Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра математики факультету інформатики

Передобумовлення для лінійних обернених задач

Текстова частина до курсової роботи

за спеціальністю «Комп'ютерні науки»- 122

Керівник курсової роботи

Кандидат фіз.-мат. наук,

Доцент

Крюкова Г. В.

_____

(підпис)

"____" _____ 2023 р.

Виконав студент КН-3:

Столяров В.С.

"____" _____ 2023 р.

Київ – 2023

ЗМІСТ

«КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра математики факультету інформатики

ЗАТВЕРДЖУЮ

Зав. кафедри математики,

д. ф.-м. н. Б. В. Олійник

_____

(підпис)

„____”_____2023 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

Студенту Столярову В.С. факультету інформатики 3 курсу

Тема: Передобумовлення для лінійних обернених задач

Зміст ТЧ до курсової роботи:

Календарний план
Вступ
Частина 1: Лінійні обернені задачі
Частина 2: Методи передобумовлення
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі “____” _____ 2023 р. Керівник _____

(підпис)

Завдання отримав _____

(підпис)

ТЕМА: «Передобумовлення для лінійних обернених задач»

Календарний план виконання роботи:

№	Назва етапу курсового проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	20.01. 2023	
2.	Огляд наукової літератури за темою роботи	02.02. 2023	
3.	Створення плану для написання теоретичної частини	07.03. 2023	
4.	Початок написання теоретичної частини	15.03. 2023	
5.	Написання основної частини курсової роботи	15.04. 2023	
6.	Написання програмного коду для практичної частини курсової роботи	30.04. 2023	
7.	Завершення написання теоретичної частини	13.05. 2023	
8.	Оформлення слайдів для доповіді	14.05.2023	
9.	Захист курсової роботи	23.05.2023	

Студент Столяров С.В. _____

Керівник Крюкова Г.В. _____
“_____” _____2023 р.

Анотація

Ця курсова робота досліджує лінійні обернені задачі, зосереджуючись на проблемах, пов’язаних з тим, що вони некоректно поставлені, і рішеннях, запропонованих методами регуляризації та ітераційними методами. Робота заглиблюється в норми регуляризації L1 і L2, демонструючи їх корисність у перетворенні некоректно поставлених задач у правильно поставлені. Далі обговорюються методи спряженого градієнта та проксимального градієнта як ефективні ітераційні рішення регуляризованих лінійних обернених задач. У статті також представлено передобумовлення як техніку для підвищення ефективності ітераційних методів, надаючи практичний приклад, який порівнює швидкості збіжності методів спряженого градієнта з попередньою умовою та без попередньої умови. Мета цієї роботи полягає в тому, щоб забезпечити повне розуміння стратегій, доступних для вирішення лінійних обернених задач, з особливим наголосом на методах передобумовлення.

Ключові слова: лінійні обернені задачі, некоректні задачі, регуляризація, метод спряженого градієнта, метод проксимального градієнта, передобумовлення.

Вступ

У сучасному світі, інформаційні технології і комп'ютерні системи зіграли значну роль у розвитку науки та індустрії. Значна кількість проблем, які раніше вважались нерозв'язними через обмежені обчислювальні можливості, зараз можуть бути розв'язані завдяки потужним комп'ютерам та ефективним алгоритмам. Одним з таких класів проблем є лінійні обернені задачі, які мають широке застосування в різних галузях, таких як обробка сигналів, оптимізація, керування ресурсами, медична діагностика та багатьох інших.

Ця курсова робота присвячена дослідженню методів розв'язання лінійних обернених задач з використанням передобумовлювачів для підвищення швидкості збіжності алгоритмів та забезпечення більш точних результатів. Вибір цієї теми є актуальним, оскільки розв'язання лінійних обернених задач є важливою складовою в багатьох наукових та практичних застосуваннях, а вивчення та застосування передобумовлювачів може значно підвищити ефективність цих розв'язків. У зв'язку з постійним розвитком технологій, особливо у галузі медичної діагностики, розвідки корисних копалин, та відновлюваної енергетики, актуальність вирішення обернених лінійних задач посилюється. Поступове збільшення розмірності даних та зміна вимог до точності розв'язання також вносять свій вклад у актуальність дослідження передобумовлення.

Метою даної роботи є розгляд перед обумовлення як методу для підвищення ефективності розв'язання обернених лінійних задач та покращення точності отриманих результатів.

Отже, завдання цієї курсової роботи :

- схарактеризувати обернені лінійні задачі;
- проаналізувати основні методи розв'язання лінійних обернених задач, зокрема ітераційні алгоритми, та їх особливості;
- обґрунтувати а значимість застосування перед обумовлення для підвищення ефективності розв'язання обернених лінійних задач.

- розкрити практичний аспект використання перед обумовлення на прикладі розв'язання лінійної оберненої задачі з використанням алгоритму спряжених градієнтів.

- виявити та оцінити вплив перед обумовлення на швидкість збіжності та точність результатів на основі проведених чисельних експериментів.

Об’єктом дослідження є процеси розв'язання лінійних систем рівнянь.

Предметом дослідження є вивчення властивостей та ефективності методів перед обумовлення для покращення процесу розв'язання обернених лінійних задач, зосереджуючись на використанні алгоритму спряжених градієнтів.

