

Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра математики

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: «**ОПТИМІЗАЦІЙНІ ЕКОЛОГО-ЕКОНОМІЧНІ
МОДЕЛІ**»

Виконала: студентка 4-го року
навчання
освітньої програми «Прикладна
математика»,
спеціальності 113 Прикладна
математика

Куриленко Олександра Андріївна

Керівник: Чорней Р. К.

доцент, кандидат фіз.-мат. наук

Рецензент:

Кваліфікаційна робота захищена

з оцінкою _____

Секретар ЕК _____

(підпис)

« _____ » _____ 20__ р.

Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра математики

ЗАТВЕРДЖУЮ

Зав.кафедри математики,
доцент, кандидат фіз.-мат. наук

_____ *Чорней Р.К.*
(підпис)

“ _____ ” _____ 2025

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

для кваліфікаційної роботи
студенту 4-го курсу, факультету інформатики
Куриленко Олександрі Андріївні

Тема: «Оптимізаційні еколого-економічні моделі»

Зміст кваліфікаційної роботи:

Анотація

1. Вступ

2. Основи еколого-економічного моделювання

3. Міжгалузева балансова модель Леонт'єва-Форда

4. Програмна реалізація оптимізаційної моделі

Висновки

Список літератури

Дата видачі “ _____ ” _____ 2025 Керівник _____
(підпис)

Завдання отримав _____
(підпис)

Графік підготовки кваліфікаційної роботи до захисту

Графік узгоджено « _____ » _____ 2024р.

№ з/п	Перелік робіт	Термін виконання етапу	Підпис наукового керівника	Дата ознайомлення наукового керівника	Примітка
1.	Отримання теми кваліфікаційної роботи.	17.09.2024			
2.	Ознайомлення з темою кваліфікаційної роботи.	24.09.2024			
3.	Розробка плану та структури роботи.	27.09.2024			
4.	Робота з науковою літературою, опис основних означень. Написання вступу та анотації.	7.10.2024			
5.	Дослідження моделі Леонтєва-Форда.	21.10.2024			
6.	Робота над текстовим оформленням теоретичної частини та одержаних результатів.	18.02.2025			
7.	Попередній аналіз кваліфікаційної роботи. Виправлення помилок.	04.04.2025			
8.	Попередній захист кваліфікаційної роботи.	14.05.2025			
9.	Захист кваліфікаційної роботи.	04.06.2025			

Науковий керівник _____
(ПІБ)

Виконавець кваліфікаційної роботи _____
(ПІБ)

ЗМІСТ

АНОТАЦІЯ	4
ВСТУП	5
РОЗДІЛ 1. Основи еколого-економічного моделювання	7
1.1 Історія становлення та розвиток екологічної економіки	7
1.2 Теоретичні основи побудови та класифікації еколого-економічних моделей	9
1.3 Балансова модель Леонтьєва та її екологічне розширення	11
РОЗДІЛ 2. Міжгалузєва балансова модель Леонтьєва-Форда	13
2.1 Класична модель Леонтьєва-Форда	13
2.2 Оптимізаційні моделі на основі моделі Леонтьєва-Форда	15
2.3 Модель з урахуванням ризиків виникнення техногенних аварій .	21
РОЗДІЛ 3. Програмна реалізація оптимізаційної моделі	23
3.1 Пояснення класу EcoEconDisasterModel	23
3.2 Ініціалізація класу	23
3.3 Методи перевірки даних	24
3.4 Розрахункові методи моделі	26
3.5 Методи для формування задачі оптимізації	27

3.6	Методи для аналізу результатів	28
3.7	Основний метод запуску моделі	29
3.8	Допоміжна функція для конвертування даних	30
3.9	Приклад використання моделі	30
3.10	Ідейні межі застосування моделі	32
ВИСНОВКИ		34
ЛІТЕРАТУРА		35
ДОДАТОК А		39
	Вихідний код програми	39

АНОТАЦІЯ

Дана робота присвячена дослідженню та аналізу еколого-економічних моделей, що враховують взаємозв'язок економічної діяльності та навколишнього середовища. Основна увага приділяється задачі мінімізації витрат економіки на екологічну складову за умов, коли виробництво кожного виду продукції та утилізація забруднювачів можуть здійснюватися різними способами. Розглядається концепція "права на забруднення", що є ефективним інструментом для стимулювання скорочення шкідливих викидів.

У теоретичному розділі узагальнено підхід еколого-економічного моделювання, що ґрунтується на моделі Леонтьєва-Форда. Наведено ключові поняття та методи, які використовуються для аналізу взаємодії економіки та довкілля. Основна увага приділяється оцінці різних стратегій регулювання впливу виробництва на навколишнє середовище: зокрема, розглянуто підхід із запровадженням плати за забруднення та підхід, що враховує ризики техногенних катастроф. Для кожного з варіантів визначено цільові функції та відповідні обмеження в рамках оптимізаційних задач.

Практична частина містить програмну реалізацію алгоритму на Python, що дозволяє розрахувати оптимальні плани виробництва та утилізації забруднювачів. Розглядається також модель Леонтьєва-Форда з урахуванням ризиків техногенних катастроф.

Отримані результати мають як теоретичну, так і прикладну цінність. Розглянуті моделі можна використовувати для прийняття обґрунтованих рішень у сфері природокористування. Водночас алгоритм, реалізований в межах дослідження, дозволяє оцінювати наслідки виробничої діяльності для довкілля та приймати більш виважені управлінські рішення з урахуванням екологічних факторів.

Ключові слова: еколого-економічна модель, оптимізація, право на забруднення, модель Леонтьєва-Форда, міжгалузевий баланс, мінімізація витрат, оптимальні стратегії виробництва.

ВСТУП

Екологічні проблеми, пов'язані із промисловим забрудненням, стають дедалі актуальнішими в сучасному світі. Головне протиріччя між економікою та екологією виникає тоді, коли розвиток економіки розглядають як збереження нинішньої структури виробництва та зайнятості, де інтереси захисту довкілля часто відсуваються на другий план заради економічної вигоди [1]. Це свідчить про необхідність формування збалансованого підходу до взаємодії економіки та природи.

В умовах зростання кількості екологічних викликів та обмеженості природних ресурсів особливої ваги набувають математичні моделі, які дозволяють кількісно описати складні зв'язки між економічною діяльністю та станом довкілля. Такі моделі є важливим інструментом для прогнозування наслідків управлінських рішень, оцінки екологічних ризиків та обґрунтування ефективної екологічної політики [2]. Вони допомагають не тільки краще розуміти взаємозалежності у системі «економіка – природа», а й знаходити реалістичні способи їх узгодження.

Особливої важливості ці зрушення набувають в Україні, адже в умовах повномасштабної війни країна зазнала значних екологічних втрат. Воєнні дії призвели до масштабного забруднення повітря, ґрунтів і водних ресурсів, руйнування екосистем та зростання ризиків техногенних катастроф. Крім того, Україна вже тривалий час стикається з екологічними викликами, спричиненими промисловою діяльністю та техногенними аваріями [3]. Усі ці аспекти зумовлюють нагальну потребу у впровадженні сучасних підходів до оцінки та регулювання впливу виробництва на довкілля, а також підкреслюють **актуальність дослідження**.

Завдання роботи полягає у вивченні теоретичних основ еколого-економічного моделювання та їх застосуванні для побудови оптимізаційної моделі. У другому розділі детально розглядається **об'єкт дослідження** — міжгалузєва модель Леонт'єва–Форда, що включає основне та допоміжне виробництво і дозволяє формалізувати взаємозв'язок між економічною системою та

навколишнім середовищем. У межах цієї моделі досліджується задача мінімізації витрат на екологічну складову, що виникають у процесі виробництва різних видів продукції та знешкодження забруднювачів. Особливу увагу приділено розширенню моделі шляхом введення плати за забруднення, а також врахуванню ризиків техногенних катастроф, які можуть додатково впливати на екологічні та економічні результати.

Метою роботи є побудова та реалізація оптимізаційної моделі в середовищі програмування Python, яка дозволяє знаходити ефективні виробничі та екологічні стратегії з урахуванням впливу на довкілля. Сформульована задача мінімізації витрат економіки на екологічну складову формалізується у вигляді задачі лінійного програмування та розв'язується за допомогою симплекс-методу.

РОЗДІЛ 1. Основи еколого-економічного моделювання

1.1 Історія становлення та розвиток екологічної економіки

Екологічна економіка як окрема наукова галузь почала формуватись у 1970-х роках, коли науковці усвідомили, що традиційна економіка не враховує обмежений екологічний потенціал природного середовища. Основні макроекономічні моделі ігнорували екологічні обмеження.

Прикладом таких моделей є модель Солоу (1956) [4]:

$$Y(t) = K(t)^\alpha \cdot [A(t)L(t)]^{1-\alpha}$$

де:

$Y(t)$ — обсяг виробництва (випуск) у момент часу t ;

$K(t)$ — обсяг капіталу;

$L(t)$ — обсяг праці;

$A(t)$ — рівень технологічного прогресу;

$\alpha \in (0, 1)$ — еластичність випуску за капіталом.

Дана модель не враховує обмеженість природних ресурсів і вплив довкілля на економічне зростання.

