

ALGEBRA W SAGEMATH

Część 2: MACIERZE

WITALI BUŁATOW

$$A^T = \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 3 & 2 \end{bmatrix}$$

$$A^n = \underbrace{A \cdot \dots \cdot A}_n$$

$$\begin{bmatrix} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 2 & -4 & -3 \end{bmatrix} \xrightarrow{w_3 + 2w_2} \begin{bmatrix} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 0 & -2 & -1 \end{bmatrix}$$

$$\begin{bmatrix} -2 & 1 & 5 & -1 \\ 2 & -4 & -2 & 4 \\ 3 & -4 & -7 & 3 \end{bmatrix} \xrightarrow{w_2 \cdot \frac{1}{2}} \begin{bmatrix} -2 & 1 & 5 & -1 \\ 1 & -2 & -1 & 2 \\ 3 & -4 & -7 & 3 \end{bmatrix}$$

1	3	$1 \cdot 2 + 3 \cdot 4$	
0	2		$0 \cdot 3 + 2 \cdot 1$

$$\begin{bmatrix} -2 & 1 & 5 & -1 \\ 1 & -2 & -1 & 2 \\ 3 & -4 & -7 & 3 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 & 2 \\ -1 & 1 & 1 \\ 2 & -4 & -3 \end{bmatrix}$$

$$B^2 = B \cdot B = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} = \begin{bmatrix} 2 \cdot 2 + 3 \cdot 4 & 2 \cdot 3 + 3 \cdot 1 \\ 4 \cdot 2 + 1 \cdot 4 & 4 \cdot 3 + 1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 16 & 9 \\ 12 & 13 \end{bmatrix}$$

WITALI BUŁATOW

ALGEBRA W SAGEMATH

CZĘŚĆ 2: MACIERZE



OFICyna WYDAWNICZA POLITECHNIKI BIAŁOSTOCKIEJ
BIAŁYSTOK 2026

Recenzent:
dr Anna Ewa Łyczkowska-Hanćkowiak

Redaktor naukowy dyscypliny matematyka:
prof. dr hab. inż. Zbigniew Bartosiewicz

Redakcja i korekta językowa:
Katarzyna Duniewska

Skład, grafika i okładka:
Marcin Dominów

© Copyright by Politechnika Białostocka, Białystok 2026

ISBN 978-83-68673-41-8 (e-Book)
DOI: 10.65731/978-83-68673-41-8



Publikacja jest udostępniona na licencji
Creative Commons Uznanie autorstwa-Użycie niekomercyjne-Bez utworów zależnych 4.0
(CC BY-NC-ND 4.0).

Pełną treść licencji udostępniono na stronie
creativecommons.org/licenses/by-nc-nd/4.0/legalcode.pl.
Publikacja jest dostępna w Internecie na stronie Oficyny Wydawniczej PB.

Spis treści

1. Wstęp	5
2. Grupy i permutacje	7
2.1. Działania i grupy	7
2.1.1. Własności działania	8
2.1.2. Definicja grupy	11
2.1.3. Tabela działania	13
2.2. Struktury algebraiczne	17
2.2.1. Ciało	18
2.2.2. Pierścień	27
2.2.3. Elementy teorii liczb	31
2.3. Permutacje	37
2.3.1. Działania na permutacjach	38
2.3.2. Parzystość permutacji	43
2.3.3. Rozkład na cykle	46
2.4. Własności permutacji	48
2.4.1. Transpozycje	48
2.4.2. Rząd permutacji	52
2.4.3. Równania permutacyjne	54
3. Macierze	57
3.1. Operacje na macierzach	57
3.1.1. Segmentacja macierzy	57
3.1.2. Podstawowe działania na macierzach	62
3.1.3. Macierze charakterystyczne	67
3.2. Układy równań liniowych	69
3.2.1. Operacje elementarne	70
3.2.2. Metoda eliminacji Gaussa	73
3.2.3. Twierdzenie Kroneckera-Capellego	78

3.3. Wyznacznik macierzy	84
3.3.1. Obliczanie wyznacznika	85
3.3.2. Własności wyznacznika	90
3.3.3. Zastosowanie wyznacznika w rozwiązywaniu układów równań	106
3.4. Równania macierzowe	111
3.4.1. Wyznaczanie macierzy z równania	111
3.4.2. Podstawienie bezpośrednie	115
3.4.3. Twierdzenie Cauchy'ego	119
4. Zadania	124
4.1. Zadania z rozwiązaniami	124
4.2. Zadania do samodzielnego rozwiązania	145
4.3. Odpowiedzi do zadań	149
Dodatek A. Spis funkcji występujących w książce	152
Literatura	154
Spis rysunków	155
Spis tabel	155
Streszczenie	157
Abstract	158

1. Wstęp

Algebra była pierwszym przedmiotem, który prowadziłem na Politechnice Białostockiej. Początkowo były to zwykłe ćwiczenia polegające na rozwiązywaniu zadań przy tablicy. Z czasem w programie studiów pojawiły się jednak pracownie specjalistyczne, których celem było pokazanie, jak radzić sobie z podobnymi problemami przy użyciu komputera. W tej książce przedstawiam oba podejścia do zadań – tradycyjne rachunki wykonywane ręcznie oraz implementację rozwiązań w systemie **SageMath**.

W czasie zajęć przekonałem się, jak wiele korzyści daje współpraca. Niektóre tematy znalazły się w moich notatkach tylko dlatego, że ktoś o nie zapytał. Całość powstawała na podstawie kontaktu ze studentami, ich potrzeb dotyczących materiałów, a nawet błędów, które popełniali. Na początku zachęcałem do zadawania pytań, a następnie umieszczałem odpowiedzi na stronie kursu. Dawało nam to motywację do tego, aby wspólnie się czegoś nauczyć. Uważam, że podobnie powinna wyglądać współpraca między rówieśnikami – warto pomagać sobie nawzajem i spędzać razem czas, ponieważ wymiana doświadczeń pozwala dostrzec różne sposoby podejścia do tego samego zagadnienia.

Jak natomiast może wyglądać samodzielna nauka? Często największym wyzwaniem jest samo rozpoczęcie tego procesu. Jest to zrozumiałe – studia wymagają wielkiego wysiłku, ponieważ w ich trakcie z reguły spotykamy się z zagadnieniami, których nigdy wcześniej nie widzieliśmy. Podzielię się metodą, która sprawdzała się u mnie, gdy przygotowywałem się do egzaminów ustnych, kiedy zazwyczaj trzeba było nauczyć się na pamięć kilkudziesięciu stron definicji, twierdzeń i ich dowodów. Postępowałem wtedy zgodnie ze schematem: 1) intuicyjne zrozumienie tematu i powiązań między pojęciami, 2) zapisanie materiału w uporządkowanej formie, 3) nauczenie się go na pamięć oraz – co najważniejsze – 4) ponowne przepisanie wszystkiego z pamięci. Dopiero ten ostatni etap pokazywał, jak wiele szczegółów umyka przy zrozumieniu wyłącznie intuicyjnym.

Każdy ma swój własny styl uczenia się – dlatego warto poszukiwać metod, które działają na nas najlepiej. W dużej mierze to od ilości poświęconego czasu zależy, jak wiele uda nam się osiągnąć. Zdarzenia z przeszłości nie mają aż tak wielkiego wpływu na cały ten proces. Co jednak zrobić, gdy napotkamy barierę, której nie potrafimy przekroczyć? Wtedy pomocna może być rada, którą usłyszałem kiedyś

od jednego z nauczycieli: „Jeśli czegoś nie rozumiesz, naucz się tego na pamięć. Zrozumiesz później”. Przede wszystkim nie warto koncentrować się na własnych ograniczeniach, lecz starać się robić tyle, na ile w danej sytuacji pozwalają nasze możliwości.

