АНАЛІЗ РЕАЛІЗАЦІЙ СИМПЛЕКС-МЕТОДУ ЛІНІЙНОГО ПРОГРАМУВАННЯ НА ПРИКЛАДІ ЗАДАЧІ ЛОГІСТИЧНОЇ ОПТИМІЗАЦІЇ

КН-4
Мудра Катерина Володимирівна

НАУКОВИЙ КЕРІВНИК
Силенко Ілля Володимирович

Мета

розробити та реалізувати модель багатоцільової оптимізації задачі гуманітарної логістики з використанням симплекс-методу

Постановка задачі

- дослідження основних засад лінійного програмування та наявних програмних інструментів
- вибір задачі та побудова математичної моделі
- реалізація задачі трьома методами: з використанням бібліотек PuLP і scipy.optimize.linprog, а також за допомогою власноруч реалізованого симплекс-методу
- тестування розробленого алгоритму на експериментальних даних
- аналіз мікроаналізу чутливості (альфа, бета, гамма)

Актуальність роботи

Гуманітарна логістика, особливо в умовах війни, вимагає моделей, які враховують не лише вартість, а й ризик та час доставки.

Для оптимізації цих показників, а також для спрощення обрахунків можна використовувати лінійне програмування.

Багатоцільова задача лінійного програмування

це задача оптимізації, у якій необхідно одночасно враховувати декілька цільових функцій, що описують різні аспекти системи

Загальна математична постановка виглядає так:

$$\begin{aligned} &\text{Оптимізувати (одночасно):} && f_1(x),\ f_2(x),\ \ldots,\ f_k(x) \\ &\text{при умові:} && Ax \leq b, \quad x \geq 0 \end{aligned}$$

де:

- $x \in \mathbb{R}^n$ — вектор змінних;
- $f_i(x) = c_i^T x, i = 1, \ldots, k$ — лінійні цільові функції;
- $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ — система обмежень.

Задача гуманітарної логістики

Тема - задача гуманітарної логістики. Питання - як доставити вантаж із m складів до n пунктів призначення, зважаючи на ризик маршрутів.

Мета - мінімізувати зважену суму трьох показників:

- вартості доставки,
- ризику маршруту,
- часу доставки.

Цільова функція:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} (\alpha \cdot c_{ij} + \beta \cdot r_{ij} + \gamma \cdot t_{ij}) \cdot x_{ij}$$

α, β, γ — вагові коефіцієнти,
що визначають важливість кожного з критеріїв (вартість, ризик, час),

c_{ij} , r_{ij} , t_{ij} – відповідні значення для кожного маршруту доставки,

x_{ij} – змінна, що визначає обсяг доставки з точки i в точку j

Умови задачі

Набори даних

- $i \in \{1, 2, ..., m\}$ — індекси складів (центрів постачання),
- $j \in \{1, 2, ..., n\}$ — індекси пунктів призначення (споживачів).

Параметри

- s_i — доступна кількість товарів на складі i,
- d_j — потреба в пункті j,
- c_{ij} — вартість доставки одиниці товару від складу i до пункту j,
- r_{ij} — ризик доставки по маршруту $i \to j$,
- t_{ij} — час доставки по маршруту $i \to j$,
- $\alpha, \beta, \gamma \in [0, 1]$ — ваги відповідно для ризику, часу, вартості, причому $\alpha + \beta + \gamma = 1$,

Змінні

- $x_{ij} \geq 0$ — кількість одиниць, які транспортуються зі складу i до пункту j.

Обмеження

Забезпечення всіх пунктів:

$$\sum_{i=1}^{m} x_{ij} \geq d_j, \quad \forall j = 1, ..., n$$

Обмеження запасів на складах:

$$\sum_{j=1}^{n} x_{ij} \leq s_i, \quad \forall i = 1, ..., m$$

Невід'ємність:

$$x_{ij} \geq 0, \quad \forall i, j$$

Реалізація алгоритму

Три реалізації алгоритму на Python:

PuLP
- Описуються змінні x_ij, які відповідають перевезенням із складів у пункти.
- Задається цільова функція як зважена сума витрат, ризиків і часу.
- Додаються обмеження на суму поставок та потреб.
- Після формулювання задача автоматично розв’язується внутрішнім симплекс-методом.

