виключення інтервалів у цілому більше ефективними, ніж методи точкового оцінювання? Чому?

5.3.6 При реалізації пошукових методів рекомендується приймати рішення про закінчення пошуку на основі перевірок як величини різниці значень змінної, так і величини різниці значень функції. Чи можлива ситуація, коли результат однієї з перевірок указує на збіжність до точки мінімуму, тоді як отримана точка в дійсності мінімуму не відповідає? Пояснить відповідь рисунком.

6 МЕТОДИ ОПТИМІЗАЦІЇ З ВИКОРИСТАННЯМ ПОХІДНИХ

6.1 Теоретичні відомості

Усі розглянуті в попередніх розділах методи пошуку ґрунтуються на припущеннях про унімодальність й у ряді випадків про безперервність досліджуваної цільової функції. Доцільно припустити, що, якщо на додаток до умови безперервності ввести вимогу диференційованості функції, то ефективність пошукових процедур можна істотно підвищити. Нагадаємо, що у розд. 1 установлено необхідну умову існування локального мінімуму функції в деякій точці x^*, відповідно до якого перша похідна функції в точці x^* повинна перевтілюватися в нуль, тобто

$$f'(x^*) = \frac{df}{dx}|_{x=x^*} = 0.$$

Якщо функція $f(x)$ містить члени, що містять x у третій і більш високих ступенях, то безпосереднє вирішення аналітичного розв'язання рівняння $f'(x) = 0$ може виявитися складним. У таких випадках використовуються наближені методи послідовного пошуку стаціонарної точки функції f. Насамперед розглянемо класичну пошукову схему, орієнтовану на перебування кореня нелінійного рівняння. Ця схема була розроблена Ньютоном і пізніше уточнена Рафсоном.

Метод Ньютона-Рафсона

У рамках схеми Ньютона-Рафсона передбачається, що функція f двічі диференційована. Робота алгоритму починається в точці x_1, яка є початковим наближенням (або початковою оцінкою) координати стаціонарної точки, або кореня рівняння $f'(x) = 0$. Потім будується лінійна апроксимація функції $f'(x)$ в точці x_1, а точка, у якій лінійна апроксимація функції перетворюється в нуль, приймається як наступне наближення. Якщо точка x_k прийнята як поточне наближення до стаціонарної точки, то лінійна функція, що апроксимує функцію $f'(x)$ в точці x_k, записується у вигляді

$$\tilde{f}'(x; x_k) = f'(x_k) + f''(x_k)(x - x_k). \qquad (6.1)$$

У точці $x_k = x_{k+1}$ $\tilde{f}'(x; x_k) = 0$

Прирівняємо праву частину рівняння (6.1) нулю і одержимо наступне наближення:

$$x_{k+1} = x_k - [f'(x_k)/f''(x_k)]. \qquad (6.2)$$

Рис. 6.1 ілюструє основні кроки реалізації методу Ньютона-Рафсона.

В залежності від вибору початкової точки і виду функції алгоритм може як сходитися до істинної стаціонарної точки, так і розходитися, що показано на рис. 6.2. Якщо початкова точка розташована правіше x_0, то одержані в наслідок послідовних наближень точки віддаляються від стаціонарної точки x_1.

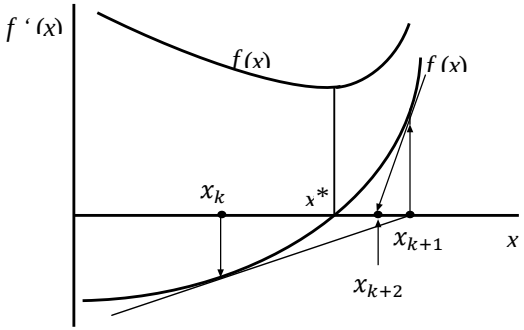


Рисунок 6.1 – Метод Ньютона-Рафсона (збіжність)

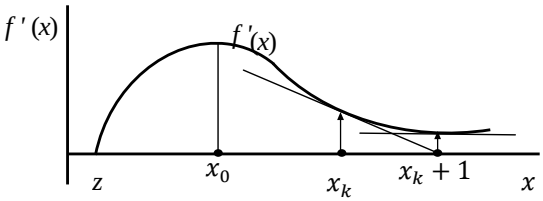


Рисунок 6.2 – Метод Ньютона-Рафсона (відсутність збіжності)

Метод середньої точки

Якщо функція $f(x)$ унімодальна у заданому інтервалі пошуку, то точкою екстремуму є точка, у якій $f'(x) = 0$. Якщо при цьому є можливість обчислювати як значення функції, так і її похідну, то для знаходження кореня рівняння $f'(x) = 0$ можна скористатися ефективним алгоритмом виключення інтервалів, на кожній ітерації якого розглядається лише одна спробна точка. Наприклад, якщо в точці x_1 виконується нерівність $f'(x_1) > 0$, то з урахуванням припущення про унімодальність природно стверджувати, що точка мінімуму не може знаходитися правіше x_1 і інтервал $x \geq x_1$ можна виключити. Приведені міркування лежать в основі логічної структури методу середньої точки, що іноді називають пошуком Больцано.

Визначимо дві точки a і b таким чином, що $f'(a) < 0$ і $f'(b) > 0$. Стаціонарна точка розташована між a і b. Обчислимо значення похідної функції в середній точці розглянутого інтервалу $x_1 = (a + b)/2$. Якщо б $f'(x_1) > 0$, то інтервал (x_1, b) слід виключити з інтервалу пошуку. З іншого боку, якщо $f'(x_1) < 0$ (див. рис. 6.3), то можна виключити інтервал (a, x_1). Нижче подано формалізований опис кроків алгоритму.

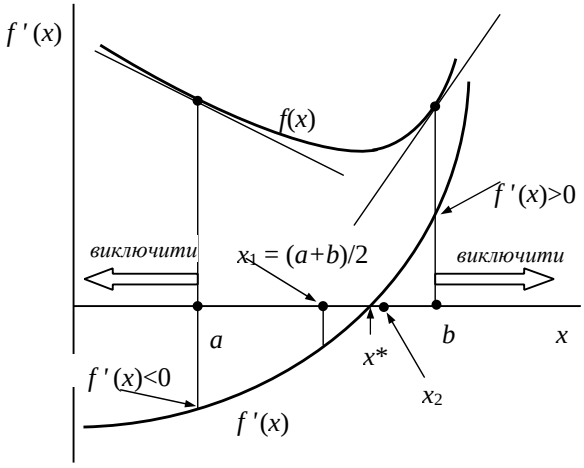


Рисунок 6.3 – Метод середньої точки

Нехай маємо обмежений інтервал $a \le x \le b$ і задану заздалегідь установлену величину припустимого відхилення – параметр ε.

Крок 1. Початковий інтервал (a, b); при цьому $f'(a) < 0$ і $f'(b) > 0$.

Крок 2. Обчислити $x_1 = (a + b)/2$ і $f'(x_1)$.

Крок 3. Якщо $|f'(x_1)| < \varepsilon$ – пошук закінчено, можна визначити точку x_1 точкою екстремуму. У протилежному випадку ($|f'(x_1)| > \varepsilon$) якщо $f'(x_1) > 0$, покласти $b = x_1$ і перейти до кроку 2. Якщо $f'(x_1) < 0$, покласти $a = x_1$ і перейти до кроку 2.

Слід зазначити, що логічна структура пошуку відповідно до викладеного методу виключення інтервалів заснована лише на дослідженні знака похідної незалежно від значень, які ця похідна приймає. У наступному підрозділі розглядається метод січних, при реалізації якого враховуються як знак похідної, так і її значення.

Метод січних

Метод січних є комбінацією методу Ньютона-Рафсона і загальної схеми виключення інтервалів та орієнтований на знаходження кореня рівняння $f(x) = 0$ у інтервалі (a, b), якщо, зрозуміло, такий корінь існує.

Припустимо, що в процесі пошуку стаціонарної точки функції $f(x)$ в інтервалі (a, b) виявлені дві точки a і b, у яких знаки похідної різні. У цьому випадку алгоритм методу січних дозволяє апроксимувати функцію $f'(x)$ «січною прямою» (прямою лінією, що з'єднує дві точки) і знайти точку, у якій січна графіка $f'(x)$ перетинає вісь абсцис (рис. 6.4). Таким чином, наступне наближення до стаціонарної точки x^* визначається із співвідношення

$$\frac{b - a}{f'(b) - f'(a)} = \frac{b - x_1}{f'(b)} \tag{6.3}$$

за формулою

$$x_1 = b - \frac{(b-a)f'(b)}{f'(b)-f'(a)} \tag{6.4}$$

Якщо $|f'(x_1)| \le \varepsilon$, пошук слід закінчити. У протилежному випадку необхідно вибрати одну з точок a або b таким чином, щоб знаки похідної в цій точці і точці x_1 були різні, а потім повторити основний крок алгоритму. Наприклад, у ситуації, що показана на рис. 4, у якості

двох наступних точок мають бути обрані точки x_1 і b. Легко бачити, що на відміну від методу середньої точки метод січних заснований на дослідженні не тільки знака, але і значень похідної у спробних точках і тому в ряді випадків дозволяє виключити більш половини інтервалу пошуку (див. рис. 6.4).

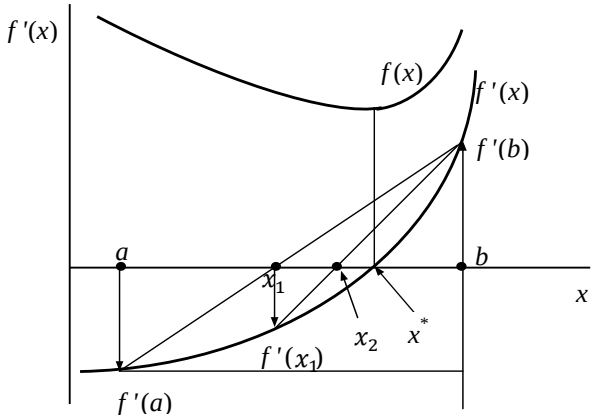


Рисунок 6.4 – Метод січних

Метод пошуку з використанням кубічної апроксимації

Відповідно до методу функція $f(\bar{x})$, що мінімізується, апроксимується поліномом третього порядку. Логічна схема методу аналогічна схемі методу з використанням квадратичної апроксимації. Однак у цьому випадку побудова полінома апроксимації проводиться на основі меншого числа точок, оскільки в кожній точці можна обчислювати і значення функції, і її похідної.

Робота алгоритму починається в довільно обраній точці x_1: знаходиться інша точка x_2, така, що похідні $f'(x_1)$ і $f'(x_2)$ мають різні знаки. Інакше кажучи, необхідно вмістити стаціонарну точку $\bar{x}$, в якій $f'(x_1) = 0$, в інтервал між x_1 і x_2. Кубічна функція полінома апроксимації записується в наступному вигляді:

$$\bar{f}(x) = a_0 + a_1(x - x_1) + a_2(x - x_1)(x - x_2) + \\ + a_3(x - x_1)^2(x - x_2) \qquad (6.5)$$

Параметри рівняння (6.5) підбираються таким чином, щоб значення $\bar{f}(x)$ і її похідної у точках x_1 і x_2 збігалися зі значеннями $f(x)$ і $f'(x)$ в цих точках. Перша похідна функції $\bar{f}(x)$ дорівнює:

$$\frac{d\bar{f}(x)}{dx} = a_1 + a_2(x - x_1) + a_2(x - x_2) + a_3(x - x_1)^2 + \\ +2a_3(x - x_1)(x - x_2) \quad (6.6)$$

Коефіцієнтиа$_0$, a_1, a_2 й a_3 кубічного полінома (6.5) визначаються за відомим значенням $f(x_1), f(x_2), f'(x_1)$і $f'(x_2)$ шляхом розв'язку наступної системи лінійних рівнянь:

$$f_1 = f(x_1) = a_0, \quad (6.7)$$
$$f_2 = f(x_2) = a_0 + a_1(x_2 - x), \quad (6.8)$$
$$f_1' = f'(x_1) = a_1 + a_2(x_1 - x_2), \quad (6.9)$$
$$f_2' = f'(x_2) = a_1 + a_2(x_2 - x_1) + a_3(x_2 - x_1)^2. \quad (6.10)$$

Помітимо, що дана система легко вирішується рекурсивним методом. Після того як коефіцієнти знайдені, діючи за аналогією з випадком квадратичної апроксимації, можна оцінити координату стаціонарної точки функції за допомогою кубічного полінома (6.5). При цьому прирівнювання до нуля похідної (6.5) приводить до квадратного рівняння. Використовуючи формулу для обчислення коренів квадратного рівняння, запишемо розв'язок, що визначає стаціонарну точку кубічного полінома апроксимації:

$$\bar{x} = \begin{cases} x_2, & \text{якщо } \mu < 0, \\ x_2 - \mu(x_2 - x_1), & \text{якщо } 0 \le \mu \le 1, \\ x_1, & \text{якщо } \mu > 1, \end{cases} \quad (6.11)$$

де:

$$\mu = \frac{f_2' + \omega - z}{f_2' - f_1' + 2\omega}, \quad (6.12)$$

$$z = \left(\frac{3(f_1 - f_2)}{x_2 - x_1}\right) + f_1' + f_2', \quad (6.13)$$

$$\omega \begin{cases} (z^2 - f_1' f_2')^{1/2}, \text{якщо } x_1 < x_2, \\ -(z^2 - f_1' f_2')^{\frac{1}{2}}, \text{якщо } x_1 > x_2. \end{cases} \quad (6.14)$$

Формула (6.14) забезпечує належний вибір одного із двох коренів квадратного рівняння; для значень μ, що знаходяться в інтервалі від 0 до 1, формула (6.11) гарантує, що отримувана точка $\bar{x}$ розташована між x_1 і x_2. Потім знову вибираються дві точки для реалізації процедури кубічної апроксимації — $\bar{x}$ і одна із точок x_1 або x_2, причому значення похідної досліджуваної функції в цих точках повинні бути протилежні за знаком, і процедура кубічної апроксимації повторюється.

Наведемо формалізований опис алгоритму. Нехай задані початкова точка x_0, додатна величина кроку Δ і параметри збіжності ε_1 й ε_2.

Крок 1. Обчислити $f'(x_0)$.

Якщо $f'(x_0) < 0$, то обчислити $x_{k+1} = x_k + 2^k\Delta$ для значень k=0,1…

Якщо $f'(x_0) > 0$, то обчислити $x_{k+1} = x_k - 2^k\Delta$ для значень k=0,1...

Крок 2. Обчислити значення $f'(x)$ в точках x_{k+1} при k=0, 1, 2, ... аж до точки x_m, у якій $f'(x_{m-1})f'(x_m) \le 0$. Потім прийняти $x_1 = x_{m-1}, x_2 = x_m$. Обчислити значення f_1, f_2, f_1' і f_2' .

Крок 3. Знайти стаціонарну точку $\bar{x}$ апроксимуючого кубічного поліному, користуючись формулами (6.11) – (6.14).

Крок 4. Якщо $f(\bar{x}) < f(x_1)$, то перейти до кроку 5. У протилежному випадку обчислювати $\bar{x}$ по формулі $\bar{x} = \bar{x} + \frac{1}{2(\bar{x} - x_1)}$ доти, поки не буде виконуватися нерівність $f(\bar{x}) \le f(x_1)$.

Крок 5. Перевірка на закінчення пошуку.

Якщо $|f'(\bar{x})| \le \varepsilon_1$ і $\left|\left(\bar{x} - \frac{x_1}{\bar{x}}\right)\right| \le \varepsilon_2$, пошук закінчити.

Інакше прийняти або

а) $x_2 = x_1$ і $x_1 = \bar{x}$, якщо $f'(\bar{x})f'(x_1) < 0$, або

б) $x_1 = \bar{x}$, якщо $f'(\bar{x})f'(x_2) < 0$.

Потім перейти до кроку 3.

Кроки 1 і 2 реалізують процедуру пошуку меж інтервалу евристичним методом, до того ж зміна знаку похідної використовується як критерій переходу через точку оптимуму. На кроці 3 проводяться обчислення координати точки оптимуму поліному апроксимації. Крок 4 асоційований з перевіркою того факту, що отримана оцінка дійсно є поліпшеним наближенням до точки оптимуму. У випадку, коли значення похідної обчислюються безпосередньо, метод пошуку з використанням кубічної апроксимації, безумовно, є більш ефективним у порівнянні з кожним із наведених вище методів пошуку. Однак якщо значення похідної обчислюються шляхом диференціювання різниць, то перевагу варто віддати методу Пауела, який базується на квадратичній апроксимації.

