Міністерство освіти і науки України
Вінницький національний технічний університет

Б. І. Мокін, В. Б. Мокін, О. Б. Мокін

НАВЧАЛЬНИЙ ПОСІБНИК
для опанування студентами способів розв’язання задач з функціонального аналізу мовою Python

Частина 2

Вінниця
ВНТУ
2023

УДК 517.98
М74

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України як навчальний посібник для студентів та аспірантів закладів вищої освіти, що спеціалізуються в галузі інформаційних технологій (протокол № 7 від «28» лютого 2023 р.).

Рецензенти:
В. Я. Данилов, д.т.н. проф. (НТУУ «КПІ ім. Сікорського»)
В. І. Клочко, д. пед. н. професор (ВНТУ)
О. С. Макаренко, д. ф-м. н. професор (НТУУ «КПІ ім. Сікорського»)

Мокін, Б. І.
М74 Навчальний посібник для опанування студентами способів розв’язання задач з функціонального аналізу мовою Python. Частина 2 : навчальний посібник / Б. І. Мокін, В. Б. Мокін, О. Б. Мокін. – Вінниця : ВНТУ, 2023. – 139 с.
ISBN 978-966-641-926-5
В навчальному посібнику викладено способи розв’язання задач з функціонального аналізу, адаптованого до прикладних проблем в галузі інформаційних технологій, згідно зі змістом однойменного навчального посібника цих же авторів, а також викладені основи програмування мовою Python і програми реалізації способів розв’язання даного класу задач цією мовою. Частина 2 є продовженням частини І і охоплює задачі з дослідження та використання різних класів операторів.
Навчальний посібник рекомендується для студентів та аспірантів, що спеціалізуються в ІТ-галузі за спеціальностями 124 «Системний аналіз» та 126 «Інформаційні системи та технології»
УДК 517.98

ISBN 978-966-641-926-5

ЗМІСТ

ВСТУП

Оскільки друга частина нашого «Навчального посібника для опанування студентами способів розв'язання задач з функціонального аналізу мовою Python» є органічним продовженням нашої першої частини цього ж навчального посібника [1], то, з метою їхнього пов'язання між собою, ми почнемо вступ до другої частини з цитування першого абзацу вступу до частини першої, який звучить так [1]: «У 2020 році ми опублікували навчальний посібник «Функціональний аналіз, адаптований до прикладних задач в галузі інформаційних технологій» [2], з використанням якого студенти, що навчаються у ВНТУ за спеціальністю 124 «Системний аналіз», опановують навчальну дисципліну «Функціональний аналіз», яка, згідно з освітньо-професійною програмою цієї спеціальності, входить до переліку обов'язкових.

Уже після першого року використання в навчальному процесі нашого навчального посібника з функціонального аналізу ми зрозуміли, що студентам важко засвоювати практичні навички з розв'язання прикладних задач з цієї навчальної дисципліни, користуючись лише навчальним посібником, в якому викладені її теоретичні основи, особливо ж в умовах реалізації навчального процесу в онлайн-режимі, зумовленому карантинними обмеженнями щодо перебування студентів групами в аудиторіях, викликаними світовою пандемією коронавірусної хвороби. А тому в нашому авторському колективі виникла задумка написати з функціонального аналізу ще один навчальний посібник, в якому викласти практичні способи розв'язання задач з функціонального аналізу, характерних для ІТ-галузі, аби цей другий навчальний посібник доповнив перший практикою застосування теоретичних основ функціонального аналізу, викладених у першому навчальному посібнику, при розв'язанні прикладних задач, характерних для ІТ-галузі. Особливо важливим аспектом цього другого навчального посібника, на наш погляд, мало стати доведення етапу розв'язання прикладних задач функціонального аналізу до застосування сучасної мови програмування Python, усі дистрибутиви якої викладені в інтернеті для вільного доступу у вигляді програмного середовища Anaconda [3]. Саме ця задумка і реалізована нами у навчальному посібнику, зміст якого подається нижче. І оскільки цей навчальний посібник покликаний доповнювати практикою попередній, присвячений теорії функціонального аналізу, то і розділи його збігаються за назвами і змістом з відповідними розділами свого теоретичного попередника.