SciPy
- Подання задачі у вигляді стандартної матриці: A_eq, b_eq, c.
- Передається масив меж змінних (усі невід’ємні).
- Алгоритм highs (HiGHS solver) виконує оптимізацію й повертає результат.

Власна реалізація
- Покрокове ручне впровадження класичного симплекс-алгоритму
- Візуалізація симплекс-таблиць та виведення пояснень
- Графічне представлення “шляху” цільової функції

Реалізація алгоритму

PuLP

```
--- alpha=1.0, beta=0.2, gamma=0.1 ---
[PuLP] Статус: Optimal
[PuLP] Z = 2551.00
    x_0_0 = 80.00
    x_0_3 = 20.00
    x_1_1 = 100.00
    x_1_3 = 50.00
    x_2_2 = 90.00
    x_2_3 = 30.00
```

Власна реалізація

```
[Власна реалізація] Z = 3974.00
    x_0_1 = 10.00
    x_0_2 = 90.00
    x_1_0 = 50.00
    x_1_3 = 100.00
    x_2_0 = 30.00
    x_2_1 = 90.00
```

SciPy

```
--- alpha=1.0, beta=0.2, gamma=0.1 ---
[SciPy] Статус: Optimization terminated successfully. (HiGHS Status 7: Optimal)
[SciPy] Z = 2551.00
    x_0_0 = 80.00
    x_0_3 = 20.00
    x_1_1 = 100.00
    x_1_3 = 50.00
    x_2_2 = 90.00
    x_2_3 = 30.00
```

Реалізація алгоритму

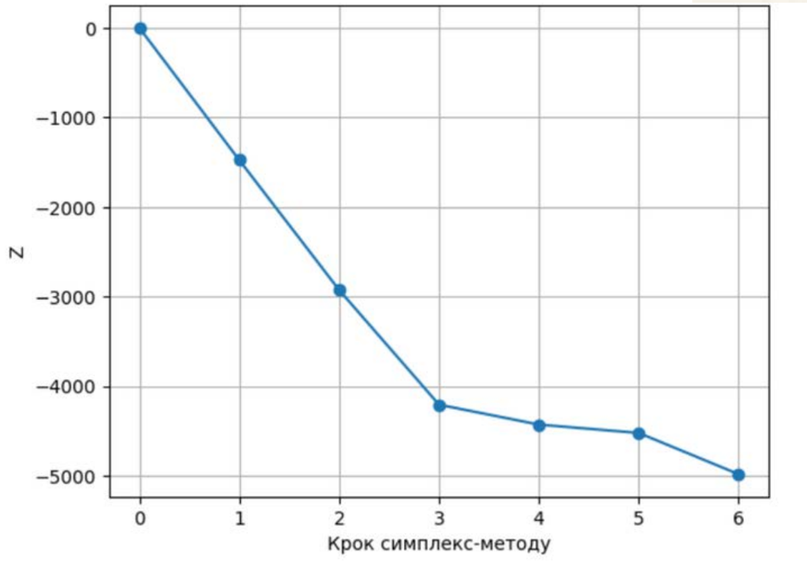
Особливості власної реалізації симплексу:

- Покроковий вивід таблиці з поясненнями
- Графік зміни цільової функції
- Можливість роботи з великомасштабною моделлю (до 80 змінних)

```
--- Симплекс-таблиця. Крок 5 ---
Базис   x1      x2      x3      x4      x5      x6      x7      x8      x9      x10     x11     s1      s2      s3      s4      s5      s6      s7      s8      RHS
x2      1.00    1.00    0.00    1.00    0.00    0.00    -1.00   0.00    0.00    0.00    -1.00   0.00    1.00    0.00    0.00    0.00    0.00    -1.00   0.00    10.00
s3      0.00    0.00    0.00    -1.00   1.00    1.00    1.00    0.00    0.00    0.00    0.00    -1.00   0.00    1.00    0.00    0.00    0.00    0.00    -1.00   50.00
x9      1.00    0.00    0.00    1.00    0.00    -1.00   -1.00   0.00    1.00    0.00    0.00    1.00    1.00    0.00    1.00    0.00    -1.00   -1.00   0.00    30.00
s5      0.00    0.00    0.00    -1.00   1.00    1.00    1.00    0.00    0.00    0.00    0.00    -1.00   -1.00   0.00    -1.00   1.00    1.00    1.00    0.00    50.00
x10     -1.00   0.00    0.00    -1.00   0.00    1.00    1.00    0.00    0.00    1.00    1.00    0.00    -1.00   0.00    0.00    0.00    1.00    1.00    0.00    90.00
x3      0.00    0.00    1.00    0.00    0.00    0.00    1.00    0.00    0.00    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00    1.00    0.00    90.00
x8      0.00    0.00    0.00    1.00    0.00    0.00    0.00    1.00    0.00    0.00    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00    1.00    100.00
z       1.60    0.00    0.00    8.80    -7.30   -4.40   -4.30   0.00    0.00    0.00    8.90    13.10   7.40    0.00    8.80    0.00    1.40    5.70    11.60   3609.00

Розрахунок θ для кожного рядка:
  θ_1 = ∞ (a_ik = 0.00 ≤ 0)
  θ_2 = 50.00 / 1.00 = 50.00 для s2
  θ_3 = ∞ (a_ik = 0.00 ≤ 0)
  θ_4 = 50.00 / 1.00 = 50.00 для s4
  θ_5 = ∞ (a_ik = 0.00 ≤ 0)
  θ_6 = ∞ (a_ik = 0.00 ≤ 0)
  θ_7 = ∞ (a_ik = 0.00 ≤ 0)

=== Пояснення переходу до наступного кроку ===
Вибрано вхідну змінну: x_5 (мінімальна оцінка приросту: -7.30)
Мінімальне θ = 50.00, отже, вихідна змінна: s2
Півот-елемент: 1.00
Оновлюємо базис: замінюємо s2 на x_5
```



Експериментальні дані

Для тестування алгоритму було обрано 2 набори експериментальних даних:

- Склади: 3
- Пункти призначення: 4
- Запаси по складах: [100, 150, 120]
- Потреби пунктів призначення: [80, 100, 90, 100]
- Матриця вартості C (грн):
  [[4, 6, 9, 7],
  [5, 4, 7, 8],
  [6, 7, 4, 5]]
- Матриця ризику R (0-10):
  [[3, 5, 8, 6],
  [4, 3, 5, 7],
  [5, 6, 3, 4]]
- Матриця часу доставки T (год):
  [[12, 18, 25, 20],
  [15, 12, 20, 22],
  [18, 20, 10, 15]]

- Склади: 6
- Пункти призначення: 8
- Запаси по складах: [120, 140, 100, 90, 130, 110]
- Потреби пунктів призначення: [80, 75, 95, 60, 85, 70, 100, 100]
- Матриця вартості C (грн):
  [[6, 8, 5, 9, 7, 6, 4, 8],
  [5, 7, 6, 8, 5, 5, 6, 9],
  [7, 6, 8, 7, 6, 7, 5, 6],
  [6, 5, 6, 9, 7, 6, 8, 7],
  [5, 8, 7, 6, 7, 5, 6, 5],
  [7, 6, 6, 7, 8, 6, 7, 6]]
- Матриця ризику R (0-10):
  [[4, 5, 4, 6, 3, 5, 3, 4],
  [5, 4, 3, 4, 4, 3, 5, 4],
  [4, 4, 5, 3, 5, 4, 3, 3],
  [3, 5, 4, 4, 3, 5, 4, 5],
  [4, 3, 5, 4, 4, 4, 3, 4],
  [5, 4, 3, 5, 5, 3, 4, 3]]
- Матриця часу доставки T (год):
  [[15, 20, 12, 18, 14, 17, 13, 16],
  [13, 17, 14, 15, 16, 12, 14, 18],
  [18, 14, 17, 16, 15, 18, 13, 17],
  [14, 16, 15, 17, 14, 16, 15, 15],
  [15, 17, 16, 14, 18, 14, 16, 14],
  [16, 14, 13, 16, 17, 15, 14, 15]]