Розділ 1: Лінійні обернені задачі

1.1 Визначення лінійних обернених задач

Обернені задачі відіграють значну роль у різних наукових та інженерних галузях, оскільки включають процес визначення основних причин, параметрів або властивостей системи на основі наявної інформації про її спостережувані ефекти, результати або вимірювання. Це відрізняє їх від прямих задач, які зосереджуються на прогнозуванні наслідків або результатів системи або ж моделі за відомих причин або параметрів. Тобто у прямих задачах відома система або ж модель, і по ній треба розрахувати дані, а в обернених навпаки – треба реконструювати модель на основі набору даних. Цю відмінність між задачами демонструє рисунок 1.1.

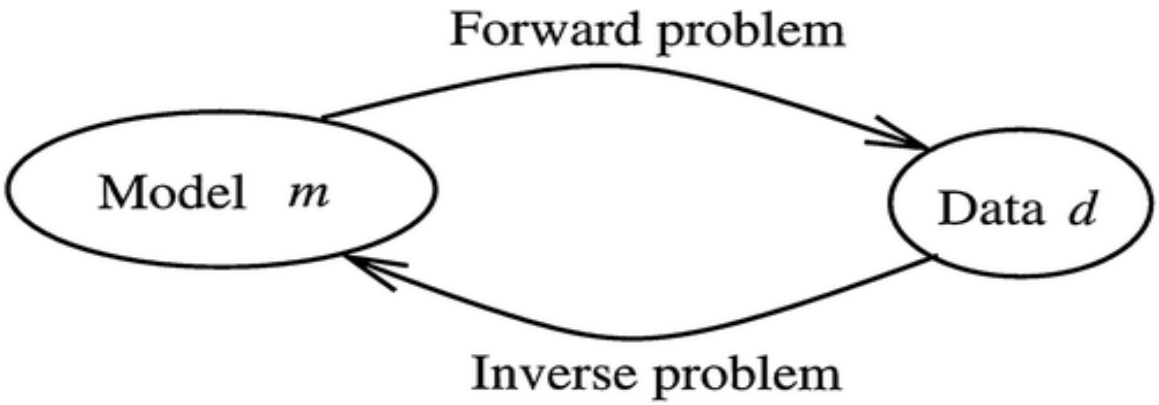


Рисунок 1.1

Математично обернена задача може бути описана як :

$$A(x) = b, \qquad (1.1)$$

де b – це дані, отримані зі спостережень;

x – це невідомі параметри моделі;

A – це функція, що робить пряме відображення невідомих параметрів моделі на обстежувані дані.

У випадку лінійного прямого відображення, тобто співвідношення між параметрами моделі та отриманими даними є лінійним, і отримані дані можна

виразити як лінійну комбінацію параметрів моделі, то така обернена задача називається лінійною та математично може бути описана як :

$$Ax = b, \qquad (1.2)$$

де b – це дані, отримані зі спостережень, які можна представити у вигляді вектору;

x – це невідомі параметри моделі, які також можна представити у вигляді вектору;

A – це матриця, що характеризує пряме відображення. У випадку не лінійного відношення між параметрами моделі та обстежуваними даними, обернена задача є не лінійною.

Концепція обернених задач у математиці та фізиці бере свій початок у 19 столітті, хоча сам термін «обернена задача» широко не використовувався до 20 століття. Одним із найвідоміших випадків розв'язування оберненої задачі було відкриття планети Нептун у нашій сонячній системі в ніч з 23 на 24 вересня 1846 року. Тоді астрономи виявили цю планету на основі математичних розрахунків її передбачуваної позиції через спостережувані пертурбації на орбіті планети Уран. Проте систематичне вивчення обернених задач і використання самого терміну «обернена задача» почало по-справжньому розвиватися в середині 20-го століття, особливо в галузях геофізики та медичної візуалізації. Експоненціальне зростання обчислювальної потужності за останні кілька десятиліть дозволило вирішувати більші та складніші обернені задачі, що було практично нездійсненним в минулому. Також розвиток машинного навчання та технологій штучного інтелекту надав нові інструменти для вирішення зворотних проблем. Ці методи часто можуть знаходити закономірності у великих наборах даних, які неможливо було б виявити людині, що призводить до нових ідей і рішень.

1.2 Некоректно поставлена задача

Для того, щоб зрозуміти, що таке некоректно поставлена задача, спочатку дамо визначення терміну «Коректно поставлена задача». Отже, термін коректно поставленої задачі був введений у 20 столітті французьким математиком Жаком

Соломоном Адамаром. Він визначив 3 критерії для будь-якої математичної задачі, які вона може мати :

1. Розв’язок : розв’язок (s) існує для кожної точки даних(d), для кожного d релевантного цій задачі.
2. Унікальний розв’язок: s унікальний для всіх d; для кожного d існує щонайбільше один значення з s.
3. Стійкий розв’язок: s залежить неперервно від d (маленька зміна в d буде вести до маленької зміни в s; і велика зміна в d буде вести пропорційно до більшої зміни в s).

Якщо поставлена задача відповідає усім цим трьом критеріям – то така задача називається коректно поставленою, у іншому випадку – це некоректно поставлена задача. Більшість задач в математичному аналізі, інженерії та математиці є коректно поставленими. Для прикладу візьмемо $f(x) = x^2 + x$. Для кожного дійсного числа x, $x^2 + x$ дійсне та визначене. Немає місця для двозначності; кожне вхідне значення k дасть рівно один розв’язок який буде дорівнювати $k^2 + k$. Ця функція неперервна, і збільшення різниці між точками даних, в даному випадку x призведе до збільшення різниці між значеннями f(x). Тобто ця задача є коректно поставленою.