В умовах глобалізації світової економіки на перший план почав виходити пріоритет забезпечення повноцінного майбутнього для світового суспільства, що зумовило активне включення екологічного аспекту у макроекономічні моделі і сприяло формуванню нової дисципліни — екологічної економіки [5].

Варто зазначити, що каталізатором для переосмислення ролі довкілля стали й історичні події. Після Другої світової війни світове господарство вступило в період стрімкої індустріалізації: відбулося різке зростання енергоспоживання, активна урбанізація, розширення виробництва та широке викори-

стання викопного палива. Наслідком цього процесу стало суттєве підвищення рівня забруднення довкілля, накопичення промислових відходів і деградація природних систем [6]. У 1950–60-х роках це вилилося в перші масштабні екологічні катастрофи, зокрема: Великий смог у Лондоні 1952 року.

Однією з перших спроб адаптації традиційних економічних моделей до нових викликів стала модифікована модель Солоу, запропонована Партха Дасгупта та Джеффри Гілом, яка враховує вичерпність природних ресурсів [7]. У цій моделі виробництво залежить не лише від капіталу та праці, а й від доступних запасів ресурсів:

$$Y(t) = F(K(t), L(t), R(t))$$

де:

$Y(t)$ — обсяг випуску продукції у момент часу t ;

$K(t)$ — обсяг капіталу;

$L(t)$ — обсяг праці;

$R(t)$ — запаси природних ресурсів.

Таким чином, модель демонструє, що сталий економічний розвиток неможливий без врахування обмеженості ресурсної бази. Вона закладає основу для подальших досліджень у сфері екологічного моделювання та формує підґрунтя для політик сталого розвитку [7].

У XXI столітті еколого-економічний підхід набуває дедалі більшої актуальності. Людство стикається з дедалі серйознішими екологічними викликами — такими як зміна клімату, забруднення водних ресурсів і повітря, накопичення відходів та зменшення запасів природних ресурсів. У таких умовах інтеграція екологічних чинників в економічну політику та виробничі стратегії постає як необхідна умова переходу до більш усвідомленої моделі господарювання, що передбачає раціональне використання ресурсів, підвищення ефективності та суттєве зменшення антропогенного навантаження [8].

1.2 Теоретичні основи побудови та класифікації еколого-економічних моделей

Розглянемо основні поняття та підходи еколого-економічного моделювання, що формують теоретичну основу дослідження.

Еколого-економічна модель — це прикладна економіко-математична модель, що використовується для аналізу та прогнозування взаємодії між економічною діяльністю та навколишнім середовищем [9]. Вона дозволяє оцінювати вплив виробництва і споживання на екологічні системи, а також зворотний вплив стану довкілля на економіку. Такі моделі зазвичай будуються у формі системи математичних рівнянь, що відображають кількісні взаємозв'язки між екологічними та економічними змінними. Їх застосування дозволяє формувати науково обґрунтовані стратегії сталого розвитку [10] з урахуванням обмеженості природних ресурсів і необхідності підтримання екологічної рівноваги.

Побудова еколого-економічних моделей спирається на низку ключових принципів і підходів. Серед них — системний підхід [11], що розглядає економічні та природні компоненти як єдину взаємопов'язану систему; врахування просторової та часової неоднорідності процесів [12]; використання балансових рівнянь для опису потоків ресурсів [10]; застосування методів оптимізації в умовах ресурсних або екологічних обмежень [13]; поєднання знань з різних наукових галузей.

Міждисциплінарний характер еколого-економічного моделювання проявляється у практичному поєднанні знань з економіки, екології, математики та інформатики [14]. Зокрема, економічні знання застосовуються для аналізу прибутковості підприємств з урахуванням екологічних податків або моделювання впливу субсидій на стале виробництво. Екологія надає параметри для моделювання — наприклад, гранично допустимі рівні концентрації забруднювачів, коефіцієнти самовідновлення екосистем або норми споживання природних ресурсів. Математика використовується для формалізації цих взаємозв'язків: створення систем рівнянь, оптимізаційних задач або дина-

мічних моделей. Інформатика забезпечує реалізацію цих моделей у вигляді комп'ютерних симуляцій — зокрема, за допомогою мов програмування, геоінформаційних систем для просторового аналізу або спеціалізованого ПЗ для сценарного прогнозування.

За рівнем охоплення еколого-економічні моделі можуть поділятися на глобальні, регіональні та локальні. Прикладом глобальної моделі може бути GCAM (Global Change Assessment Model), в якій моделюються взаємозв'язки між енергетикою, кліматом, землекористуванням та економікою у світовому масштабі [15]. Регіональною моделлю можна вважати LEAP, яка широко використовується для моделювання енергетичних систем і викидів парникових газів на рівні країн або окремих регіонів [16].

За характером процесів моделі можуть поділятися на детерміновані — де результати повністю визначаються початковими умовами та параметрами, — та стохастичні, які враховують випадковість і невизначеність у змінних або впливах, що діють на систему [17].

Враховуючи різноманіття еколого-економічних моделей, постає логічне питання стосовно їх складності та характеристик. Варто почати з двох найскладніших для моделювання - системно-динамічних та стохастичних моделей.

Системно-динамічні моделі, такі як World3 [18] чи GUMBO [19], базуються на системах нелінійних диференціальних рівнянь і описують взаємодію між численними змінними: населенням, виробництвом, ресурсами, забрудненням та іншими. Їх складність полягає у потребі врахування запізнь у часі, зворотних зв'язків, меж зростання, а також у високій чутливості до параметрів, що робить моделювання дуже залежним від вихідних припущень.

Стохастичні моделі враховують випадковість і невизначеність у вхідних даних або параметрах, таких як коливання цін, кліматичні аномалії чи ризики технологічних аварій. Вони будуються з використанням розподілів ймовірностей, імовірнісних сценаріїв або методу Монте-Карло [23], що значно ускладнює як математичну реалізацію, так і інтерпретацію результатів. Крім

того, для їх надійного калібрування потрібні великі обсяги якісних статистичних даних.

Прикладом більш простої, але водночас прикладної моделі є LEAM — Land Use Evolution and Impact Assessment Model. LEAM базується на алгоритмічному підході, де поведінка землекористування моделюється як послідовність змін, зумовлених просторовими правилами, соціально-економічними параметрами та політиками регіонального розвитку [16]. Модель використовує сіткове подання простору та оцінює ймовірність зміни використання кожної клітинки залежно від сусідніх умов. Завдяки простоті реалізації та інтеграції з геоінформаційними системами LEAM є зручним інструментом для місцевого екологічного планування та оцінки впливу інфраструктурних рішень на довкілля.

Проте в порівнянні з балансовими та оптимізаційними підходами LEAM має обмежені можливості щодо кількісного аналізу економічних взаємозв'язків і не дозволяє точно оцінювати ресурсні потоки чи фінансові витрати в межах виробничих систем. Її алгоритмічна структура більше орієнтована на просторове планування, а не на глибоку економіко-екологічну оцінку. Саме тому темою даного дослідження було обрано модель Леонт'єва–Форда як більш формалізований і аналітично потужний інструмент моделювання взаємодії між економікою та довкіллям.

1.3 Балансова модель Леонт'єва та її екологічне розширення

У загальному вигляді міжгалузєва балансова модель описує взаємозв'язки між різними галузями економіки через систему лінійних рівнянь. Вона показує, як продукція однієї галузі використовується іншими для забезпечення власного випуску і широко застосовується для кількісного аналізу й прогнозування економічних систем [20].

Однією з найвідоміших моделей у сфері економічного моделювання є між-

галузева балансова модель Леонт'єва, що також має назву модель «витрати-випуск». Згодом модель Леонт'єва була доповнена екологічним компонентом і трансформована у модель Леонт'єва – Форда. Доповнена модель дозволяє враховувати не лише виробничі взаємозв'язки, а й екологічні фактори, зокрема вплив економічної діяльності на навколишнє середовище та витрати на його очищення, так звана оплата «права на забруднення» [13]. У такій моделі економіка доповнюється допоміжними галузями, що відповідають за утилізацію й нейтралізацію шкідливих відходів, а також вводяться параметри, що відображають обсяги забруднення на одиницю продукції кожної галузі.

Математично еколого-економічні моделі здебільшого подаються у вигляді систем рівнянь:

$$\mathbf{x} = \mathbf{Ax} + \mathbf{y}, \quad \mathbf{e} = \mathbf{Ex},$$

де \mathbf{x} — вектор обсягів випуску, \mathbf{A} — матриця прямих витрат, \mathbf{y} — вектор кінцевого попиту, \mathbf{E} — матриця екологічного впливу, а \mathbf{e} — обсяги забруднення [10]. Таке формалізоване подання дає змогу здійснювати кількісний аналіз сценаріїв розвитку, оцінювати вплив галузей на довкілля та розробляти збалансовані політики сталого управління.

Проте, важливо зазначити, що балансова модель Леонт'єва сама по собі не є оптимізаційною: розв'язок її матричного рівняння отримується без формулювання функції мети та без урахування конкретних обмежень, що можуть виникати в реальному виробництві [21]. Такий підхід дозволяє лише описати взаємозалежності між галузями та оцінити потребу в обсягах виробництва для задоволення кінцевого попиту, однак не дозволяє знайти оптимальну виробничу структуру.