6.2 Контрольні запитання

6.2.1 Розглянуті раніше методи оптимізації засновані на припущенні про унімодальність й у ряді випадків про безперервність досліджуваної цільової функції. Виконання якої додаткової умови необхідно для реалізації методів оптимізації з використанням похідних?

6.2.2 Опишіть пошукову процедуру, що реалізує метод Ньютона-Рафсона.

6.2.3 Як впливає вибір початкової точки на збіжність алгоритму методу Ньютона-Рафсона. Пояснить відповідь малюнками.

6.2.4 Опишіть схему пошуку з використанням методу середньої точки (пошуку Больцано).

6.2.5 Опищіть схему алгоритму методу січних (методу хорд).

6.2.6 Наведіть опис алгоритму методу пошуку з використанням методу кубічної апроксимації.

7 МЕТОДИ ПРЯМОГО ПОШУКУ В ЗАДАЧАХ БАГАТОВИМІРНОЇ БЕЗУМОВНОЇ ОПТИМІЗАЦІЇ. МОДИФІКОВАНА ПРОЦЕДУРА ПОШУКУ ПО СИМПЛЕКСУ НЕДЛЕРА-МІДА

Мета роботи - вивчити метод безумовної багатовимірної оптимізації Нелдера-Міда.

7.1 Короткі теоретичні відомості

Метод пошуку по симплексу (s2 - метод)

При вирішенні задач з двома змінними можна скористатися квадратним зразком. Найкраща (F_{min}) з 5 досліджуваних точок вибирається як наступна базова точка, навколо якої будується наступний зразок. Якщо жодна з кутових точок не має переваг перед базовою, розміри зразка слід зменшити, після чого продовжити пошук. Цей тип еволюційної оптимізації був використаний для аналізу функціонування промислових підприємств, коли ефект варіювання значень змінних, що описують виробничі процеси, вимірюється з помилкою. У задачі великої розмірності обчислення значень цільової функції проводиться у всіх вершинах, а також в центрі тяжіння гіперкуба, тобто в точках так званого кубічного зразка. Гіперкубом називається куб в N-вимірному евклідовому просторі, тобто безліч $X=(X1,X2.,XN)Є N$, $a_i \le x_i \le b_i$, i=1,N, де a і b - задані числа.

Якщо кількість змінних (розмірність простору, в якому ведеться пошук) рівна N, то пошук за кубічним зразком вимагає 2N+1 обчислень значення функції для даного зразка. При збільшенні розмірності задачі необхідна кількість обчислень значення цільової функції зростає надзвичайно швидко. Таким чином, не дивлячись на логічну простоту пошуку за кубічним зразком, виникає необхідність використання ефективніших методів прямого пошуку для вирішення задач оптимізації, що виникають на практиці.

Одна із стратегій пошуку покладена в основу методу пошуку по симплексу. Процедура симплексного пошуку базується на тому, що

експериментальним зразком, який містить найменшу кількість точок, є регулярний симплекс. **Регулярний симплекс** в N-мірному просторі-багатогранник, утворений N+1 рівновіддаленими одна від одної точками (вершинами). В разі двох змінних, симплексом є рівносторонній трикутник, в тривимірному просторі, симплекс є тетраедром. У алгоритмі симплексного методу використовується властивість симплексу, згідно якій новий симплекс можна побудувати на будь-якій грані початкового симплексу шляхом перенесення обраної вершини на належну відстань вздовж прямої, проведеної через центр тяжіння решти вершин початкового симплексу. Отримана таким чином точка є вершиною нового симплексу, а обрану при побудові початкового симплексу вершину виключають. Потрібне одне обчислення значення цільової функції (при переході до нового симплексу).

Робота алгоритму починається з побудови регулярного симплексу в просторі незалежних змінних і оцінювання значень цільової функції в кожній з вершин симплексу. При цьому визначається вершина, якою відповідає найбільше значення цільової функції. Потім знайдена вершина проектується через центр тяжіння решти вершин в нову точку, яка використовується як вершина нового симплексу. Якщо функція зменшується достатньо плавно, ітерації продовжуються до тих пір, поки або не буде накрита точка мінімуму, або не почнеться циклічний рух по симплексу. У таких ситуаціях можна скористатися наступними трьома правилами:

Правило 1. «*Накриття» точки мінімуму.*

Якщо вершина, якій відповідає найбільше значення цільової функції, побудована на попередній операції, то замість неї береться вершина, якій відповідає наступне за величиною значення цільової функції.

Правило 2. *Циклічний рух.*

Якщо деяка вершина симплексу не виключається впродовж більш, ніж М ітерацій, то необхідно зменшити розмір симплексу за допомогою коефіцієнта редукції і побудувати новий симплекс, вибравши як базову точку, якій відповідає мінімальне значення цільової функції.

Запропоновано обчислювати M за формулою:

$$M = 1.65 * N + 0.05 * N^2 \quad (7.1)$$

де N - розмірність задачі;

M - округляється до найближчого цілого.

Для застосування даного правила потрібно встановити величину коефіцієнта редукції.

Правило 3. *Критерій закінчення пошуку.*

Пошук завершується, коли розміри симплексу або різниці між значеннями функції у вершинах стають достатньо малими. Щоб можна було застосовувати ці правила, необхідно задати величину параметра закінчення пошуку. Реалізація алгоритму, що вивчається, заснована на обчисленнях двох типів:

1. Обчислення координат регулярного симплексу при заданих базовій точці і масштабному множнику.

2. Розрахунку координат віддзеркаленої точки.

При заданій початковій точці X(0) і масштабному множнику α координати решти N вершин симплексу в N-мірному просторі обчислюються за формулою (7.2):

$$x^{(i)} = \begin{cases} x_j{}^{(0)} + \delta_1, \text{якщо } i \neq j, \\ x_j{}^{(0)} + \delta_2, \text{якщо } i = j \end{cases} \quad (7.2)$$

для i, j=1, 2,…,N,

де i – номер вершини;

j – номер координати.

Прирости δ_1 і δ_2, які залежать тільки від N і вибраного множника α, обчислюються за наступними формулами (7.3) і (7.4).

$$\delta_1 = \left[\frac{(N+1)^{1/2}+N-1}{N\sqrt{2}}\right] \cdot a \tag{7.3}$$

$$\delta_2 = \left[\frac{(N+1)^{1/2}-1}{N\sqrt{2}}\right] \cdot a \tag{7.4}$$

Значення масштабного множника α вибирається виходячи з характеристик вирішуваного завдання.

При α=1 ребра регулярного симплексу мають одиничну довжину. Центр тяжіння решти N точок розташований в точці x_c:

$$x_c = \frac{1}{N}\sum_{\substack{i=1 \\ i\neq j}}^{N} x^{(i)} \tag{7.5}$$

Всі точки прямої, яка проходить через $x^{(j)}$ і x_c, задаються формулою:

$$x_c = x^{(j)} + \lambda * (x_c - x^{(j)}) \tag{7.6}$$

При λ=1 отримуємо вихідну точку $x^{(j)}$, тоді як значення λ=1 відповідає центру тяжіння x_c. Для того, щоб побудований симплекс мав властивість регулярності, віддзеркалення має бути симетричним. Отже, нова вершина отримується при λ=2.

$$x^{(j)}{}_{\text{нове}} = 2 * x_c - x^{(j)}$$

Приклад. *Обчислення відповідно до методу пошуку по симплексу.*

Мінімізувати $f(x)=(1-x_1)^2+(2-x_2)^2$.

Розв’язок:

1. Для побудови початкового симплексу потрібно задати початкову точку і масштабувальний множник.

Хай $x^{(0)}=[0;0]^T$, α=2, тоді

$\delta_1 = \frac{\sqrt{3}+2-1}{2\sqrt{2}} \cdot 2 = \frac{\sqrt{3}+1}{\sqrt{2}} = 1.9318$;

$\delta_2 = \frac{\sqrt{3}-1}{2\sqrt{2}} \cdot 2 = \frac{\sqrt{3}-1}{\sqrt{2}} = 0.5176$.

2.Використовуючи ці два параметри, обчислимо координати двох основних вершин симплексу:

$x^{(1)}=[0+0.5176;0+1.9318]^T=[0.5176;1.9318]^T$;

$x^{(2)}=[0+1.9318;0+0.5176]^T=[1.9318;0.5176]^T$.

3.Даним точкам $x^{(1)}$ і $x^{(2)}$ відповідають значення цільової функції:

$f(x^{(1)})$ =0.2374 ;

$f(x^{(2)})$ =3.0658 .

4.Оскільки $f(x^{(0)})$=5, необхідно відобразити точку $x^{(0)}$ через центр тяжіння решти двох вершин.

$$x_c = \frac{1}{2}\sum_{i=1}^{2} x^{(i)} = \frac{1}{2}(x^{(1)} + x^{(2)}) \qquad (7.7)$$

5.Використовуючи формулу (7.6), отримуємо:

$x^{(3)}=x^{(1)}+x^{(2)}-x^{(0)}$

$x^{(3)}=[0.5176+1.9318-0;1.9318+0.5176-0]^T=[2.4494;2.4494]^T$

У отриманій точці $f(x^{(3)})$=2,3027.

Тобто, спостерігається зменшення цільової функції. Новий симплекс утворений точками $x^{(1)}$, $x^{(2)}$, $x^{(3)}$. Відповідно до алгоритму, слід відобразити точку $x^{(2)}$, якій відповідає найбільше значення цільової

функції, через центр тяжіння точок $x^{(1)}$ і $x^{(3)}$. Ітерації продовжуються, поки не буде потрібно застосування сформульованих вище правил.

Викладений вище алгоритм S^2-методу має декілька очевидних переваг:

1. Розрахунки і логічна структура методу відрізняються порівняльною простотою, і, отже, відповідна програма виявляється відносно короткою.

2. Рівень вимог до обсягу пам'яті невисокий. Масив має розмірність (N+1, N+2).

3. Використовується порівняно невелике число заздалегідь встановлених параметрів (α, коефіцієнт редукції і параметр закінчення пошуку).

4. Алгоритм виявляється ефективним навіть в тих випадках, коли помилка обчислення значень цільової функції велика, оскільки при його реалізації оперують найбільшими значеннями функції у вершинах, а не найменшими.

Алгоритм має ряд недоліків:

1. Не виключено виникнення складностей, пов'язаних з масштабуванням, оскільки всі координати вершин симплексу залежать від одного і того ж масштабованого множника α. Щоб обійти складності такого роду в практичних завданнях, слід промасштабувати всі змінні для того, щоб їх значення були порівнянними за величиною.

$$Z = \frac{X_i - X_{i,0}}{\triangle X_i} \tag{7.8}$$

2. Алгоритм працює дуже повільно, оскільки отримана на попередніх ітераціях інформація не використовується для прискорення пошуку.

3. Не існує простого способу розширення симплексу, що не вимагає перерахунку значень цільової функції в усіх точках зразка.

Таким чином, якщо α з якої-небудь причини зменшується, наприклад, якщо зустрічає зону з вузькою западиною або хребтом, то пошук повинен продовжуватися із зменшеною величиною кроку.

Симплексний метод рекомендується для використання при безперервній оптимізації промислових об'єктів в умовах високого рівня шумів і дрейфу екстремальної точки цільової функції.

Модифікована система пошука по симплексу Нелдера – Міда

Хоча формула для визначення регулярного симплексу виявляється доволі зручною при побудові початкового зразка, проте вірних підстав для збереження властивостей регулярності симплексу в процесі пошуку немає. Отже, при віддзеркаленні симплексу існує можливість, як його розтягування, так і стискування. З цієї причини процедуру Нелдера- Міда інколи називають методом пошуку по деформованому багатограннику. При розрахунках по методу Нелдера-Міда використовуються вершини симплексу $x_{(h)}$, яким відповідає найбільше значення цільової функції $f_{(h)}$, вершина $x_{(g)}$, якій відповідає наступне по величині значення цільової функції $f_{(g)}$ і $x_{(e)}$, якій відповідає найменше значення цільової функції $f_{(e)}$.

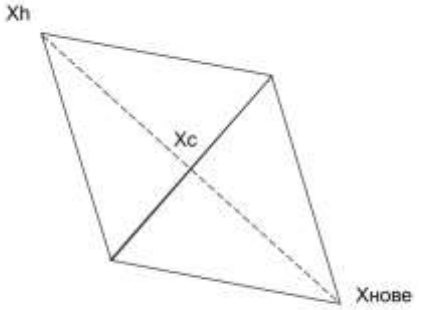
Віддзеркалення вершини симплексу здійснюється за прямою:

$$x = x_{(h)} = x * (x_c - x_{(h)}) \tag{7.9}$$

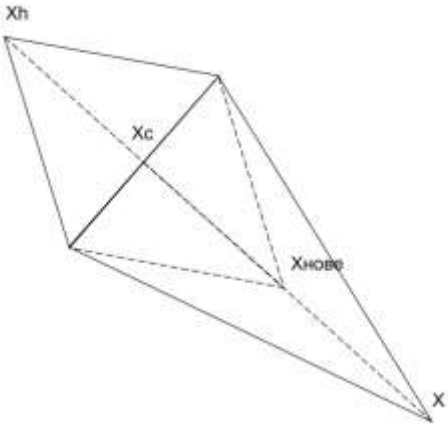
$$x = x_{(h)} + (1 + \theta) * (x_c - x_{(h)}) \tag{7.10}$$

При θ=1 має місце нормальне віддзеркалення симплексу, оскільки точка $x_{нове}$ розташовується на відстані (x_c-$x_{(h)}$)) від точки x_c, при -1<=θ <=+1 спостерігається стисле віддзеркалення або стиск симплексу. Вибір θ =>1забезпечує розтягнуте віддзеркалення або розтягування симплексу.

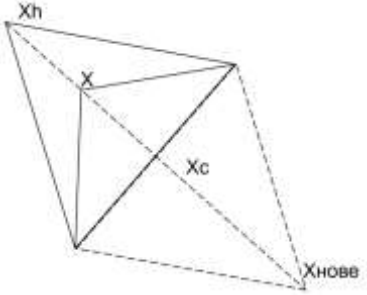
Різні види відображення представлені на рис. 7.1



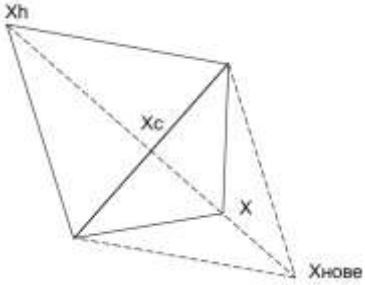
а)



б)



в)



д)

а) нормальне відображення ($\theta=\alpha=1$), $f_{нове}<f_{(g.)}$;

б) розтягування ($\theta=\alpha>1$), $f(x_{нове})<f_{(e)}$;

в) стиск ($\theta=\beta<0$), $f(x_{нове})>f_{(g)}$, $f(x_{нове})>f_{(h)}$; д) стиск ($\theta=\beta>0$), $f_{(g)}<f(x_{нове})<f_{(h)}$

Рисунок 7.1 - Розтягування та зжимання симплексу

Три значення параметру θ, що використовуються при нормальному віддзеркаленні, стиску та розтягуванні, позначаються (λ,β,γ).

Реалізація методу починається з побудови початкового симплексу і визначення точок $x_{(h)}$, $x_{(g)}$, $x_{(e)}$, x_c, після нормального віддзеркалення здійснюється перевірка значень цільової функції за критерієм закінчення пошуку в точках відображеного симплексу, якщо пошук не закінчений за допомогою тестів, обирається одна з операцій: нормальне віддзеркалення, розтягування або стиск. Ітерації продовжуються, поки зміни значень цільової функції в вершинах симплексу не стануть незначними. Як задовільні значення параметрів (λ,β,γ) Нелдер і Мід рекомендують використовувати λ =1, β=0,5, γ =2.

Метод Нелдера-Міда володіє достатньою ефективністю і високою надійністю в умовах наявності випадкових збурень або помилок при визначенні значень цільової функції.