Характерною особливістю нашого навчального посібника, вміст якого подається нижче, є те, що ми спочатку демонструємо способи і алгоритми «ручного» розв'язання прикладних задач функціонального аналізу на конкретних простих прикладах, а уже потім показуємо, як ці задачі розв'язувати за допомогою мови Python з використанням дистрибутивів і підпрограм, що допускають їх комп'ютерну реалізацію в консолі Spyder.

Такий підхід зумовлений тим, що, як показала практика роботи зі студентами під час вивчення функціонального аналізу, в разі, якщо для розв'язання задачі використати готову програму, написану мовою Python, то студенти, використовуючи цю програму, отримують розв'язок задачі, не розуміючи суті закладених в програму алгоритмів, що, по-перше, надовго не залишається в пам'яті, а по-друге, не сприяє осмисленому застосуванню цих же алгоритмів при розв'язанні інших задач, умови яких в чомусь відрізняються.»

Але, оскільки у цьому навчальному посібнику ми використовуватимемо мову програмування Python, то спочатку, з посиланням на популярний [4] та науковий [5] варіанти її викладення, здійснимо екскурс в історію створення цієї мови та в загальних рисах розглянемо основні дистрибутиви, що входять до її структури, використавши для цього матеріал, уже викладений нами у вступі до першої частини навчального посібника [1].

Цитувати історію створення мови Python ми не будемо, бо ті, кого вона цікавить, можуть ознайомитись з цією історією, прочитавши вступ до першої частини навчального посібника [1], а ось інформацію про основні дистрибутиви цієї мови, аби їх пригадати, ми наведемо знову ж таки зі вступу до першої частини цього навчального посібника [1]:

«…сконцентруємо увагу на 64-бітовому інтегрованому середовищі Anaconda, на яке ви можете зустріти посилання в різних літературних джерелах і як на дистрибутив, і як на пакет прикладних програм (ППП) мовою Python. Встановлювати на своєму комп'ютері дистрибутив Anaconda вигідно ще й тому, що одночасно та автоматично на вашому комп'ютері встановлюється і бібліотека пакетів прикладних програм, які використовуються для наукових досліджень і носять назви: numpy, sympy, scipy, matplotlib.

ППП numpy (або NumPy – числовий Пайтон) – це пакет програм, який використовується для числових розрахунків.

ППП sympy (або SymPy - символьний Пайтон) – це пакет програм, який використовується для всіляких перетворень виразів, заданих у символьній формі (зокрема і для взяття похідних та невизначених інтегралів від складних функцій).

ППП scipy (або SciPy – науковий Пайтон) – це пакет програм, який зручно застосовувати при обробленні результатів наукових досліджень у сукупності з пакетами sympy та numpy.

ППП matplotlib – це пакет програм для одно-, дво- і тривимірних графічних зображень результатів розрахунків, виконаних з застосуванням пакетів numpy, sympy та scipy, тобто, це графічний редактор, пристосований до роботи з програмами, написаними мовою Python.

Зміст усіх цих ППП та порядок їх застосування ми будемо розкривати в процесі їх використання для розв'язання задач функціонального аналізу, а у цій вступній частині навчального посібника згадаємо ще лише про те, що реалізовувати ці ППП в разі, якщо ви встановили на своєму комп'ютері дистрибутив Anaconda, можна за допомогою інтерпретаторів (або, що одне і те ж, консолей) IPython, IPython Notebook, який ще називають Jupyter Notebook, та Spyder, головне вікно останнього з яких на екрані розділене на три частини, причому ліва частина призначена для набору і редагування програм у вигляді файлів, які можна або виконувати або відправляти в пам'ять з можливістю виклику у подальшому, права нижня частина призначена для набору програм за допомогою командного рядка, їх тестування та виведення результатів їх дії і результатів тестування на екран, а права верхня частина призначена для виведення на екран рисунків, що супроводжують обчислення.

Ми в нашому навчальному посібнику будемо використовувати саме консоль Spyder, назва якої є абревіатурою від Scientific PYthon Development EnviRonment, що перекладається як "Наукового Пайтону середовище розвитку», і яка є найбільш узагальненою, оскільки інтегрує в собі основні риси інших двох інтерпретаторів, згаданих вище» – кінець цитати зі вступу до першої частини навчального посібника [1].