Що стосується некоректно поставлених задач, то наприклад багато диференціальних рівнянь першого порядку та обернених задач є некоректно поставленими задачами. Велика кількість обернених задач є некоректно поставленими, оскільки або вони не мають розв’язку для кожної точки даних, або цей розв’язок не унікальний, або він не стійкий, тобто неперервний.

Класичним прикладом є обернена задача теплообміну, де розподіл температури поверхні твердого тіла виводиться з інформації про внутрішню площу поверхні. Хоча пряме рівняння теплопровідності (за допомогою якого можна отримати внутрішню теплопровідність з даних поверхні) добре визначене частковими похідними, обернена задача не є стійкою. Найменші зміни в даних температури поверхні можуть призвести до як завгодно великих відмінностей у розрахованому внутрішньому розподілі тепла. Це пояснюється тим, що обернена задача є дуже чутливою до шуму та помилок у вхідних даних, які можуть бути посилені під час процесу реконструкції розподілу

температури поверхні. Рівняння теплоти – це рівняння дифузії, яке означає, що високочастотні компоненти розподілу температури мають тенденцію швидко затухати з часом. Отже, при спробі реконструювати розподіл температури на поверхні за внутрішніми вимірюваннями, високочастотні компоненти важко точно відновити, і будь-які невеликі помилки в даних можуть призвести до великих помилок в оцінених високочастотних компонентах. Нестабільність оберненої теплової задачі може мати практичні наслідки в реальних застосуваннях, наприклад, неточності реконструкції температури або труднощі в проектуванні ефективних систем охолодження.

Найпоширенішою причиною того, що обернена задача - це некоректно поставлена задача є те, що вона не задовольняє критерію стійкості або ж неперервності , який є третім критерієм коректно поставленої задачі. Це означає, що невеликі збурення або зміни у вхідних даних можуть призвести до як завгодно великих відмінностей у розв’язках, роблячи задачу нестійкою. У багатьох практичних застосуваннях вхідні дані (наприклад, вимірювання або спостереження) схильні до шуму, помилок або невизначеностей. Якщо обернена задача нестабільна, ці невеликі збурення у вхідних даних можуть бути значно посилені в розв’язанні, що призведе до ненадійних або неточних результатів.

Отже, оскільки на практиці вхідні дані для обернених лінійних задач часто містять шум або помилки, то формулу (1.2) можна точніше представити як :

$$Ax + e = b. \qquad (1.3)$$

Тут e – це помилки або шум при вимірюванні даних, які можна представити у вигляді вектору. Це представлення враховує шум або невизначеності, присутні в даних реального світу, роблячи проблему більш реалістичною.

Для усунення нестабільності та некоректної постановки обернених задач часто використовуються методи регуляризації та інші методи для стабілізації розв’язку, обмеження простору розв’язку та підвищення точності та надійності результатів.

1.3 Прийоми регуляризації для стабілізації лінійних обернених задач

З точки зору функціонального аналізу обернена задача представлена відображенням між метричними просторами. Нехай H і K – це 2 гілбертові простори і A : H → K – це лінійний обмежений оператор.

Тоді розглянемо рівняння (1.2) як :

$$Ax = b_\delta , \tag{1.4}$$

де b_δ, $b \in K$ і $||b- b_\delta||_K \leq \delta$.

Тут b представляє точні невідомі дані, а b_δ – це доступні, неточні дані. Тоді знаходження x, що задовольняє наведене вище рівняння, коли задані b_δ та A, і є лінійною оберненою задачею асоційованою з рівнянням (1.2). Оскільки вищезазначена задача є некоректно поставленою, то унікальність може бути відновлена за допомогою псевдообернення Мура-Пенроуза $x^+ = A^+b$ визначеного як мінімальна норма розв'язку задачі

$$\min_{x \in H} ||Ax - b||_K^2. \tag{1.5}$$

Проте оператор A^+ зазвичай не обмежений, таким чином, щоб забезпечити неперервну залежність розв'язку від даних до задачі (1.5) додається регуляризуючий вираз R(x) :

$$\min_{x \in H} ||Ax - b||_K^2 + \lambda R(x), \tag{1.6}$$

де λ – це параметр, що визначає важливість доданка;

R(x) – це регуляризуючий вираз, який може приймати різні форми в залежності від того який прийом регуляризації буде використано.

Отже регуляризацією можна назвати додавання додаткової інформації (регуляризуючого виразу), щоб задовольнити третій критерій Адамара для коректно поставлених задач.

Існує два основних метода регуляризації : це регресія LASSO або ж l1 регуляризація та гребенева регресія або ж регуляризація Тихонова або ж l2 регуляризація. Їхня відмінність у регуляризуючому виразі, що вони додають, у l1 регуляризації – це абсолютне значення величини, або ж модуль, а в регуляризації Тихонова це квадрат величини.

Тепер замість узагальненої формули (1.5) використаємо гребеневу регресію :

$$\min_{x\in H}\{||Ax-b||_K^2+\lambda||x||_H^2\}, \tag{1.7}$$

у якій унікальний мінімізатор задано як

$$x_\delta^\lambda=(A^*A+\ \lambda I)^{-1}A^*b_\delta\ , \tag{1.8}$$

де A^* позначає приєднану матрицю до A.