W książce omówimy wiele pojęć dotyczących podstaw algebry. Przykładowo w kolejnym rozdziale przyjrzymy się działaniom i ich własnościom, takim jak łączność czy przemienność. Dlaczego warto to robić? Po pierwsze, aby takie pojęcia uporządkować. Pamiętam, że w szkole średniej słyszałem podobne zwroty, jednak wtedy wydawały mi się one bardziej „rozmyte”. Miałem też tendencję, by skupiać się na pytaniu: Co to robi? Obecnie natomiast, gdy spotykam się z nową definicją, zaczynam od pytania: Co to jest? Czy jest to zbiór, liczba, wektor, a może funkcja? Dlatego teraz w definicji działania najważniejszą informacją jest dla mnie to, że jest ono funkcją, a dopiero później zaczynam się zastanawiać, jakie ma własności.

Drugim powodem jest dążenie do uogólnienia tych pojęć. Kiedyś działanie kojarzyło mi się głównie z dodawaniem i odejmowaniem. Okazuje się natomiast, że można je definiować w różny sposób na dowolnych zbiorach (pod warunkiem, że spełnia definicję) – wówczas możemy wykonywać działania nawet na obiektach, które nie są liczbami. Dobrym przykładem są podzielność liczb i podzielność wielomianów – pojęcia te są analogiczne, a jedno stanowi uogólnienie drugiego. Czasem studenci mówią, że na studiach pojawia się jakaś „inna matematyka”. Może tak się wydawać, ponieważ materiał wprowadzany jest szybko i w dużych porcjach. W rzeczywistości jest to jednak naturalna kontynuacja treści ze szkoły średniej, tylko ujęta bardziej ogólnie. Dlatego tak ważne jest dokładne zrozumienie podstaw, aby później nie zgubić się w gąszczu nowych zagadnień.

Skrypt ten powstał z myślą o studentach kierunków ścisłych i stanowi uzupełnienie wykładu z algebry. Szczególne podziękowania kieruję do profesora Ryszarda Mazurka, którego bogate materiały, porządna organizacja zajęć i spokojne, życzliwe podejście wiele mnie nauczyły oraz stanowiły zachętę do dalszego rozwoju.

Rozpoczniemy od przeanalizowania podstawowych struktur algebraicznych, takich jak grupa, ciało i pierścień. Pozwolą one wprowadzić organizację pojęć, co umożliwi nam później tworzenie obiektów o wspólnych cechach. Następnie omówimy macierze oraz ich własności, dzięki czemu poznamy narzędzia potrzebne do analizy bardziej złożonych zagadnień w matematyce wyższej. Materiał na studiach może momentami wydawać się przytłaczający, ale warto podchodzić do niego spokojnie i przyswajać go regularnie, w niewielkich porcjach i we własnym tempie. Warto spojrzeć na proces uczenia się nie jak na obowiązek, lecz jak na okazję do poznania czegoś nowego i ciekawego.

Jesteś wyjątkową i wartościową osobą! 🥰

2. Grupy i permutacje

2.1. Działania i grupy

Jednym z podstawowych narzędzi w matematyce jest zbiór. Jest to tzw. pojęcie pierwotne, którego się nie definiuje. Zbiór rozumiemy intuicyjnie jako zestaw pewnych elementów, którymi mogą być dowolne obiekty (liczby, figury, funkcje itp.). Zbiory możemy podzielić na skończone i nieskończone. Na przykład zbiór $X = \{1, 2, 3\}$ jest skończony i składa się z trzech elementów – są to liczby 1, 2 oraz 3 (piszemy wówczas $1, 2, 3 \in X$). Przykładem zbioru nieskończonego jest natomiast zbiór liczb naturalnych¹ $\mathbb{N} = \{0, 1, 2, \dots\}$.

Działania wprowadzamy, aby wykonywać operacje na elementach danego zbioru. Od tej pory działanie będziemy rozumieć jako konkretne odwzorowanie – czyli funkcję. Zobaczmy to na przykładzie – dobrze wiemy, że spełniona jest równość $4 + 3 = 7$. Możemy odczytać ją następująco: funkcja o nazwie „+” przyjmuje dwa argumenty będące liczbami naturalnymi 4 i 3, a następnie zwraca wynik równy 7. Zatem standardowe działanie „+” można zapisać jako odwzorowanie:

$$+: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N},$$

które jako argumenty przyjmuje dwie dowolne liczby naturalne² i zwraca wynik należący do zbioru liczb naturalnych.

Wyróżniamy dwa istotne rodzaje działań: addytywne i multiplikatywne. Pierwsze z nich zachowuje się jak standardowe działanie dodawania „+”. Powiemy, że w zbiorze X , na którym jest ono określone, istnieje pewien odpowiednik liczby 0, który nie zmienia wyniku działania. Ponadto dla każdego elementu $x \in X$ można wskazać element przeciwny $(-x)$, spełniający warunek $x + (-x) = 0$. Działanie multiplikatywne jest natomiast podobne do standardowego działania mnożenia „·” – w zbiorze X znajduje się wtedy element 1, który nie wpływa na wynik tego działania,

¹ Czasem przyjmuje się, że zbiór liczb naturalnych nie zawiera liczby 0, dlatego zawsze należy doprecyzować to pojęcie.

² Produkt kartezjański $X \times Y = \{(x, y): x \in X \wedge y \in Y\}$ jest zbiorem par uporządkowanych, więc elementem zbioru $\mathbb{N} \times \mathbb{N}$ jest dowolna para liczb naturalnych (m, n) , gdzie $m, n \in \mathbb{N}$.

oraz dla każdego elementu $x \in X$ (różnego od 0) istnieje element odwrotny x^{-1} , taki że $x \cdot x^{-1} = 1$.

W tym rozdziale uporządkujemy podstawowe pojęcia związane z działaniami i na tej podstawie wprowadzimy definicję grupy. Dzięki temu zobaczymy, że różne obiekty (takie jak liczby, funkcje czy macierze) mogą mieć pewne wspólne własności. Umożliwi nam to traktowanie ich w podobny sposób i posługiwanie się jednym wspólnym językiem.

2.1.1. Własności działania

Definicja. Działaniem wewnętrznym w zbiorze X nazywamy odwzorowanie:

$$\circ : X \times X \rightarrow X.$$

Wynik działania³ dla dwóch elementów $x, y \in X$ to wartość tego odwzorowania, którą zapisujemy jako $\circ(x, y)$ lub prościej: $x \circ y$.

Przykład 2.1

Działanie standardowego dodawania „+” nie jest wewnętrzne w zbiorze $X = \{0, 1, 2\}$, ponieważ możemy wskazać dwa elementy $1, 2 \in X$ należące do tego zbioru, dla których wynik działania, czyli:

$$1 + 2 = 3 \notin X,$$

nie należy do zbioru X .

Wprowadzimy teraz nazewnictwo dotyczące własności działania wewnętrznego „ $\circ$ ”:

Definicja. Mówimy, że działanie „ $\circ$ ” określone na zbiorze X :

- jest przemienne, gdy:

$$\forall x, y \in X: x \circ y = y \circ x;$$

- jest łączne, gdy:

$$\forall x, y, z \in X: (x \circ y) \circ z = x \circ (y \circ z);$$

³ Działanie można oznaczyć dowolną literą lub symbolem. Tutaj stosujemy symbol „ $\circ$ ”, aby uniknąć skojarzeń ze znanymi symbolami „+” i „ $\cdot$ ”.

- ma element neutralny $e \in X$, gdy:

$$\exists e \in X \quad \forall x \in X: x \circ e = e \circ x = x.$$

Uwaga.