Для подолання цих обмежень застосовують оптимізаційні міжгалузеві моделі, які розширюють аналітичні можливості класичних балансових моделей [22]. Вони враховують не лише умови міжгалузевого балансу, а й додаткові економічні, технологічні або екологічні обмеження, а також включають функцію мети — наприклад, мінімізацію витрат, максимізацію прибутку або

скорочення рівня забруднення.

Математично оптимізаційні міжгалузеві моделі можуть бути подані у вигляді:

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{за умови} \quad \mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{y}, \quad \mathbf{G}\mathbf{x} \leq \mathbf{b}, \quad \mathbf{x} \geq 0,$$

де \mathbf{x} — вектор обсягів випуску, \mathbf{A} — матриця прямих витрат, \mathbf{y} — вектор кінцевого попиту, $f(\mathbf{x})$ — функція мети (наприклад, загальні витрати, викиди або їх поєднання), $\mathbf{G}\mathbf{x} \leq \mathbf{b}$ — додаткові обмеження (ресурсні, екологічні, технологічні) [23].

Такий підхід дозволяє не лише описати виробничі зв'язки між галузями, а й оптимізувати структуру випуску відповідно до обраного критерію ефективності за наявності обмежень.

Практично модель Леонтьєва–Форда може використовуватись для розрахунку впливу змін у структурі споживання на обсяги забруднення, для визначення необхідного обсягу утилізації шкідливих речовин при заданому попиті, або як інструмент формування політик екологічного регулювання на галузевому рівні.

РОЗДІЛ 2. Міжгалузєва балансова модель Леонтьєва-Форда

2.1 Класична модель Леонтьєва-Форда

Цей розділ носить реферативний характер і містить теоретичний матеріал з роботи [24].

Першою міжгалузєвою моделлю, що описує взаємозв'язок економіки та навколишнього середовища, є модель Леонтьєва-Форда, яка складається із двох груп галузей: основного виробництва - галузі матеріального виробництва, та допоміжного виробництва - галузі, що знищує шкідливі відходи [10].

Модель Леонтьєва–Форда не лише відображає економічні зв'язки, але й дозволяє враховувати екологічні наслідки господарської діяльності [20]. Вона показує, що досягнення економічного зростання можливе лише за умови належного контролю за обсягами забруднення та ефективного функціонування секторів утилізації відходів.

Математично модель виражається системою рівнянь:

$$\begin{cases} x^1 = A_{11}x^1 + A_{12}x^2 + y^1, \\ x^2 = A_{21}x^1 + A_{22}x^2 - y^2. \end{cases}$$

Ця система рівнянь також може бути записана в матричній формі [25]:

$$\begin{pmatrix} x^1 \\ x^2 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} x^1 \\ x^2 \end{pmatrix} + \begin{pmatrix} y^1 \\ -y^2 \end{pmatrix}$$

або за допомогою одиничної матриці E , яка використовується для спрощення структури моделі:

$$\left(E - \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \right) \cdot \begin{pmatrix} x^1 \\ x^2 \end{pmatrix} = \begin{pmatrix} y^1 \\ -y^2 \end{pmatrix}$$

У моделі застосовуються такі позначення:

$x^1 = (x_i^1)_{i \in I}$ — вектор обсягів виробництва продукції, де $I = \{1, 2, \dots, m\}$ — множина індексів видів продукції.

$x^2 = (x_j^2)_{j \in J}$ — вектор обсягів знищених забруднювачів, де $J = \{1, 2, \dots, n\}$ — множина індексів видів забруднювачів.

$y^1 = (y_i^1)_{i \in I}$ — вектор обсягів кінцевого споживання продукції.

$y^2 = (y_j^2)_{j \in J}$ — вектор гранично допустимих обсягів незнищених забруднювачів.

$A_{11} = (a_{ij}^{11})_{i \in I, j \in I}$ — матриця коефіцієнтів прямих витрат продукції i на виробництво одиниці продукції j ;

$A_{12} = (a_{ij}^{12})_{i \in I, j \in J}$ — матриця коефіцієнтів витрат продукції i на знищення одиниці забруднювача j ;

$A_{21} = (a_{ij}^{21})_{i \in J, j \in I}$ — матриця коефіцієнтів утворення забруднювача i при виробництві одиниці продукції j ;

$A_{22} = (a_{ij}^{22})_{i \in J, j \in J}$ — матриця коефіцієнтів утворення забруднювача i при знищенні одиниці забруднювача j .

Усі коефіцієнти a_{ij} лежать у відрізку $[0; 1]$. Необхідною умовою для існування невід'ємного розв'язку є дотримання нерівності $A_{21}y^1 \geq y^2$.

Модель Леонтьєва–Форда слугує інструментом для аналізу взаємодії між виробничими процесами та утворенням і нейтралізацією забруднень у системі міжгалузевого балансу.

Її економічна інтерпретація передбачає невід'ємність усіх змінних, а саме: $x_i^1 \geq 0$, $x_j^2 \geq 0$, $y_i^1 \geq 0$, $y_j^2 \geq 0$ для кожного $i \in I$ та $j \in J$.

Структура міжгалузевих взаємозв'язків у цій моделі задається блочною матрицею:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

2.2 Оптимізаційні моделі на основі моделі Леонтьєва–Форда

Цей розділ носить реферативний характер і містить теоретичний матеріал з роботи [24].

Економіко-математичні моделі використовуються для обґрунтованого планування виробництва, вибору ефективних технологій та підтримки пріоритетних галузей. У зв'язку з цим виникає потреба у вдосконаленні моделі Леон-

тьєва–Форда та перетворенні її на оптимізаційну.

Оптимізаційні міжгалузеві моделі поєднують баланс ресурсів із мінімізацією витрат або впливу на довкілля. Вони дозволяють знаходити найкращі варіанти розвитку з урахуванням економічних і екологічних критеріїв.

Розглянемо оптимізаційну еколого-економічну модель з альтернативними способами виробництва продукції та знищення забруднювачів. Цільова функція описує витрати на ці процеси. Перед формулюванням моделей окреслимо змінні та базові теоретичні положення. Кожен вид продукції $i \in I$ може виготовлятися кількома способами $\varphi_i \in P_i$, при цьому кожен спосіб відповідає лише одному продукту. Аналогічно, кожен забруднювач $j \in J$ знищується способами $\psi_j \in Q_j$, що також однозначно відповідають конкретному виду забруднення.

Введемо наступні позначення:

$x_{i\varphi_i}^1$ — обсяг вироблення продукції i способом φ_i ;

$x_{j\psi_j}^2$ — обсяг знищення забруднювача j способом ψ_j ;

y_i^1 — обсяг кінцевого споживання продукції i ;

y_j^2 — допустимий обсяг незнищення забруднювача j (гранична норма незнищення забруднювача);

$a_{ij\varphi_j}^{11}$ — коефіцієнт прямих затрат продукції i на вироблення одиниці продукції j способом φ_j ;

$a_{ij\psi_j}^{12}$ — коефіцієнт затрат продукції i на знищення одиниці забруднювача j способом ψ_j ;

$a_{ij\varphi_j}^{21}$ — коефіцієнт випуску забруднювача i на вироблення одиниці продукції j способом φ_j ;

$a_{ij\psi_j}^{22}$ — коефіцієнт випуску забруднювача i на знищення одиниці забруднювача j способом ψ_j .

c_j — плата за кожну одиницю незнищеного забруднювача j ;

$c_j\psi_j$ — вартість знищення одиниці забруднювача j способом ψ_j ;

Запишемо наступну цільову функцію:

$$\sum_{k \in J} c_k \left(\sum_{i \in I} \sum_{\varphi_j \in P_j} a_{kj\varphi_j}^{21} x_{j\varphi_j}^1 + \sum_{g \in J} \sum_{\psi_g \in Q_g} \sigma_{kg\psi_g} x_{k\psi_g}^2 \right) \rightarrow \min,$$

у якій перший доданок відображає вартісну оцінку обсягу забруднювачів, що утворюються в процесі виробництва продукції, а другий — вартісний еквівалент забруднень, які виникають у результаті процесу їх знищення, з урахуванням як обсягів утилізованих забруднювачів, так і витрат на їхнє усунення, де

$$\sigma_{kg\psi_g} = \begin{cases} a_{kg\psi_g}^{22}, & g \neq k, \\ a_{kk\psi_k}^{22} - 1 + \frac{c_k\psi_k}{c_k}, & g = k. \end{cases}$$

Вимога забезпечення кінцевого споживання продукції призводить до таких обмежень:

$$\sum_{j \in I} \sum_{\varphi_j \in P_j} (\delta_{ij} - a_{ij\varphi_j}^{11}) x_{j\varphi_j}^1 - \sum_{g \in J} \sum_{\psi_g \in Q_g} a_{ig\psi_g}^{12} x_{g\psi_g}^2 \geq y_i^1, \quad i \in I; \quad (2.1)$$

Щоб обсяг незнищених забруднювачів не перевищував допустиму норму, вводяться такі обмеження:

$$-\sum_{j \in I} \sum_{\varphi_j \in P_j} a_{kj\varphi_j}^{21} x_{j\varphi_j}^1 + \sum_{g \in J} \sum_{\psi_g \in Q_g} (\delta_{kg} - a_{kg\psi_g}^{22}) x_{g\psi_g}^2 \geq -y_k^2, \quad k \in J; \quad (2.2)$$

В обмеженнях (2.1) та (2.2) δ_{ij} є позначенням дельти Кронекера:

$$\delta_{ij} = \begin{cases} 1, & \text{якщо } i = j, \\ 0, & \text{якщо } i \neq j. \end{cases}$$

оскільки елементи квадратної матриці, визначені дельта-функцією Кронекера, утворюють одиничну матрицю [26].

