

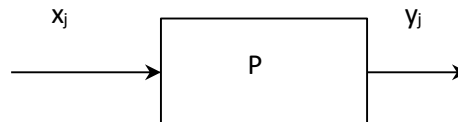
Київський національний університет імені Тараса Шевченка
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем
Моделювання складних систем

Звіт
з лабораторної роботи №2
студента 3-го курсу групи
ІПС-31
Іванченко Владислав
Віталійович
Варіант №3

Побудова лінійної моделі з допомогою псевдообернених операторів

Постановка задачі

Будемо вважати, що на вхід системи перетворення, математична модель якої невідома, поступають послідовно дані у вигляді $m - 1$ вимірних векторів \mathbf{x}_j . На виході системи спостерігається сигнал у вигляді вектора \mathbf{y}_j розмірності p .



Постановка задачі: Для послідовності вхідних сигналів \mathbf{x}_j , $j = 1, 2, \dots, n$ та вихідних сигналів \mathbf{y}_j , $j = 1, 2, \dots, n$ знайти оператор P перетворення вхідного сигналу у вихідний.

Будемо шукати математичну модель оператора об'єкту в класі лінійних операторів

$$\mathbf{A} \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix} = \mathbf{y}_j, \quad j = 1, 2, \dots, n. \quad (1)$$

Невідома матриця \mathbf{A} математичної моделі об'єкту розмірності $p \times n$. Систему (1) запишемо у матричній формі

$$\mathbf{A} \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \\ 1 & 1 & \dots & 1 \end{pmatrix} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n),$$

або

$$\mathbf{A}\mathbf{X} = \mathbf{Y}, \quad (2)$$

де $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \\ 1 & 1 & \dots & 1 \end{pmatrix}$ – матриця вхідних сигналів розмірності $m \times n$,

$\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n)$ – матриця вихідних сигналів розмірності $p \times n$.

Матрицю \mathbf{X} будемо інтерпретувати як двовимірне вхідне зображення, а матрицю \mathbf{Y} вихідне зображення.

Тоді

$$\mathbf{A} = \mathbf{Y}\mathbf{X}^+ + \mathbf{V}\mathbf{Z}^T(\mathbf{X}^T),$$

де матриця

$$\mathbf{V} = \begin{pmatrix} \mathbf{v}^T \\ \mathbf{v}^{(1)} \\ \vdots \\ \mathbf{v}^{(p)} \end{pmatrix},$$

розмірності $p \times m$, $\mathbf{Z}(\mathbf{X}^T) = \mathbf{I}_m - \mathbf{X}\mathbf{X}^+$.

Формула Гревіля для псевдообернення матриці:

Якщо для матриці A відома псевдообернена (обернена) матриця A^+ , то для розширеної матриці

$$\begin{pmatrix} A \\ a^T \end{pmatrix}$$

справедлива формула

a^T

$$\begin{pmatrix} A \\ a^T \end{pmatrix}^+ = \begin{cases} \begin{pmatrix} A^+ - \frac{Z(A)aa^TA^+}{a^TZ(A)a} : \frac{Z(A)a}{a^TZ(A)a} \end{pmatrix}, & \text{if } a^TZ(A)a > 0 \\ \begin{pmatrix} A^+ - \frac{R(A)aa^TA^+}{1+a^TR(A)a} : \frac{R(A)a}{1+a^TR(A)a} \end{pmatrix}, & \text{if } a^TZ(A)a = 0 \end{cases},$$

де $Z(A) = E - A^+A$, $R(A) = A^+(A^+)^T$.

Формула Мура - Пенроуза для знаходження оберненої (псевдооберненої) матриці:

$$A^+ = \lim_{\delta^2 \rightarrow 0} \left\{ \left(A^T A + \delta^2 E_n \right)^{-1} A^T \right\} = \lim_{\delta^2 \rightarrow 0} \left\{ A^T \left(A A^T + \delta^2 E_m \right)^{-1} \right\}.$$

матриця A розмірності $m \times n$.

Теоретичні відомості

Псевдообернена матриця A^+ - це узагальнення поняття оберненої матриці для випадків, коли

A :

- не квадратна,
- або вироджена ($\det = 0$),
- або має ранг, менший за мінімальний розмір.