На цьому ми у вступі до другої частини нашого навчального посібника поставимо крапку і приступимо до конкретизації її змісту, зберігши ту ж послідовність розділів, що і в базовому навчальному посібнику [2] та розпочавши нумерацію розділів і програм у другій частині у послідовностях, що доповнюють їх до послідовностей, використаних у частині першій [1]. Єдине, в чому ми відступимо від вищевикладеного, це те, що восьмий розділ базового навчального посібника [2] ми у другій частині, до написання якої приступаємо, розіб'ємо на два розділи, у першому з яких, тобто у «Розділі 8», викладемо задачі і програми, пов'язані з неперервними операторами, а у другому з них, тобто у «Розділі 9», викладемо задачі і програми, пов'язані з дискретними операторами.

Розділ 6 ОПЕРАТОРИ ТА ЇХ ОСНОВНІ ХАРАКТЕРИСТИКИ (В ПРИКЛАДАХ І ПРОГРАМАХ)

6.1 Загальні характеристики операторів, резольвента і спектр та група

У кінці однойменного підрозділу у базовому навчальному посібнику [2] серед інших стосовно дослідження операторів сформульовані і такі питання:

1. Дайте означення оператора.
2. Який оператор є лінійним?
3. Що собою являє норма оператора?
4. Як визначити обернений оператор?
5. Що є резольвентою неоднорідного операторного рівняння?
6. Які значення оператора є регулярними, а які складають його спектр?
7. За яких умов неоднорідне операторне рівняння має коректний розв’язок?
8. Як визначити власні числа та власні вектори оператора?
9. Що розуміють під групою операторів, і які властивості такої групи ви знаєте?

Тож із відповідей на ці питання ми і розпочнемо викладення змісту другої частини нашого «Навчального посібника для опанування студентами способів розв’язання задач з функціонального аналізу мовою Python».

Що стосується суті відповідей на кожне з цих питань, то її ми розкриватимемо послідовно, використовуючи матеріал, викладений в роботі [2], у вигляді розлогої, але з пропусками та деякими перефразуваннями відфільтрованої цитати. Отже,

Оператор *A*, який здійснює перетворення множини *X* у множину *Y*, ***називається лінійним, якщо:***

$$A\cdot\left(x_1+x_2\right)=A\cdot x_1+A\cdot x_2,\ \forall\, x_1,x_2\in X\,, \tag{6.1}$$

$$A\cdot x_n\to A\cdot x_0,\quad \{x_n\}\subset \mathrm{X},\ \ x_0\in\mathrm{X},\ \ x_n\to x_0, \tag{6.2}$$

$$A\cdot 0=0,\ 0\in X; \tag{6.3}$$

$$A(-x)=-A\cdot x,\ \forall\, x\in X\cdot \tag{6.4}$$

Лінійний оператор, для кого виконується співвідношення

$$A\cdot\left(t\cdot x\right)=t\cdot A\cdot x,\ \forall\, x\in X, t - \text{скаляр}, \tag{6.5}$$

називається ***однорідним***.

Число

$$K_0=\sup_{\|x\|\le 1}\|A\cdot x\|=\|A\|=\sup_{x\ne 0}\frac{\|A\cdot x\|}{\|x\|}. \tag{6.6}$$

називають нормою оператора A і позначають

$$K_0=\|A\|\cdot \tag{6.7}$$

З нерівності трикутника для норми випливає, що для $\forall\, x\in S$ справедливо

$$\|A\cdot x\|\le\|A\|\cdot\|x\|. \tag{6.8}$$

Отже, ***множина лінійних операторів A, які перетворюють множину X у множину*** Y, що символічно позначається як

$$A:(X \to Y) \tag{6.9}$$

або

$$A \in (X \to Y), \tag{6.10}$$

є лінійним нормованим простором $(X \to Y)$, в якому

$$\| A \cdot x_n - A \cdot x_0 \| = \| A \cdot (x_n - x_0) \| \le K \cdot \| x_n - x_0 \| = 0, \tag{6.11}$$

оскільки з того, що $x_n \to x_0$, випливає, що

$$\| x_n - x_0 \| = 0 . \tag{6.12}$$

Оператор $A^{-1} \in (Y \to X)$**,** який задовольняє рівняння

$$\begin{cases} A^{-1} \cdot (A \cdot x) = x, \ \forall\, x \in X, \\ A \cdot \left(A^{-1} \cdot y\right) = y, \ \forall\, y \in Y. \end{cases} \tag{6.13}$$

що матиме місце лише за умови

$$A^{-1} \cdot A = A \cdot A^{-1} = I \tag{6.14}$$

називають ***оберненим до оператора A***.