Згідно з економічною інтерпретацією моделі Леонтьєва–Форда, додатково накладаються умови невід’ємності змінних $x_{j\varphi_j}^1$ та $x_{g\psi_g}^2$, які доповнюють раніше сформульовані обмеження:

$$\begin{aligned} x_{j\varphi_j}^1 &\geq 0, & j \in I, \varphi_j \in P_j; \\ x_{g\psi_g}^2 &\geq 0, & g \in J, \psi_g \in Q_g. \end{aligned} \quad (2.3)$$

Нижче подано теорему, що стосується вищезазначеної задачі:

Теорема 1. [24] В оптимальному плані за умови додатності y^1 та невід’ємності y^2 виробляється кожен вид продукції та знищується кожен вид забруднювачів, причому вироблення продукту може відбуватися лише одним способом, і відповідно знищення забруднювача може відбуватися лише одним способом.

Властивість 1. У моделі оптимізації екологічно-економічного балансу вводяться додаткові змінні:

$$\Delta y_i^1 \geq 0 \quad - \text{надлишок кінцевої продукції над попитом } y_i^1, \quad i \in I,$$

$$\Delta y_k^2 \geq 0 \quad - \text{надлишок знищених забруднювачів над граничною нормою } y_k^2, \quad k \in J.$$

Ці змінні дозволяють перетворити нерівності задачі у рівності. У кожному рівнянні модель приймає вигляд:

$$\sum_{j \in I} \sum_{\varphi_j \in P_j} \left(\delta_{ij} - a_{ij\varphi_j}^{11} \right) x_{\varphi_j}^1 - \sum_{g \in G} \sum_{\psi_g \in Q_g} a_{ig\psi_g}^{12} y_g^2 x_{\psi_g}^2 - \Delta y_i^1 = y_i^1, \quad i \in I,$$

де в оптимальному плані активними можуть бути лише змінні $x_{\varphi_j}^1$, для яких $x_{\varphi_j}^1 > 0$.

Звідси випливає, що кожен вид продукції i повинен вироблятися принаймні одним способом:

$$\sum_{\varphi_i \in P_i} x_{i\varphi_i}^1 > 0, \quad i \in I. \quad (2.4)$$

Аналогічно, для знищення забруднювачів маємо:

$$-\sum_{j \in I} \sum_{\varphi_j \in P_j} a_{kj\varphi_j}^{21} x_{\varphi_j}^1 + \sum_{g \in J} \sum_{\psi_g \in Q_g} (\delta_{kg} - a_{kg\psi_g}^{22}) y_g^2 x_{\psi_g}^2 - \Delta y_k^2 = -y_k^2, \quad k \in J,$$

що за умови досяжності y_k^2 призводить до виконання:

$$\sum_{j \in I} \sum_{\varphi_j \in P_j} a_{kj\varphi_j}^{21} x_{\varphi_j}^1 \geq y_k^2,$$

а отже, для кожного $k \in J$ виконується:

$$\sum_{\psi_k \in Q_k} x_{k\psi_k}^2 > 0. \quad (2.5)$$

Це означає, що кожен вид забруднювача також знищується принаймні одним способом.

Введемо позначення $N(A)$ — кількість змінних у множині A , а $N(I) + N(J)$ — максимальна кількість додатних змінних в оптимальному плані. Звідси випливає, що в кожній сумі (2.4) та (2.5) може бути додатною лише одна змінна.

Наслідок. Оскільки в оптимальному плані лише одним способом виробляється кожен вид продукції та лише одним способом знищується кожен вид забруднювачів, то всі $\Delta y_i^1 = \Delta y_k^2 = 0$.

Означення 1. Позначимо через x_1^* та x_2^* вектори, що відповідають оптимальному плану моделі. Кожен елемент цих векторів відображає інтенсивність використання певного способу основного або допоміжного виробництва.

На основі вибраних у плані способів формуємо матрицю коефіцієнтів A^* , яка має блочну структуру, подібну до матриці A еколого-економічної моделі міжгалузевого балансу.

Однак на відміну від A , матриця A^* містить лише коефіцієнти тих способів виробництва продукції або знищення забруднювачів, які увійшли до оптимального плану. Матриця A^* зберігає економіко-математичні властивості, аналогічні до властивостей початкової матриці A .

Система рівнянь виглядає наступним чином:

$$\begin{cases} x_1^* = A_{11}^* x_1^* + A_{12}^* x_2^* + y_1^0, \\ x_2^* = A_{21}^* x_1^* + A_{22}^* x_2^* - y_2^0. \end{cases}$$

Розглянемо наступну теорему:

Теорема 2.[24] За допустимих змін векторів y_1^0 та y_2^0 базис оптимального плану, від якого залежить вибір «кращих» способів, не змінюється.

Пояснення. Базис оптимального плану в задачі лінійного програмування зберігається сталим доти, доки змінні, що входять до плану, залишаються невід'ємними. Оскільки ці змінні відповідають обраним «кращим» способам виробництва або знищення забруднювачів, то за умови їх невід'ємності структура базису не порушується.

Таким чином, допустимі варіації правих частин не призводять до зміни базису оптимального розв'язку.

2.3 Модель з урахуванням ризиків виникнення техногенних аварій

Цей розділ носить реферативний характер і містить теоретичний матеріал з роботи [27].

Розглянемо модель з урахуванням ризиків техногенних катастроф у виробничій діяльності. Маємо цільовою функцією, що виражає витрати економіки на виробництво та знищення забруднювачів, за умови, що будь-який виробничий спосіб може зазнати «техногенної екологічної катастрофи». Введемо додаткові змінні для визначення моделі з урахуванням ризиків техногенних катастроф:

Нехай кожен вид продукції $i \in I$ виробляється кількома способами $\varphi_i \in P_i$, де кожен спосіб забезпечує випуск лише одного продукту. Аналогічно, кожен вид забруднювача $j \in J$ знищується кількома способами $\psi_j \in Q_j$, причому кожен спосіб призначений для усунення лише одного забруднювача.

p_{φ_i} — ймовірність “техногенної екологічної катастрофи” при застосуванні способу $\varphi_i \in P_i$ виробництва продукції $i \in I$;

p_{ψ_j} — ймовірність “техногенної екологічної катастрофи” при застосуванні способу $\psi_j \in Q_j$ знищення забруднювача $j \in J$;

$b_{j\varphi_i}^1$ — коефіцієнт очікуваного викиду забруднювача $j \in J$ при виробництві одиниці продукції $i \in I$ способом $\varphi_i \in P_i$ у випадку технологічної аварії;

$b_{j\psi_l}^2$ — коефіцієнт очікуваного викиду забруднювача $j \in J$ при знищенні одиниці забруднювача $l \in J$ способом $\psi_l \in Q_l$ у випадку технологічної аварії.

Введемо додаткові позначення::

$$\tilde{a}_{ik\varphi_k}^{11} := (1 - p_{\varphi_k})a_{ik\varphi_k}^{11};$$

$$\tilde{a}_{ij\psi_j}^{12} := (1 - p_{\psi_j})a_{ij\psi_j}^{12};$$

$$\begin{aligned}\tilde{a}_{ji\varphi_i}^{21} &:= (1 - p_{\varphi_i})a_{ji\varphi_i}^{21} + p_{\varphi_i}b_{ji\varphi_i}^1; \\ \tilde{a}_{jl\psi_l}^{22} &:= (1 - p_{\psi_l})a_{jl\psi_l}^{22} + p_{\psi_l}b_{jl\psi_l}^2.\end{aligned}$$

Тоді можна записати цільову функцію, що має наступний вигляд:

$$\sum_{j \in J} c_j \left(\sum_{i \in I} \sum_{\varphi_i \in P_i} a_{ji\varphi_i}^{21} x_{i\varphi_i}^1 + \sum_{l \in J} \sum_{\psi_l \in Q_l} \sigma_{jl\psi_l} x_{l\psi_l}^2 \right) \rightarrow \min,$$

у якій перший доданок відображає обсяг викинутих забруднювачів при виробництві продукції, а другий - обсяг викинутих забруднювачів при їх знищенні мінус обсяг знищених забруднювачів плюс витрати на їх знищення, усе у вартісному вигляді, де

$$\sigma_{j\psi_l} := \begin{cases} \tilde{a}_{jl\psi_l}^{22}, & j \neq l, \\ \tilde{a}_{jl\psi_l}^{22} - 1 + \frac{c_j \psi_j}{c_j}, & j = l. \end{cases}$$

Це дозволяє сформулювати задачу лінійного програмування, задавши цільову функцію як мінімізацію втрат економічної системи.