Вирішальним кроком у наведеному вище алгоритмі – це вибір параметра регуляризації $\lambda=\lambda(\delta,b_\delta)$, як функцію рівня шуму δ і даних b_δ таким чином що

$$\lim_{\delta\to 0}||x_\delta^{\lambda(\delta,b_\delta)}-x^+||_H=0\ , \tag{1.9}$$

тобто регуляризований розв'язок $x_\delta^{\lambda(\delta,b_\delta)}$ збігається до узагальненого розв'язку $x^+=A^+b$ коли шум δ прямує до нуля.

Також варто зазначити, що також формули (1.4,1.5,1.6, 1.7,1.8,1.9) працюють і для евклідових просторів.

На евклідових просторах формула з регуляризуючим виразом (1.6) матиме вигляд :

$$\operatorname*{argmin}_x||Ax-b||_2^2+\lambda R(x). \tag{1.10}$$

Розділ 2: Методи передобумовлення

2.1 Ітераційні методи для розв’язування регуляризованих лінійних обернених задач

Регуляризовані лінійні обернені задачі можна представити як задачі оптимізації, де метою буде мінімізувати між даними спостереження, та даними передбаченими моделлю, з урахуванням обмежень. Однією з причин для переформулювання лінійних обернених задач як задач оптимізації полягає в тому, що це дозволяє використовувати широкий спектр потужних методів оптимізації для пошуку рішень як от ітераційні методи.

Ітераційні методи це методи наближеного розв'язування, що базуються на послідовному наближенні до розв'язку шляхом багатократного застосування деякої обчислювальної процедури, при цьому вихідними даними для кожної наступної процедури є результати застосування попередніх процедур. Існує багато різних ітераційних методів, але більшість мають деякі спільні риси, а саме :

- початкове припущення: алгоритм починається з початкового припущення для рішення. Це може бути нульовий вектор, випадковий вектор або якийсь інший вектор на основі попередньої інформації про проблему.

- ітерація: потім алгоритм входить у цикл, у якому поточне наближення рішення оновлюється для створення нового наближення. Конкретний спосіб, у який виконується оновлення, залежить від конкретного ітераційного методу, який використовується.

- перевірка збіжності: після кожного оновлення алгоритм перевіряє, чи є нове наближення «достатньо хорошим», тобто чи воно достатньо близьке до справжнього рішення, чи зміна порівняно з попереднім наближенням досить мала. Якщо апроксимація достатньо хороша, алгоритм зупиняється та повертає поточну апроксимацію як рішення. Якщо ні, він виконує ще одну ітерацію.

- критерії зупинки: якщо наближення не збігається до достатньо точного рішення протягом певної кількості ітерацій, алгоритм зупиняється та повертає найкраще

наближення, знайдене на даний момент. Це запобігає нескінченній роботі алгоритму, якщо рішення не сходиться.

Ітераційні методи особливо корисні для вирішення великомасштабних лінійних обернених задач, де прямі методи обчислювально дорогі або навіть нездійсненні. У обчислювальній магнітно-резонансній томографії (МРТ) ітераційні проксимальні методи [7] з'явилися як робочі алгоритми для розв'язання лінійних обернених задач, які поставлені у формі (1.10). Можливо, це тому, що:

1. Ці алгоритми є дуже узагальненими, оскільки вони не накладають жодних обмежень на використовувану функцію регуляризації (R) (якщо можна обчислити проксимальний оператор (R), визначений у подальшому).

2. Ці алгоритми використовують безматричні реалізації моделі A (тобто окремі координати або записи A невідомі або зберігаються в пам'яті) для обчислювально ефективної обробки проблем великої розмірності.

Завдяки вищезазначеним перевагам реконструкції МРТ зазвичай представляють як (1.10) і розв'язують за допомогою швидкого ітераційного алгоритму визначення порогу скорочення (FISTA) [8], який має теоретично оптимальну збіжність. Однак ітераційна збіжність FISTA та інших ітераційних проксимальних методів значною мірою обмежена умовою A, як визначеною власними значеннями A^*A. Це обмежує інтеграцію додатків МРТ із некоректно поставленою A, які використовують (1.10) у клінічній практиці, оскільки некоректне обумовлення призводить до тривалого часу реконструкції. В проксимальних ітераційних методах важливої частиною є проксимальний оператор або ж оператор близькості про що більш детально буде розказано у наступному підрозділі.

Також одним з ітераційних методів, що використовується при розв'язуванні лінійних обернених задач є метод спряжених градієнтів. Він особливо корисний при роботі з великими розрідженими системами. Цей метод застосовується в обчислювальній електромагнетиці для вирішення матричного рівняння, отриманого за допомогою методу скінченних елементів або методу скінченних різниць, а також в обчислювальній гідродинаміці для вирішення лінійних систем, які виникають під час моделювання потоку рідини.

2.2 Визначення передобумовлення і його використання в ітераційних методах

Передобумовлення – це техніка, яка використовується для прискорення збіжності ітераційних методів, які використовуються для розв'язування систем лінійних рівнянь. Основна ідея передобумовлення полягає в тому, щоб перетворити вихідну систему в еквівалентну, яку можна більш ефективно розв'язувати ітераційним методом. Це можна продемонструвати проаналізувавши метод проксимального градієнтного спуску.