Зауважимо, що під добутком будь-яких операторів ми розуміємо їх послідовне застосування до елемента, що стоїть з правого боку від них.

Після введення оберненого оператора A^{-1} зрозуміло, що розв'язок рівняння

$$A \cdot x = y, \tag{6.15}$$

де $x \in X$, $y \in Y$, $A \in (X \to Y)$, матиме вигляд

$$x = A^{-1} y, \tag{6.16}$$

а розв'язок рівняння

$$A \cdot x - \lambda \cdot x = y, \tag{6.17}$$

де λ – скаляр, матиме вигляд

$$x = (A - \lambda \cdot I)^{-1} \cdot y. \tag{6.18}$$

Оператор R_λ, обернений до $(A - \lambda \cdot I)$, тобто,

$$R_\lambda = (A - \lambda \cdot I)^{-1} = \frac{I}{A - \lambda I} = -\left(\frac{1}{\lambda}\, \frac{I}{I - \frac{1}{\lambda} A}\right) = -\frac{1}{\lambda}\left(I + \frac{1}{\lambda} A + \frac{1}{\lambda^2} A^2 + \frac{1}{\lambda^3} A^3 + \cdots\right) \tag{6.19}$$

називають ***резольвентою оператора A або розв'язувальним оператором для рівняння*** (6.17).

Якщо ж рівняння (6.17) матиме вигляд

$$A\, x - \lambda\, x = 0, \tag{6.20}$$

то розв'язок рівняння (6.20) буде мати вигляд

$$x = \left(A - \lambda \cdot I\right)^{-1} \cdot 0 = 0 \cdot \qquad (6.21)$$

Ті значення параметра λ, які дозволяють мати резольвенту R_λ, називають регулярними значеннями оператора A. Всі інші значення параметра λ, які не є регулярними, складають спектр оператора A, а значення λ, за яких однорідне рівняння* (6.20) *має розв'язок, відмінний від нульового, тобто для яких не виконується* (6.21), *називають характеристичними числами або власними значеннями λ_i, $i = 1,2,,,,n$ оператора A з порядком n, які теж є точками спектра цього оператора.

Нагадаємо, що множина ***власних значень λ_i, $i = 1,2,,,,n$ оператора*** *A*, яка знаходиться з рівняння

$$|A - \lambda\, I| = 0, \qquad (6.22)$$

породжує множину власних векторів w_i, $i = 1,2,\dots,n$ цього оператора, яка знаходиться з системи рівнянь

$$(A - \lambda_i\, I)w_i = 0,\;\; i = 1,2,\dots,n\,, \qquad (6.23)$$

доповненої рівнянням нормування

$$\sum_{i=1}^{n} w_i = 1 \qquad (6.24)$$

Але потрібно пам'ятати, що до складу спектра оператора *A* можуть входити значення *λ*, які не є його характеристичними числами, тобто потужність спектра оператора є більшою потужності множини його характеристичних чисел.

Потрібно також пам'ятати, що, якщо оператор *A* є обмеженим, то весь його спектр лежить у крузі

$$\|\lambda\| \le \|A\|, \qquad (6.25)$$

а за межами цього круга, тобто при

$$\|\lambda\| > \|A\|, \qquad (6.26)$$

рівняння* (6.17) *має однозначний розв'язок, якщо відповідне йому однорідне рівняння

$$A \cdot x = 0 \qquad (6.27)$$

має лише нульовий розв'язок, і має коректний розв'язок, якщо для $\forall\, x \in X$ виконується співвідношення

$$\|x\| \le K \cdot \|A \cdot x\|,\;\; A \in \left(X \to X\right). \qquad (6.28)$$

Під групою $u(t)$ операторів A розуміють таку їх сукупність, якій притаманні властивості:

1) $$u(0) = I\,; \qquad (6.29)$$

2) $$u(t+\tau) = u(t) \cdot u(\tau), \quad -\infty < t, \tau < \infty; \qquad (6.30)$$

3) $$\frac{du}{dt} = A \cdot u \cdot \qquad (6.31)$$

Зрозуміло, що ці три властивості мають лише ті оператори A, які є обмеженими і задовольняють співвідношення

$$u(t) = e^{t \cdot A}. \tag{6.32}$$

З виразу (6.32) випливає, що, по-перше,

$$A = \frac{\ln u(t)}{t}. \tag{6.33}$$

а по друге, оператор A можна визначити як похідну від групи $u(t)$ при $t = 0$.