Вводимо обмеження на обсяги виробництва продукції:

$$\sum_{k \in I} \sum_{\varphi_k \in P_k} (\delta_{ik} - \tilde{a}_{ik\varphi_k}^{11}) x_{k\varphi_k}^1 - \sum_{j \in J} \sum_{\psi_j \in Q_j} \tilde{a}_{ij\psi_j}^{12} x_{j\psi_j}^2 \geq y_i^1, \quad i \in I,$$

та обмеження на обсяги незнищених забруднювачів:

$$-\sum_{i \in I} \sum_{\varphi_i \in P_i} \tilde{a}_{ji\varphi_i}^{21} x_{i\varphi_i}^1 + \sum_{l \in J} \sum_{\psi_l \in Q_l} (\delta_{jl} - \tilde{a}_{jl\psi_l}^{22}) x_{l\psi_l}^2 \geq -y_j^2, \quad j \in J.$$

Наступні обмеження гарантують що обсяги виробництва продукції та знищення забруднювачів будуть невід'ємними:

$$x_{i\varphi_i}^1 \geq 0, \quad i \in I, \varphi_i \in P_i, \quad x_{j\psi_j}^2 \geq 0, \quad j \in J, \psi_j \in Q_j.$$

РОЗДІЛ 3. Програмна реалізація оптимізаційної моделі

3.1 Пояснення класу `EcoEconDisasterModel`

Клас `EcoEconDisasterModel` є розширенням моделі Леонт'єва-Форда, яка враховує ризики технологічних катастроф у процесі виробництва та нейтралізації забруднювачів. Ця модель дозволяє оптимізувати виробничі процеси з урахуванням екологічних обмежень та потенційних катастрофічних подій.

3.2 Ініціалізація класу

```
def __init__(
    self, A11, A12, A21, A22, y1, y2,
    c, c_psi, p_phi, p_psi, b1, b2
):
```

Конструктор класу приймає наступні параметри:

- $A11$ – 3D масив коефіцієнтів прямих витрат продукції i на виробництво одиниці продукції j кожним способом
- $A12$ – 3D масив коефіцієнтів витрат продукції i на нейтралізацію одиниці забруднювача j кожним способом
- $A21$ – 3D масив коефіцієнтів викиду забруднювача i при виробництві одиниці продукції j кожним способом
- $A22$ – 3D масив коефіцієнтів викиду забруднювача i при нейтралізації одиниці забруднювача j кожним способом
- $y1$ – вектор кінцевого споживання

- y_2 – вектор порогів толерантності до забруднення
- c – вектор витрат для ненейтралізованих забруднювачів
- c_psi – матриця витрат на нейтралізацію забруднювачів різними методами
- p_phi – ймовірність катастрофи під час виробництва для кожного методу виробництва
- p_psi – ймовірність катастрофи під час нейтралізації для кожного методу нейтралізації
- b_1 – очікувані викиди забруднювачів під час виробничої катастрофи
- b_2 – очікувані викиди забруднювачів під час катастрофи нейтралізації

3.3 Методи перевірки даних

Метод `is_feasibility_condition_met`

```
@staticmethod
```

```
def is_feasibility_condition_met(A21, y1, y2):
```

Статичний метод, який перевіряє, чи виконується достатня умова для існування невід’ємних рішень згідно з моделлю Леонт’єва-Форда. Розглядаються всі можливі комбінації викидів для кожного забруднювача та перевіряється виконання обмеження:

$$A_{21} \cdot y_1 \geq y_2 \tag{1}$$

Параметри:

- A_{21} – 3D масив коефіцієнтів викиду забруднювача

- y_1 – вектор кінцевого споживання
- y_2 – вектор порогів толерантності

Повертає булеве значення `True`, якщо умова виконується, та `False` в іншому випадку.

Метод `range_is_correct`

```
@staticmethod  
def range_is_correct(matrix):
```

Перевіряє, чи всі значення в матриці знаходяться в діапазоні $[0, 1]$. Використовується для валідації матриць коефіцієнтів, оскільки вони повинні бути в цьому діапазоні для коректності моделі.

Параметри:

- `matrix` – матриця для перевірки

Повертає булеве значення `True`, якщо всі елементи в діапазоні $[0, 1]$, та `False` в іншому випадку.

Метод `validate_data`

```
def validate_data(self):
```

Комплексна перевірка всіх вхідних даних для забезпечення коректного функціонування моделі. Перевіряє достатню умову та діапазони значень матриць. Також перевіряє параметри ризику катастроф:

- Ймовірності катастроф (`p_phi` та `p_psi`) повинні бути в діапазоні $[0, 1]$
- Очікувані викиди від катастроф (`b1` та `b2`) повинні бути невід'ємними

3.4 Розрахункові методи моделі

Метод `get_sigma`

```
def get_sigma(self, g, k, psi_g):
```

Розраховує скоригований коефіцієнт забруднювача (сигма) для задачі оптимізації. Якщо забруднювач і нейтралізатор однакові ($k = g$), враховуються витрати на нейтралізацію.

Формула розрахунку:

$$\sigma_{kg}^{\psi_g} = \begin{cases} a_{kg}^{\psi_g} & \text{якщо } k \neq g \\ a_{kg}^{\psi_g} - 1 + \frac{c_{\psi_g}^g}{c_k} & \text{якщо } k = g \end{cases} \quad (2)$$

Параметри:

- `g` – індекс забруднювача, який нейтралізується
- `k` – індекс забруднювача, для якого обчислюється коефіцієнт
- `psi_g` – метод нейтралізації забруднювача `g`

Метод `add_arrays`

```
@staticmethod
```

```
def add_arrays(result, temp):
```

Допоміжний метод для поелементного додавання двох масивів. Використовується при побудові коефіцієнтів цільової функції.

Параметри:

- `result` – результуючий масив

- `temp` – тимчасовий масив для додавання

Метод `k_delta`

```
@staticmethod
def k_delta(i, j):
```

Реалізує функцію дельти Кронекера, яка повертає 1, якщо $i = j$, та 0 в іншому випадку. Використовується при побудові матриці коефіцієнтів обмежень.

3.5 Методи для формування задачі оптимізації

Метод `get_disaster_adjusted_obj_func_coefs`

```
def get_disaster_adjusted_obj_func_coefs(self):
```

Формує коефіцієнти для цільової функції з урахуванням ризику катастроф. Включає:

- Звичайні витрати на виробництво та нейтралізацію забруднювачів
- Додаткові очікувані витрати від потенційних катастроф під час виробництва
- Додаткові очікувані витрати від потенційних катастроф під час нейтралізації

Цільова функція має форму:

$$\begin{aligned}
 f(x, \psi) = & \sum_{k=1}^m \sum_{j=1}^n \sum_{\phi_j=1}^{s_j} c_k a_{kj}^{\phi_j} x_j^{\phi_j} + \sum_{k=1}^m \sum_{g=1}^m \sum_{\psi_g=1}^{t_g} c_k \sigma_{kg}^{\psi_g} \psi_g^{\psi_g} + \\
 & + \sum_{k=1}^m \sum_{j=1}^n \sum_{\phi_j=1}^{s_j} c_k p_{\phi_j}^j b_{1,kj}^{\phi_j} x_j^{\phi_j} + \sum_{k=1}^m \sum_{g=1}^m \sum_{\psi_g=1}^{t_g} c_k p_{\psi_g}^g b_{2,kg}^{\psi_g} \psi_g^{\psi_g}
 \end{aligned} \tag{3}$$

де дві останні суми відображають очікувані витрати від катастроф.

Метод `get_disaster_adjusted_constraints`

```
def get_disaster_adjusted_constraints(self, obj_coefs):
```

Формує праву частину обмежень нерівності з урахуванням ризику катастроф. Коригує вектори y_1 та y_2 для компенсації потенційних втрат виробництва під час катастроф.

Метод `get_disaster_adjusted_inequalities_coefs`

```
def get_disaster_adjusted_inequalities_coefs(self):
```

Генерує матрицю коефіцієнтів для обмежень нерівності з урахуванням ризику катастроф. Матриця включає:

- Коефіцієнти для виробничих вимог (з урахуванням ймовірності катастроф)
- Коефіцієнти для обмежень забруднення (з урахуванням додаткових викидів від катастроф)
- Одиничну матрицю для обмежень невід'ємності

3.6 Методи для аналізу результатів

Метод `count_methods`

```
def count_methods(self):
```

Підраховує кількість методів виробництва та нейтралізації викидів. Це необхідно для правильного розділення вектора рішення після оптимізації.