Вона визначається 4 умовами Мура–Пенроуза:

1. $AA^+A = A$
2. $A^+AA^+ = A^+$
3. AA^+ — симетрична
4. A^+A — симетрична

У лабораторній роботі використовуються два підходи:

Ітераційна формула Мура–Пенроуза

$$A^+ = \lim_{\delta^2 \rightarrow 0} (A^T A + \delta^2 E)^{-1} A^T$$

Метод Гревіля

Побудова псевдооберненої матриці поелементно, додаючи рядки один за одним.

Модель оператора A через X^+

Після знаходження псевдооберненої матриці оператор відновлюється формулою:

$$A=YX^+.$$

Це головна формула, яка використовується у лабораторній роботі для побудови моделі. У загальному випадку допускається довільна матриця V:

$$A=YX^++VZ(X^T),$$

де $Z(X^T)=I-XX^+$.

У реалізації ми використовуємо найпростіший варіант: $V=0$.

Метрики точності відновлення: MSE та RMSE

MSE (Mean Squared Error)

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y}_i)^2$$

RMSE (Root Mean Square Error) — це метрика для оцінки помилок у прогнозах, яка вимірює середню різницю між прогнозованими та фактичними значеннями.

$$RMSE = \sqrt{MSE}$$

Хід роботи

1. Завантаження вхідного та вихідного зображень.

Отримано матриці X та Y, що представляють відповідно вхідний та вихідний

сигнали системи. Проведено їхнє попереднє опрацювання (перетворення у числові матриці, нормалізація).

2. Формування розширеної матриці X

До матриці X додано рядок одиниць для урахування вільного члена у моделі лінійного оператора.

3. Обчислення псевдооберненої матриці двома методами.

- методом Мура–Пенроуза (ітераційний підхід на основі граничного переходу);
- методом Гревіля (побудова псевдооберненої матриці шляхом поступового додавання рядків).

Для кожного методу перевірено виконання умов Мура–Пенроуза.

4. Побудова лінійного оператора A .

Оператор моделі визначено за формулою:

$$A=YX^+,$$

де X^+ - псевдообернена матриця, отримана відповідним методом.

5. Обчислення відновленого вихідного сигналу.

Для кожного методу отримано апроксимацію вихідного сигналу:

$$Y = AX$$

6. Оцінювання точності та порівняння методів.

Для результатів обох методів обчислено:

- $L1$ -норму похибки,
- MSE (середньоквадратичну помилку),
- RMSE (корінь середньоквадратичної помилки),
- час виконання.

На основі цих метрик проведено порівняльний аналіз точності та ефективності методів.

Програмна реалізація

Програму реалізовано мовою *Python* та розділено на три логічні модулі:

- `main.py` – основний керуючий файл;
- `image_io.py` – модуль роботи із зображеннями;
- `pseudoinverse.py` – модуль обчислення псевдообернених матриць двома методами (Мура–Пенроуза та Гревіля).

Для реалізації використано бібліотеки **NumPy** (лінійна алгебра), **Pillow** (зчитування/запис зображень), **Matplotlib** (побудова графіків), а також стандартні модулі `time` та `os` для вимірювання часу та роботи з файлами.

1. У модулі `image_io.py` реалізовано дві функції:

`read_grayscale_image(path)` – зчитує BMP-файл, переводить його у формат *grayscale* та перетворює на матрицю типу `float` з нормалізацією пікселів у діапазон $[0,1][0,1][0,1]$. Саме ці матриці використовуються як матриці XXX та YYY у задачі $AX=YAX=YAX=Y$.

`save_grayscale_image(path, matrix)` – перетворює отриману матрицю назад у 8-бітне зображення (обрізаючи значення до $[0,1][0,1][0,1]$) і зберігає результат на диск. Це дозволяє візуально порівняти вихідні та відновлені зображення.

2. Модуль `pseudoinverse.py` містить реалізацію двох методів:

`pseudo_inverse_moore_penrose(A:)` – ітераційно обчислює псевдообернену матрицю за формулою Мура–Пенроуза. На кожній ітерації будується матриця

$$(A^T A + \delta^2 I)^{-1} \text{ або } (A A^T + \delta^2 I)^{-1},$$

параметр δ поступово зменшується, а цикл зупиняється при досягненні заданої точності ϵ .

`pseudo_inverse_greville(A:)` – реалізує метод Гревіля, який формує псевдообернену матрицю послідовно, додаючи рядки A та оновлюючи A^+ за рекурсивними формулами для випадків лінійно незалежного та майже залежного рядка.

Окремою функцією **`is_pseudoinverse(A, A_plus)`** перевіряються чотири умови Мура–Пенроуза для отриманої матриці A^+ що дозволяє переконатися у коректності реалізації алгоритмів.