Тому ***оператор A називають породним оператором для групи u(t) або, що одне і те саме, інфінітезимальним.***

Розкладаючи експоненту у виразі (6.32) у степеневий ряд, отримаємо

$$u(t) = 1 + t \cdot A + \frac{t^2}{2!} \cdot A^2 + \frac{t^3}{3!} \cdot A^3 + \cdots, \tag{6.34}$$

тобто, група $u(t)$ може бути заданою не лише через експоненту (6.32), а й у вигляді степеневого ряду (6.34).

А завершуючи цей матеріал, ми ще раз звертаємо увагу на те, що за класичним означенням оператор – це закон, за яким одній множині функцій ставиться у відповідність інша множина функцій, про що не потрібно забувати, обчислюючи за виразом (6.6) норму оператора. Тобто, не потрібно забувати, що вираз $||x||$, який стоїть у знаменнику, задає норму функції, наприклад, $x(\theta),\ \theta \in [a,b]$, яка може бути заданою або на множині функцій $C[a,b]$ з нормою у вигляді

$$||x|| = \max_{\theta \in [a,b]} |x(\theta)|, \tag{6.35}$$

або на множині функцій $L_2[a,b]$ з нормою у вигляді

$$||x|| = \sqrt{\int_a^b x^2(\theta) d\theta}. \tag{6.36}$$

6.2 Додаткові відомості з мови програмування Python, достатні для розв'язання задач, пов'язаних з обчисленням характеристик операторів

Аналізуючи характер формул, викладених у попередньому підрозділі 6.1, бачимо, що для їх реалізації в програмах мовою Python нам будуть потрібні деякі нові знання функцій і методів цієї мови програмування, додаткові до тих, про які уже йшла мова в підрозділах, присвячених викладенню додаткових відомостей з технології програмування мовою Python у першій частині цього навчального посібника, які ми дублювати не будемо.

Отже, для програмної реалізації мовою Python виразів (6.6),(6.14),(6.19), (6.22), (6.23), (6.24),(6.34) до тих знань, що ми уже маємо з цієї мови, у першу чергу потрібно додати інформацію про те, як реалізувати операції лінійної алгебри, які зосереджені в ППП numpy в модулі **numpy.linalg**, в модулі **numpy.matlib** та модулі **numpy.matrix** (numpy.mat), а також в ППП scipy в модулі **scipy.linalg** та в ППП sympy у вигляді функції **Matrix**, яка містить і атрибути та методи, потрібні для реалізації символьного варіанта матричної алгебри. З прикладів застосування функції **Matrix**, використовуючи інформацію, наведену в [5], ми і продовжимо. Отже,