1. Jeżeli w zbiorze X z określonym działaniem „ $\circ$ ” istnieje element neutralny, to jest on wyznaczony jednoznacznie.
2. W definicji elementu neutralnego kwantyfikator „istnieje” ($\exists$) znajduje się przed kwantyfikatorem „dla każdego” ($\forall$). Oznacza to, że powinien istnieć jeden element ($e \in X$) zbioru X , taki że dla dowolnego elementu ($x \in X$) zbioru X zachodzi warunek definicji. W szczególności element neutralny e nie zależy od x , ponieważ kwantyfikator $\exists$ znajduje się przed wystąpieniem zmiennej x .

Przykład 2.2

Znajdziemy element neutralny działania określonego dla dowolnych liczb $x, y \in \mathbb{R}$ wzorem:

$$x \circ y \stackrel{\text{def}}{=} x + y - 2.$$

Rozwiązujemy równanie $x \circ e = x$, aby określić zestaw kandydatów na element neutralny. Korzystając z definicji działania „ $\circ$ ”, otrzymujemy zależność:

$$x + e - 2 = x.$$

Po wyznaczeniu zmiennej e z tej równości⁴ uzyskujemy:

$$e = 2.$$

W **Sage** zaczynamy od zdefiniowania działania „ $\circ$ ”. W poniższym kodzie oznaczamy je literą d :

```
d(x, y) = x + y - 2
d
Out: (x, y) |--> x + y - 2
```

⁴ Stąd wynika, że tylko liczba $e = 2$ może być elementem neutralnym. Aby to udowodnić, należy sprawdzić obie równości w definicji, ponieważ ogólnie działanie nie musi być przemienne.

W tym przypadku nie ma potrzeby definiowania zmiennych x i y jako symboli za pomocą polecenia `var('x y')`, ponieważ argumenty funkcji `d` są automatycznie traktowane jako symbole. Następnie stosujemy funkcję `solve()`, aby znaleźć rozwiązanie równania w definicji elementu neutralnego:

```
solve( d(x, e) == x, e )
```

```
Out: [e == 2]
```

Definicja. Jeżeli istnieje element neutralny $e \in X$ działania „ $\circ$ ” określonego w zbiorze X , to elementem odwrotnym⁵ do elementu $x \in X$ nazywamy element $x^{-1} \in X$, taki że:

$$x \circ x^{-1} = x^{-1} \circ x = e.$$

Istnienie elementu odwrotnego można zapisać za pomocą kwantyfikatorów w postaci:

$$\forall x \in X \exists x^{-1} \in X: x \circ x^{-1} = x^{-1} \circ x = e.$$

Uwaga.

1. Jeżeli działanie „ $\circ$ ” jest łączne w zbiorze X i istnieje element neutralny $e \in X$ tego działania, to element odwrotny $x^{-1} \in X$ do danego elementu $x \in X$ jest wyznaczony jednoznacznie oraz zachodzi wzór $(x^{-1})^{-1} = x$.

2. W definicji elementu odwrotnego kwantyfikator „dla każdego” ($\forall$) znajduje się przed kwantyfikatorem „istnieje” ($\exists$). Oznacza to, że dla dowolnego elementu ($x \in X$) zbioru X powinien istnieć osobny element odwrotny ($x^{-1} \in X$), spełniający warunek definicji. W szczególności element odwrotny x^{-1} zależy od x , ponieważ kwantyfikator $\exists$ znajduje się po wystąpieniu zmiennej x .

Twierdzenie. Jeżeli działanie „ $\circ$ ” określone w zbiorze X jest łączne, posiada element neutralny $e \in X$ i istnieją elementy odwrotne do $x, y \in X$, to istnieje też element odwrotny do $x \circ y$ i zachodzi wzór:

$$(x \circ y)^{-1} = y^{-1} \circ x^{-1}.$$

⁵ Dla działania addytywnego stosujemy nazwę element przeciwny, natomiast dla działania mnożliwego – element odwrotny.

Przykład 2.3

Znajdziemy element odwrotny do dowolnego elementu $x \in \mathbb{R}$ dla działania zdefiniowanego w przykładzie 2.2.

Zaczynamy od rozwiązania równania $x \circ x^{-1} = e$ występującego w definicji elementu odwrotnego. Z przykładu 2.2 wiemy, że element neutralny jest równy $e = 2$. Zatem na podstawie definicji działania „ $\circ$ ” mamy do rozwiązania równanie:

$$x + x^{-1} - 2 = 2.$$

Wyznaczamy z niego zmienną x^{-1} i otrzymujemy wzór pozwalający obliczyć element odwrotny do dowolnego elementu $x \in \mathbb{R}$ jako:

$$x^{-1} = 4 - x.$$

W **Sage** ponownie znajdujemy ten wzór za pomocą funkcji **solve()**. Tutaj element odwrotny oznaczamy literą y :

```
solve( d(x, y) == 2, y )
```

```
Out: [y == -x + 4]
```

2.1.2. Definicja grupy

Definicja. Grupą nazywamy parę $(X, \circ)$, gdzie X jest zbiorem, a „ $\circ$ ” – działaniem wewnętrznym w X , gdy spełnione są następujące warunki:

1. Działanie „ $\circ$ ” jest **łączne**:

$$\forall x, y, z \in X: (x \circ y) \circ z = x \circ (y \circ z).$$

2. W zbiorze X istnieje **element neutralny** $e \in X$:

$$\exists e \in X \quad \forall x \in X: x \circ e = e \circ x = x.$$

3. Dla każdego elementu $x \in X$ istnieje element odwrotny $x^{-1} \in X$:

$$\forall x \in X \quad \exists x^{-1} \in X: x \circ x^{-1} = x^{-1} \circ x = e.$$

Jeżeli dodatkowo zachodzi warunek:

4. Działanie „ $\circ$ ” jest **przemienne**:

$$\forall x, y \in X: x \circ y = y \circ x,$$

to grupę $(X, \circ)$ nazywamy grupą abelową (lub grupą przemianą).

Przykład 2.4

Zbiór $X = \mathbb{R} \setminus \left\{-\frac{5}{2}\right\}$ jest grupą z działaniem określonym dla dowolnych liczb $x, y \in X$ wzorem:

$$x \circ y \stackrel{\text{def}}{=} 2xy + 5x + 5y + 10.$$

Znajdziemy wynik działania $2^{-1} \circ 1$.

Aby wyznaczyć element neutralny działania „ $\circ$ ”, należy rozwiązać równanie $x \circ e = x$. Na podstawie definicji działania otrzymujemy:

$$2xe + 5x + 5e + 10 = x.$$

Niewiadomą w tym równaniu jest zmienna e , dlatego wyłączamy ją przed nawias po lewej stronie i przenosimy pozostałe składniki na stronę prawą:

$$e \cdot (2x + 5) = -4x - 10.$$

Zgodnie z definicją zbioru X , liczba $-\frac{5}{2}$ nie należy do tego zbioru, zatem możemy podzielić równanie stronami przez czynnik $(2x + 5)$. Uzyskujemy:

$$e = \frac{-2 \cdot (2x + 5)}{2x + 5}.$$

Skracamy ułamek, ponieważ element neutralny nie może zależeć od zmiennej x . Stąd:

$$e = -2.$$

Od tej pory element $e = -2 \in X$ ma podobną interpretację jak liczba 1 dla standardowego mnożenia. Teraz możemy napisać wzór na element odwrotny do dowolnej liczby $x \in X$. W tym celu rozwiązujemy równanie $x \circ x^{-1} = e$ i otrzymujemy kolejno:

$$2xx^{-1} + 5x + 5x^{-1} + 10 = -2 \quad \rightarrow \quad x^{-1} \cdot (2x + 5) = -5x - 12.$$