З [7] випливає, що проксимальний оператор замкнутої функції R визначений як :

$$prox_{\alpha R}(v) = \{\underset{x}{\operatorname{argmin}} \frac{1}{2}||x - v||_2^2 + \alpha R(x). \tag{2.1}$$

Цей проксимальний оператор використовується в алгоритмі проксимального градієнтного спуску (ПГС). для вирішення оптимізаційних алгоритмів виду (1.3) [7]. З припущенням, що індукована норма A є унікальною, алгоритм ПГС для вирішення (1.10) зазначений як Алгоритм 2.1.

Алгоритм 2.1. Проксимальний Градієнтний Спуск

Вхідні дані :

- пряма модель A
- дані з вимірювань b
- проксимальний оператор $prox_{\alpha R}$
- $x_0 = 0$

Крок k : ($k \geq 0$) Обчислити

$$x_{k+1} = prox_{\lambda R}(x_k - A^*(Ax_k - b)). \tag{2.2}$$

Ітерації алгоритму 2.1 сходяться до оптимального розв'язку (1.3), позначеного x^*, який є стаціонарною точкою.

Отже розв'язок x^* оптимальний тоді й тільки тоді, коли

$$x^* = prox_{\lambda R}(x^* - A^*(Ax^* - b)). \quad (2.3)$$

Припустивши, що $prox_{\alpha R}$ має закриту форму або ж легке для реалізації рішення, ПГС є надійним і простим алгоритмом, який досягає ітеративної збіжності O(1/k). Корисна властивість проксимальних операторів, яка буде використана далі, — властивість твердої непоширюваності, яка може бути описана таким чином :

Для всіх $x, y \in C^n$,

$$||prox_{\lambda R}(x) - prox_{\lambda R}(y)||_2 \leq ||x - y||_2. \quad (2.4)$$

Тобто проксимальний оператор не збільшує відстань між точками. Це гарантує, що відображення зберігає близькість вхідних точок навіть після застосування проксимального оператора. Також ця властивість показує як від діапазону власних значень A страждає збіжність ітерацій ПГС.

Теорема 2.2. Похибка ітерацій x_k в алгоритмі 2.1 відносно x^* зверхньо-обмежена діапазоном $(I - A^*A)$, де I – це одинична матриця.

Доведення.

Нехай $e_k = x_k - x^*$ буде похибкою за ітерації. Віднімаємо (2.2) з (2.3), беручи l2 норму вектору, та з використанням властивості твердої непоширюваності маємо :

$$||e_{k+1}||_2 \leq ||(I - A^*A)e_k||_2. \quad (2.5)$$

Зокрема розбиття e_k на $s_k + t_k$, де $s_k \in null(A)$ а $t_k \in null(A)^{\perp}$ призводить до:

$$||e_{k+1}||_2^2 \leq \left||(I - A^*A)t_k|\right|_2^2 + ||s_k||_2^2. \quad (2.6)$$

$Null(A)$ – це нульовий простір матриці, $null(A)^{\perp}$ - це ортогональне доповнення до цього нульового простору матриці A.

Тому зменшення похибки e_k в межах t_k обмежене зверху діапазоном власних значень $I - A^*A$. Ця межа є строгою коли A^*A ін'єктивна. Тобто якщо A^*A не має нульових власних значень нерівність у (2.6) є строгою.

Таким чином обернені задачі, в яких A^*A мають маленькі власні значення, можуть страждати від повільної ітеративної збіжності. Щоб зрозуміти чому це відбувається можна розглянути число обумовленості – величину, що характеризує стійкість по відношенню до обчислювальної похибки, тобто чим більше значення числа обумовленості матриці системи, то менш точними будуть розв'язки системи. Число обумовленості матриці – це співвідношення між найбільшим сингулярним значенням матриці та найменшим. Сингулярним значенням матриці називають абсолютно величину власного значення матриці. Тому коли найменше сингулярне значення матриці мале, то число обумовленості стає дуже великим, що й призводить або до повільної збіжності, або навіть розбіжності. Саме тому використовуються різні методи передобумовлення, які зменшують число обумовленості ,тим самим пришвидшуючи збіжність. Але передобумовлювачі не тільки пришвидшують збіжність в ітераційних алгоритмах, але й також покращують точність розв'язку.

Одна з найпростіших форм передобумовлення – це діагональне передобумовлення, в якому передобумовлювач – це діагональна матриця матриці A : $P = diag(A)$. Припускаючи, що $A_{ii} \neq 0$, $P_{ii} = \frac{1}{A_{ii}}$. Всі ж елементи, що не лежать на головній діагоналі матриці P мають значення 0. Такий метод передобумовлення ефективний для діагонально панівних матриць.

З таким передобумовленням формула (1.10) матиме вигляд :

$$\underset{x}{\operatorname{argmin}} ||P(Ax - b)||_2^2 + \lambda R(x). \tag{2.7}$$

На основі такого методу передобумовлення існують так звані діагональноподібні методи передобумовлення, в яких матриця P розроблена так, що матриця A^*PA краще обумовлена, що значить що вона має менше число обумовленості, ніж матриця A.

Передобумовленну систему у вигляді (2.7) можна використовувати в такому ітераційному методі як метод спряжених градієнтів.