Мікроаналіз чутливостей

Були обрані такі конфігурації:
:

Сценарій	α	β	γ	Пріоритет
Економічний	1.0	0.2	0.1	Мін. вартості
Швидка доставка	0.3	0.1	1.0	Мін. часу
Максимальна безпека	0.2	1.0	0.1	Мін. ризику
Збалансований (нейтральний)	1.0	1.0	1.0	Компроміс

α – ваговий коефіцієнт для вартості доставки,
β – ваговий коефіцієнт для ризику маршруту,
·γ – ваговий коефіцієнт для часу транспортування.

Мікроаналіз чутливостей

Результати тестувань:

Сценарій – економічний

$$\alpha = 1.0, \beta = 0.2, \gamma = 0.1$$

Набір даних	Версія	Z	Коментар
Мала	PuLP	2551	
Мала	SciPy	2551	
Мала	Власна	3974	
Велика	PuLP	5112	Статус infeasible, результат перевищує попит
Велика	SciPy	-	No solution, статус infeasible
Велика	Власна	6935	

Сценарій – максимальна безпека

$$\alpha = 0.2, \beta = 1.0, \gamma = 0.1$$

Набір даних	Версія	Z	Коментар
Мала	PuLP	2220	
Мала	SciPy	2220	
Мала	Власна	3678	
Велика	PuLP	3931	Статус infeasible, результат перевищує попит
Велика	SciPy	-	No solution, статус infeasible
Велика	Власна	5324	

Сценарій – швидка доставка

$$\alpha = 0.3, \beta = 0.1, \gamma = 1.0$$

Набір даних	Версія	Z	Коментар
Мала	PuLP	5861	
Мала	SciPy	5861	
Мала	Власна	8775	
Велика	PuLP	10606	Статус infeasible, результат перевищує попит
Велика	SciPy	-	No solution, статус infeasible
Велика	Власна	13301.50	

Сценарій – збалансований (нейтральний)

$$\alpha = 1.0, \beta = 1.0, \gamma = 1.0$$

Набір даних	Версія	Z	Коментар
Мала	PuLP	8180	
Мала	SciPy	8180	
Мала	Власна	12810	
Велика	PuLP	15525	Статус infeasible, результат перевищує попит
Велика	SciPy	-	No solution, статус infeasible
Велика	Власна	19645	

Мікроаналіз чутливостей - висновки

- Зміна вагових коефіцієнтів суттєво впливає на структуру оптимального плану і значення цільової функції.
- Збалансовані сценарії дають найвищі значення Z, що підтверджує: одночасна оптимізація суперечливих критеріїв вимагає компромісів.
- Найефективніші (у плані витрат) рішення – у економічному сценарії, але вони можуть не задовольняти безпекові чи часові обмеження гуманітарних місій.
- Вибір значень має не лише математичне, але й етичне значення в контексті гуманітарних місій.
- Надмірна концентрація на одному критерії може спричинити небажані наслідки: дешеві, але повільні або небезпечні маршрути.

Реалізація симплексу - висновки

Малий набір даних:

- Реалізації PuLP, SciPy дали ідентичне значення цільової функції.
- Власна реалізація в деяких випадках обирала інший базис, що призводило до трохи гіршого результату

Великий набір даних:

SciPy (linprog з методом HiGHS):

- Повертає статус infeasible – не існує оптимального розв′язку.
- Не видає жодного плану доставки.
- Не дозволяє проаналізувати ситуацію – повертає лише res.success = False.

PuLP (розв’язувач CBC):

- Повертає розв’язок незважаючи на статус "Infeasible".
- План доставки порушує обмеження на потреби – перевищено обсяги доставки до деяких пунктів.
- Навіть за додаткової перевірки на жорсткість обмежень розв’язувач не може знайти оптимального розв’язку.

Власна реалізація:

- Успішно знаходить допустиме рішення.
- Не порушує жодного з обмежень.

Дякую за увагу!