7.2 Контрольні запитання

7.2.1 7аведіть класифікацію методів безумовної багатовимірної оптимізації.

7.2.2 Опишіть метод пошуку за симплексом.

7.2.3 Охарактеризуйте ситуацію накриття точки мінімуму, що виникає під час симплексного пошуку.

7.2.4 Охарактеризуйте ситуацію циклічного руху під час симплексного пошуку.

7.2.5 Що є критерієм закінчення симплексного пошуку?

7.2.6 Як визначаються координати вершин початкового симплексу?

7.2.7 Як визначаються координати відображеної вершини симплексу?

7.2.8 Переваги та недоліки методу пошуку за симплексом.

7.2.9 Опишіть модифіковану процедуру пошуку за симплексом Нелдера-Міда.

8 МЕТОД ПОШУКУ ХУКА-ДЖИВСА. МОДИФІКАЦІЇ ПРОЦЕДУРИ ХУКА-ДЖИВСА

Мета роботи - вивчити метод безумовної багатовимірної оптимізації Хука-Дживса.

8.1 Короткі теоретичні відомості

Процедура Хука-Дживса є комбінацією дослідницького пошуку і прискореного пошуку за зразком з використанням певних евристичних правил. Дослідницький пошук орієнтований на виявлення характеристик локальної поведінки цільової функції і визначення напряму уздовж «ярів». Отримана в результаті дослідницького пошуку інформація потім використовується в процесі пошуку за зразком при русі по «ярах».

Для проведення **дослідницького пошуку** необхідно задати величину кроку, яка може бути різною для різних координат і напрямів і змінюватися в процесі пошуку. Дослідницький пошук починається, якщо значення цільової функції в пробній точці не перевищує значення функції в початковій точці. Цей крок пошуку розглядається як успішний. Інакше необхідно повернутися в попередню точку, і зробити крок в протилежному напрямі з подальшою перевіркою значення цільової функції.

Якщо в результаті виходить точка з меншим значенням цільової функції, чим в точці $X^{(к)}$, то вона розглядається як нова базова точка $X^{(к+1)}$. З іншого боку, якщо дослідницький пошук невдалий, необхідно повернутися в точку $X^{(к)}$ і провести дослідницький пошук з метою виявлення нового напряму мінімізації. Зрештою виникає ситуація коли такий пошук не приводить до успіху. В цьому випадку потрібно зменшити величину кроку шляхом введення деякого множника і відновити дослідницький пошук. Пошук завершується коли величина кроку стає досить малою. Послідовність точок, отриманих в процесі реалізації методу, можна записати в такому вигляді:

$X^{(к)}$ – поточна базова точка;

$X^{(к-1)}$ – попередня базова точка;

$X_p^{(к+1)}$ – точка побудови в наближенні до зразка;

$X^{(к+1)}$ – наступна нова базова точка.

Приведена послідовність характеризує логічну структуру пошуку по методу Хука-Дживса.

Алгоритм методу пошуку Хука-Дживса

Крок 1: Визначити початкову точку X_0, прирощення Δ_i, i=1,…,N коефіцієнт зменшення кроку $\alpha>1$ і параметр закінчення пошуку $\varepsilon>0$.

Крок 2: Провести дослідницький пошук.

Крок 3: Перевірити, чи був дослідницький пошук вдалий (чи знайдена точка з меншим значенням цільової функції)?

«Так»: перейти до кроку 5.

«Ні» : продовжити.

Крок 4: перевірка закінчення пошуку. Чи виконується нерівність $||\Delta x||<\varepsilon$?;

«Так» : припинити пошук, поточна точка апроксимує точку оптимуму X*.

«Ні» : зменшити приріст $\Delta_i = \Delta_i/\alpha_i$, i=1,…,N. Перейти до кроку 2.

Крок 5 : Провести **пошук за зразком**: $X_p^{(к+1)=} X^{(к)}+ (X^{(к)}- X^{(к-1)})$

Крок 6 : Провести дослідницький пошук, використовуючи $X_p^{(к+1)}$ як базову точку. Хай $X^{(к+1)}$ – отримана в результаті точка.

Крок 7 : Чи виконується умова $f(X^{(к+1)})<f(X^{(к)})$?.

«Так» : покласти $X^{(к-1)} = X^{(к)}$; $X^{(к)}= X^{(к+1)}$. Перейти до кроку 5.

«Ні» : перейти до кроку 6.

Приклад пошуку за методом Хука-Дживса.

Знайти точку мінімуму функції $f(x)=8x_1^2+4x_1x_2+5x_2^2$, використовуючи точку $x^{(0)}[-4;-4]^T$.

Розв'язок.

Для того, щоб використати метод прямого пошуку Хука-Дживса, необхідно знати такі векторні величини:

- векторна величина прирощення $\Delta x=[-1;1]^T$;

- коефіцієнт зменшення кроку $\alpha=2$;

- параметр закінчення пошуку $\varepsilon=10^{-4}$.

Ітерації починаються з дослідницького пошуку навколо точки x_0, що відповідає значенню $f(x^{(0)})=272$.

Фіксуючи x_2, дамо приріст x_1.

$x_2= -4$.

$x_1= -4+1 \rightarrow f(-3;-4)=200< f(x^{(0)})=272 \rightarrow$ успіх.

Це означає, що необхідно зафіксувати $x_1=-3$ та дати приріст змінній x_2.

$x_2=-3$

$x_1=-4+1 \rightarrow f(-3;-3)=153<200 \rightarrow$ успіх.

Таким чином, в результаті дослідницького пошуку знайдемо точку $x^{(1)}=[-3;-3]^T$.

$f(x_1^{(1)})=153$.

Оскільки дослідницький пошук був успішнім, переходимо до пошуку за зразком.

$X^{(2)}=x^{(1)}+(x^{(1)}-x^{(0)})=[-2;-2]^T$.

$F(x_p^{(2)})=68$.

Далі проводимо дослідницький пошук навколо точки $x_p^{(2)}$, який виявляється успішним при застосуванні додатних змінних x_1, x_2.

В результаті отримуємо точку $x^{(2)}=[-1;-1]^T$.

$f(x_2)=17$.

$F(x^{(2)})< f(x^{(1)})$, пошук за зразком можемо вважати успішним, $x^{(2)}$ стає новою базовою точкою при проведенні нового пошуку за зразком.

Ітерації продовжуються доки зменшення величини кроку не вкаже на закінчення пошуку в околиці точки.

Модифікації процедури Хука-Дживса

Метод Хука-Дживса характеризується нескладною стратегією пошуку, відносною простотою обчислення і невисоким рівнем вимог до об'єму пам'яті, який виявляється нижчим, ніж при використанні методу пошуку за симплексом.

Завдяки цьому алгоритм Хука-Дживса знаходить широке застосування на практиці.

Особливо зручний метод з використанням штрафних функцій, проте заснований на циклічному проходженні по координатах алгоритм у ряді випадків може закінчувати роботу передчасно, а при отриманні значних нелінійних дефектів вироджується в послідовність дослідницьких пошуків без переходу до прискорювального пошуку за зразком.

Відомий цілий ряд підходів до удосконалення методу Хука-Дживса. Відома модифікація процедури Хука-Дживса шляхом введення додаткових правил збільшення і зменшення приросту змінних, а також правило зменшення кроку за зразком, який застосовується в тих випадках, коли звичайний крок виявляється невдалим.

Експерименти дозволили допрацювати іншу фазу реалізації алгоритму, яку називають використанням зразка.

Якщо рух за зразком проходить до успіху, є певні підстави для того, щоб повністю використовувати можливості пошуку уподовж прямої, визначеної зразком, або принаймні звеличити довжину кроку за зразком. Така дія часто дозволяє істотно прискорити збіжність методу.

При іншій модифікації методу змінюється фаза дослідницького пошуку шляхом введення системи ортогональних напрямів пошуку, орієнтація якої випадковим чином міняється на кожній ітерації.

8.2 Контрольні запитання

8.2.1. Опишіть метод пошуку Хука-Дживса.

8.2.2. Яка мета дослідного пошуку в процедурі Хука-Дживса?

8.2.3. В чому полягає пошук за зразком в процедурі Хука-Дживса?

8.2.4. Які існують модифікації процедури Хука-Дживса?

9 МЕТОД СПРЯЖЕНИХ НАПРЯМКІВ ПАУЕЛЛА

Мета роботи - вивчити метод спряжених напрямків Пауелла.

9.1 Короткі теоретичні відомості

Найбільш ефективним з алгоритмів прямого пошуку є метод, розроблений Пауеллом. При роботі цього алгоритму інформація, отримана на попередніх ітераціях, використовується для побудови векторів напряму пошуку, а також для усунення зациклення послідовності координатних пошуків.

Метод орієнтований на вирішення завдань з квадратичними цільовими функціями і грунтується на фундаментальних теоретичних результатах.

Завдання з квадратичними цільовими функціями займають важливе місце в теорії оптимізації з двох причин:

1. Квадратична функція є найбільш простим типом нелінійних функцій, для яких може бути сформульовано завдання безумовної оптимізації. Лінійні функції не мають внутрішнього оптимуму. Отже, якщо за допомогою того або іншого методу успішно вирішуються завдання оптимізації з цільовими функціями загального вигляду, то такий метод повинен виявитися ефективним при вирішенні завдань з квадратичними функціями.

2. В околі точки оптимуму будь-яку нелінійну функцію можна апроксимувати квадратичною функцією, оскільки лінійний член розкладання Тейлора звертається в нуль. Отже, робота алгоритму при вирішенні завдань з квадратичними функціями дозволяє отримати певні уявлення про збіжність алгоритму у разі, коли мінімізується функція загального вигляду.

Основна ідея алгоритму полягає в тому, що якщо квадратична функція N змінних приведена до виду суми повних квадратів, то її оптимум може бути знайдений в результаті реалізації N одновимірних пошуків по перетворених координатних напрямах.

Процедура перетворення квадратичної функції

$$q(x) = a + b^T x + 1/2 x^T C x \quad (9.1)$$

до вигляду суми повних квадратів еквівалентна знаходженню такої матриці перетворення T, що приводить матрицю квадратичної форми до діагонального вигляду. Таким чином, задана квадратична форма

$$Q(x) = x^T C x \quad (9.2)$$

шляхом перетворення

$$x = TZ \quad (9.3)$$

приводиться до вигляду

$$Q(x) = Z^T T^T C T_Z = Z^T D Z, \quad (9.4)$$

де D — діагональна матриця, тобто елементи D відмінні від нуля тільки при i=j.

Нехай t_j — j-й стовпець матриці T. Тоді перетворення дозволяє записати кожен вектор x у вигляді лінійної комбінації стовпців t_j:

$$x = TZ = t_1 z_1 + t_2 z_2 + \ldots + t_N z_N \quad (9.5).$$

Іншими словами, замість координат вектора x в стандартній координатній системі, визначеній безліччю векторів, використовуються координати вектора в новій координатній системі, яка задана векторами t_j. Крім того, система векторів t_j відповідає головним осям даної квадратичної форми, оскільки матриця квадратичної форми приводиться до діагонального вигляду.

За допомогою перетворення змінних квадратичної функції будується нова система координат, співпадаючих з головними осями квадратичної функції. Отже, одновимірний пошук точки оптимуму в просторі перетворених змінних z еквівалентний пошуку уздовж кожної з головних осей квадратичної функції. Оскільки напрям головних осей визначається векторами t_j, одновимірний пошук проводиться в напрямах, заданих цими векторами.

Проілюструємо вищевикладене прикладом.

Перетворення до вигляду суми квадрата.

Розглянемо функцію:

$f(x)=4x_1^2+3x_2^2-4x_1x_2+x_1$

і перетворення: $x_1=z_1+0.5*z_2$; $x_2=z_2$

Перетворена квадратична функція набирає такого вигляду:

$f(z)=4z_1^2+2z_2^2+z_1+0.5z_2$

Це перетворення не є єдиним, зокрема перетворення

$$x = \begin{pmatrix} 1 & 0 \\ 2/3 & 1 \end{pmatrix} z$$

також приводить матрицю квадратичної форми до діагонального вигляду.

Задаючи початкову точку $x_0=[0,0]^T$ і два стовпці матриці перетворення $t_1=[1,0]^T$, $t_2=[0.5,1]^T$ можна знайти точку оптимуму в результаті проведення двох послідовних пошуків в напрямах t_1, t_2. Пошук у напряму t_1 по формулі:

$x_{(1)}=x(0)+\lambda*t_1=[0,0]^T+\lambda*[1,0]^T$

дозволяє отримати $\lambda=1/8$ і точку $x_{(1)}=[-1/8, 0]^T$.

Далі $x_{(2)} = x_{(1)}+\lambda*t_2= [-1/8, 0]^T+\lambda*[0.5,1]^T=[-1/8, 0]^T-1/8*[0.5,1]^T=$

$=[-3/16, -1/8]^T$.

З розглянутого прикладу виходить, що якщо система векторів t_j, $j=1,\ldots,N$ або система зв'язаних напрямів побудована, то точку оптимуму квадратичної функції можна знайти в результаті реалізації N одновимірних пошуків, які проводяться уздовж кожного з N напрямів t_j. Таким чином, невирішеними залишаються лише питання, пов'язані з побудовою системи векторів t_j. Якщо матриця C відома, то матрицю перетворення T можна знайти за допомогою методу Гауса-Жордана.

Метод Гауса-Жордана дозволяє представити матрицю C у вигляді перетворення

$$C = P^T DP \quad (9.6),$$

$$(P^{-1})^T C(P^{-1}) = D, T = P^{-1} \quad (9.7).$$

Проте матриця C або її оцінка в даному випадку невідома, оскільки мова йде про побудову методу вирішення завдань безумовної оптимізації, при реалізації якого використовуються тільки значення функції і не використовуються значення перших і тим більше других похідних.

Проте, в цьому випадку можна побудувати систему зв'язаних напрямів на основі властивості квадратичних функцій.

Приклад мінімізації на основі властивості паралельного підпростору.

Знову розглянемо квадратичну функцію

$q(x)=4x_1^2+3x_2^2-4x_1x_2+x_1$

Нехай задано дві точки $x_{(1)}=[0,0]^T$, $x_{(2)}=[1,0]^T$ і напрям $d=[1,1]^T$.

Перший пошук проводиться вздовж прямої $x=[0,0]^T + \lambda[1,1]^T$ і приводить до точки:

$y_{(1)} = [-1/6 , -1/6]^T$,

($\lambda^* = -1/6$).

Другий пошук проводиться вздовж прямої $x=[1,0]^T + \lambda[1,1]^T$ і дозволяє отримати точку: $\lambda^*=-5/6$, відповідно

$y_{(2)}=[1,0]^T – 5/6^* [1,1]^T = [1,0]^T + [-5/6,-5/6]^T = [1/6,-5/6]^T$

$y_{(2)}=[1/6,-5/6]^T$

Згідно властивості паралельного підпростору напрям

$y_{(2)} - y_{(1)} = [1/6,-5/6] - [-1/6,-1/6]^T = [1/3,-2/3]^T$ пов'язаний з d=[1,1],

тобто за визначенням $[1,1]^T$ C $[1/3,-2/3]^T$ =0.

Вище розглядалося, що в разі двох змінних, оптимум q(x) можна знайти шляхом проведення пошуку вздовж прямої заданого напряму ($y_{(2)}$-$y_{(1)}$). Цей факт не важко перевірити, оскільки мінімум q(x) вздовж прямої

$x= [-1/6,-1/6]^T + \lambda[1/3,-2/3]^T$

досягається в точці $x^* = [-3/16,-1/8]^T$, ($\lambda= -1/16$), яка збігається з раніше отриманим рішенням.

9.2 Контрольні запитання

9.2.1 До якого виду приводиться квадратична функція?

9.2.2 Як проводиться пошук оптимуму в просторі перетворених змінних?

9.2.3 Як знаходиться матриця перетворення?

10 ГРАДІЄНТНІ МЕТОДИ БАГАТОВИМІРНОЇ ОПТИМІЗАЦІЇ. МЕТОД КОШІ

Мета роботи - вивчити градієнтні методи безумовної багатовимірної оптимізації.

10.1 Короткі теоретичні відомості

Важливість прямих методів велика, бо в ряді практичних задач інформація про значення цільової функції є єдиною надійною інформацією, якою володіє дослідник.