Ponownie możemy wykonać dzielenie przez wyrażenie $(2x + 5)$, gdyż liczba $-\frac{5}{2}$ nie należy do zbioru X . Tym razem element odwrotny x^{-1} zależy od zmiennej x . Mamy:

$$x^{-1} = \frac{-5x - 12}{2x + 5}.$$

Możemy teraz przejść do wyznaczenia wyniku działania $2^{-1} \circ 1$. Zaczynamy od znalezienia liczby 2^{-1} na podstawie uzyskanego wzoru na element odwrotny:

$$2^{-1} = \frac{-5 \cdot 2 - 12}{2 \cdot 2 + 5} = \frac{-22}{9} = -\frac{22}{9}.$$

Następnie wykonujemy żądany rachunek, wykorzystując definicję działania:

$$2^{-1} \circ 1 = \left(-\frac{22}{9}\right) \circ 1 \stackrel{\text{def}}{=} 2 \cdot \left(-\frac{22}{9}\right) \cdot 1 + 5 \cdot \left(-\frac{22}{9}\right) + 5 \cdot 1 + 10 = -\frac{19}{9}.$$

W **Sage** zaczynamy od definicji działania, a następnie wyznaczamy element neutralny, korzystając z funkcji **solve()**:

```
d(x, y) = 2*x*y + 5*x + 5*y + 10
```

```
solve( d(x, e) == x, e )
```

```
Out: [e == -2]
```

Następnie zapisujemy wzór funkcji **f** pozwalającej wyznaczyć element odwrotny do przekazanego argumentu. Indeks [0] oznacza, że bierzemy równość będącą pierwszym elementem listy zwracanej przez funkcję **solve()** (zobacz przykład 2.3). Funkcja **rhs()** zwraca natomiast prawą stronę tej równości – jest to wzór na element odwrotny:

```
f(x) = solve( d(x, y) == -2, y )[0].rhs()
```

```
f
```

```
Out: x |--> -(5*x + 12)/(2*x + 5)
```

Teraz możemy obliczyć wartość 2^{-1} , stosując zapis **f(2)**. Zatem szukany wynik działania otrzymujemy za pomocą polecenia:

```
d( f(2), 1 )
```

```
Out: -19/9
```

2.1.3. Tabela działania

Przeanalizujemy przykłady grup nieskończonych – są to struktury, dla których zbiór **X** w definicji grupy ma nieskończenie wiele elementów. Podzielimy je ze względu na rodzaj działania. Wyniki zostały zebrane w tabeli 2.1:

Tabela 2.1. Przykłady grup nieskończonych

	Przykłady grup	Element neutralny	Element odwrotny
Standardowe dodawanie „+”	$(\mathbb{Z}, +), (\mathbb{Q}, +),$ $(\mathbb{R}, +), (\mathbb{C}, +)$	$e = 0$	$x^{-1} = -x$
Standardowe mnożenie „·”	$(\mathbb{Q} \setminus \{0\}, \cdot),$ $(\mathbb{R} \setminus \{0\}, \cdot),$ $(\mathbb{C} \setminus \{0\}, \cdot)$	$e = 1$	$x^{-1} = \frac{1}{x}$
Złożenie funkcji	$(\mathcal{F}, \circ)$, gdzie $\mathcal{F}$ – zbiór bijekcji ⁶ $f: \mathbb{R} \rightarrow \mathbb{R}$	funkcja identycznościowa $f: x \mapsto x$	funkcja odwrotna ⁷ f^{-1}

Uwaga. Grupa $(\mathcal{F}, \circ)$ przedstawiona w ostatnim wierszu tabeli nie jest grupą abelową. Oznacza to, że działanie złożenia funkcji „ $\circ$ ” nie jest przemienne. W dalszej części książki poznamy grupę permutacji, która również wykorzystuje działanie złożenia.

Jeżeli grupa jest skończona, to możemy przedstawić tabelę działania w tej grupie. Jest to zestawienie wszystkich możliwych wyników działań dla każdej pary elementów ze zbioru X . Przykładem grupy skończonej jest tzw. grupa reszt modulo n , czyli struktura $(\mathbb{Z}_n, +_n)$, gdzie

$$\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$$

jest zbiorem wszystkich reszt z dzielenia liczby całkowitej przez dodatnią liczbę n , natomiast działanie „ $+_n$ ” jest określone wzorem:

$$x +_n y \stackrel{\text{def}}{=} (x + y) \bmod n, \quad \text{dla } x, y \in \mathbb{Z}_n.$$

W praktyce oznacza to, że najpierw wykonujemy działanie standardowego dodawania, a następnie podajemy resztę z dzielenia (czyli modulo) uzyskanego wyniku przez n .

Przykład 2.5

Narysujemy tabelę działania w grupie $(\mathbb{Z}_5, +_5)$.

⁶ Bijekcja to funkcja różnowartościowa $f: X \rightarrow Y$, której zbiór wartości $f(X)$ jest równy przeciwdziedzinie Y .

⁷ Wykres funkcji odwrotnej f^{-1} jest symetryczny do wykresu funkcji f względem prostej $y = x$.

Grupa modulo n jest reprezentowana w **Sage** za pomocą obiektu **Integers()**. Dlatego, aby zdefiniować zbiór $\mathbb{Z}_5$, zapisujemy:

```
R = Integers(5)
```

```
R
```

```
Out: Ring of integers modulo 5
```

Otrzymujemy komunikat, że jest to pierścień liczb całkowitych modulo 5. Więcej o strukturze pierścienia powiemy w następnym rozdziale. Teraz zobaczmy, jak wygląda operator standardowego dodawania:

```
operator.add
```

```
Out: <built-in function add>
```

System **SageMath** rozumie domyślnie, że w połączeniu ze zbiorem R jest to działanie dodawania modulo 5. Podobnie możemy zdefiniować działanie standardowego mnożenia, pisząc: `operator.mul`⁸. Powyższe obiekty – zbiór i działanie – umieszczamy na koniec jako kolejne argumenty funkcji **OperationTable()**, która rysuje tabelę działania w powstałej grupie:

```
from sage.matrix.operation_table import OperationTable
OperationTable( R, operator.add, names='elements' )
```

Stosujemy tutaj składnię z języka **Python**, wykorzystywaną przy importowaniu obiektów z bibliotek. Najpierw piszemy, skąd (**from**) importujemy dane narzędzie, a następnie podajemy jego nazwę po słowie kluczowym **import**. Ostatni argument funkcji **OperationTable()** – `names='elements'` – określa jedynie, że wyświetlane wartości będą elementami zbioru R (domyślnie są to małe litery alfabetu). Otrzymujemy wynik:

```
Out: +  0 1 2 3 4
      +-----
```

⁸ Nazwy działań wynikają z angielskich tłumaczeń słów: dodawanie (**Addition**) i mnożenie (**Multiplication**).

0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Z uzyskanej tabeli możemy m.in. odczytać, że:

- Wynik działania $3 +_5 4$ jest równy 2 – jest to liczba znajdująca się na przecięciu wiersza z liczbą 3 i kolumny z liczbą 4 (ogólnie: pierwszy argument działania znajduje się w wierszu, a drugi – w kolumnie).
- Elementem neutralnym działania jest liczba $e = 0$ – w wierszu dla liczby 0 wartości są takie same jak nazwy kolumn.
- Elementem odwrotnym do liczby $x = 2$ jest liczba $x^{-1} = 3$ – na przecięciu wiersza dla 2 i kolumny dla 3 znajduje się element neutralny $e = 0$.
- Grupa jest abelowa – tabela jest symetryczna względem głównej przekątnej (od lewego górnego rogu do prawego dolnego).