Метод `get_results`

```
def get_results(self, obj_coefs, constraints, ineq_coefs):
```

Розв'язує задачу лінійного програмування з використанням методу симплекс (`linprog` з бібліотеки `SciPy`). Параметри задачі:

- `obj_coefs` – коефіцієнти цільової функції, яка мінімізується
- `constraints` – права частина обмежень нерівностей
- `ineq_coefs` – матриця коефіцієнтів обмежень

З отриманого вектора рішення `res.x` виділяються:

- `prod_indexes` – індекси обраних методів виробництва та обсяги продукції
- `emis_indexes` – індекси обраних методів нейтралізації та обсяги нейтралізованих забруднювачів

3.7 Основний метод запуску моделі

Метод `run`

```
def run(self):
```

Виконує всі етапи роботи моделі:

1. Перевіряє коректність вхідних даних через `validate_data()`
2. Формує цільову функцію та обмеження з урахуванням ризику катастроф
3. Розв'язує задачу оптимізації через `get_results()`

4. Виводить результати:

- Мінімальне значення цільової функції (загальна вартість)
- Оптимальний обсяг виробництва кожного продукту та обраний метод
- Оптимальний обсяг нейтралізації кожного забруднювача та обраний метод
- Інформацію про ймовірності катастроф та очікувані втрати для кожного методу
- Очікувані додаткові витрати від технологічних катастроф

3.8 Допоміжна функція для конвертування даних

```
def convert_dict_to_array(data_dict, shape):
```

Конвертує вхідні дані з формату словника у формат 3D масиву NumPy для використання в моделі. Це дозволяє задавати вхідні дані у зручнішому форматі, ніж багатовимірні масиви.

Параметри:

- `data_dict` – словник вхідних даних
- `shape` – кортеж (рядки, стовпці, глибина) для результуючого масиву

3.9 Приклад використання моделі

Моделі дозволяє знаходити оптимальні методи виробництва та нейтралізації забруднювачів з урахуванням не лише прямих витрат, але й потенційних витрат від технологічних катастроф, що є важливим розширенням класичної моделі Леонт'єва-Форда.

У код включено тестові дані для демонстрації застосування моделі:

- Матриці міжгалузевих взаємодій (A11, A12, A21, A22)
- Вектори кінцевого споживання та порогів забруднення (y1, y2)
- Вартісні параметри (c, c_psi)
- Параметри ризику катастроф:
 - Ймовірності катастроф для методів виробництва (p_phi)
 - Ймовірності катастроф для методів нейтралізації (p_psi)
 - Очікувані викиди під час виробничих катастроф (b1)
 - Очікувані викиди під час катастроф нейтралізації (b2)

Вхідні дані виглядають наступним чином:

```
A11 = np.array([[0.94, 0.21], [0.2, 0.43]], [[0.09, 0.29], [0.11, 0.26]])
A12 = np.array([[0.78, 0.95], [0.24, 0.11]], [[0.01, 0.03], [0.13, 0.23]])
A21 = np.array([[0.16, 0.23], [0.39, 0.14]], [[0.02, 0.15], [0.33, 0.12]])
A22 = np.array([[0.41, 0.27], [0.98, 0.17]], [[0.01, 0.03], [0.65, 0.23]])
y1 = np.array([989, 621])
y2 = np.array([57, 25])
c = np.array([76, 73])
c_psi = np.array([[18, 28], [39, 55]])
p_phi = np.array([[0.01, 0.006], [0.014, 0.011]])
p_psi = np.array([[0.01, 0.02], [0.015, 0.025]])
b1 = np.array([
    [[0.3, 0.25], [0.2, 0.15]],
    [[0.4, 0.35], [0.3, 0.25]]
]).reshape(2, 2, 2)
b2 = np.array([
    [[1.1, 1.0], [0.9, 0.8]],
    [[1.0, 0.9], [1.1, 1.0]]
]).reshape(2, 2, 2)
```

Рис. 1: Початкові дані, використані для ілюстрації роботи реалізованої програми

Виведення результатів, створення та запуск оптимізаційного розрахунку здійснюється через ініціалізацію екземпляра класу EcoEconDisasterModel, якому передаються вхідні параметри, після чого викликається метод run(), що забезпечує виконання основного обчислювального процесу.

Вивід програми виглядає наступним чином:

```

Необхідні умови для невід'ємних розв'язків виконуються
Мінімальне значення цільової функції (загальна вартість з урахуванням ризику катастроф): 765450.5203171985
Обсяг виробництва продукту 1 за методом 2: 26913.9711
Ймовірність катастрофи для цього методу: 0.0060
Очікувані втрати виробництва: 161.4838

Обсяг виробництва продукту 2 за методом 2: 14345.1754
Ймовірність катастрофи для цього методу: 0.0110
Очікувані втрати виробництва: 157.7969

Обсяг нейтралізації забруднювача 1 за методом 1: 16888.4895
Ймовірність катастрофи для цього методу: 0.0100
Очікувані втрати нейтралізації: 168.8849

Обсяг нейтралізації забруднювача 2 за методом 2: 8278.1947
Ймовірність катастрофи для цього методу: 0.0250
Очікувані втрати нейтралізації: 206.9549

Очікувані додаткові витрати від технологічних катастроф: 66010.7193
Базові витрати без урахування ризику катастроф: 699439.8010

```

Рис. 2: Результати обчислення лінійної оптимізації, а також розрахунок загальних витрат з урахуванням ризику техногенних катастроф

3.10 Ідейні межі застосування моделі

Тип задачі: Багатовимірна оптимізація з множинними обмеженнями

Складність: $O(n^3)$ для симплекс-методу, де n — кількість змінних

Умови застосування:

- Лінійні залежності між змінними
- Детерміновані параметри системи
- Відомі ймовірності катастроф
- Стабільні технологічні процеси

Обмеження:

- Пам'ять: $O(n^2)$ для зберігання матриць
- Час виконання: $O(n^3)$ для розв'язання задачі

Можливості реалізації:

- Оптимальна робота з:
 - 2 типами продукції
 - 2 типами забруднювачів
 - 2 методами виробництва / нейтралізації для кожного типу

Застосування: Підходить для моделювання еколого-економічних систем середньої складності, де важливо враховувати як економічні, так і екологічні аспекти.

Незважаючи на ефективність моделі, її практичне застосування обмежується швидким зростанням обчислювальної складності при збільшенні кількості змінних. При цьому загальна кількість змінних зростає пропорційно добутку кількості методів і типів об'єктів системи.

ВИСНОВКИ

У дипломній роботі було розглянуто проблему досягнення балансу між економічним зростанням і збереженням природного середовища. Проведено аналіз теоретичних засад еколого-економічного моделювання, здійснено історичний огляд становлення цієї наукової галузі та обґрунтовано актуальність застосування балансових моделей.

Особливу увагу приділено моделі Леонтьєва–Форда, на основі якої сформовано оптимізаційну еколого-економічну модель. Вона враховує різні методи виробництва та очищення, а також дозволяє мінімізувати сукупні витрати на ці процеси завдяки відповідній цільовій функції. Окремо розглянуто модифікацію моделі з урахуванням ризиків техногенних аварій.

Реалізовано практичну задачу мінімізації витрат економіки на екологічну складову з урахуванням обсягів виробництва, процесів нейтралізації шкідливих викидів та ризиків техногенних катастроф. Для цього розроблено програмну реалізацію еколого-економічної моделі Леонтьєва–Форда мовою Python. Проведено аналіз отриманих результатів і перевірено ефективність роботи моделі.

Перспективи подальших досліджень полягають у розвитку динамічних та стохастичних модифікацій моделі, а також у розширенні її застосування для моделювання реальних регіональних еколого-економічних систем.

ЛІТЕРАТУРА

- [1] Онищенко І., Онищенко Є. Теоретичні засади екологічної економіки // *Економіка і регіон*. 2016. № 2. С. 81–84.
- [2] Петровська С. А. *Моделювання сталого розвитку: процесні й еколого-економічні аспекти проблеми*. Суми: Сумський державний університет, 2021.
- [3] Мягченко О. П. *Основи екології: підручник*. Київ: Центр учбової літератури, 2022. 312 с.
- [4] Solow R. M. A Contribution to the Theory of Economic Growth // *The Quarterly Journal of Economics*. 1956. Vol. 70, No. 1. P. 65–94.
- [5] Онищенко А. М. Методологічні основи еколого-економічного моделювання положень Кіотського протоколу // *Економіка та держава*. 2008. № 8. С. 21–25.
- [6] Hjärpe M., Linnér B.-O. *Environmental management since World War II*. Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), 2006. P. 24–34.
- [7] Dasgupta P. S., Heal G. M. The optimal depletion of exhaustible resources // *The Review of Economic Studies*. 1974. Vol. 41 (Symposium on the Economics of Exhaustible Resources). P. 3–28.
- [8] Химинець В. В. *Еколого-економічні проблеми в контексті сталого розвитку Закарпаття*. Ужгород: Ужгородський національний університет, 2004.
- [9] Тадеєв Ю. П. Еколого-економічна модель оптимального керування з лінійною функцією корисності // *Проблеми економіки*. 2013. № 2. С. 284–287.