3. В основному файлі `main.py` виконується послідовність дій:

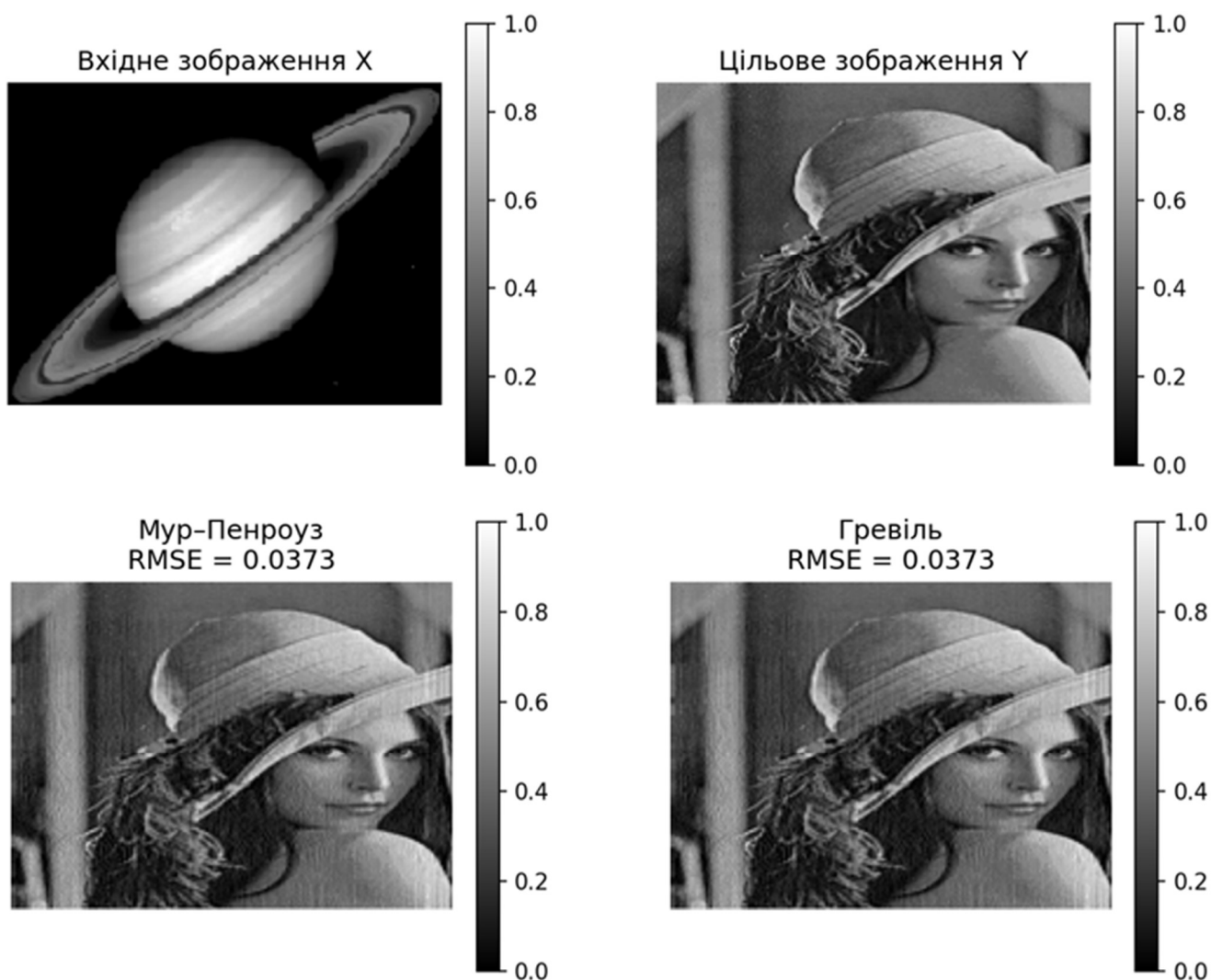
- 1) Зчитуються вхідне зображення **`x1.bmp`** та відповідне вихідне **`y3.bmp`**. Перевіряється узгодженість розмірностей.
- 2) Для урахування вільного члена формується розширена матриця за допомогою функції **`augment_with_ones`**.