Przykład 2.6

Narysujemy tabelę działania dla grupy $(\mathbb{Z}_{11}, \oplus)$, gdzie działanie „ $\oplus$ ” jest określone dla $x, y \in \mathbb{Z}_{11}$ wzorem:

$$x \oplus y \stackrel{\text{def}}{=} (x + y + 2) \bmod 11.$$

W celu utworzenia tabeli korzystamy z poleceń:

```
R = Integers(11)
OperationTable( R, lambda x, y: x + y + 2, names='elements' )
```

Pierwszym argumentem funkcji **OperationTable()**, której nie musimy już ponownie importować, jest zbiór **Integers(11)**. Drugim argumentem powinna być funkcja reprezentująca działanie w grupie. Ponieważ działanie jest niestandardowe, wykorzystujemy w tym celu funkcję anonimową **lambda** – po dwukropku podajemy definicję działania. System **SageMath** wie domyślnie, że to działanie wykonywane jest modulo 11, ponieważ działamy na elementach zbioru $\mathbb{Z}_{11}$. Otrzymujemy wynik:

```

Out: .    0 1 2 3 4 5 6 7 8 9 10
        +-----+
0| 2 3 4 5 6 7 8 9 10 0 1
1| 3 4 5 6 7 8 9 10 0 1 2
2| 4 5 6 7 8 9 10 0 1 2 3
3| 5 6 7 8 9 10 0 1 2 3 4
4| 6 7 8 9 10 0 1 2 3 4 5
5| 7 8 9 10 0 1 2 3 4 5 6
6| 8 9 10 0 1 2 3 4 5 6 7
7| 9 10 0 1 2 3 4 5 6 7 8
8| 10 0 1 2 3 4 5 6 7 8 9
9| 0 1 2 3 4 5 6 7 8 9 10
10| 1 2 3 4 5 6 7 8 9 10 0

```

2.2. Struktury algebraiczne

Poznaliśmy już najprostszą strukturę algebraiczną – grupę, czyli parę składającą się ze zbioru oraz funkcji zwanej działaniem. Teraz wprowadzimy dwie bardziej złożone konstrukcje: pierścień i ciało. W przeciwieństwie do grupy są one opisywane jako trójki, w skład których wchodzi: zbiór, działanie addytywne i działanie multiplikatywne.

Ciało to jedna z najważniejszych struktur algebraicznych, ponieważ pozwala wykonywać wszystkie podstawowe operacje arytmetyczne: dodawanie, odejmowanie, mnożenie i dzielenie (z wyjątkiem dzielenia przez zero). W pierwszej części omawialiśmy już przykład ciała liczb zespolonych, czyli struktury złożonej ze zbioru liczb zespolonych $\mathbb{C}$ oraz standardowych działań dodawania i mnożenia. Ciała pojawiają się jednak w wielu innych dziedzinach, takich jak:

- analiza matematyczna – przy całkach, pochodnych i szeregach potęgowych;
- kryptografia – gdzie ważną rolę odgrywają ciała skończone i algorytmy szyfrujące;
- informatyka – na przykład w kodach korekcyjnych czy arytmetyce komputerowej;
- fizyka – zarówno klasyczna, jak i kwantowa.

W samej algebrze pojęcie ciała będzie dla nas niezbędne do wprowadzenia przestrzeni liniowych, które będziemy omawiać w kolejnych częściach.

W tym rozdziale przedstawimy również sposoby wykonywania rachunków w ciałach modulo n , które otrzymujemy poprzez rozszerzenie wcześniej omawianej grupy $(\mathbb{Z}_n, +_n)$ o działanie „ $\cdot_n$ ” mnożenia modulo n . Omówimy także algorytm Euklidesa, służący do wyznaczania elementów odwrotnych w danym ciele.

2.2.1. Ciało

Definicja. Ciałem nazywamy trójkę $(\mathbb{K}, +, \cdot)$, gdzie $\mathbb{K}$ jest zbiorem, a „ $+$ ” i „ $\cdot$ ” są działaniami wewnętrznymi w $\mathbb{K}$, gdy spełnione są następujące warunki:

1. Aksjomaty **addytywne**:

(A1) Działanie „ $+$ ” jest **łączne**:

$$\forall x, y, z \in \mathbb{K}: (x + y) + z = x + (y + z).$$

(A2) Działanie „ $+$ ” jest **przemienne**:

$$\forall x, y \in \mathbb{K}: x + y = y + x.$$

(A3) W zbiorze $\mathbb{K}$ istnieje **element neutralny** $0 \in \mathbb{K}$ działania addytywnego:

$$\exists 0 \in \mathbb{K} \forall x \in \mathbb{K}: x + 0 = x.$$

(A4) Do każdego elementu $x \in \mathbb{K}$ istnieje **element przeciwny** $(-x) \in \mathbb{K}$:

$$\forall x \in \mathbb{K} \exists (-x) \in \mathbb{K}: x + (-x) = 0.$$

2. Aksjomaty **multiplikatywne**:

(M1) Działanie „ $\cdot$ ” jest **łączne**:

$$\forall x, y, z \in \mathbb{K}: (x \cdot y) \cdot z = x \cdot (y \cdot z).$$

(M2) Działanie „ $\cdot$ ” jest **przemienne**:

$$\forall x, y \in \mathbb{K}: x \cdot y = y \cdot x.$$

(M3) W zbiorze $\mathbb{K}$ istnieje **element neutralny** $1 \in \mathbb{K}$ działania multiplikatywnego:

$$\exists 1 \in \mathbb{K} \forall x \in \mathbb{K}: x \cdot 1 = x.$$

(M4) Do każdego elementu $x \in \mathbb{K} \setminus \{0\}$ istnieje **element odwrotny** $x^{-1} \in \mathbb{K}$:

$$\forall x \in \mathbb{K} \setminus \{0\} \exists x^{-1} \in \mathbb{K}: x \cdot x^{-1} = 1.$$

3. Aksjomat **rozdzielności**:

(D) Mnożenie „ $\cdot$ ” jest **rozdzielne** względem dodawania „ $+$ ”:

$$\forall x, y, z \in \mathbb{K}: x \cdot (y + z) = x \cdot y + x \cdot z.$$

Uwaga. Struktury $(\mathbb{K}, +)$ oraz $(\mathbb{K} \setminus \{0\}, \cdot)$ są grupami abelowymi.

Twierdzenie. Weźmy zbiór $\mathbb{Z}_p = \{0, 1, 2, \dots, p-1\}$ z określonymi na nim działaniami:

- „+” – dodawania modulo p ;
- „ $\cdot$ ” – mnożenia modulo p .

Wtedy trójka $(\mathbb{Z}_p, +, \cdot)$ jest ciałem reszt modulo p tylko wtedy, gdy p jest liczbą pierwszą.

Przykład 2.7

W ciele⁹ $\mathbb{Z}_{23}$ wykonamy działanie:

$$11 \cdot 5^8.$$

Podczas wykonywania rachunków będziemy wielokrotnie stosowali następującą zasadę: w ciele $\mathbb{Z}_p$ możemy bezkarnie dodawać lub odejmować dowolną wielokrotność liczby p – operacja ta nie zmienia wyniku działania. Tę regułę można stosować na dowolnym etapie obliczeń, z pominięciem wykładnika potęgi. Zatem w ciele $\mathbb{Z}_{23}$ możemy zawsze dodać lub odjąć 23 dla uproszczenia rachunków. Zaczniemy od rozpisania potęgi 5^8 , tak aby w jej rozkładzie na czynniki znaleźć liczbę w pobliżu pewnej wielokrotności 23:

$$11 \cdot 5^8 = 11 \cdot (5^2)^4 = 11 \cdot 25^4.$$

Teraz zmniejszamy podstawę potęgi o 23 i kontynuujemy wykonywanie działań:

$$\dots = 11 \cdot 2^4 = 11 \cdot 16.$$

W tym momencie możemy odjąć od liczby 16 wartość 23, aby znaleźć się bliżej zera:

$$\dots = 11 \cdot (-7) = -77.$$

Jako wynik działania zawsze należy podać liczbę ze zbioru $\mathbb{Z}_{23}$. Możemy to zrobić, dodając wielokrotność $4 \cdot 23$ do uzyskanego wyniku:

$$\dots = -77 + 4 \cdot 23 = -77 + 92 = 15.$$

⁹ Dla uproszczenia ciałem będziemy nazywać sam zbiór – bez zapisywania trójki z działaniami.