З іншого боку, при використанні навіть самих ефективних прямих методів для отримання рішення іноді необхідна надзвичайно велика кількість обчислень значень функції. Ця обставина разом з прагненням реалізувати можливості знаходження стаціонарних точок (точок, що задовольняють умові $f(x^*)=0$) призводить до необхідності розгляду методів, які основані на використанні градієнта цільової функції. Вказані методи носять ітеративний характер, тому що компоненти градієнта виявляються нелінійними функціями керуючих змінних. Далі вважається, що $f(x)$, $\nabla f(x)$, $\nabla^2 f(x)$ існують та безперервні. Методи з використанням як 1-х, так і 2-х похідних проглядаються в їх зв‘язку з більш корисними методами.

Особливе значення приділяється застосуванню методів *спряжених градієнтів*, в основі яких лежать введені вище поняття спряженості напрямків і так званих квазіньютонівських методів, які аналогічні методу Ньютона, але використовують інформацію тільки перших похідних. Припускається, що компоненти градієнта можуть бути записані в аналітичному вигляді або з достатньо високою точністю розрахунків за допомогою чисельних методів. Крім того, розглядаються способи чисельної апроксимації градієнтів. Всі методи, що описані, основані на ітераційній процедурі, яка реалізується у відповідності з формулою

$$X^{(k+1)} = X^{(k)} + \alpha^{(k)} * S^{(k)} \qquad (10.1),$$

де $X^{(k)}$ - поточне ближнє до рішення X^*,

$\alpha^{(k)}$ - параметр, який характеризує довжину кроку.

$S^{(k)} = S^{'}(X^{(k)})$ - напрямок пошуку в N – мірному просторі керуючих змінних $X_i (i = \overline{1, N})$.

Спосіб визначення $S^{(k)}$ та α на кожній ітерації пов‘язаний з особливостями методу, що застосовується. Звичайно, вибір $\alpha^{(k)}$ здійснюється шляхом рішення задачі мінімізації $f(x)$ в напрямку $S(X^{(k)})$. Тому при реалізації методів, що вивчаються, необхідно використовувати ефективні алгоритми одновимірної мінімізації.

Метод Коші

Припустимо, що в деякій точці $\bar{x}$ простору керуючих змінних необхідно визначити напрямок найшвидшого локального спуску, тобто найбільшого локального зменшення цільової функції. Розложимо цільову функцію в окружності точки $\bar{x}$ в ряд Тейлора,

$$f(x) = f(\bar{x}) + \nabla f(\bar{x})^T \Delta x + ... \qquad (10.2)$$

та відкинемо члени другого порядку та вище. Можна побачити, що локальне зменшення цільової функції визначається 2-м доданком, оскільки значення $f(\bar{x})$ фіксовано. Найбільше зменшення f асоціюється з вибором такого напрямку $S(\bar{x})$ виразу (10.1), якому відповідає найбільша від’ємна величина скалярного множення, який є другим доданком розкладення (1.2). Вказаний вибір забезпечується при

$$S(\bar{x}) = -\nabla f(\bar{x}) , \qquad (10.3)$$

а другий доданок розкладення (10.2) приймає вигляд: $\Delta x = \alpha S(\bar{x}) = -\alpha \nabla f(\bar{x})$; $\nabla f(\bar{x})^T \Delta x = -\alpha \nabla f(\bar{x})^T \nabla f(\bar{x})$. Розглянутий випадок відповідає найшвидшому локальному спуску. Тому в основі найпростішого градієнтного методу лежить формула

$$x^{(k+1)} = x^{(k)} - \alpha \nabla f(x^{(k)}) \qquad (10.4),$$

де α - заданий додатній параметр.

Метод має два недоліки:

- виникає необхідність вибору значення α;

- методу властива повільна збіжність до точки мінімуму внаслідок малості ∇f навколо цієї точки.

Таким чином, правильно буде визначити значення α на кожній ітерації.

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)}) \qquad (10.5).$$

Значення $\alpha^{(k)}$ обчислюється шляхом розв'язання задачі мінімізації $f(x^{(k+1)})$ в напрямку $\nabla f(x^{(k)})\alpha$ методом одновимірного пошуку.

Даний градієнтний метод пошуку носить назву методу найшвидшого спуску або методу *Коші*, оскільки Коші першим використав аналогічний алгоритм для рішення систем лінійних рівнянь. Пошук вздовж прямої у відповідності з формулою (10.5) забезпечує більш високу надійність методу Коші в порівнянні з найпростішим градієнтним методом (коли α не змінюється на кожній ітерації), однак швидкість його збіжності при розв'язку ряду практичних задач залишається недопустимо низькою. Це пояснюється тим, що змінення змінних безпосередньо залежить від величини градієнта, який прямує до 0 в околі точки мінімуму та відсутній механізм прискорення руху до точки мінімуму на останніх ітераціях. Одна з головних переваг методу Коші пов'язана з його стійкістю. Метод має важливу властивість, яка полягає в тому, що при достатньо малій довжині кроку ітерації забезпечують виконання нерівності:

$$f(x^{(k+1)}) \leq f(x^{(k)}) \qquad (10.6).$$

Метод Коші, як правило, дозволяє достатньо зменшити значення цільової функції прямуванням з точок, що розміщені на значних відстанях від точки мінімуму, і тому часто використовується при реалізації градієнтних методів в якості початкової процедури. На прикладі методу Коші можна продемонструвати окремі заходи, які використовуються при реалізації різних градієнтних алгоритмів.

Не зважаючи на те, що метод Коші має відносно невелике практичне застосування, він реалізує найважливіші кроки більшості градієнтних методів.

Робота алгоритму завершується, коли модуль градієнту або модуль вектора Δx стають достатньо малим.

Алгоритм роботи методу Коші

Крок 1. Ввести $M, N, X^{(0)}, \varepsilon_1, \varepsilon_2$.

Крок 2. Задати $K = 0$.

Крок 3. Обчислити градієнт $\nabla f(x^{(k)})$.

Крок 4. $\left\|\nabla f(x^{(k)})\right\| \leq \varepsilon_1$. Якщо *так* – друк результатів, кінець. Якщо *ні* – перехід на наступний крок.

Крок 5. $k > M$. Якщо *так* – друк результатів, кінець. Якщо *ні* – перехід на наступний крок.

Крок 6. Обчислити $f^{(k)}$ на основі пошуку вздовж прямої, використовуючи ε_2.

Крок 7. Покласти $x^{(k+1)} = x^{(k)} - a^{(k)}\nabla f(x^{(k)})$.

Крок 8. $\frac{\left\|x^{(k+1)} - x^{(k)}\right\|}{\left\|x^{(k)}\right\|} \leq \varepsilon_1$. Якщо *так* – друк результатів, кінець. Якщо *ні* – $K = K + 1$ та перехід на 4 крок.

Приклад використання методу Коші.

Розглянемо функцію $f(x) = 8x_1^2 + 4x_1x_2 + 5x_2^2$, використовуючи метод Коші для розв'язку задачі її мінімізації.

Рішення: Обчислимо компоненти градієнту:

$$\frac{\partial f}{\partial x_1} = 16x_1 + 4x_2 ; \frac{\partial f}{\partial x_2} = 10x_2 + 4x_1 .$$

Для того, щоб застосувати метод найшвидшого спуску, задамо початкове наближення $x^{[0]} = [10;10]^T$ та за допомогою формули (10.5) побудуємо нове наближення.

$x^{(1)} = x^{(0)} - \alpha^{(0)} * \nabla f(x^{(0)})$. Оберемо $\alpha^{(0)}$ таким чином, щоб $f(x^{(1)}) \to \min$. $\alpha^{(0)} = 0.056$.

$$\nabla f(x^{(0)}) = [16x_1 + 4x_2;10x_2 + 4x_1] = [200;140]$$

$$\alpha^{(0)}\nabla f(x^{(0)}) = 0.056 * [200;140] = [11.2;7.84]$$

$$x^{(1)} = [10;10] - [11.2;7.84] = [-1.20;2.16]$$

Далі знаходимо точку $x^{(2)} = x^{(1)} - \alpha^{(1)} * \nabla f(x^{(1)})$, обчислюємо градієнт в точці $x^{(1)}$ та проводимо пошук вздовж прямої

$\nabla f(x^{(1)}) = [16x_1 + 4x_2;10x_2 + 4x_1] = [-19.2 + 8.64;21.6 - 4.8] =$
$[-10.56;16.8]$.

В таблиці 10.1 надані дані, отримані при проведені ітерацій на основі одномірного пошуку методом кубічної апроксимації.

Таблиця 10.1 – Результати обчислень по методу Коші.

К	$X_1^{(k)}$	$X_2^{(k)}$	$f(x^{(k)})$
1	-1,200	2,1600	24,4800
2	0,1441	0,1447	0,3540
3	-0,0181	0,0309	0,0052
4	0,0021	0,0021	0,0000

10.2 Контрольні запитання

10.2.1 Які недоліки має метод Коші?
10.2.2 Які переваги методу Коші?
10.2.3 Алгоритм методу Коші.

11ГРАДІЄНТНІ МЕТОДИ БАГАТОВИМІРНОЇ ОПТИМІЗАЦІЇ. МЕТОД НЬЮТОНА

Мета роботи - вивчити градієнтні методи безумовної багатовимірної оптимізації.

11.1 Короткі теоретичні відомості

Метод Ньютона

У методі Коші застосовується найкраща локальна стратегія пошуку з використанням градієнту. Однак, рух у напрямку, протилежному градієнту, приводить до точки мінімуму лише в тому випадку, коли лінії рівня функції *f* являють собою кола. Таким чином, напрямок, протилежний градієнту, загалом кажучи, не може бути придатним глобальним напрямком пошуку точок оптимуму нелінійних функцій. Метод Коші засновується на послідовній лінійній апроксимації цільової функції та потребує обчислень значень функції та її перших похідних на кожній ітерації. Для того, щоб побудувати більш загальну стратегію пошуку слід підключити інформацію про другі похідні цільової функції. Розкладемо цільову функцію в ряд Тейлора.

$$f(x) = f(x^{(k)}) + \nabla f(x^{(k)})^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(x^{(k)})\Delta x + 0(x^3) \quad (11.1)$$

Відкидаючи всі члени розкладу третього порядку й вище, отримаємо квадратичну апроксимацію *f(x)*.

$$\tilde{f}(x; x^{(k)}) = f(x^{(k)}) + \nabla f(x^{(k)})^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(x^{(k)})\Delta x, \quad (11.2)$$

де $\tilde{f}(x; x^{(k)})$ - апроксимуюча функція змінної *x*, побудована в точці $x^{(k)}$.

На основі квадратичної апроксимації функції $f(x)$ сформулюємо послідовність ітерацій таким чином, щоб у знов здобутій

точці $x^{(k+1)}$ градієнт апроксимуючої функції, тобто $\nabla\tilde{f}$ обертався на 0.

$$\nabla\tilde{f}(x;x^{(k)}) = \nabla f(x^{(k)}) + \nabla^2 f(x^{(k)})\Delta x = 0. \qquad (11.3)$$

Одержуємо:

$$\Delta x = -\nabla^2 f(x^{(k)})^{-1} \cdot \nabla f(x^{(k)}). \qquad (11.4)$$

Послідовне використання схеми квадратичної апроксимації приводить до реалізації оптимізаційного методу Ньютона за формулою:

$$x^{(k+1)} = x^{(k)} - \nabla^2 f(x^{(k)})^{-1} \cdot \nabla f(x^{(k)}). \qquad (11.5)$$

Властивості збіжності.

Наведемо аналіз властивостей збіжності методу Ньютона.

Доведено, що метод Ньютона виявляє квадратичну швидкість збіжності, тобто виконується умова

$$\left\|\varepsilon^{(k+1)}\right\| \le c\left\|\varepsilon^{(k)}\right\|^2, \qquad (11.6)$$

де константа *c* зв'язана з обумовленістю матриці Гессе $\nabla^2 f$.

Метод Ньютона збігається всякий раз, коли вибір $x^{(0)}$ здійснюється згідно умові

$$\left\|\varepsilon x^0\right\| < \frac{1}{c}. \qquad (11.7)$$

В цьому випадку виконується нерівність (11.6).

Квадратична швидкість збіжності пояснюється тією обставиною, що метод заснований на квадратичній апроксимації.

При мінімізації довільних функцій доцільно вважати, що у випадку вибору початкової точки, яка задовольняє нерівності $\left\|\varepsilon x^0\right\| > \frac{1}{c}$, використання методу не приводить до розв'язку.

Модифікований метод Ньютона

Досвід показує, що при дослідженні неквадратичних функцій метод Ньютона не відрізняється високою надійністю.

Дійсно, якщо точка $x^{(0)}$ находиться на значній відстані від точки x^*, крок за методом Ньютона часто виявляється досить великим, що може привести до відсутності збіжності. Метод можна досить легко модифікувати для того, щоб забезпечити зменшення цільової функції від ітерації до ітерації та здійснити пошук уздовж прямої як в методі Коші. Послідовність ітерацій будується згідно з формулою:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \cdot \nabla^2 f\left(x^{(k)}\right)^{-1} \cdot \nabla f\left(x^{(k)}\right). \qquad (11.8)$$

Вибір $\alpha^{(k)}$ здійснюється таким чином, щоб $f\left(x^{(k+1)}\right) \to \min$. Це гарантує виконання нерівності $f\left(x^{(k+1)}\right) \le f\left(x^{(k)}\right)$. Такий метод носить назву модифікований метод Ньютона. У випадках, коли розрахунок точних значень перших та других похідних не зв'язаний з суттєвими труднощами, він є надійним та ефективним. Однак, при використанні модифікованого методу Ньютона виникає необхідність будування та рішення лінійного рівняння, що містить елементи матриці Гессе.

11.2 Контрольні запитання

11.2.1 Відмінності методу Ньютона від методу Коші.
11.2.2 Наведіть аналіз властивостей збіжності методу Ньютона.
11.2.3 В чому полягає модифікація методу Ньютона?

12 ГРАДІЄНТНІ МЕТОДИ БАГАТОВИМІРНОЇ ОПТИМІЗАЦІЇ. МЕТОД МАРКВАРДТА

Мета роботи - вивчити градієнтні методи безумовної багатовимірної оптимізації.

12.1 Короткі теоретичні відомості

Метод Марквардта

Даний метод є комбінацією методів Коші та Ньютона, в якому добре сполучаються позитивні властивості обох методів. При використанні методу Марквардта необхідна інформація про значення других похідних цільової функції. Градієнт вказує напрямок найбільш локального збільшення цільової функції, а рух у напрямку, протилежному градієнту з точки $x^{(0)}$, розташованої на значній відстані від точки мінімуму x^*, зазвичай приводить до суттєвого зменшення цільової функції. З іншого боку, напрямки ефективного пошуку в околі точки мінімуму, визначаються методом Ньютона. Ідея об'єднання методів Коші та Ньютона була покладена в основу алгоритму, розробленого Марквардтом. Згідно цього методу напрямок пошуку визначається рівнянням:

$$S^{(k)} = -\left[H^{(k)} + \alpha^{(k)} \cdot I\right]^{-1} \cdot \nabla f(x)^{(k)}. \qquad (12.1)$$

При цьому в формулі (10.1) покласти $\alpha^{(k)} = 1$, так як параметр λ дозволяє не тільки змінювати напрямок пошуку, але й регулювати довжину кроку.

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot S\left(x^{(k)}\right),$$

де $x^{(k)}$ – поточне наближення до рішення x^*;

$\alpha^{(k)}$ – параметр, що характеризує довжину кроку;

$S\left(x^{(k)}\right)$ – це напрямок пошуку в n-мірному просторі управляючих змінних;

I – це одинична матриця, тобто матриця, всі елементи якої дорівнюють 0, за винятком діагональних елементів, що дорівнюють 1.