Пояснювальний приклад № 1

```
In [1]: import sympy                              # Виклик ППП sympy
In [2]: from sympy import*                        # Виклик усіх функцій ППП sympy
In [3]: x,y,z,w = symbols('x y z w')              # Оголошення x,y,z,w
                                                  ….символьними
In [4]: M = Matrix([[x,y],[z,w]]); M              # Створення матриці (2x2)
Out[4]:                                           ….у символьному вигляді
Matrix([
[x, y],
[z, w]])
In [5]: M1 = Matrix([[1,2],[3,4]]); M1            # Створення матриці (2x2)
Out[5]:                                           ….у числовому вигляді
Matrix([
[1, 2],
[3, 4]])
In [6]: M2 = Matrix([5,6,7]);M2                   # Створення матриці-стовпця
Out[6]:
Matrix([
[5],
[6],
[7]])
In [7]: M.det()                                   # Обчислення визначника
                                                  ….матриці в символьному
Out[7]:                                           …. вигляді
w*x - y*z
In [8]: M1.det()                                  # Обчислення визначника
Out[8]:                                           …. матриці в числовому
-2                                                …. вигляді
In [9]: M.T                                       # Транспонування матриці
Out[9]:                                           ….в символьному вигляді
Matrix([
[x, z],
[y, w]])
In [10]: M1.T                                     # Транспонування матриці
Out[10]:                                          ….в числовому вигляді
Matrix([
[1, 3],
[2, 4]])
In [11]: M2.T                                     # Транспонування матриці-
                                                  ….стовпця у матрицю-рядок
Out[11]:
Matrix([[5, 6, 7]])
In [12]: M3 = simplify(M1.inv());M3               # Обчислення оберненої матриці
Out[12]:
Matrix([
[ -2,    1],
[3/2, -1/2]])
```

```
In [13]: M4 = M1**(-1);M4                    # Інший варіант обчислення
Out[13]:                                     ….оберненої матриці
Matrix([
[ -2,    1],
[3/2, -1/2]])
In [14]: M1*M3                               # Результат перемноження
                                             ….матриць
Out[14]:
Matrix([
[1, 0],
[0, 1]])
In [15]: M1+M3                               # Результат додавання матриць
Out[15]:
Matrix([
[ -1,   3],
[9/2, 7/2]])
In [16]: M1**2                               # Результат піднесення
Out[16]:                                     …. матриці до квадрату
Matrix([
[ 7, 10],
[15, 22]])
In [17]: M1.eigenvals()                      # Обчислення власних
Out[17]: {5/2 - sqrt(33)/2: 1,               …. чисел матриці
5/2 + sqrt(33)/2: 1}
In [18]: M1.eigenvects()                     # Обчислення власних
Out[18]:                                     …. векторів матриці
[(5/2 - sqrt(33)/2,1,
  [Matrix([
   [-2/(-3/2 + sqrt(33)/2)],[1]])]),
 (5/2 + sqrt(33)/2,1,
  [Matrix([
   [-2/(-sqrt(33)/2 - 3/2)],[1]])])]
In [19]: M5 = Matrix([3,2,1])                # Створення матриці
In [20]: M2.dot(M5)                          # Обчислення скалярного
                                             ….добутку двох матриць
Out[20]:
34
In [21]: M2.cross(M5)                        # Обчислення векторного
                                             ….добутку двох матриць
Out[21]:
Matrix([
[-8],
[16],
[-8]])
In [22]: a = symbols('a:2:2');a              # Символьне внесення
                                             ….послідовності з подвійним
Out[22]: (a00, a01, a10, a11)                ….індексуванням
```

```
In [23]: b = symbols('b:2:2');b                      # Символьне внесення
                                                     ….послідовності з подвійним
Out[23]: (b00, b01, b10, b11)                        …. індексуванням
In [24]: A = Matrix([[a[0],a[1]],[a[2],a[3]]]);A     # Індексна трансформація
                                                     ….послідовності в матрицю
Out[24]:
Matrix([
[a00, a01],
[a10, a11]])
In [25]: B = Matrix([[b[0],b[1]],[b[2],b[3]]]);B     # Індексна трансформація
                                                     ….послідовності в матрицю
Out[25]:
Matrix([
[b00, b01],
[b10, b11]])
In [26]: A*B                                         # Результат перемноження
                                                     ….індексно трансформованих
Out[26]:                                             …. матриць
Matrix([
[a00*b00 + a01*b10, a00*b01 + a01*b11],
[a10*b00 + a11*b10, a10*b01 + a11*b11]])
```

Кінець пояснювального прикладу № 1

А далі звертаємо увагу на те, що модуль **scipy.linalg** надає такі ж можливості у виконанні операцій лінійної алгебри, як і модуль **numpy.linalg**, але містить в собі і низку таких операцій, яких немає в модулі **numpy.linalg**, а тому використовується частіше. Однак, оскільки вхідні дані в програмах мовою Python задаються у вигляді числових масивів, які є атрибутами ППП numpy, то навіть при використанні для реалізації операцій лінійної алгебри модуля **scipy.linalg**, що входить до ППП scipy, потрібно перед цим внести в програму команду, якою викликати ППП numpy. Заслуговує бути відзначеним також те, що в рамках ППП numpy операції лінійної алгебри здійснюються з прямокутними числовими масивами за правилами здійснення матричних операцій, залишаючись однак в результатах теж числовими масивами. Якщо ж в рамках ППП numpy є необхідність від масивів перейти до матриць, потрібно трансформувати масиви певної розмірності в матриці тієї ж розмірності, використавши клас функцій **matrix**, до символу якого можна застосовувати скорочення **mat**. Отже,