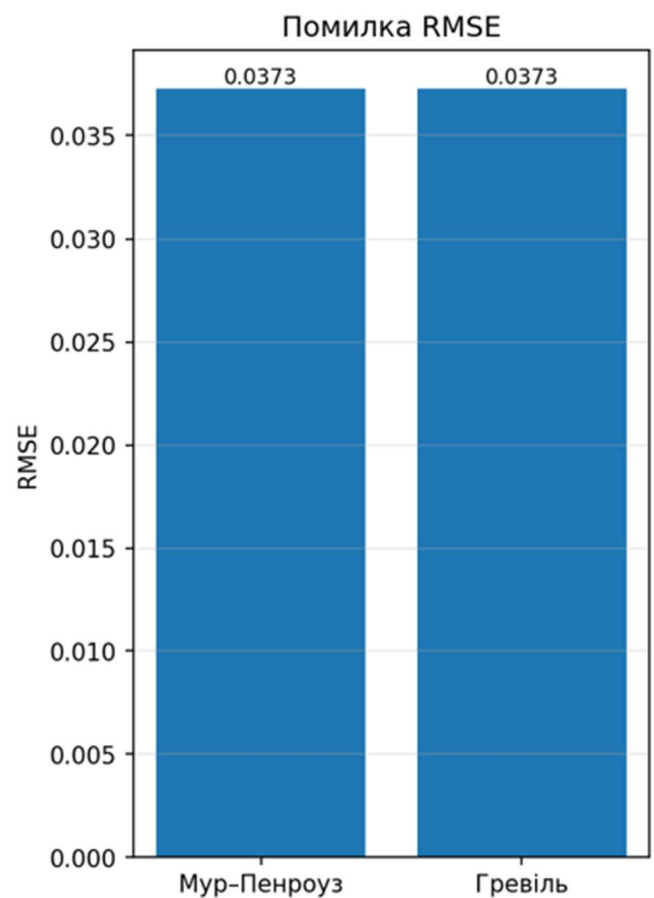
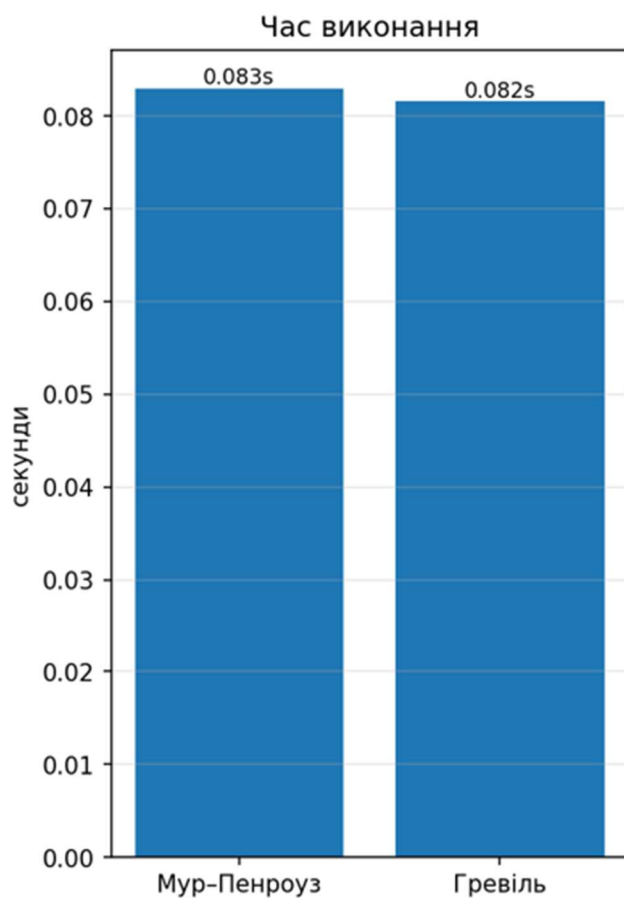
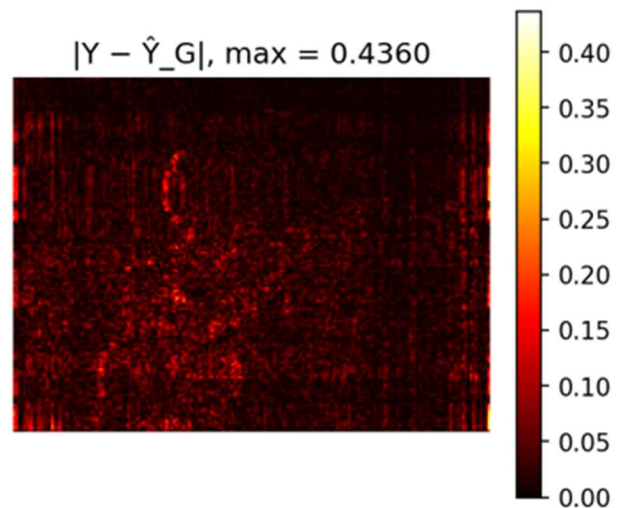
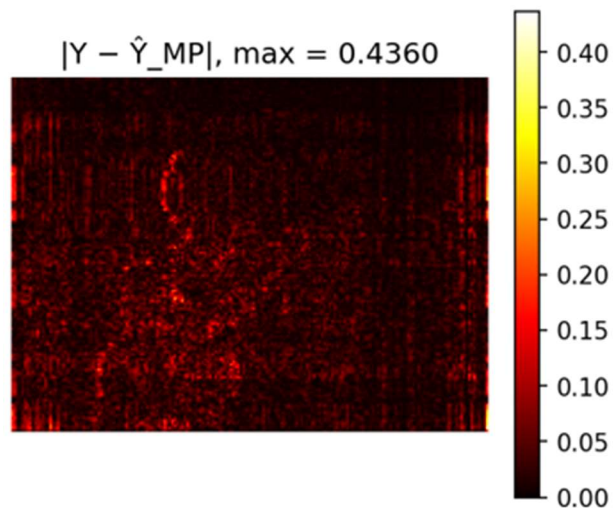
- 3) Функція **build_operator_and_predict()** для кожного з методів: обчислює псевдообернену X^+ , перевіряє умови Мура–Пенроуза, будує оператор $A=YX^+$, відновлює вихідне зображення, обчислює метрики похибки: L1-норму, MSE та RMSE; фіксує час виконання кожного методу.
- 4) Отримані результати для методів Мура–Пенроуза та Гревіль зберігаються у вигляді зображень **result_moore_penrose.bmp** та **result_greville.bmp**, що дає можливість якісно оцінити відновлення.