W **Sage** ciało modulo p reprezentowane jest za pomocą obiektu **Zmod()**. Poniżej definiujemy funkcję R , która umożliwi wykonywanie dowolnych rachunków w tym ciele:

```
R = Zmod(23)
R( 11 * 5^8 )
Out: 15
```

Twierdzenie (małe twierdzenie Fermata). Dla każdej liczby $a \in \mathbb{Z}_p \setminus \{0\}$, gdzie p jest liczbą pierwszą, zachodzi równość:

$$a^{p-1} = 1.$$

Przykład 2.8

W ciele $\mathbb{Z}_{73}$ obliczymy:

$$3^{160}.$$

Liczba 73 jest liczbą pierwszą, zatem możemy skorzystać z małego twierdzenia Fermata. Ponieważ $p = 73$, mamy $p - 1 = 72$. Twierdzenie można rozumieć w następujący sposób: do wykładnika potęgi możemy bezkarnie dodawać (lub odejmować) dowolną wielokrotność liczby $(p - 1)$ – ta operacja nie zmienia wyniku działania. W ten sposób zmniejszamy wykładnik potęgi:

$$3^{160} = 3^{160-2 \cdot 72} = 3^{16}.$$

Następnie stosujemy sposoby poznane w przykładzie 2.7 – do podstawy potęgi zawsze możemy dodać (lub odjąć) wartość 73:

$$\dots = (3^4)^4 = 81^4 = 8^4 = (8^2)^2 = 64^2 = (-9)^2 = 9^2 = 81 = 8.$$

W **Sage** wszystkie rachunki w ciele modulo p możemy wykonywać za pomocą poniższego kodu:

```
R = Zmod(73)
R( 3^160 )
Out: 8
```

Przykład 2.9

Narysujemy tabelę dla działania multiplikatywnego w ciele $\mathbb{Z}_{17}$.

Wykorzystujemy kod z przykładu 2.5, jednak tym razem stosujemy operator mnożenia, czyli operator.mul:

```
from sage.matrix.operation_table import OperationTable
R = Integers(17)
OperationTable( R, operator.mul, names='elements' )
Out: * 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
      +-----+
0| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1| 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
2| 0 2 4 6 8 10 12 14 16 1 3 5 7 9 11 13 15
3| 0 3 6 9 12 15 1 4 7 10 13 16 2 5 8 11 14
4| 0 4 8 12 16 3 7 11 15 2 6 10 14 1 5 9 13
5| 0 5 10 15 3 8 13 1 6 11 16 4 9 14 2 7 12
6| 0 6 12 1 7 13 2 8 14 3 9 15 4 10 16 5 11
7| 0 7 14 4 11 1 8 15 5 12 2 9 16 6 13 3 10
8| 0 8 16 7 15 6 14 5 13 4 12 3 11 2 10 1 9
9| 0 9 1 10 2 11 3 12 4 13 5 14 6 15 7 16 8
10| 0 10 3 13 6 16 9 2 12 5 15 8 1 11 4 14 7
11| 0 11 5 16 10 4 15 9 3 14 8 2 13 7 1 12 6
12| 0 12 7 2 14 9 4 16 11 6 1 13 8 3 15 10 5
13| 0 13 9 5 1 14 10 6 2 15 11 7 3 16 12 8 4
14| 0 14 11 8 5 2 16 13 10 7 4 1 15 12 9 6 3
15| 0 15 13 11 9 7 5 3 1 16 14 12 10 8 6 4 2
16| 0 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

Przykład 2.10

W ciele $\mathbb{Z}_{17}$ znajdziemy element odwrotny do liczby 10, czyli 10^{-1} .

Element odwrotny można łatwo odczytać z wyniku uzyskanego w przykładzie 2.9. Wystarczy znaleźć kolumnę, dla której zachodzi równość:

$$10 \cdot \square = 1.$$

Dostrzegamy, że jedynka znajduje się na przecięciu wiersza z liczbą 10 i kolumny z liczbą 12. Wobec tego szukany element odwrotny to:

$$10^{-1} = 12.$$

Co jednak zrobić w sytuacji, gdy nie możemy odgadnąć, jaką liczbę należy wpisać w puste pole w definicji elementu odwrotnego? Wówczas możemy wyznaczyć tę wartość za pomocą algorytmu Euklidesa. Podzielimy ten proces na dwa kroki:

1. Dzielenie z resztą.

Wykonujemy kolejne dzielenia z resztą liczby n przez liczbę d , stosując zapis:

$$n = d \cdot k + r, \text{ gdzie } r = 0, 1, 2, \dots, d - 1.$$

W pierwszym kroku $n = 17$ jest indeksem dolnym w ciele, natomiast $d = 10$ jest liczbą, dla której szukamy elementu odwrotnego. Dzielenie wykonujemy do momentu uzyskania reszty równej 1:

$$17 = 10 \cdot 1 + 7$$

$$10 = 7 \cdot 1 + 3$$

$$7 = 3 \cdot 2 + 1.$$

2. Rekonstrukcja.

Wykonujemy na przemian dwie czynności, zaczynając od ostatniej równości z pierwszego kroku: (1) wyznaczamy resztę oraz (2) upraszczamy wyrażenie:

$$1 \stackrel{(1)}{=} 7 - 2 \cdot 3 \stackrel{(1)}{=} 7 - 2 \cdot (10 - 7) \stackrel{(2)}{=} -2 \cdot 10 + 3 \cdot 7$$

$$\stackrel{(1)}{=} -2 \cdot 10 + 3 \cdot (17 - 10) \stackrel{(2)}{=} 3 \cdot 17 - 5 \cdot 10.$$

Ostatecznie otrzymujemy równość:

$$1 = 3 \cdot 17 - 5 \cdot 10.$$

W ciele $\mathbb{Z}_{17}$ liczba $3 \cdot 17$ jest równa 0, ponieważ jest to wielokrotność liczby 17. Zatem algorytmicznie uzyskaliśmy równość zapisaną na początku przykładu, czyli:

$$10 \cdot \boxed{-5} = 1.$$

Wynik powinien należeć do zbioru $\mathbb{Z}_{17}$, stąd szukany element odwrotny jest równy:

$$10^{-1} = -5 + 17 = 12.$$

W **Sage** za pomocą kodu przedstawionego w poprzednich przykładach można także znaleźć element odwrotny. Wystarczy zastosować polecenia:

```
R = Zmod(17)
```

```
R( 10^-1 )
```

```
Out: 12
```

W definicji ciała jawnie wyróżnia się tylko dwa działania: dodawanie i mnożenie. Aksjomaty (A4) i (M4) zapewniają jednak pośrednio istnienie odejmowania i dzielenia. Operacje te rozumiemy odpowiednio jako dodawanie elementu przeciwnego oraz mnożenie przez element odwrotny (różny od zera).

Przykład 2.11

W ciele $\mathbb{Z}_{29}$ rozwiążemy równanie:

$$24x = 26.$$

Zaczynamy od podzielenia równania stronami przez 2. Formalnie tę operację rozumiemy jako mnożenie przez element odwrotny: 2^{-1} . Wówczas po obu stronach równania pojawia się wyrażenie $2^{-1} \cdot 2$, a to jest liczba 1, czyli element neutralny mnożenia, który nie zmienia wyniku działania:

$$12x = 13.$$

Równanie liniowe w ciele postaci $ax = b$, gdzie $a \neq 0$, ma zawsze jedno rozwiązanie. Otrzymujemy je, mnożąc równanie stronami przez element odwrotny a^{-1} . W naszym przypadku stosujemy mnożenie przez liczbę 12^{-1} :

$$x = 12^{-1} \cdot 13.$$

Element 12^{-1} możemy wyznaczyć za pomocą algorytmu Euklidesa. Czasami jednak da się odgadnąć tę wartość inaczej, stosując pewne przekształcenia. Zaczniemy od zapisania rozwiązania równania w postaci:

$$x = 3^{-1} \cdot 4^{-1} \cdot 13.$$

Na podstawie definicji elementu odwrotnego liczba 3^{-1} stanowi brakujący czynnik w poniższej równości:

$$3 \cdot \square = 1.$$

Zgadujemy, że jest to liczba 10, ponieważ w ciele $\mathbb{Z}_{29}$ mamy $3 \cdot 10 = 30 = 1$. Zasadniczo w ciele $\mathbb{Z}_p$ łatwo można odgadnąć element odwrotny poprzez znalezienie liczby, dla której wynik działania jest równy $p + 1$. Dla liczby 4^{-1} natomiast stosujemy inne podejście – zapisujemy tę wartość w postaci ułamka i korzystamy z tego, że w liczniku możemy dodawać lub odejmować dowolną wielokrotność liczby 29:

$$4^{-1} = \frac{1}{4} = \frac{1 - 29}{4} = \frac{-28}{4} = -7 = 22.$$

Na koniec wstawiamy uzyskane elementy odwrotne do równości, w której wyznaczyliśmy zmienną x , oraz wykonujemy rachunki. Warto zaznaczyć, że rozwiązanie równania musi być elementem zbioru $\mathbb{Z}_{29}$ – oznacza to, że w odpowiedzi nie możemy podać na przykład ułamka czy liczby ujemnej:

$$\begin{aligned} x &= 10 \cdot 22 \cdot 13 = 10 \cdot (-7) \cdot 13 = 10 \cdot (-91) \\ &= 10 \cdot (-91 + 87) = 10 \cdot (-4) = -40 = 18. \end{aligned}$$

Sprawdzamy uzyskany wynik, korzystając z systemu **SageMath**. Wprowadzamy wartość $x = 24^{-1} \cdot 26$, którą uzyskujemy poprzez przekształcenie początkowego równania:

```
R = Zmod(29)
R( 24^-1 * 26 )
Out: 18
```

Definicja. Mówimy, że liczba $a \in \mathbb{Z}_p$ jest resztą kwadratową modulo p , jeżeli istnieje element $x \in \mathbb{Z}_p$ spełniający równość:

$$x^2 = a.$$

Jeśli warunek ten nie zachodzi, liczbę a nazywamy nieresztą kwadratową.

Liczba $a \in \mathbb{Z}_p$ jest resztą kwadratową modulo p (gdzie $p > 2$) tylko wtedy, gdy:

$$a^{\frac{p-1}{2}} = 1.$$

Jeśli natomiast spełniony jest warunek $p \equiv 3 \pmod{4}$ (czyli reszta z dzielenia liczby p przez 4 jest równa 3), to pierwiastki wyrażają się wzorem (zob. [5]):

$$x = \pm a^{\frac{p+1}{4}}.$$

Przykład 2.12

Wyznamy reszty kwadratowe modulo 31.

Wystarczy znaleźć wszystkie liczby postaci x^2 , gdzie $x \in \mathbb{Z}_{31}$. Wykonujemy ten rachunek, korzystając ze składni listy składanej. Tutaj zamiast nawiasów kwadratowych ($[]$) oznaczających listę stosujemy nawiasy klamrowe ($\{\}$) określające zbiór (Set) w języku **Python**. To pozwala nam pominąć powtarzające się reszty:

```
reszty = {x^2 for x in Zmod(31)}
```

```
reszty
```

```
Out: {0, 1, 2, 4, 5, 7, 8, 9, 10, 14, 16, 18, 19, 20, 25, 28}
```

Równanie kwadratowe $ax^2 + bx + c = 0$, gdzie $a, b, c \in \mathbb{Z}_p$, ma rozwiązanie w ciele $\mathbb{Z}_p$ tylko wtedy, gdy wyróżnik $\Delta = b^2 - 4ac$ należy do zbioru reszt kwadratowych modulo p . W ciele $\mathbb{Z}_p$ równanie wielomianowe stopnia n może mieć co najwyżej n rozwiązań.

Przykład 2.13

W ciele $\mathbb{Z}_{31}$ rozwiążemy równania kwadratowe:

a) $x^2 + 3x + 7 = 0$;

b) $x^2 + 5x + 9 = 0$.

a) Liczymy wyróżnik trójmianu kwadratowego. Podczas wykonywania rachunków w ciele $\mathbb{Z}_{31}$ możemy zawsze dodawać lub odejmować wartość 31:

$$\Delta = 3^2 - 4 \cdot 1 \cdot 7 = 9 - 28 = -19 = 12.$$

Równanie jest sprzeczne, ponieważ liczba 12 nie jest resztą kwadratową. Poniżej sprawdzamy w **Sage** za pomocą słowa kluczowego **in**, że rzeczywiście wartość ta nie należy do zbioru uzyskanego w przykładzie 2.12:

```
a, b, c = 1, 3, 7
R = Zmod(31)

Delta = R( b^2 - 4*a*c )
Delta in reszty
Out: False
```

b) Liczymy wyróżnik trójmianu $x^2 + 5x + 9$. Otrzymujemy:

$$\Delta = 5^2 - 4 \cdot 1 \cdot 9 = 25 - 36 = -11 = 20.$$

Na podstawie przykładu 2.12 wiemy, że liczba 20 jest resztą kwadratową, zatem równanie ma rozwiązania. Należy teraz znaleźć w ciele $\mathbb{Z}_{31}$ dowolny pierwiastek kwadratowy z liczby 20, czyli rozwiązanie równania:

$$x^2 = 20.$$

Równanie kwadratowe $x^2 = a$ w ciele $\mathbb{Z}_p$, gdzie a jest resztą kwadratową modulo p różną od 0, ma zawsze 2 rozwiązania. Jeżeli liczba x jest pierwiastkiem równania, to także liczba przeciwna $(-x) = p - x$ jest jego rozwiązaniem. Wartości te można znaleźć metodą prób i błędów, wypisując kolejno w tabeli drugie potęgi kolejnych elementów zbioru $\mathbb{Z}_{31}$. Jeżeli jednak zachodzi warunek $p \equiv 3 \pmod{4}$ (a tak jest w naszym przykładzie dla $p = 31$), to jeden z pierwiastków kwadratowych możemy obliczyć ze wzoru:

$$x = a^{\frac{p+1}{4}}.$$

W naszym przypadku otrzymujemy następujący wynik:

$$x = 20^{\frac{31+1}{4}} = 20^8 = (-11)^8 = 11^8 = (11^2)^4 = 121^4.$$

Teraz wykorzystujemy metodę z przykładu 2.7 – w podstawie potęgi zawsze możemy dodać lub odjąć dowolną wielokrotność liczby 31:

$$\dots = (121 - 4 \cdot 31)^4 = (-3)^4 = 3^4 = 3^3 \cdot 3 = 27 \cdot 3 = (-4) \cdot 3 = -12 = 19.$$

Gdy odgadniemy jeden z pierwiastków, możemy stosować zwykle wzory na wyznaczanie rozwiązań równania kwadratowego, czyli:

$$x_1 = \frac{-b - \sqrt{\Delta}}{2a} \quad \text{lub} \quad x_2 = \frac{-b + \sqrt{\Delta}}{2a},$$

gdzie przyjmujemy $\sqrt{\Delta} = 19$. Otrzymujemy rozwiązania:

$$x_1 = \frac{-5 - 19}{2 \cdot 1} = \frac{-24}{2} = -12 = 19$$

$$x_2 = \frac{-5 + 19}{2 \cdot 1} = \frac{14}{2} = 7.$$

W **Sage** równania w ciele modulo p możemy rozwiązywać za pomocą funkcji **solve_mod()**. Pierwszym argumentem tej funkcji jest równość reprezentująca równanie, które chcemy rozwiązać (prawa strona domyślnie jest równa 0), a drugim – liczba p :