- [10] Лях І. М., Кляп М. М., Вовканич С. В. Математичні моделі міжгалузевого балансу // *International Scientific Herald*. 2015. № 1. С. 373–380.
- [11] Вергунова І. М. *Системне моделювання в економіці (блок 2)*. Київ: ФОП Корзун Д. Ю., 2013. 106 с.
- [12] Пичура В. І. Методика просторово-часового моделювання агрохімічних показників меліорованих ґрунтів з використанням ГІС та нейротехнологій // *Таврійський науковий вісник*. 2012. № 77. С. 118–123.
- [13] Чорней Н. Б. Еколого-економічна оптимізаційна модель та оплата права на забруднення // *Науковий вісник*. 2015.
- [14] Бубенко П. Т., Димченко О. В., Бурак О. М., Величко В. В., Тітяєв В. В., Єсіна В. О., Матвєєва Н. М., Владимірова М. С., Сухонос М. К., Славута О. І., Волгіна Н. О., Водка Н. В., Гайдєнко С. М., Покуца І. В., Дворкін С. В., Телятник С. В., Лук'янов В. І. *Економіка довкілля і природних ресурсів: навчальний посібник*. Харків: ХНУМГ, 2014. 280 с.
- [15] Calvin K., Patel P., Clarke L., Asrar G., Bond-Lamberty B., Cui R., Di Vittorio A., Dorheim K., Edmonds J., Hartin C., Hejazi M., Horowitz R., Iyer G., Kyle P., Kim S., Link R., McJeon H., Smith S. J., Snyder A., Waldhoff S., Wise M. GCAM v5.1: Representing the linkages between energy, water, land, climate, and economic systems // *Geoscientific Model Development*. 2019. Vol. 12. P. 677–698. DOI: 10.5194/gmd-12-677-2019.
- [16] Sun Z., Deal B. M., Pallathucheril V. G. The land-use evolution and impact assessment model: A comprehensive urban planning support system // *URISA Journal*. 2009. Vol. 21, No. 1. P. 57–68.
- [17] Грабовецький Б. Є. Класифікація методів прогнозування // *Основи економічного прогнозування: навчальний посібник*. Вінниця: ВФ ТАНГ, 2000.
- [18] Nebel A., Kling A., Willamowski R., Schell T. Recalibration of limits to growth: An update of the World3 model // *Journal of Industrial Ecology*. 2023. Vol. 27, No. 6. P. 1342–1355.

- [19] Boumans R., Costanza R., Farley J., Wilson M. A., Portela R., Rotmans J., Villa F., Grasso M. Modeling the dynamics of the integrated earth system and the value of global ecosystem services using the GUMBO model // *Ecological Economics*. 2002. Vol. 41, No. 3. P. 529–560.
- [20] Хрущ Л. З., Коржевська О. П. Розширення міжгалузевої еколого-економічної моделі Леонт'єва – Форда // *Бізнес Інформ*. 2012. № 3. С. 75–78.
- [21] Пономаренко О. Ш., Перестюк М. О., Бурим В. М. Статична модель «витрати–випуск» Леонт'єва // *Основи математичної економіки*. Київ: Інформтехніка, 1995. С. 163–194.
- [22] Кутковецький В. Я. *Оптимізація потоків міжгалузевого балансу*. Миколаїв, 2016. 5 с.
- [23] Буреннікова Н. В., Кузнецова І. С., Салов О. О. *Оптимізаційні методи та моделі: навчальний посібник*. Вінниця: ВНТУ, 2019. 242 с.
- [24] Чорней Н. Б., Чорней Р. К. Еколого-економічна оптимізаційна модель з урахуванням оплати права на забруднення // *Сучасні проблеми математичного моделювання, прогнозування та оптимізації*. Київ–Кам'янець–Подільський, 2004. С. 75–79.
- [25] Ляшенко І. М. *Економіко-математичні методи та моделі сталого розвитку*. Київ: Вища школа, 1999. 236 с.
- [26] Clapham C., Nicholson J. Kronecker delta // *The Concise Oxford Dictionary of Mathematics*. 2009.
- [27] Чорней Р. К. Оптимізаційна еколого-економічна модель з урахуванням технологічних аварій // *Наукові записки НаУКМА. Фізико-математичні науки*. 2014. Т. 152. С. 52–55.
- [28] `scipy.optimize.linprog`.
- [29] Nelder A. J., Mead R. A Simplex Method for Function Minimization // *The Computer Journal*. 1965. No. 4. P. 308–313.

[30] ACME Lab. *The Simplex Method*. Brigham Young University.

[31] Mudit Gupta. Simplex Method for Linear Programming // *Medium*.

ДОДАТОК А

Вихідний код програми

```
import numpy as np
import itertools
from scipy.optimize import linprog

class EcoEconDisasterModel:
    def __init__(self, A11, A12, A21, A22, y1, y2, c, c_psi, p_phi, p_psi, b1, b2):
        """
        Ініціалізація еколого-економічної моделі з додатковими параметрами ризику
        → технологічних катастроф.

        Параметри:
        - A11, A12, A21, A22: 3D масиви numpy, що представляють міжгалузеві матриці
        → та матриці забруднення
        - y1: вектор кінцевого споживання
        - y2: вектор порогів толерантності до забруднення
        - c: вектор витрат для ненейтралізованих забруднювачів
        - c_psi: матриця витрат на нейтралізацію забруднювачів різними методами
        - p_phi: ймовірність катастрофи під час виробництва для кожного методу
        → виробництва
        - p_psi: ймовірність катастрофи під час нейтралізації для кожного методу
        → нейтралізації
        - b1: очікувані викиди забруднювачів під час виробничої катастрофи
        - b2: очікувані викиди забруднювачів під час катастрофи нейтралізації
        """
        self.A11 = A11
        self.A12 = A12
        self.A21 = A21
        self.A22 = A22
        self.y1 = y1
        self.y2 = y2
        self.c = c
        self.c_psi = c_psi

        # Нові параметри для ризику технологічних катастроф
        self.p_phi = p_phi # Ймовірність катастрофи для методів виробництва
        self.p_psi = p_psi # Ймовірність катастрофи для методів нейтралізації
        self.b1 = b1 # Очікувані викиди під час виробничої катастрофи
        self.b2 = b2 # Очікувані викиди під час катастрофи нейтралізації
```

```

@staticmethod
def is_feasibility_condition_met(A21, y1, y2):

    rows, cols = len(A21), len(A21[0])
    res_matrix = [list(itertools.product(*A21[i])) for i in range(rows)]
    res_matrix = np.array(res_matrix).reshape((-1, rows, cols))
    return np.all(res_matrix @ y1 >= y2)

@staticmethod
def range_is_correct(matrix):
    return np.all(np.logical_and(matrix >= 0, matrix <= 1))

def get_sigma(self, g, k, psi_g):
    return self.A22[k][g][psi_g] if k != g else self.A22[k][g][psi_g] - 1 +
    ↪ self.c_psi[k][psi_g] / self.c[k]

@staticmethod
def add_arrays(result, temp):
    if not result:
        return temp
    return [result[i] + temp[i] for i in range(len(result))]

def get_disaster_adjusted_obj_func_coefs(self):
    result = []
    for k in range(len(self.c)):
        coefs_prod = []
        for j in range(len(self.A21[k])):
            for phi_j in range(len(self.A21[k][j])):
                coef = self.A21[k][j][phi_j] * self.c[k]
                if j < self.p_phi.shape[0] and phi_j < self.p_phi.shape[1]:
                    if k < self.b1.shape[0] and j < self.b1.shape[1] and phi_j <
                    ↪ self.b1.shape[2]:
                        coef += self.p_phi[j][phi_j] * self.b1[k][j][phi_j] *
                        ↪ self.c[k]
                coefs_prod.append(coef)

        coefs_ext = []
        for g in range(len(self.A22[k])):
            for psi_g in range(len(self.A22[k][g])):
                coef = self.get_sigma(g, k, psi_g) * self.c[k]
                if g < self.p_psi.shape[0] and psi_g < self.p_psi.shape[1]:
                    if k < self.b2.shape[0] and g < self.b2.shape[1] and psi_g <
                    ↪ self.b2.shape[2]:
                        coef += self.p_psi[g][psi_g] * self.b2[k][g][psi_g] *
                        ↪ self.c[k]
                coefs_ext.append(coef)

```

```

        result = self.add_arrays(result, coefs_prod + coefs_ext)
    return result

def get_disaster_adjusted_constraints(self, obj_coefs):
    y1_adjusted = np.zeros_like(self.y1)
    for i in range(len(self.y1)):
        max_prob = 0.0
        for j in range(len(self.A11[i])):
            for phi_j in range(len(self.A11[i][j])):
                if j < self.p_phi.shape[0] and phi_j < self.p_phi.shape[1]:
                    max_prob = max(max_prob, self.p_phi[j][phi_j])

        adjustment_factor = 1.0 - max_prob
        if adjustment_factor <= 0:
            adjustment_factor = 0.01
        y1_adjusted[i] = self.y1[i] / adjustment_factor

    y2_adjusted = -self.y2
    y = np.concatenate((y1_adjusted, y2_adjusted))
    return np.concatenate((y, np.zeros(len(obj_coefs))))

@staticmethod
def k_delta(i, j):
    return int(i == j)

def get_disaster_adjusted_inequalities_coefs(self):
    coefs_res = []

    for i in range(len(self.y1)):
        coefs_y1 = []
        for j in range(len(self.A11[i])):
            for phi_j in range(len(self.A11[i][j])):
                disaster_prob = 0.0
                if j < self.p_phi.shape[0] and phi_j < self.p_phi.shape[1]:
                    disaster_prob = self.p_phi[j][phi_j]

                coefficient = self.kronecker_delta(i, j) * (1 - disaster_prob) -
                    ↪ self.A11[i][j][phi_j]
                coefs_y1.append(coefficient)

        for g in range(len(self.A12[i])):
            for psi_g in range(len(self.A12[i][g])):
                disaster_prob = 0.0
                if g < self.p_psi.shape[0] and psi_g < self.p_psi.shape[1]:
                    disaster_prob = self.p_psi[g][psi_g]