На початковій стадії пошуку $\lambda^{(0)}$ присвоюється велике значення, наприклад $\lambda^{(0)} = 10^4$. В цьому випадку

$$\left[H^{(0)} + \lambda^{(0)} \cdot I\right]^{-1} = \left[\lambda^{(0)} \cdot I\right]^{-1} = \frac{1}{\lambda^{(0)}} \cdot I . \qquad (12.2)$$

При великому значенні $\lambda^{(0)}$ ($\frac{1}{\lambda^{(0)}}$ мале) напрям пошуку:

$$S^{(0)} = -\nabla f(x^{(0)}) . \qquad (12.3)$$

З виразу (2.8) можна заключити, що при зменшенні λ до 0 $S(x)$ змінюється від напрямку, протилежному градієнту до напрямку, визначеному за методом Ньютона. Якщо після першого кроку отримана точка з меншим значенням цільової функції ($f(x^{(1)} < f(x^{(0)})$, слід обрати $\lambda^{(1)} < \lambda^{(0)}$ та реалізувати ще один крок. Інакше слід покласти $\lambda^{(0)} = \beta\lambda^{(0)}$, де $\beta > 0$, та знову реалізувати попередній крок.

Алгоритм Марквардта

Крок 1. Задати $x^{(0)}$ початкове наближення до x^*, М – максимально припустима кількість ітерацій, *E* – параметр збіжності.

Крок 2. Покласти К = 0, $\lambda^{(0)} = 10^4$.

Крок 3. Обчислити $\nabla f(x^{(\kappa)})$

Крок 4. Перевірити чи виконується нерівність $\left\|\nabla f(x^{(\kappa)})\right\| < E$.

Якщо так – перейти до кроку 11, ні – перейти до наступного кроку.

Крок 5. Перевірити чи виконується нерівність К ≥ М.

Так - перейти до кроку 11, ні - перейти до наступного кроку.

Крок 6. Обчислити $S(x^{(к)})$ за формулою (12.1).

Крок 7. Покласти $x^{(к+1)} = x^{(к)} + S(x^{(к)})$.

Крок 8. Чи виконується нерівність $f(x^{(к+1)}) < f(x^{(к)})$.

Так - перейти до кроку 9, ні - перейти до кроку 10.

Крок 9. Покласти $\lambda^{(к+1)}=1/2\,\lambda^{(к)}$, К=К+1. Перейти до кроку 3.

Крок 10. Покласти $\lambda^{(К+1)}= 2\,\lambda^{(к)}$. Перейти до кроку 6.

Крок 11. Друк результатів і зупинка.

Метод Марквардта характеризується відносною простотою, властивістю спадання цільової функції при переході від ітерації до ітерації, високою швидкістю збіжності в околі точки мінімуму x^*, а також відсутністю процедури пошуку уздовж прямої. Головний недолік методу полягає в необхідності обчислення матриці Гессе й наступного рішення системи лінійних рівнянь, що відповідають рівнянню (1.16). Цей метод широко використається при рішенні задач, у яких f(x) записується у вигляді суми повних квадратів. Задача такого типу виникає, наприклад, у регресійному аналізі:

$$f(x) = f_1^2(x) + f_2^2(x) + \cdots + \cdots + f_m^2(x) \qquad (12.4)$$

Метод Марквардта відрізняється високою ефективністю при рішенні задач такого типу.

12.2 Контрольні запитання

12.2.1 Відмінності методу Марквардта від методів Коші та Ньютона.

12.2.2 Чим характеризується метод Марквардта?

12.2.3 Алгоритм методу Марквардта.

13 МЕТОДИ СПРЯЖЕНИХ ГРАДІЄНТІВ. МЕТОД ФЛЕТЧЕРА – РІВСА

13.1 Короткі теоретичні відомості

Метод Флетчера – Рівса

Метод дозволяє знайти мінімум нелінійної цільової функції багатьох змінних вигляду

$$M = F(x_1, x_2, \dots, x_N)$$

при відсутності обмежень. Метод заснований на застосуванні часткових похідних цільової функції по незалежним змінним і перевизначений для дослідження унімодальних функцій. За його допомогою можна досліджувати і мультимодальні функції, однак в цьому випадку слід брати декілька вхідних точок і перевіряти, чи в усіх випадках однакове рішення.

Алгоритм Флетчера – Рівса

Крок 1. Вибір початкової точки.

Крок 2. Визначення напрямку градієнту у точці.

Крок 3. Одномірний пошук в напрямку градієнту з метою визначення «найкращої» точки.

Крок 4. Перевірити чи відповідає знайдена «найкраща» точка рішенню задачі.

Якщо *так*, то кінець алгоритму. Якщо *ні*, перехід на наступний крок.

Крок 5. Перевірити чи виконано пошук в усіх N+1 напрямках.

Якщо *так*, перейти до кроку 2. Якщо *ні*, перейти на наступний крок.

Крок 6. Визначення спряженого градієнту в новій кращій точці.

Перехід до 3 кроку.

13.4 Контрольні запитання

13.4.1 На чому заснований метод Флетчера – Рівса?
13.4.2 Які функції можна досліджувати методом Флетчера - Рівса?
13.4.3 Алгоритм методу Флетчера – Рівса.

14 КВАЗІНЬЮТОНІВСЬКІ МЕТОДИ. МЕТОД ДЕВІДОНА-ФЛЕТЧЕРА-ПАУЕЛЛА

14.1 Короткі теоретичні відомості

Квазіньютонівські методи подібні до методів спряжених градієнтів, оскільки також засновані на властивостях квадратичної функції. Відповідно до методів спряжених градієнтів пошук рішень здійснюється по системі спряжених напрямків, тоді як квазіньютонівські методи мають позитивні риси методу Ньютона, однак використовують тільки перші похідні.

Побудова векторів напрямків пошуку здійснюється за допомогою формули:

$$x_{k+1} = x_k + \lambda^{(k)} s(x^{(k)}) \qquad (*)$$

де x_k – поточне наближення до x^*, $\lambda^{(k)}$ – параметр що характеризує довжину кроку, $s(x^{(k)})$ – напрямок пошуку в *n*-мірному просторі. $s(x^{(k)})$ записується у вигляді

$$s(x^{(k)}) = -A^{(k)} \nabla f(x^{(k)}), \qquad (14.1)$$

де $A^{(k)}$ - матриця метрики, порядку $n \times n$.

Методи пошуку уздовж напрямків, визначених цією формулою, мають назву змінної метрики, оскільки матриця *A* змінюється на кожній ітерації.

Метод змінної метрики являє собою квазіньютонівський метод, якщо відповідно до нього переміщення пробної точки задовольняє наступній умові:

$$\Delta x = C^{-1} \Delta g, \qquad (14.2)$$

де Δx - переміщення *x*.

У методі Ньютона $\Delta x = -\nabla^2 f(x^{(k)})^{-1} \nabla f(x^{(k)})$ – це аналог C^{-1} у квазіньютонівському методі.

$\Delta g = g(x_{(1)} - x_{(0)})$, Δg - зміна градієнта при переході від $x_{(0)}$ до $x_{(1)}$.

Градієнт квадратичної функції: $\nabla f(x) = Cx + b = g(x)$

$$g(x_{(0)}) = Cx^{(0)} + b,\ g(x^{(1)}) = Cx^{(1)} + b,\ \Delta x = C^{-1}\Delta g \quad (14.3)$$

$\nabla^2 f(\bar{x})$ - матриця Гессе, симетрична матриця порядку $n \times n$ других похідних $f(x)$, що обчислюється у точці $\bar{x}$. Елемент матриці Гессе, розташований на перетині *i*-го рядка *j*-го стовпця, дорівнює $\partial^2 x / \partial x_i \partial x_j$

Для апроксимації матриці, зворотної до матриці Гессе, скористаємося наступною рекурентною формулою:

$$A^{k+1} = A^{(k)} + A^{k+1} c, \quad (14.4)$$

$A^k c$ – коригувальна матриця.

Матриця $A^{(k)}$ буде використовуватися в (14.1) і (*). Задача полягає в тому, щоб побудувати матрицю $A^{(k)}$ в такий спосіб щоб послідовність $A^{(0)}, A^{(1)}, ..., A^{(k+1)}$ давала наближення до $H^{-1} = \nabla^2 f(x^*)^{-1}$, при цьому для одержання рішення x^* потрібний один додатковий пошук уздовж прямої, якщо $f(x)$ - квадратична функція.

Припустимо, що матриця C^{-1} апроксимується по формулі $\beta A^{(k)}$, де β - скалярна величина. Кращім є наближення, що задовольняє (14.2), тобто

$$\Delta x^{(k)} = \beta A^{(k)} \Delta g^{(k)} \quad (14.5)$$

Однак побудувати таку апроксимацію неможливо, оскільки для того щоб знайти $\Delta g^{(k)}$, необхідно знати матрицю $A^{(k)}$.

Будемо використовувати наступні позначення:

$$\Delta g^{(k)} = g^{(k+1)} - g^{(k)} \tag{14.6}$$

$$\Delta x^{(k)} = x^{(k+1)} - x^{(k)} \tag{14.7}$$

Можлива вимога, щоб нове наближення задовольняло формулі (14.2)

$$\Delta x^{(k)} = \beta A^{(k+1)} g^{(k)} \tag{14.8}$$

Після підстановки (14.3) в (14.8) одержимо

$$\Delta x^{(k)} = \beta A^{(k+1)} * g^{(k)} = \beta (A^{(k)} + A^{(k)}{}_c)\Delta g^{(k)} = \beta A^{(k)} \Delta g^{(k)} + \beta A^{(k)}{}_c \Delta g^{(k)}$$

$$A^{(k)}{}_c \Delta g^{(k)} = \frac{1}{\beta}\Delta x^{(k)} - A^{(k)} \Delta g^{(k)} \tag{14.9}$$

Можна переконатися, що матриця

$$A^{(k)}{}_c = \frac{1}{\beta}\left(\frac{\Delta x^{(k)} y^T}{y^T \Delta g^{(k)}}\right) - \frac{\left(A^{(k)} \Delta g^{(k)} z^T\right)}{z^T \Delta g^{(k)}},$$

тут всі значення крім β – вектори, є рішеннями рівняння, *y,z* – довільні вектори, тобто наступна формула визначає деякий рід рішень.

Якщо покласти

$$y = \Delta x^{(k)} \text{ і } z = A^{(k)} \Delta g^{(k)}, \tag{14.10}$$

то отримаємо формулу, що реалізує метод Девідона-Флетчера-Пауелла :

$$A_c^{(k)} = \frac{1}{\beta}\left(\frac{\Delta x^{(k-1)}\Delta x^{(k)^T}}{\Delta x^{(k)^T}\Delta g^{(k)}}\right) - \frac{A^{(k)}\Delta g^{(k)}\Delta g^{(k)^T}A^{(k)}}{\Delta g^{(k)^T}A^{(k)}\Delta g^{(k)}} \quad (14.11`)$$

$$A_{c-1}^{(k)} = \frac{1}{\beta}\left(\frac{\Delta x^{(k-1)}\Delta x^{(k-1)^T}}{\Delta x^{(k-1)^T}\Delta g^{(k-1)}}\right) - \frac{A^{(k-1)}\Delta g^{(k-1)}\Delta g^{(k-1)^T}A^{(k-1)}}{\Delta g^{(k-1)^T}A^{(k-1)}\Delta g^{(k-1)}}$$

$$A^{(k+1)} = A^{(k)} + A_c^{(k)}$$

$$A^{(k)} = A^{(k-1)} + A_c^{(k-1)}$$

$$A^{(k)} = A^{(k-1)} + \left(\frac{\Delta x^{(k-1)}\Delta x^{(k-1)^T}}{\Delta x^{(k-1)^T}\Delta g^{(k-1)}}\right) - \frac{A^{(k-1)}\Delta g^{(k-1)}\Delta g^{(k-1)^T}A^{(k-1)}}{\Delta g^{(k-1)^T}A^{(k-1)}\Delta g^{(k-1)}} \quad (14.12)$$

Ця рекурентна формула має властивість симетрії й позитивної визначеності матриці.

Перша варіація (приріст, зміна):

$$\Delta f(x) = \nabla f(x^{(k)})^T \Delta x \quad (14.13)$$

Використовуючи формули (*) і (5.1) одержуємо

$$x^{k+1} = x^k + \lambda^{(k)} s(x^{(k)}) \text{ та } s(x^{(k)}) = -A^{(k)}\nabla f(x^{(k)}).$$

$\Delta x = x^{(k+1)} - x^{(k)} = \lambda^{(k)} s(x^{(k)})$, звідси випливає що

$$\Delta f(x) = \nabla f(x^{(k)})^T \lambda^{(k)} s(x^{(k)}) = -\nabla f(x^{(k)})^T \lambda^{(k)} A^{(k)} \nabla f(x^{(k)}), \quad (14.14)$$

звідки

$$\Delta f(x) = -\lambda^{(k)} \nabla f(x^{(k)})^T A^{(k)} \nabla f(x^{(k)}) \quad (14.15)$$

З (14.15) випливає, що $f(x^{(k+1)}) < f(x^{(k)})$, тобто $\Delta f(x) < 0$ виконується при будь-яких значеннях $\lambda^{(k)} > 0$, якщо $A^{(k)}$ - позитивно визначена матриця. У такий спосіб алгоритм забезпечує подвоєння цільової функції від ітерації до ітерації.

Метод Девідона-Флетчера-Пауелла є широко застосовуваним градієнтним методом. Він відрізняється стабільністю й успішно застосовується при вирішені різних задач, що виникають на практиці. Основним недоліком методів такого типу є необхідність зберігати в пам'яті матрицю порядку $n \times n$.

Алгоритм Девідона-Флетчера-Пауелла

Крок 1. Ввести вхідні дані $X^{(0)}, \varepsilon > 0, f(X), n.$

Крок 2. Почати з точки x_i

$$A_i = I(i = 0)$$

Крок 3. Покласти $d_i - A * g_i.$

Крок 4. Знайти значення λ_i, що мінімізує функцію $f(x_i + \lambda d_i)$.

Крок 5. Покласти $x_{i+1} = x_i + \lambda_i d_i = x_i + v_i.$

Крок 6. Знайти g_{i+1}.

Крок 7. Використати A_i, y_i і $u_i = g_{i+1} - g_i$ для формування A_{i+1}.

Крок 8. Перевірити умову $|v_i| < \varepsilon$ або $|g_{i+1}| < \delta$.

Якщо *так*, перейти на наступний крок. Якщо *ні*, покласти $i = i + 1,$ перейти на крок 3.

Крок 9. Вивести результати оптимізації.

14.2 Контрольні запитання

14.2.1 В чому полягає суть методу Девідона-Флетчера-Пауелла?

14.2.2 Які критерії завершення пошуку використовуються в роботі?

14.2.3 Поясніть, які методи називають методами змінної метрики?

15 ЗАСТОСУВАННЯ ЗАСОБІВ МОВИ ПРОГРАМУВАННЯ PYTHON ДЛЯРОЗВ'ЯЗКУ ЗАДАЧ ОПТИМІЗАЦІЇ

15.1 Можливості пакету SciPy

Python – сучасна потужна високорівнева кросплатформна мова програмування, яка стає все більш популярною завдяки прозорому і логічному синтаксису та наявності кросплатформних високоефективних пакетів розширення. Одним з них є пакет SciPy. Це пакет з відкритим вихідним кодом, призначений для вирішення наукових та математичних задач. Він побудований на базі NumPy (пакет для збереження даних масивів та операції з ними) та дозволяє керувати даними, а також візуалізувати їх за допомогою різних високорівневих команд.

SciPy використовують фахівці з Data Science, Big Data, аналітики даних, а також математики та вчені:

– для складних математичних розрахунків, які важко здійснити вручну або за допомогою калькулятора;

– проведення наукових досліджень, де потрібне використання поглибленої математики;

– глибокого аналізу даних, інтерполяції та інших методів роботи з інформацією;

– машинного навчання та створення моделей штучного інтелекту, прогнозування та побудови моделей;

– формування двовимірних та тривимірних графіків, які можна потім візуалізувати (за допомогою інших бібліотек).

Можливості бібліотеки розподілені за кількома модулями або пакетами, які об'єднані призначенням. Це потрібно, щоб фахівець міг підключити замість повної бібліотеки потрібну частину для економії ресурсів. Так код буде ефективнішим, а його написання – зручнішим.

Основні модулі SciPy:

– кластерний модуль (scipy.cluster). Кластеризація — популярний метод категоризації даних шляхом об'єднання їх у групи;

– модуль констант (scipy.constants). Модуль містить у собі безліч фізико-математичних констант та одиниць, дуже зручно, адже не треба вручну прописувати їх значення, а використовувати вже готові.