Алгоритм 2.3. Метод спряжених градієнтів для систем вигляду (1.10)

Вхідні дані :

- пряма модель A
- дані з вимірювань b
- початкове припущення x_0 (випадковий або нульовий вектор)
- $r_0 = b - Ax_0$
- $p_0 = r_0$

Крок k : ($k \geq 0$) Обчислити

$$\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}, \tag{2.8a}$$

$$x_{k+1} = x_k + \alpha_k p_k, \tag{2.8b}$$

$$r_{k+1} = r_k - \alpha_k A p_k, \tag{2.8c}$$

$$\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}, \tag{2.8d}$$

$$p_{k+1} = r_{k+1} + \beta_k p_k. \tag{2.8e}$$

Тут x_k – це наближений розв'язок на k-тій ітерації;

r_k – це похибка на k-тій ітерації.

Алгоритм зупиняється коли похибка стає меншою за норму похибки, заздалегідь визначену. Алгоритм 2.3 також може працювати для передобумовленних систем.

Алгоритм 2.4. Метод передобумовлених спряжених градієнтів для формули (2.7)

Вхідні дані :

- пряма модель A
- дані з вимірювань b

- початкове припущення x_0 (випадковий або нульовий вектор)
- $r_0 = b - Ax_0$
- $z_0 = Pr_0$
- $p_0 = z_0$

Крок k : ($k \geq 0$) Обчислити

$$\alpha_k = \frac{r_k^T z_k}{p_k^T A p_k}, \tag{2.9a}$$

$$x_{k+1} = x_k + \alpha_k p_k, \tag{2.9b}$$

$$r_{k+1} = r_k - \alpha_k A p_k, \tag{2.9c}$$

$$z_{k+1} = P r_{k+1}, \tag{2.9d}$$

$$\beta_k = \frac{r_{k+1}^T z_{k+1}}{r_k^T z_k}, \tag{2.9e}$$

$$p_{k+1} = z_{k+1} + \beta_k p_k. \tag{2.9f}$$

Також одним із методів передобумовлення є використання поліноміального передобумовлювача. Такий передобумовлювач, як можна побачити із назви, будується у вигляді поліному $P = p(A^*A)$, де p розроблений таким чином, щоб $p(A^*A)$ збільшувала внески маленьких власних значень A^*A протягом ітерації для більш швидкої збіжності. Коефіцієнти p розраховуються шляхом оптимізації (2.10).

Поліном p має вигляд :

$$p = \underset{q}{\operatorname{argmin}} \int_{z=0}^{1} (1 - q(z)z)^2 dz. \tag{2.10}$$

Потім поліном з (2.10) використовується для отримання передобумовлювача $p(A^*A)$ який включено в алгоритм 2.1, щоб прийти до алгоритму 2.5.

Алгоритм 2.1. Поліноміальне передобумовлення для ПГС

Вхідні дані :

- пряма модель A
- дані з вимірювань b
- проксимальний оператор $prox_{\alpha R}$
- оптимізований поліном з (2.8)
- $x_0 = 0$

Крок k : ($k \geq 0$) Обчислити

$$x_{k+1} = prox_{\lambda R}(x_k - p(A^*A)A^*(Ax_k - b)). \quad (2.11)$$

Як і у випадку з алгоритмом 2.4 цей алгоритм використовує передобумовлювач при кожній ітерації, з яким можна буде отримати швидшу збіжність та більш точний результат.

2.3 Приклад використання передобумовлювача при розв'язку лінійної оберненої задачі на Python

В цьому підрозділі буде показано використання методу спряжених градієнтів для розв'язку лінійної оберненої задачі, а також буде показано різницю між розв'язками ітераційного методу з передобумовлювачем та без нього. Для цього було використано мову програмування Python і вбудовані бібліотеки. Сам код має такий вигляд :