```

```

        coefficient = -self.A12[i][g][psi_g] * (1 - disaster_prob)
        coefs_y1.append(coefficient)

    coefs_res.append(coefs_y1)

for k in range(len(self.y2)):
    coefs_y2 = []

    for j in range(len(self.A21[k])):
        for phi_j in range(len(self.A21[k][j])):
            coefficient = -self.A21[k][j][phi_j]

            if j < self.p_phi.shape[0] and phi_j < self.p_phi.shape[1]:
                if k < self.b1.shape[0] and j < self.b1.shape[1] and phi_j <
                    ↪ self.b1.shape[2]:
                    coefficient -= self.p_phi[j][phi_j] *
                    ↪ self.b1[k][j][phi_j]

            coefs_y2.append(coefficient)

    for g in range(len(self.A22[k])):
        for psi_g in range(len(self.A22[k][g])):
            if k == g:
                coefficient = self.kronecker_delta(k, g) -
                ↪ self.A22[k][g][psi_g]
            else:
                coefficient = -self.A22[k][g][psi_g]

            if g < self.p_psi.shape[0] and psi_g < self.p_psi.shape[1]:
                if k < self.b2.shape[0] and g < self.b2.shape[1] and psi_g <
                    ↪ self.b2.shape[2]:
                    coefficient -= self.p_psi[g][psi_g] *
                    ↪ self.b2[k][g][psi_g]

            coefs_y2.append(coefficient)
    coefs_res.append(coefs_y2)

identity_matrix = np.identity(len(coefs_res[0]))
return np.concatenate((coefs_res, identity_matrix))

def count_methods(self):
    return [len(i) for i in self.A11[0]] + [len(j) for j in self.A22[0]]

def get_results(self, obj_coefs, constraints, ineq_coefs):
    res = linprog(obj_coefs, A_ub=-ineq_coefs, b_ub=-constraints, method='highs')

```

```

if not res.success:
    print(f"\nОптимізація не вдалася: {res.message}")
    return None, None, None

subarrays_sizes = self.count_methods()
split_points = np.cumsum(subarrays_sizes)[: -1]
subarrays = np.split(res.x, split_points)

prod_indexes = [(np.nonzero(array)[0][0] if np.any(array > 1e-8) else 0,
                 array[np.nonzero(array)[0][0]] if np.any(array > 1e-8) else
                 ↪ 0)
                for array in subarrays[:len(self.A11)]]

emis_indexes = [(np.nonzero(array)[0][0] if np.any(array > 1e-8) else 0,
                 array[np.nonzero(array)[0][0]] if np.any(array > 1e-8) else
                 ↪ 0)
                for array in subarrays[len(self.A11):]]

return res.fun, prod_indexes, emis_indexes

def validate_data(self):
    basic_validation = (self.data_is_sufficient(self.A21, self.y1, self.y2) and
                       all(map(self.range_is_correct, [self.A11, self.A12,
                                                       ↪ self.A21, self.A22])))

    p_phi_valid = np.all((self.p_phi >= 0) & (self.p_phi < 1))
    p_psi_valid = np.all((self.p_psi >= 0) & (self.p_psi < 1))

    b1_valid = np.all(self.b1 >= 0)
    b2_valid = np.all(self.b2 >= 0)

    return basic_validation and p_phi_valid and p_psi_valid and b1_valid and
    ↪ b2_valid

def run(self):
    if not self.validate_data():
        print("\nНеобхідні умови для невід'ємних розв'язків не виконуються або
        ↪ параметри ризику катастроф недійсні")
        return

    print("\nНеобхідні умови для невід'ємних розв'язків виконуються")

    obj_coefs = self.get_disaster_adjusted_obj_func_coefs()
    constraints = self.get_disaster_adjusted_constraints(obj_coefs)
    ineq_coefs = self.get_disaster_adjusted_inequalities_coefs()

```

```

func_val, prod_indexes, emis_indexes = self.get_results(obj_coefs,
↳ constraints, ineq_coefs)

if func_val is None:
    print("\nНе вдалося знайти оптимальне рішення")
    return

print('\nМінімальне значення цільової функції (загальна вартість з
↳ урахуванням ризику катастроф):', func_val)

for i, (idx, val) in enumerate(prod_indexes):
    if val > 1e-8: # Виведення лише значних ненульових значень
        print(f"\nОбсяг виробництва продукту {i + 1} за методом {idx + 1}:
↳ {val:.4f}")
        if i < self.p_phi.shape[0] and idx < self.p_phi.shape[1]:
            disaster_prob = self.p_phi[i][idx]
            print(f" Ймовірність катастрофи для цього методу:
↳ {disaster_prob:.4f}")
            print(f" Очікувані втрати виробництва: {disaster_prob *
↳ val:.4f}")

for j, (idx, val) in enumerate(emis_indexes):
    if val > 1e-8:
        print(f"Обсяг нейтралізації забруднювача {j + 1} за методом {idx +
↳ 1}: {val:.4f}")
        if j < self.p_psi.shape[0] and idx < self.p_psi.shape[1]:
            disaster_prob = self.p_psi[j][idx]
            print(f" Ймовірність катастрофи для цього методу:
↳ {disaster_prob:.4f}")
            print(f" Очікувані втрати нейтралізації: {disaster_prob *
↳ val:.4f}")

expected_disaster_costs = 0
for k in range(len(self.c)):
    for i, (idx_i, val_i) in enumerate(prod_indexes):
        if val_i > 1e-8:
            if i < self.p_phi.shape[0] and idx_i < self.p_phi.shape[1]:
                if k < self.b1.shape[0] and i < self.b1.shape[1] and idx_i <
↳ self.b1.shape[2]:
                    expected_disaster_costs += self.p_phi[i][idx_i] *
↳ self.b1[k][i][idx_i] * val_i * self.c[k]

    for j, (idx_j, val_j) in enumerate(emis_indexes):
        if val_j > 1e-8:
            # Витрати на катастрофу нейтралізації (якщо розміри параметрів
↳ дозволяють)

```

```

        if j < self.p_psi.shape[0] and idx_j < self.p_psi.shape[1]:
            if k < self.b2.shape[0] and j < self.b2.shape[1] and idx_j <
                ↪ self.b2.shape[2]:
                    expected_disaster_costs += self.p_psi[j][idx_j] *
                        ↪ self.b2[k][j][idx_j] * val_j * self.c[k]

print(f"Очікувані додаткові витрати від технологічних катастроф:
    ↪ {expected_disaster_costs:.4f}")
print(f"Базові витрати без урахування ризику катастроф: {func_val -
    ↪ expected_disaster_costs:.4f}")

def convert_dict_to_array(data_dict, shape):
    result = np.zeros(shape)

    for i in data_dict:
        if isinstance(data_dict[i], dict):
            for j in data_dict[i]:
                if isinstance(data_dict[i][j], dict):
                    for k in data_dict[i][j]:
                        if i-1 < shape[0] and j-1 < shape[1] and k-1 < shape[2]:
                            result[i-1][j-1][k-1] = data_dict[i][j][k]
                else:
                    if i-1 < shape[0] and j-1 < shape[1]:
                        result[i-1][j-1] = data_dict[i][j]

    return result

if __name__ == "__main__":
    A11 = np.array([[0.94, 0.21], [0.3, 0.51]], [[0.08, 0.30], [0.09, 0.35]])
    A12 = np.array([[0.78, 0.95], [0.29, 0.15]], [[0.02, 0.04], [0.11, 0.21]])
    A21 = np.array([[0.16, 0.23], [0.33, 0.18]], [[0.01, 0.17], [0.35, 0.14]])
    A22 = np.array([[0.41, 0.27], [0.89, 0.14]], [[0.02, 0.04], [0.66, 0.22]])
    y1 = np.array([989, 621])
    y2 = np.array([57, 25])
    c = np.array([81, 77])
    c_psi = np.array([[18, 28], [39, 55]])

    p_phi = np.array([[0.01, 0.006], [0.014, 0.011]])
    p_psi = np.array([[0.01, 0.02], [0.015, 0.025]])

    b1 = np.array([
        [[0.3, 0.25], [0.2, 0.15]], # Вплив на забруднювач 1
        [[0.4, 0.35], [0.3, 0.25]] # Вплив на забруднювач 2
    ]).reshape(2, 2, 2)

    b2 = np.array([

```

```
[[[1.1, 1.0], [0.9, 0.8]]], # Вплив на забруднювач 1
[[[1.0, 0.9], [1.1, 1.0]]] # Вплив на забруднювач 2
]).reshape(2, 2, 2)
```

```
disaster_model = EcoEconDisasterModel(A11, A12, A21, A22, y1, y2, c, c_psi, p_phi,
→ p_psi, b1, b2)
disaster_model.run()
```

```
second_model = EcoEconDisasterModel(A11, A12, A21, A22, y1, y2, c, c_psi, p_phi,
→ p_psi, b1, b2)
second_model.run()
```