Константи використовуються при фізичних або математичних розрахунків;

– набори даних (scipy.datasets). Модуль використовує та залежить від Pooch , пакета Python, створеного для спрощення отримання файлів даних. Pooch використовує ці сховища для отримання відповідних файлів набору даних під час виклику функції набору даних;

– оптимізація та пошук кореня (scipy.optimize). Модуль надає функції для мінімізації (або максимізації) цільових функцій, можливо з обмеженнями. Він містить розв'язування нелінійних задач (з підтримкою як локальних, так і глобальних алгоритмів оптимізації), лінійне програмування, обмежені та нелінійні найменші квадрати, пошук кореня та підгонку кривої. Цей процес може бути корисним в різних областях, таких як: машинне навчання, оптимізація виробничих процесів, фінансовий аналіз.

Scipy.optimize містить ряд корисних методів для оптимізації різних типів функцій:

– minimize_scalar() і minimize() – мінімізація функції однієї змінної та багатьох змінних відповідно;

– curve_fit() – метод оптимізації, який знаходить параметри моделі, що найкращим чином апроксимує задані дані;

– root_scalar() і root() – пошук нулів функції однієї і багатьох змінних відповідно;

– linprog() – мінімізація лінійної цільової функції з лінійною системою обмежень.

15.2 Метод linprog()

Метод linprog() в бібліотеці SciPy є інструментом для розв'язування задач лінійного програмування (linear programming). Метод знаходить мінімальне значення лінійної функції лінійного програмування з лінійними обмеженнями на змінні:

A_ub * x <= b_ub
A_eq * x == b_eq
lb <= x <= ub.

Метод може бути застосований з такими параметрами:

scipy.optimize.linprog(c, A_ub=None, b_ub=None, A_eq=None, b_eq=None, bounds=None, method='highs', callback=None, options=None, x0=None, integrality=None):

c - одновимірний масив коефіцієнтів лінійної цільової функції, обов'язковий параметр;

A_ub - двовимірний масив, необов'язковий - матриця обмежень-нерівностей;

b_ub - одновимірний масив, необов'язковий - вектор обмежень-нерівностей. Кожен елемент є верхнею межею відповідного значення *A_ub * x*;

A_eq - двовимірний масив, необов'язковий - матриця обмежень-рівностей;

b_eq - одновимірний масив, необов'язковий - вектор обмежень-рівностей. Кожен елемент має дорівнювати відповідному елементу *A_eq * x*;

bounds, необов'язковий - послідовність пар для кожного елемента вектору x, що визначає мінімальне та максимальне значення цієї змінної рішення x_i. Якщо надається єдиний кортеж , то і буде служити межами для всіх змінних рішення. Використовуйте , щоб вказати, що обмеження немає. Пара меж за замовчуванням (0, None) означає, що всі змінні рішення є невід'ємними;

method, необов'язковий - алгоритм розв'язання задачі стандартної форми. Підтримуються 'highs', 'highs-ds' , 'highs-ipm';

callback, необов'язковий - функція зворотного виклику, якщо вона надається, буде викликана принаймні один раз за ітерацію алгоритму;

options, необов'язковий - словник опцій;

x0 - одновимірний масив, необов'язковий – початкові значення змінних рішення, які будуть уточнені алгоритмом оптимізації. Цей аргумент наразі використовується лише «переглянутим симплексним» методом і може бути використаний, лише якщо x0 представляє базове можливе рішення;

integrality - одновимірний масив або int, необов'язковий, вказує тип обмеження цілісності для кожної змінної рішення, використовується лише методом 'highs'і ігнорується в інших випадках.

Далі подано приклад роботи методу linprog(), що розв'язує таку задачу лінійного програмування:

$f=x_1+2*x_2 \rightarrow max;$
$x_1+ x_2 <= 6$
$x_1+10* x_2 <= 26$
$x_1+11* x_2 <= 20;$
$x_1, x_2 >=0.$

```
c=[-1,-2]
A=[[1,1],[1,10],[1,11]]
b=[6,26,20]
res=scipy.optimize.linprog(c,A,b)
print(res)
```

Результат роботи методу:

```
con: array([], dtype=float64)
    fun: -7.399999993266249
 message: 'Optimization terminated successfully.'
    nit: 5
  slack: array([6.83198920e-09, 7.40000001e+00, 5.84961057e-09])
 status: 0
success: True
      x: array([4.59999999, 1.4       ])
```

Результат роботи методу *res* - об'єкт типу OptimizeResult складається з наведених нижче полів (типи результатів полів можуть залежати від успішності оптимізації, тому рекомендується спершу перевірити OptimizeResult.status):

x – одновимірний масив - значення змінних рішення, які мінімізують цільову функцію при дотриманні обмежень;

fun - оптимальне значення цільової функції;

stack - одновимірний масив - значення змінних b_ub - A_ub * x;

con - одновимірний масив – значення обмежень-рівностей b_eq - A_eq * x;

success має значення *True*, коли алгоритму вдається знайти оптимальне рішення;

status - ціле число, що представляє статус виходу з алгоритму, 0, якщо оптимізацію успішно завершено;

nit - рагальна кількість ітерацій, виконаних на всіх етапах$

message - повідомлення статусу виходу з алгоритму.

15.3 Метод мінімізації функції однієї змінної minimize_scalar()

Математична функція, яка приймає одне число та дає один результат, називається скалярною функцією.

Мета даної функції пакету - знайти мінімальне значення заданої одновимірної функції. Для прикладу візьмемо функцію y = 3x⁴ - 2x + 1, яку зображено на рисунку 15.1, у діапазоні X від 0 до 1.

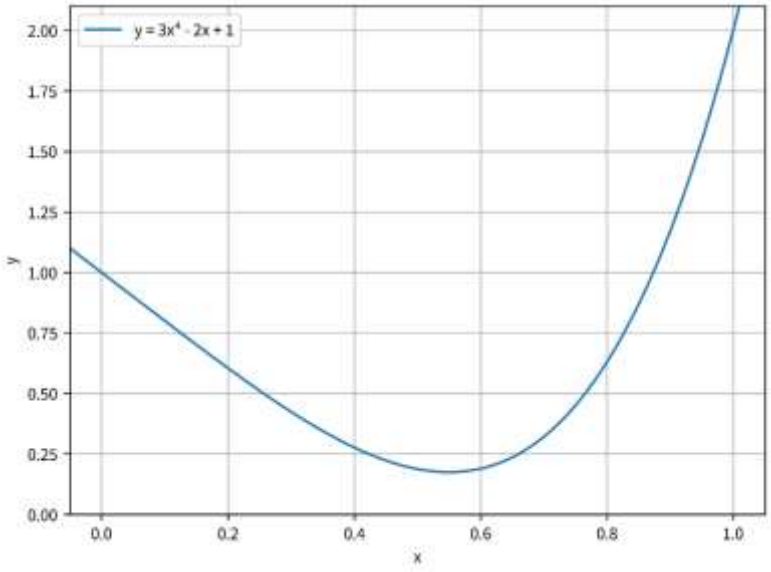


Рисунок 15.1 – Поліном четвертої степені

На рисунку можна побачити, що мінімальне значення цієї функції становить приблизно x = 0,55. Можна використовувати minimize_scalar() для визначення точних координат x і y мінімуму. Спочатку треба імпортувати minimize_scalar() з scipy.optimize. Потім потрібно визначити цільову функцію, яку потрібно мінімізувати:

```
from scipy.optimize import minimize_scalar

def objective_function(x):
    return 3 * x ** 4 - 2 * x + 1
```

objective_function() приймає вхідні дані x та застосовує до них необхідні математичні операції, а потім повертає результат. У визначенні функції можна використовувати будь-які математичні функції. Єдине обмеження полягає в тому, що функція повинна повертати одне число в кінці.

minimize_scalar() має лише один обов'язковий параметр, який є назвою визначення цільової функції:

```
res = minimize_scalar(objective_function)
```

Результатом minimize_scalar() є екземпляр OptimizeResult. Цей клас збирає разом багато відповідних деталей із запуску оптимізатора, включно з тим, чи була оптимізація успішною чи ні, і, якщо успішно, яким був кінцевий результат. Результат роботи minimize_scalar() для заданої функції:

```
fun: 0.17451818777634331
   nfev: 16
    nit: 12
success: True
      x: 0.5503212087491959
```

Усі ці результати є атрибутами OptimizeResult:

- *success* - логічне значення, яке вказує, чи успішно завершилася оптимізація;
- *x* – точка мінімуму;
- *fun* - це значення цільової функції в точці *x*;
- *nfev* – кількість обчислень досліджуваної функції;
- *nit* – кількість ітерацій.

З отриманого результату можна побачити, що, як і очікувалося, оптимальне значення для цієї функції було близько x = 0,55.

Якщо функція не має мінімуму (наприклад, x^3), то робота minimize_scalar() закінчиться помилкою OverflowError (занадто велике число) оскільки оптимізатор зрештою намагається отримати число, яке є занадто великим, щоб його обчислив комп'ютер.

З іншого боку, для функцій, що мають більше одного мінімуму, робота minimize_scalar() не гарантує знаходження глобального мінімуму функції.

Функція minimize_scalar() може використовувати не один метод знаходження мінімуму. Щоб вказати яким саме методом треба провести оптимізацію, необхідно при виклику функції вказати параметр method='назва методу':

```
def objective_function(x):
    return 3*x**4-2*x*x
res = minimize_scalar(objective_function, method='bounded', bounds=(-1, 0)),
```

де bounds=(-1, 0) – границі пошуку.

Графік досліджуваної функції на інтервалі [-1;1] представлений на рисунку 15.2.

Результат роботи методу:

```
fun: -0.3333333333330272
 message: 'Solution found.'
    nfev: 10
  status: 0
 success: True
       x: -0.5773499925310644
```

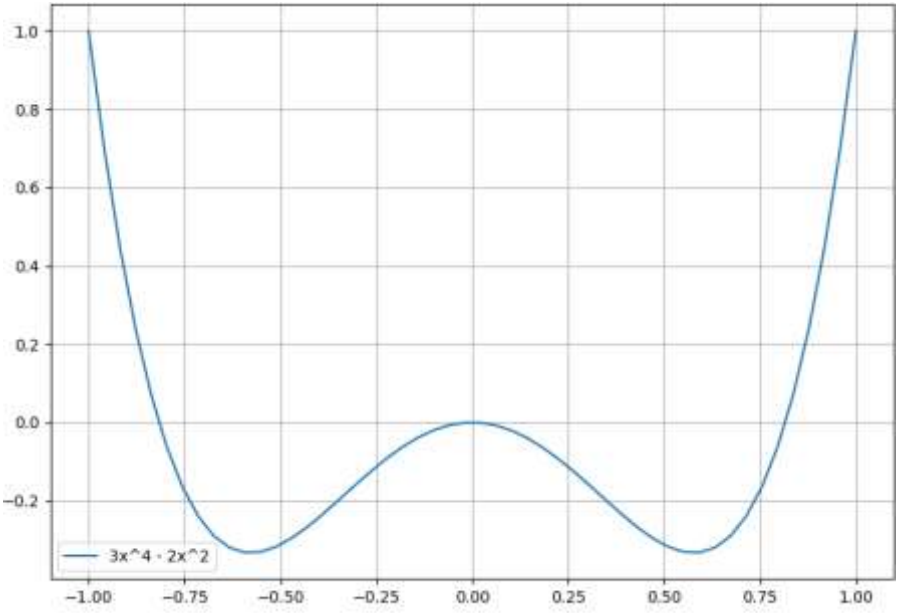


Рисунок 15.2 – Графік $y = 3x^4 - 2x^2$ на інтервалі [-1;1]

Всього існує три вбудованих у функцію методи:

– метод Brent використовує алгоритм Брента для знаходження локального мінімуму. Алгоритм використовує зворотну параболічну інтерполяцію, коли це можливо, щоб прискорити збіжність методу золотого перетину;

– метод Golden використовує метод пошуку золотого перетину. Він використовує аналог методу поділу навпіл для зменшення інтервалу в дужках;

– метод Bounded може виконувати обмежену мінімізацію. Він використовує метод Брента для знаходження локального мінімуму в інтервалі $x_1 < x_{opt} < x_2$.

За замовчанням застосовується метод Bounded, якщо задані межі пошуку, або метод Brent, якщо межі не вказані.

15.4 Метод мінімізації функції багатьох змінних (minimize())

Ця функція може обробляти багатоваріантні вхідні та вихідні дані та має більш складні алгоритми оптимізації. Крім того, minimize() може обробляти обмеження щодо вирішення поставленої задачі. Можна вказати три типи обмежень:

– LinearConstraint. Розв'язок обмежується шляхом взяття внутрішнього добутку значень x рішення з масивом, введеним користувачем, і порівняння результату з нижньою та верхньою межами;

– NonlinearConstraint. Розв'язок обмежений шляхом застосування наданої користувачем функції до значень x рішення та порівняння повернутого значення з нижньою та верхньою межею;

– Bounds. Значення x розв'язку обмежені нижньою та верхньою межею.

Використання цих обмежень може звузити конкретний вибір методу оптимізації, який можна застосувати, оскільки не всі доступні методи підтримують обмеження таким чином.

Далі представлені методи, які можуть бути задані як параметр функції minimize().

Метод Nelder-Mead використовує симплексний алгоритм Нелдера-Міда. Цей алгоритм надійний у багатьох програмах. Однак, якщо можна довіряти числовому обчисленню похідної, інші алгоритми, що використовують інформацію про першу та/або другу похідну, можуть бути кращими для кращої продуктивності в цілому.

Метод Powell є модифікацією методу спряжених напрямків Пауелла. Він виконує послідовну одновимірну мінімізацію вздовж кожного вектора набору напрямків, який оновлюється на кожній ітерації основного циклу мінімізації. Функція не повинна бути диференційованою.

Метод CG шукає мінімум скалярної функції однієї чи декількох змінних за допомогою алгоритму спряженого градієнта.

Метод BFGS використовує алгоритм Бройдена-Флетчера-Голдфарба-Шенно. Щоб швидше наблизитися до розв'язку, ця процедура використовує градієнт цільової функції. Якщо градієнт не задано користувачем, він оцінюється за допомогою перших різниць. Метод Бройдена-Флетчера-Голдфарба-Шанно (BFGS) зазвичай

вимагає меншої кількості викликів функцій, ніж симплексний алгоритм, навіть якщо потрібно оцінити градієнт.

Метод Newton-CG - модифікований метод Ньютона, який використовує алгоритм спряженого градієнта для інвертування локального гессіана.

Метод TNC мінімізує скалярну функцію однієї чи декількох змінних за допомогою усіченого алгоритму Ньютона.

Метод COBYLA використовує метод обмеженої оптимізації за допомогою лінійної апроксимації. Алгоритм заснований на лінійних наближеннях цільової функції та кожного обмеження.

Метод SLSQP використовує послідовне програмування найменших квадратів для мінімізації функції однієї чи декількох змінних із будь-якою комбінацією обмежень, рівності та нерівності.

Метод trust-constr – це алгоритм мінімізації скалярної функції з урахуванням обмежень, які задані системою нерівностей.

Методи Newton-CG, trust-ncgi trust-krylov придатні для вирішення масштабних задач із тисячами змінних. Оскільки для отримання розв'язку потрібен лише добуток матриці Гессе з довільним вектором, алгоритм має низькі вимоги до пам'яті та дозволяє значну економію часу для цих розріджених задач.

Якщо метод не вказано, вибирається один із BFGS, L-BFGS-B, SLSQP, залежно від того, чи має задача обмеження чи межі.