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.sparse.linalg import cg, LinearOperator
np.random.seed(42)
size = 100
A = np.random.rand(size, size)
A = A @ A.T
```

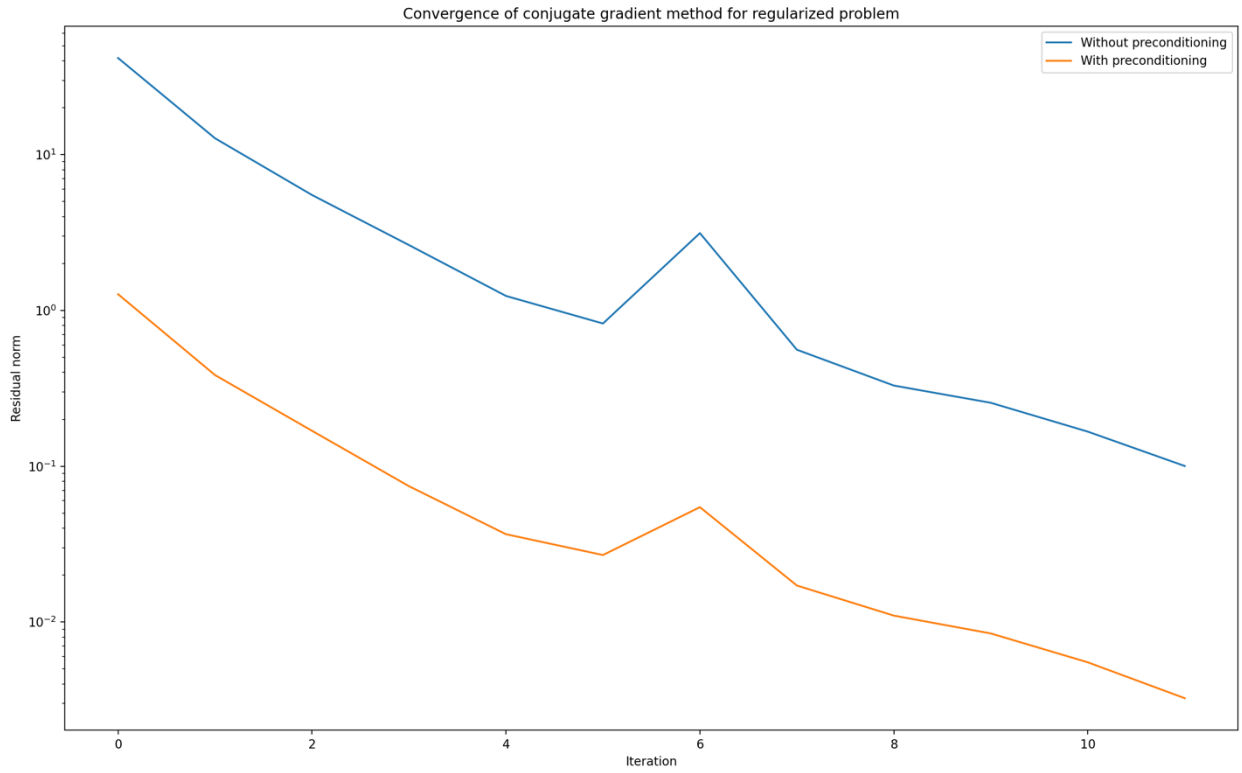
```
x_true = np.random.rand(size)
b = A @ x_true + 0.01 * np.random.randn(size)
alpha = 1
A_reg = A + alpha * np.eye(size)
residuals_no_precond = []
residuals_precond = []
def callback_no_precond(xk):
    residuals_no_precond.append(np.linalg.norm(b - A_reg @ xk))
def callback_precond(xk):
    residuals_precond.append(np.linalg.norm(M_inv @ (b - A_reg @ xk)))
x_no_precond, _ = cg(A_reg, b, callback=callback_no_precond)
M_inv = np.diag(1 / np.diag(A_reg))
precond_op = LinearOperator(A_reg.shape, matvec=lambda x: M_inv @ x)
x_precond, _ = cg(A_reg, b, M=precond_op, callback=callback_precond)
plt.figure()
plt.semilogy(residuals_no_precond, label='Without preconditioning')
plt.semilogy(residuals_precond, label='With preconditioning')
plt.xlabel('Iteration')
plt.ylabel('Residual norm')
plt.legend()
plt.title('Convergence of conjugate gradient method for regularized problem')
plt.show()
```

Отже, як працює це код :

1. Спочатку імпортуються необхідні бібліотеки : NumPy для числових обчислень, Matplotlib для побудови графіків і модуль розрідженої лінійної алгебри SciPy для методу спряженого градієнта.
2. Потім генерується випадкова квадратична матриця A розміру 100 x 100.
3. Ця матриця множиться на транспоновану їй матрицю і результуюча матриця присвоюється до A. Ця операція гарантує, що A є додатно визначеною матрицею, що є вимогою для методу спряженого градієнта.

4. Після цього генерується випадковий вектор x_true довжини 100. Цей вектор представляє істинний розв’язок.
5. Вектор даних, отриманих зі спостережень, b генерується шляхом множення A на x_true і додавання Гаусового шуму, масштабованого 0,01.
6. Далі визначається параметр регуляризації який дорівнює 1 і використовується гребенева регуляризація.
7. Потім ініціалізуються порожні списки residuals_no_precond і residuals_precond для зберігання норм похибок під час ітерацій методу спряженого градієнта без передобумовлювача та з передобумовлювачем.
8. Визначаються 2 функції зворотного виклику callback_no_precond та callback_precond які будуть додавати до списків residuals_no_precond і residuals_precond відповідно норми похибок на кожній ітерації.
9. Пізніше визначається LinearOperator precond_op, який представляє передобумовлювач для методу сполученого градієнта. Тут використовується діагональне передобумовлення, тому визначається M_inv як діагональна матриця матриці A.
10. Далі застосовується метод спряженого градієнта для розв’язання лінійної оберненої задачі без передобумовлення і з діагональним передобумовленням. Отримані рішення зберігаються в x_no_precond та x_precond відповідно, а функції зворотного виклику використовуються для обстеження збіжності цього методу.
11. В кінці будуються графіки збіжності методу спряженого градієнту з передобумовленням та без нього шляхом побудови норм похибок у логарифмічній шкалі.

    Графік побудований в результаті виконання цього коду можна побачити на рисунку зображеному нижче.



Малюнок 2.1

На малюнку 2.1 можна побачити два графіки у вигляді ломаних ліній синього та жовтого кольорів, які зображають збіжність кожної ітерації методу спряжених градієнтів. Синій графік – це збіжність ітераційного методу без передобумовлення, жовтий – збіжність з передобумовленням. Із графіку можна побачити, що обидва методи зробили по 13 ітерацій, і поведінка цих ламаних ліній однакова, оскільки ці методи вирішували одну й ту саму лінійну обернену задачу. Також видно, що жовтий графік на всьому проміжку знаходиться нижче по осі ординат, ніж синій, що означає, що норма похибки у методі з передобумовленням завжди нижча за норму похибки у методі без передобумовлення. Більше того, початкова похибка методу з передобумовленням трохи більша за 10^0 – тобто десь приблизно дорівнює 1, що досить добре при початковому кроці, а оптимальним є рішення з похибкою менше ніж 0.05, що показує досить гарну точність при використанні передобумовлювача.

Отримані результати легко порівняти з методом без передобумовлення, в якого початкова похибка більша за 10^1 – вона дорівнює приблизно трохи більше 40, тобто набагато більше за початкову похибку методу з передобумовленням, оптимальним же є рішення з похибкою приблизно в 0.1, що приблизно в 2 рази більше ніж похибка оптимального рішення з передобумовленням. Без передобумовлення виходить теж непогана точність, але вона відчутна більша за похибку з передобумовленням.