Пояснювальний приклад № 2

```
In [1]: import numpy as np                           # Виклик ППП numpy np
In [2]: import scipy.linalg as la                    # Виклик модуля scipy.linalg як la
In [3]: x = np.array([[1,3],[4,6]])                  # Внесення масиву x
In [4]: la.det(x)                                    # Обчислення визначника
Out[4]: -6.0                                         ….двовимірного масиву x
In [5]: la.norm(x)                                   # Обчислення норми масиву x
Out[5]: 7.874007874011811                            ….у двовимірному просторі
In [6]: np.round_(la.norm(x),2)                      # Обмеження числа знаків у
Out[6]: 7.87                                         ….нормі масиву x
In [7]: x1=la.inv(x);x1                              # Обчислення масиву,
Out[7]:                                              ….оберненого до масиву x
```

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Мокін Б. І., Мокін В. Б., Мокін О. Б. Навчальний посібник для опанування студентами способів розв’язання задач з функціонального аналізу мовою Python, частина 1 : навч. пос. Вінниця : ВНТУ, 2022. 124 с.

2. Мокін Б. І., Мокін В. Б., Мокін О. Б. Функціональний аналіз, адаптований до прикладних задач в галузі інформаційних технологій : навчальний посібник.– Вінниця : ВНТУ, 2020 192 с.

3. Python. [Електронний ресурс]. Режим доступу : https://python.org/downloads/.

4. Бріггс Джейсон Р. Python для дітей (веселий вступ до програмування) / пер. з англ. Олександри Гордійчук. Львів : Видавництво старого Лева, 2019. 400 с.

5. Доля П. Г. Введение в научный Python. Харків : ХНУ ім. Каразіна, 2016. 265 с.

6. Мокін Б. І., Мокін О. Б., Шалагай Д. О. Про один із підходів наближеного обчислення інтегралів Стілтьєса і Лебега на мові Python в задачах системного аналізу з дискретними моделями. *Вісник Вінницького політехнічного інституту*, 2021, № 3 С. 61–68. DOI https://doi.org/10.31649/1997-9266-2021-156-3-61-68

7. Деч Густав. Руководство к практическому применению преобразования Лапласа и Z-преобразования. М. : Наука, 1971. 288 с.

8. Мокін О. Б., Мокін В. Б., Мокін Б. І. Метод ідентифікації моделі авторегресії-ковзного середнього АРКС(p,q) з довільними значеннями порядків p, q, який узагальнює методику Юла – Уокера. *Наукові праці Вінницького національного технічного університету.* 2014. № 2. С. 1–6. Режим доступу : https://praci.vntu.edu.ua/index.php/praci/article/view/406/404.

9. Бокс Дж., Дженкинс Г. Анализ временных рядов. *Прогноз и управление*. Вып. 1. М. : Мир. 1974. 408 с.

10. Бокс Дж., Дженкинс Г. Анализ временных рядов. *Прогноз и управление*. Вып. 2. М. : Мир. 1974. 197 с

11 Мокін О. Б., Мокін В. Б., Мокін Б. І. Алгоритм методу ідентифікації моделі авторегресії-ковзного середнього, який узагальнює методику Юла – Уокера, та його програмна Python-реалізація. *Вісник Вінницького політехнічного інституту*. 2022. № 4. С. 61–68. DOI https://doi.org/10.31649/1997-9266-2022-163-4-41-55

Навчальне видання

Борис Іванович Мокін
Віталій Борисович Мокі
Олесандр Борисович Мокін

НАВЧАЛЬНИЙ ПОСІБНИК
для опанування студентами способів розв'язання задач з функціонального аналізу мовою Python
Частина 2

Навчальний посібник

Рукопис оформив *Б. Мокін*

Редактор *В. Дружиніна*

Оригінал-макет підготовлено у *Редакційно-видавничому відділі ВНТУ*

Підписано до друку 5.04.2023 р.
Формат 29,7×42 ¼. Папір офсетний.
Гарнітура Times New Roman.
Друк різографічний. Ум. друк. арк. 16,.17.
Наклад 30 пр. Зам. № 2023-007

Видавець та виготовлювач
Вінницький національний технічний університет,
редакційно-видавничий відділ.
ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95,
м. Вінниця, 21021.
press.vntu.edu.ua;
E-mail: kivc.vntu@gmail.com
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.