Приклади роботи minimize для отпимізації функції двох змінних $f(x) = 4x_1^2 + 3x_2^2 - 4x_1x_2 + x1$, $x(0) = [5, 3]$ із застосуванням різних методів:

```
def fun(x):
   return 4*x[0]**2 + 3*x[1]**2  - 4*x[0]*x[1] +x[0]

print(scipy.optimize.minimize(fun,[5,3],method='Nelder-Mead')

fun: -0.09374999370453975
     message: 'Optimization terminated successfully.'
       nfev: 87
        nit: 47
     status: 0
     success: True
          x: array([-0.18747256, -0.12501951])
```

print(scipy.optimize.minimize(fun,[5,3],method='Powell'))

```
fun: -0.09374999951636215
 message: 'Optimization terminated successfully.'
    nfev: 68
     nit: 3
  status: 0
 success: True
       x: array([-0.18749869, -0.12498649])
```

print(scipy.optimize.minimize(fun,[5,3],method='CG'))

```
fun: -0.09374999999368269
     jac: array([-2.84519047e-06, -5.30947000e-06])
 message: 'Optimization terminated successfully.'
    nfev: 21
     nit: 3
    njev: 7
  status: 0
 success: True
       x: array([-0.18750121, -0.1250017 ])
```

print(scipy.optimize.minimize(fun,[5,3],method='BFGS'))

```
fun: -0.09374999999997366
 hess_inv: array([[0.18789427, 0.12502498],
       [0.12502498, 0.25000161]])
     jac: array([-5.77419996e-07,  2.71014869e-07])
  message: 'Optimization terminated successfully.'
     nfev: 21
      nit: 6
     njev: 7
   status: 0
  success: True
        x: array([-0.18750009, -0.12500002])
```

print(scipy.optimize.minimize(fun,[5,3],method='TNC'))

```
fun: -0.09374999509441738
     jac: array([ 0.00010177, -0.00023541])
 message: 'Converged (|f_n-f_(n-1)| ~= 0)'
    nfev: 87
     nit: 8
  status: 1
 success: True
       x: array([-0.18751036, -0.12504615])
```

print(scipy.optimize.minimize(fun,[5,3],method='COBYLA'))

```
fun: -0.09374986190931787
   maxcv: 0.0
 message: 'Optimization terminated successfully.'
    nfev: 43
  status: 1
 success: True
       x: array([-0.18748544, -0.12477619])
```

print(scipy.optimize.minimize(fun,[5,3],method='SLSQP'))

```
fun: -0.09374999999999942
     jac: array([-2.88709998e-08,  6.14672899e-08])
 message: 'Optimization terminated successfully'
    nfev: 11
     nit: 3
    njev: 3
  status: 0
 success: True
       x: array([-0.18750001, -0.12500001])
```

print(scipy.optimize.minimize(fun,[5,3],method='trust-constr'))

```
fun: -0.0937499999999991
            grad: array([ 9.31322575e-10, -9.31322575e-10])
             jac: []
  lagrangian_grad: array([ 9.31322575e-10, -9.31322575e-10])
```

```
  message: '`gtol` termination condition is satisfied.'
   method: 'equality_constrained_sqp'
     nfev: 36
     nhev: 0
      nit: 12
    niter: 12
     njev: 12
optimality: 9.313225746154785e-10
   status: 1
  success: True
tr_radius: 14.0
        v: []
        x: array([-0.18749998, -0.12499998])
```

Результати роботи функції *minimize()* із застосуванням різних методів відрізняються несуттєво. Відрізняється кількість ітерацій, обчислень функції та додаткових матриць.

15.5 Метод curve_fit()

Метод scipy.optimize.curve_fit є одним з методів оптимізації, який знаходить параметри моделі, яка найкращим чином апроксимує задані дані.

Конкретно, метод curve_fit знаходить найкращі значення параметрів моделі, яка найбільш точно відтворює задані дані. Цей метод можна використовувати для побудови кривих регресії, функцій апроксимації та інших математичних моделей.

Метод curve_fit вимагає задати функцію, яку треба апроксимувати, а також вхідні дані, які складаються з x та y координат. Також можна вказати початкові значення параметрів, які метод буде використовувати під час пошуку найкращих значень параметрів моделі.

Після запуску методу curve_fit, він повертає кортеж з оптимальними значеннями параметрів моделі та коваріаційною матрицею. Коваріаційна матриця вказує на точність оцінки параметрів моделі.

Отже, метод scipy.optimize.curve_fit допомагає виконати апроксимацію даних та знайти параметри моделі, які найбільш точно відтворюють задані дані.

Приклад використання наведеного методу.

Задача: Нехай дано деякий набір точок (x, y), який наближено відповідає параболічній залежності $y = a*x^2 + b*x + c$. Необхідно знайти значення параметрів a, b та c, що найкраще описують цю залежність.

Розв'язок: Спочатку імпортуємо необхідні бібліотеки та створимо функцію, яка описує нашу параболічну залежність:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

def parabola(x, a, b, c):
    return a*x**2 + b*x + c
```

Далі задаємо наші дані як масиви NumPy:

```
x_data = np.array([-2, -1, 0, 1, 2])
y_data = np.array([6.4, 2.1, 0.9, 1.7, 6.2])
```

Тепер можна використати метод curve_fit з бібліотеки scipy.optimize для знаходження найкращих значень параметрів a, b та c. Цей метод приймає функцію, яка описує залежність (у нашому випадку - parabola), а також початкові значення параметрів, які ми можна приблизно визначити з графіка. Для нашого випадку вибрано a=1, b=1 та c=1:

```
popt, pcov = curve_fit(parabola, x_data, y_data, p0=[1, 1, 1])
```

Результатом роботи методу будуть оптимальні значення параметрів a, b та c, які можна отримати таким чином:

```
a_opt, b_opt, c_opt = popt
```

Також можна побачити, як парабола наблизилась до даних, нанісши їх на графік разом з параболою:

```
plt.plot(x_data, y_data, 'bo', label='data')
plt.plot(x_data, parabola(x_data, a_opt, b_opt, c_opt), 'r-', label='fit')
plt.legend()
plt.show()
```

Результат виконання приведеного в приклад коду на рисунку 15.3.

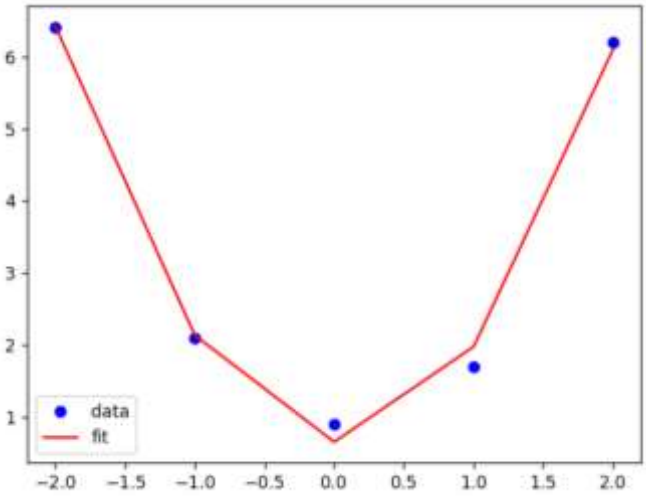


Рисунок 15.3 – Результат наближення параболи до наведених даних

15.6 Висновки

Описаний перелік можливостей пакету не є вичерпним. Використання SciPy є дуже корисним і для поглибленого вивчення математики, і для проведення складних розрахунків. Наукові програми, які використовують SciPy, отримують переваги від розробки додаткових модулів у багатьох нішах програмного забезпечення розробниками з усього світу.

СПИСОК ЛІТЕРАТУРИ

1. Barry, P. Head First Python: A Brain-Friendly Guide [Text] / P. Barry, D. Griffiths. – Sebastopol, California : O'Reilly Media, 2016. – 622 p.

2. Beazley, D. Python Essential Reference [Text] / D. Beazley. – Boston: Addison-Wesley Professional, 2009. – 717 p.

3. Beazley, D. Python Cookbook [Text] / D. Beazley, B. Jones. – Sebastopol, California : O'Reilly Media, 2013. – 704 p.

4. Beazley, D. Python Distilled [Text] / D. Beazley. – London : Pearson, 2021. – 352 p.

5. Bird, S. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit [Text] / S. Bird, E. Klein, E. Loper. – Sebastopol, California : O'Reilly Media, 2009. – 502 p.

6. Bunday, B. Basic Optimisation Methods [Text] / B. Bunday. – London : Edward Arnold, 1984. – 136 p.

7. Burley, T. A. Operational research [Text] / T. A. Burley, G. O'Sullivan. – Basingstoke : Macmillan, 1990. – 227 p.

8. Caulkins, J. P. The Role of Operations Research in Public Policy [Text] / J. P. Caulkins // Operations Research Proceedings 2002 / [eds.: U. Leopold-Wildburger, F. Rendl, G. Wäscher]. – Berlin : Springer-Verlag, 2002. – Section 1. – P. 1-13.

9. Chollet, F. Deep Learning with Python [Text] / F. Chollet. – Shelter Island, New York : Manning, 2017. – 384 p.

10. Chopra, D. Mastering Natural Language Processing with Python [Text] / D. Chopra. – Birmingham : Packt Publishing, 2016. – 238 p.

11. Churchman, C. West Introduction to Operations Research [Text] / C. West Churchman, Russell L. Ackoff, E. L. Arnoff. – New York: J. Wiley and Sons, 1957. – 645 p.

12. Cielen, D. Introducing Data Science : Big Data, Machine Learning, and more, using Python tools [Text] / D. Cielen, A. Meysman, M. Ali. – Shelter Island, New York : Manning, 2016. – 320 p.

13. Conley, W. Computer optimization techniques [Text] / W. Conley. – New York : Petrocelli Books, 1984. – 244 p.

14. Crina, G. Stigmergic Optimization: Inspiration, Technologies and Perspectives [Text] / G. Crina, A. Ajith // Stigmergic Optimization.

Studies in Computational Intelligence, vol 31 / [eds.: A. Ajith, G. Crina , R. Vitorino]. – Berlin : Springer-Verlag, 2006. – Section 1. – P. 1-24.

15. Daellenbach, H. G. Introduction to operations research techniques [Text] / H. G. Daellenbach, J. A. George. – Boston : Allyn and Bacon, 1978. – 624 p.

16. Downey, A. B. Think Python : How to Think Like a Computer Scientist [Text] / A. B. Downey. – Sebastopol, California : O'Reilly Media, 2015. – 292 p.

17. Dubrovin V. Loss Function as Tolerance Optimization Criteria [Text] / V. Dubrovin, G. Tabunshchik // International Conference on Modern Problems of Telecommunications, Computer Science and Engineers Training TCSET'2000, Slavsko, Ukraine, 2000, 14-19 February 2000. – Lviv-Slavsko, 2002. – P. 45-46.

18. Ernesti, J. Python 3: The Comprehensive Guide [Text] / J. Ernesti, P. Kaiser. – Bonn : Rheinwerk Computing, 2023. – 1036 p.

19. Fabrycky, W. J. Operations Economy : Industrial Applications of Operations Research [Text] / W. J. Fabrycky. – Englewood Cliffs, N.J. : Prentice-Hall, 1966. – 486 p.

20. Gorelick, M. High Performance Python [Text] / M. Gorelick, I. Ozsvald. – Sebastopol, California : O'Reilly Media, 2020. – 466 p.

21. Gries, P. Practical Programming : An Introduction to Computer Science Using Python 3 (Pragmatic Programmers) [Text] / P. Gries, J. Campbell, J. Montojo. – Raleigh, N.C. : Pragmatic Bookshelf, 2013. – 402 p.

22. Guzdial, M. Introduction to Computing and Programming in Python [Text] / M. Guzdial, B. Ericson. – London : Pearson, 2015. – 528 p.

23. Harvey, C. M. Operations Research: an Introduction to Linear Optimization and Decision Analysis [Text] / C. M. Harvey. – New York : North Holland, 1979. – 453 p.

24. Hattingh, C. Using Asyncio in Python [Text] / C. Hattingh. – Sebastopol, California : O'Reilly Media, 2020. – 163 p.

25. Hellmann, D. Python 3 Standard Library by Example, The (Developer's Library) [Text] / D. Hellmann. – Boston : Addison-Wesley, 2017. – 1456 p.

26. Hetland, M. L. Beginning Python: From Novice to Professional [Text] / M. L. Hetland. – New York : Apress, 2005. – 640 p.

27. Hillier, F. S. Introduction to Mathematical Programming [Text] / F. S. Hillier, G. J. Lieberman. – New York : McGraw-Hill, 1995. – 716 p.

28. Hillier, F. S. Introduction to Operations Research [Text] / F. S. Hillier, G. J. Lieberman. – San Francisco : Holden-Day, 1980. – 829 p.

29. Hillier, F. S. Introduction to Stochastic Models in Operations Research [Text] / F. S. Hillier, G. J. Lieberman. – New York : McGraw-Hill, 1990. – 555 p.

30. Hossain, S. A. Operations Research: Recent Advances [Text] / S. A. Hossain, S. Kar. – New Delhi ; Chennai ; Mumbai ; Kolkata : Narosa Publishing House, 2014. – 164 p.

31. Introduction to Algorithms [Text] / [eds.: T. H. Cormen, C. E. Leiserson, R. L. Rivest]. – Cambridge, Mass. : MIT Press, 2001. – 1056 p.

32. Karumanchi, N. Data Structure and Algorithmic Thinking with Python: Data Structure and Algorithmic Puzzles [Text] / N. Karumanchi – Hyderabad: CareerMonk, 2015. – 436 p.

33. Kochenderfer, M. J. Algorithms for Optimization [Text] / M. J. Kochenderfer, T. A. Wheeler. – Boston: The MIT Press, 2019. – 521 p.

34. Langtangen H. P. A Primer on Scientific Programming with Python (Texts in Computational Science and Engineering) [Text] / H. P. Langtangen. – Berlin : Springer, 2012. – 798 p.

35. Linear Multi-Objective Particle Swarm Optimization [Text] / [eds.: M. Sanaz, M. Sanaz, H. Werner] // Stigmergic Optimization. Studies in Computational Intelligence, vol 31 / [eds.: A. Ajith, G. Crina , R. Vitorino]. – Berlin : Springer-Verlag, 2006. – Section 9. – P. 209-238.

36. Liu, Y. H. Python Machine Learning By Example: The easiest way to get into machine learning [Text] / Y. H. Liu. – Birmingham : Packt Publishing, 2017. – 254 p.

37. Lott, S. F. Modern Python Cookbook: The latest in modern Python recipes for the busy modern programmer [Text] / S. F. Lott. – Birmingham : Packt Publishing, 2016. – 824 p.

38. Luangpaiboon, P. Process Optimization via Conventional Factorial Designs and Simulated Annealing on the Path of Steepest Ascent for a CSTR [Text] / P. Luangpaiboon // Operations Research Proceedings 2002 / [eds.: U. Leopold-Wildburger, F. Rendl, G. Wäscher]. – Berlin : Springer-Verlag, 2002. – Section 57. – P. 347-352.

39. Lubanovic, B. Introducing Python [Text] / B. Lubanovic. – Sebastopol, California : O'Reilly Media, 2019. – 630 p.

40. Lutz, M. Learning Python [Text] / M. Lutz. – Sebastopol, California : O'Reilly Media, 2009. – 502 p.

41. Lutz, M. Programming Python: Powerful Object-Oriented Programming [Text] / M. Lutz. – Sebastopol, California : O'Reilly Media, 2011. – 1626 p.

42. Lutz, M. Python Pocket Reference: Python In Your Pocket [Text] / M. Lutz. – Sebastopol, California : O'Reilly Media, 2014. – 262 p.

43. Makower, M. S. Operational research : problems, techniques, and exercises [Text] / M. S. Makower, E. Williamson. – London : Teach Yourself Books, 1975. – 276 p.

44. Martelli, A. Python in a Nutshell: A Desktop Quick Reference [Text] / A. Martelli, A. Ravenscroft, S. Holden, P. McGuire. – Sebastopol, California : O'Reilly Media, 2023. – 735 p.

45. Mathematical Optimization of a Magnetic Ruler Layout with Rotated Pole Boundaries [Text] / [eds.: M. Fügenschuh, A. Fügenschuh, M. Ludszuweit] // Operations Research Proceedings 2015 / [eds.: K. F. Dörner, I. Ljubic, G. Pflug]. – Berlin : Springer-Verlag, 2017. – Section 16. – P. 117-123.

46. Matthes, E. Python Crash Course : A Hands-On, Project-Based Introduction to Programming [Text] / E. Matthes. – San Francisco : No Starch Press, 2015. – 560 p.

47. McKinney, W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython [Text] / W. McKinney. – Sebastopol, California : O'Reilly Media, 2017. – 547 p.

48. Moder, J. J. Handbook of Operations Research : Foundations and Fundamentals [Text] / J. J. Moder, S. E. Elmaghraby. – New York : Van Nostrand Reinhold Co., 1978. – 622 p.

49. Moder, J. J. Handbook of Operations Research : Models And Applications [Text] / J. J. Moder, S. E. Elmaghraby. – New York : Van Nostrand Reinhold Co., 1978. – 659 p.

50. Mohammed, EA. Cooperative Particle Swarm Optimizers: A Powerful and Promising Approach [Text] / EA. Mohammed, K. Mohammed // Stigmergic Optimization. Studies in Computational Intelligence, vol 31 / [eds.: A. Ajith, G. Crina , R. Vitorino]. – Berlin : Springer-Verlag, 2006. – Section 10. – P. 239-259.

51. Moore, P. G., Basic Operational Research [Text] / P. G. Moore. – London : Pitman, 1968. – 185 p.

52. Morse Philip M., Kimball George E. Methods of Operations Research [Text] / Philip M. Morse, George E. Kimball. – New York, MIT Press and John Wiley & Sons, 1951

53. Müller J. P. Beginning Programming with Python For Dummies [Text] / J. P. Muller. – New York, For Dummies, 2014. – 408 p.

54. Müller A. Introduction to Machine Learning with Python: A Guide for Data Scientists [Text] / A. Müller, S. Guido. – Sebastopol, California : O'Reilly Media, 2016. – 398 p.

55. Osborne, M. R. Finite Algorithms in Optimization and Data Analysis [Text] / M. R. Osborne. – Chichester ; New York : Wiley. – 383 p.

56. Panneerselvam, R. Operations research [Text] / R. Panneerselvam. – New Delhi : Prentice-Hall of India, 2006. – 608 p.

57. Parallel Particle Swarm Optimization Algorithms with Adaptive Simulated Annealing [Text] / [eds.: C. Shu-Chuan, T. Pei-Wei, P. Jeng-Shyang] // Stigmergic Optimization. Studies in Computational Intelligence, vol 31 / [eds.: A. Ajith, G. Crina , R. Vitorino]. – Berlin : Springer-Verlag, 2006. – Section 11. – P. 261-279.

58. Pilgrim, M. Dive into Python 3 [Text] / M. Pilgrim. – New York : Apress, 2009. – 360 p.

59. Ramalho, L. Fluent Python: Clear, Concise, and Effective Programming [Text] / L. Ramalho. – Sebastopol, California : O'Reilly Media, 2015. – 790 p.

60. Reith-Ahlemeier, G. Resource-orientated Purchase Planning and Supplier Selection - Models and Algorithms for Supply Chain Optimization and E-Commerce [Text] / G. Reith-Ahlemeier // Operations Research Proceedings 2002 / [eds.: U. Leopold-Wildburger, F. Rendl, G. Wäscher]. – Berlin : Springer-Verlag, 2002. – Section 3. – P. 20-25.

61. Schienle, A. Solving the Time-Dependent Shortest Path Problem Using Super-Optimal Wind [Text] / A. Schienle // Operations Research Proceedings 2017 / [eds.: N. Kliewer, J. Ehmke, R. Borndörfer]. – Berlin : Springer-Verlag, 2018. – Section 2. – P. 3-9.

62. Schwefel, HP. Evolution and Optimum Seeking [Text] / HP. Schwefel. – New York : Wiley, 1995. – 456 p.

63. Schwefel, HP. Numerical Optimization of Computer Models [Text] / HP. Schwefel. – Chichester ; New York : Wiley, 1981. – 389 p.

64. Scopatz, A. Effective Computation in Physics: Field Guide to Research with Python [Text] / A. Scopatz, K. Huff. – Sebastopol, California : O'Reilly Media, 2015. – 550 p.

65. Seizinger, M. The Two Dimensional Bin Packing Problem with Side Constraints [Text] / M. Seizinger // Operations Research Proceedings

2017 / [eds.: N. Kliewer, J. Ehmke, R. Borndörfer]. – Berlin : Springer-Verlag, 2018. – Section 7. – P. 45-50.

66. Shaw, Z. A. Learn Python 3 the Hard Way : A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code [Text] / Z. A. Shaw. – Boston : Addison-Wesley, 2017. – 320 p.

67. Slatkin B. Effective Python [Text] / B. Slatkin. – Boston : Addison-Wesley, 2019. – 480 p.

68. Sweigart A. Automate the Boring Stuff with Python, 2nd Edition: Practical Programming for Total Beginners [Text] / A. Sweigart. – San Francisco : No Starch Press, 2019. – 592 p.

69. Sweigart A. Invent Your Own Computer Games with Python [Text] / A. Sweigart. – San Francisco : No Starch Press, 2016. – 376 p.

70. Taha, H. A. Operations research: An Introduction [Text] / H. A. Taha. – Upper Saddle River, New Jersey : Pearson Education, 2003. – 830 p.

71. Thanaski, J. Python Testing Cookbook [Text] / J. Thanaski. – Birmingham : Packt Publishing, 2017. – 486 p.

72. Thareja, R. Python Programming: Using Problem Solving Approach [Text] / R. Thareja. – Oxford : Oxford University Press, 2017. – 560 p.

73. Theiler, U. Risk-Return Optimization of the Bank Portfolio [Text] / U. Theiler // Operations Research Proceedings 2002 / [eds.: U. Leopold-Wildburger, F. Rendl, G. Wäscher]. – Berlin : Springer-Verlag, 2002. – Section 2. – P. 14-19.

74. Thierauf, R. J. An Introductory Approach to Operations Research [Text] / R. J. Thierauf. – Santa Barbara, Calif. : Wiley, 1978. – 412 p.

75. Thierauf, R. J. Decision Making Through Operations Research [Text] / R. J. Thierauf, R. A. Grosse. – Santa Barbara, Calif. : Wiley, 1970. – 723 p.

76. Traub, J. F. A General Theory of Optimal Algorithms [Text] / J. F. Traub, H. Woźniakowski. – New York : Academic Press, 1980. – 341 p.

77. Traub, J. F. Information, Uncertainty, Complexity [Text] / J. F. Traub, G. W. Wasilkowski, H. Woźniakowski. – Reading, Mass. : Addison-Wesley Pub. Co., Advanced Book Program/World Science Division, 1983. – 176 p.

78. Turnquist, G. L. Python Testing Cookbook [Text] / G. L. Turnquist. – Birmingham : Packt Publishing, 2011. – 364 p.

79. Vanderplas, J. Python Data Science Handbook [Text] / J. Vanderplas. – Sebastopol, California : O'Reilly Media, 2016. – 546 p.

80. Wagner H. M. Principles of Operations Research / H. M. Wagner. – Englewood Cliffs : Prentice-Hall, 1969. – 937 p.

81. Warren, H. S., Jr. Hacker's Delight [Text] / H. S. Warren, Jr. – Boston : Addison-Wesley, 2002. – 306 p.

82. Zaychenko, Yu. P. Problem of fuzzy portfolio optimization and its solution with application of forecasting methods [Text] / Yu. P. Zaychenko, I. A. Sidoruk // Системні дослідження та інформаційні технології : міжнародний науково-технічний журнал. – 2016. – № 1. – С. 85-98.

83. Zelle, J. M. Python programming : An Introduction to Computer Science [Text] / J. M. Zelle. – Wilsonville : Franklin, Beedle, 2003. – 528 p.

84. Zgurovsky, M. Z. Big Data : Conceptual Analysis and Applications [Text] / M. Z. Zgurovsky, Yu. P. Zaychenko. – Luxembourg : Springer Cham, 2019. – 277 p.

85. Zgurovsky, M. Z. The Fundamentals of Computational Intelligence : System Approach [Text] / M. Z. Zgurovsky, Yu. P. Zaychenko. – Luxembourg : Springer Cham, 2016. – 375 p.

86. Бартіш М. Я. Дослідження операцій. Ч. 1. Лінійні моделі [Текст] / М. Я. Бартіш, І. М. Дудзяний. — Львів: Видавничий центр Львівського національного університету ім. І. Франка, 2007. — 168 с.

87. Бартіш М. Я. Дослідження операцій. Ч. 2. Алгоритми оптимізації на графах [Текст] / М. Я. Бартіш, І. М. Дудзяний. — Львів: Видавничий центр Львівського національного університету ім. І. Франка, 2007. — 120 с.

88. Бартіш М. Я. Дослідження операцій. Ч. 3. Ухвалення рішень і теорія ігор [Текст] / М. Я. Бартіш, І. М. Дудзяний. — Львів: Видавничий центр Львівського національного університету ім. І.Франка, 2009. — 277 с.

89. Бартіш М. Я. Дослідження операцій. Ч. 4. Нелінійне програмування: підручник [Текст] / М. Я. Бартіш, І. М. Дудзяний. — Львів: Видавництво Львівського університету ім. І. Франка, 2011. — 207 с.

90. Бартіш М. Я. Дослідження операцій: підручник. Ч. 5 : Моделі з чинником часу [Текст] / М. Я. Бартіш, І. М. Дудзяний; М-во освіти і науки, молоді та спорту України, Львів. нац. ун-т ім. І. Франка. — Л. : Вид-во ЛНУ, 2012. — 256 с.

91. Боровик О. Л. Дослідження операцій в економіці : навч. посіб. [Текст] / О. Л. Боровик, Л. В. Боровик. – Київ : Центр учбової літератури, 2007. – 424 с.

92. Боровська Т. М. Основи теорії управління та дослідження операцій : навч. посіб. [Текст] / Т. М. Боровська, І. С. Колеснік, В. А. Северілов. – Вінниця : УНІВЕРСУМ-Вінниця, 2008. – 242 с.

93. Доненко, В. І. Сучасні методи мотивації та управління командою розробки продукту ІТ-стартапу : монографія [Текст] / В.І. Доненко, К.О. Водолазкіна, В.І. Дубровін. – Запоріжжя : ЗНТУ, 2019. – 85 с.

94. Дубровін, В. І. Методи оптимізації та їх застосування в задачах навчання нейронних мереж : навчальний посібник [Текст] / В. І. Дубровін, С. О. Субботін. – Запоріжжя : ЗНТУ, 2003. – 136 с.

95. Дубровін, В. І. Прийняття рішень у процесі управління ризиками проектів : навчальний посібник [Текст] / В. І. Дубровін, В. М. Льовкін. – Запоріжжя : ЗНТУ, 2012. – 196 с.

96. Дубровін, В. І. Оптимальний розподіл коштів між компонентами медіаплану [Текст] / Ю. П. Ванюшанік, В. І. Дубровін, К. О. Фандєєва // Тиждень науки : наук.-практ. конф., м. Запоріжжя, 9-13 квітня 2013 р. : тези доповідей. – Запоріжжя : ЗНТУ, 2012. – С. 342-343.

97. Дубровін, В. І. Оптимізація автоматизованого збирання інтегральних мікросхем [Текст] / В. І. Дубровін // Вісник Державного університету «Львівська політехника». Серія: Комп’ютерні системи проектування. Теорія і практика. – 1998. – №327. – С.157-160.

98. Ємець, О. О. Методи оптимізації та дослідження операцій : навчальний посібник [Текст] / О. О. Ємець. – Полтава : ПУЕТ, 2019. – 139 с.

99. Катренко А. В. Дослідження операцій: підручник [Текст] / А. В. Катренко. — Львів: Магнолія Плюс, 2004. — 549 с.

100. Клебанова Т. С. Дослідження операцій : навч. посіб. для студентів напряму підготовки 6.030502 "Економічна кібернетика" всіх форм навчання [Текст] / Т. С. Клебанова, О. Ю. Полякова, Л. О. Чаговець та ін. – Харків : Вид. ХНЕУ, 2013. – 192 с.

101. Кунда Н. Т. Дослідження операцій у транспортних системах : навч. посіб. для студентів напряму "Транспортні технології" вищих навчальних закладів [Текст] / Н. Т. Кунда. – Київ : Вид. Дім "Слово", 2008. – 400 с.

102. Ларіонов Ю. І. Дослідження операцій в інформаційних системах [Текст] / Ю. І. Ларіонов, В. М. Левикін, М. А. Хажмурадов – Харків : Компанія СМІТ, 2005. – 364 с.

103. Зайченко, Ю. П. Аналіз багатокритеріальної задачі оптимізації інвестиційного портфеля на основі прогнозування прибутковості акцій [Текст] / Зайченко Ю. П., Сидорук І. А. // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : збірник наукових праць. – 2015. – Вип. 62. – С. 79-88.

104. Зайченко Ю. П. Дослідження операцій: підручник [Текст] / Ю. П. Зайченко. — 5-е вид., перероб. і доп. — К. : ЗАТ «ВІПОЛ», 2001. — 688 с.

105. Зайченко, Ю. П. Метод вирішення задачі аналізу ситуацій системою підтримки прийняття рішення для прогнозування [Текст] / Ю. П. Зайченко, М. Ю. Медін // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : збірник наукових праць. – 2008. – № 48. – С. 89-93.

106. Зайченко, Ю. П. Удосконалення методу оптимізації нечіткого фондового портфелю з новими функціями ризику [Текст] / Ю. П. Зайченко, М. О. Мурга // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : збірник наукових праць. – 2011. – № 54. – С. 54-63.

107. Методи оптимізації та дослідження операцій : навчальний посібник [Текст] / Я. Б. Сікора, А. Й. Щехорський, Е. Л. Якимчук. – Житомир : Вид-во ЖДУ ім. Івана Франка, 2019. – 148 с.

108. Методи оптимізації та дослідження операцій : рекомендації до практичних завдань [Текст] / С. В. Прокопович, О. В. Панасенко, Л. О. Чаговець. – Харків : ХНЕУ ім. С. Кузнеця, 2019. – 64 с.

109. Оптимізація оподаткування [Текст] / [В. І. Дубровін, Л. Ю. Дейнега, Н. В. Джрагацпанян та ін.] // Тиждень науки : наук.-практ. конф., м.Запоріжжя, 15-19 квітня 2019 р. : тези доповідей. – Запоріжжя : ЗНТУ, 2019. – С. 161-162.

110. Роман Л. Л. Дослідження операцій. Курс лекцій [Текст] / Л. Л. Роман. – Львів : Видавництво Тараса Сороки, 2008. – 272 с.

111. Дубровін В.І. Методичні вказівки до лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів спеціальностей 121 “Інженерія програмного забезпечення” та 122 «Комп’ютерні науки» денної форми навчання / В.І. Дубровін, Л.Ю. Дейнега. – Запоріжжя: НУ «Запорізька політехніка», 2024. – 54 с.

112. SciPy User Guide [Electronic resource]. - Access mode: https://docs.scipy.org/doc/scipy/tutorial/index.html#user-guide.

ДОДАТОК. ПУТІВНИК ЗА СПИСКОМ ЛІТЕРАТУРИ

Тема	Номер літературного джерела
Методи оптимізації	6, 7, 13, 14, 31, 33, 35, 50, 55, 57, 60, 62, 63, 73, 76, 77, 81, 82, 84, 85, 93, 103, 105, 106
Дослідження операцій	8, 11, 15, 19, 23, 27-30, 38, 43, 48, 49, 51, 52, 56, 61, 65, 70, 74, 80, 86-90, 92, 100, 104, 110
Застосування методів оптимізації та дослідження операцій	17, 45, 75, 91, 94-98, 101, 102, 107-109, 111
Засоби мови програмування Python	1-5, 9, 10, 12, 16, 18, 20-22, 24-26, 32, 34, 36, 37, 39-42, 44, 46, 47, 53, 54, 58, 59, 64, 66-69, 71, 72, 78, 79, 83, 112

Valerii Dubrovin, Larysa Deineha

Optimization Methods And Their Implementation Using The Tools Of The Python Programming Language

SUMMARY

Валерій ДУБРОВІН

Лариса ДЕЙНЕГА

МЕТОДИ ОПТИМІЗАЦІЇ ТА ЇХ РЕАЛІЗАЦІЯ ЗАСОБАМИ МОВИ ПРОГРАМУВАННЯ PYTHON

Навчальний посібник

Рецензенти:

С.І. Гоменюк, доктор технічних наук, професор кафедри програмної інженерії, декан математичного факультету Запорізького національного університету

А.В. Переверзєв, доктор технічних наук, професор, проректор з наукової роботи Запорізького інституту економіки та інформаційних технологій