Після аналізу цих графіків можна дійти до висновку, що використання передобумовлення важливе при розв’язанні лінійних обернених і впливає на швидкість та на точність збіжності ітераційних методів, в даному випадку методу спряженого градієнту, оскільки навіть використавши один з найпростіших методів передобумовлення, а саме діагональне передобумовлення, точність результатів при використанні передобумовлення відчутно краща ніж без його використання.

Висновки

У цій роботі було досліджено лінійні обернені задачі, їхнє використання та методи, розроблені для їх вирішення. Обернені проблеми, які виникають у багатьох галузях, за своєю суттю є некоректно поставленими та вимагають обережного поводження, щоб отримати значущі та стабільні рішення. Історичний приклад відкриття Нептуна шляхом вирішення оберненої задачі підкреслює практичне значення та потенціал цих математичних проблем. Також було розказано про концепцію коректно та некоректно поставлених задач, і показано, чому обернені задачі часто належать до останньої категорії. У ньому детально описано потребу в методах регуляризації, зокрема у використанні норм l1 і l2, для стабілізації рішень і перетворення неправильно поставлених проблем у правильно поставлені. Неможливо переоцінити критичну роль цих методів регуляризації в управлінні нестабільністю, властивою оберненим задачам.

Крім того, у роботі розглядаються ітераційні методи, зокрема метод спряженого градієнта та метод проксимального градієнта, як ефективні інструменти для вирішення регуляризованих лінійних обернених задач. Ці ітераційні методи пропонують можливість поступово підходити до рішень, забезпечуючи певний рівень контролю та точності, що може бути дуже важливим при вирішенні складних і потенційно нестабільних проблем. Особливо ретельно була досліджена концепція передобумовлення як безцінна стратегія для підвищення ефективності ітераційних методів. Було розглянуто такі два методи передобумовлення як діагональне та поліноміальне передобумовлення. Методи передобумовлення відіграють вкрай важливу роль в оптимізації продуктивності ітераційних методів шляхом зменшення числа обумовленості, що призводить до швидшої збіжності ітераційних методів.

Практичне застосування цієї концепцій було продемонстровано на прикладі порівняння збіжності передобумовленого та непереобумовленого методу спряженого градієнта. Значна різниця в швидкостях та в точності збіжності підкреслює ефективність передобумовлення в покращенні процесу вирішення лінійних обернених задач.

Підсумовуючи, зазначу, що ця курсова робота пропонує дослідження проблем і стратегій, пов’язаних із вирішенням лінійних обернених задач, роблячи акцент на передобумовленні, як у необхідному інструменті для оптимального вирішення таких задач.

Джерела

1. Snieder, Trampet. “Linear and nonlinear inverse problems”, pp. 1-5. URL: http://www.geo.uu.nl/~jeannot/JTweb/pdfs/springer99-inv.pdf
2. Well Posed and Ill Posed problems & Tikhonov Regularization [Електронний ресурс]. – Режим доступу: https://www.statisticshowto.com/well-posed-ill/
3. Rosasco, Caponnetto, De Vito, De Giovannini, Odone. “Learning, Regularization and Ill-Posed Inverse Problems”, pp. 1-7.
   URL: https://web.mit.edu/lrosasco/www/publications/lip_nips2004.pdf
4. L1 and L2 Regularization Methods, Explained [Електронний ресурс]. – Режим доступу: https://builtin.com/data-science/l2-regularization
5. Benning, Burger. “Modern Regularization Methods for Inverse Problems” pp. 1-14. URL : https://arxiv.org/pdf/1801.09922.pdf
6. Iyer, Ong, Cao, Liao, Daniel, Tamir, Setsompop “Polynomial Preconditioners for Regularized Linear Inverse Problems”, pp. 1- 10.
   URL: https://arxiv.org/pdf/2204.10252.pdf
7. Parikh and S. Boyd, Proximal algorithms, Foundations and Trends in Optimization, 1 (2014), pp. 127–239.
8. A. Beck and M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.
9. Капорин. “Предобуславливание итерационных методов решения систем линейных алгебраических уравнений”, pp. 12–27.
   URL: https://core.ac.uk/download/pdf/197427253.pdf
10. 175 Years Ago: Astronomers Discover Neptune, the Eighth Planet [Електронний ресурс]. – Режим доступу: https://www.nasa.gov/feature/175-years-ago-astronomers-discover-neptune-the-eighth-planet
11. Документація модулю розрідженої лінійної алгебри бібліотеки SciPy для Python [Електронний ресурс]. – Режим доступу: https://docs.scipy.org/doc/scipy/reference/sparse.linalg.html

12. Документація бібліотеки MatPlotLib для Python [Електронний ресурс]. – Режим доступу: https://matplotlib.org/stable/tutorials/introductory/pyplot.html
13. Документація бібліотеки NumPy для Python [Електронний ресурс]. – Режим доступу: https://numpy.org/doc/stable/user/basics.html