4. Візуалізація результатів: побудова узагальнюючого звіту засобами matplotlib.

Результат

Початкові дані: вхідний сигнал – x1.bmp, вихідний сигнал – y3.bmp. Розміри матриць: X - (140, 188), Y - (200, 200). Матрицю Y було приведено до (140, 188).





```
=== Метод Мур-Пенроуз ===
Розмір  $X^{\sim}$ : (141, 188), розмір  $Y$ : (140, 188)
Розмір  $X^{\sim+}$ : (188, 141)
Ітерацій: 27
Час: 0.082982 c
Розмір оператора  $A$ : (140, 141)
Розмір  $\hat{Y}$ : (140, 188)
 $L1 = 16.684605$ 
 $MSE = 1.391000e-03$ 
 $RMSE = 3.729612e-02$ 
```

```
=== Метод Гревіль ===
Розмір  $X^{\sim}$ : (141, 188), розмір  $Y$ : (140, 188)
Розмір  $X^{\sim+}$ : (188, 141)
Ітерацій: 140
Час: 0.081557 c
Розмір оператора  $A$ : (140, 141)
Розмір  $\hat{Y}$ : (140, 188)
 $L1 = 16.684605$ 
 $MSE = 1.391000e-03$ 
 $RMSE = 3.729612e-02$ 
```


Висновки

Під час виконання лабораторної роботи було реалізовано два методи побудови псевдообернених матриць (ітераційний Мура–Пенроуза та рекурсивний Гревіля) для задачі відновлення лінійного оператора A з пари зображень (вхід X та вихід Y).

На основі отриманих результатів ($RMSE \approx 0.0373$ для обох методів) можна зробити висновок, що обидва алгоритми відновили вихідне зображення з однаково високою точністю. Візуально, відновлені зображення `result_moore_penrose.bmp` та `result_greville.bmp` майже не відрізняються від цільового `u3.bmp` (після його масштабування).

Найпомітніша різниця полягає у часі виконання. У даному експерименті метод Гревіля (0.0816 с) виявився незначно швидшим за метод Мура–Пенроуза (0.0830 с). Хоча різниця невелика, це підтверджує теоретичну перевагу методу Гревіля, який уникає "дорогих" обчислень інверсії матриць, які використовуються у методі Мура–Пенроуза.

Обидва методи коректно пройшли перевірку на 4 умови Мура–Пенроуза, що підтверджує правильність їх програмної реалізації.