```
solve_mod( x^2 + 5*x + 9, 31 )
Out: [(7,), (19,)]
```

2.2.2. Pierścień

Definicja. Pierścieniem nazywamy trójkę $(\mathcal{R}, \circ, \star)$, gdzie $\mathcal{R}$ jest zbiorem, a „ $\circ$ ” i „ $\star$ ” są działaniami wewnętrznymi w $\mathcal{R}$, gdy spełnione są następujące warunki:

1. Para $(\mathcal{R}, \circ)$ jest **grupą abelową**, czyli spełnione są aksjomaty dla ciała **(A1)–(A4)**.
2. Działanie „ $\star$ ” jest **łączne**:

$$\forall x, y, z \in \mathcal{R}: (x \star y) \star z = x \star (y \star z).$$

3. Działanie multiplikatywne „ $\star$ ” jest **rozdzielne** względem działania addytywnego „ $\circ$ ”:

$$\forall x, y, z \in \mathcal{R}: [(x \circ y) \star z = (x \star z) \circ (y \star z) \wedge x \star (y \circ z) = (x \star y) \circ (x \star z)].$$

Jeżeli dodatkowo zachodzi warunek:

4a. Działanie „ $\star$ ” jest **przemienne**:

$$\forall x, y \in \mathcal{R}: x \star y = y \star x,$$

to pierścień $(\mathcal{R}, \circ, \star)$ nazywamy pierścieniem przemennym.

Lub jeżeli dodatkowo zachodzi warunek:

4b. W zbiorze $\mathcal{R}$ istnieje **element neutralny** $1 \in \mathcal{R}$ działania „ $\star$ ”:

$$\exists 1 \in \mathcal{R} \quad \forall x \in \mathcal{R}: x \star 1 = 1 \star x = x,$$

to pierścień $(\mathcal{R}, \circ, \star)$ nazywamy pierścieniem z jedynką.

Uwaga. Pierścień można traktować jako osłabioną wersję ciała – jest to struktura, w której brakuje aksjomatów (M2), (M3) i (M4). Ponadto:

- pierścień przemienny może nie spełniać aksjomatów (M3) i (M4);
- pierścień z jedynką może nie spełniać aksjomatów (M2) i (M4).

Każde ciało jest również pierścieniem.

Struktura złożona ze zbioru $\mathbb{Z}_p$ oraz działań dodawania i mnożenia modulo p jest pierścieniem dla każdej liczby całkowitej $p \geq 2$. Ciałem natomiast jest tylko wtedy, gdy p jest liczbą pierwszą.

Przykład 2.14

W pierścieniu $\mathbb{Z}_{100}$ znajdziemy wynik działania $40 \cdot 50$.

Ponieważ $p = 100$ nie jest liczbą pierwszą, struktura $\mathbb{Z}_{100}$ jest pierścieniem, ale nie ciałem. W tym pierścieniu nadal możemy dodawać lub odejmować dowolną wielokrotność liczby 100, natomiast nie możemy korzystać z małego twierdzenia Fermata. Co więcej, nie każdy element posiada element odwrotny względem mnożenia, dlatego nie można stosować operacji dzielenia. W przykładzie najpierw wykonujemy standardowe mnożenie, a następnie odejmujemy taką liczbę wielokrotności 100, aby wynik był liczbą ze zbioru $\mathbb{Z}_{100}$:

$$40 \cdot 50 = 2000 = 2000 - 20 \cdot 100 = 0.$$

W **Sage** pierścienie modulo p definiujemy podobnie jak ciała, czyli:

```
R = Zmod(100)
R( 40*50 )
Out: 0
```

Przykład 2.14 pokazuje, że w pierścieniu może zdarzyć się szczególna sytuacja – iloczyn dwóch elementów niezerowych daje w wyniku 0, czyli element neutralny dodawania. Zasadniczo, jeżeli w pierścieniu $\mathcal{R}$ istnieją elementy $x, y \in \mathcal{R} \setminus \{0\}$, dla których zachodzi równość:

$$x \star y = 0,$$

to elementy x i y nazywamy dzielnikami zera w tym pierścieniu. Na przykład w pierścieniu $\mathbb{Z}_{100}$ dzielnikami zera są liczby 40 i 50.

Dzielnikami zera w pierścieniu $\mathbb{Z}_p$ są wszystkie liczby $x \in \mathbb{Z}_p$, które nie są względnie pierwsze z p . Czyli równoważnie, gdy spełniają warunek:

$$\text{NWD}(x, p) > 1.$$

Przykład 2.15

Znajdziemy wszystkie dzielniki zera w pierścieniu $\mathbb{Z}_{10}$.

Sprawdzamy po kolei wszystkie elementy zbioru $\mathbb{Z}_{10}$ za pomocą warunku z największym wspólnym dzielnikiem, który w **Sage** uzyskujemy za pomocą funkcji **gcd()**. Stosujemy składnię listy składanej wewnątrz zbioru:

```
dzielniki_zera = {x for x in Zmod(10)
                  if x != 0 and gcd(x, 10) > 1}
print('Dzielniki zera:', dzielniki_zera)
Out: Dzielniki zera: {2, 4, 5, 6, 8}
```

Pierścień przemienny z jedyneką, w którym nie występują dzielniki zera, nazywamy pierścieniem całkowitym. Przykładem takiej struktury jest trójka $(\mathbb{Z}, +, \cdot)$,

w skład której wchodzi zbiór liczb całkowitych $\mathbb{Z}$ oraz standardowe działania dodawania i mnożenia.

Pierścień całkowity może nie spełniać jedynie aksjomatu (**M4**), który dotyczy istnienia elementu odwrotnego działania multiplikatywnego. Pierścień $\mathbb{Z}_p$ natomiast jest całkowity tylko wtedy, gdy p jest liczbą pierwszą.

Co więcej, w ciele dowolne równanie wielomianowe n -tego stopnia ma co najwyżej n rozwiązań, natomiast w pierścieniu własność ta nie występuje.

Przykład 2.16

W pierścieniu $\mathbb{Z}_{30}$ rozwiążemy równanie:

$$x^5 - 3x^3 = 2x^2 + 6.$$

W ciele równanie wielomianowe piątego stopnia może mieć co najwyżej pięć rozwiązań. W pierścieniu natomiast nie ma już takiej zasady, dlatego należy sprawdzić osobno prawdziwość równości dla wszystkich elementów zbioru $\mathbb{Z}_{30}$. Wykonujemy to w **Sage** za pomocą pętli **for**:

```
for x in Zmod(30):  
    if x^5 - 3*x^3 == 2*x^2 + 6:  
        print(f'Liczba {x} jest rozwiązaniem równania.')
```

Out: Liczba 6 jest rozwiązaniem równania.
Liczba 11 jest rozwiązaniem równania.
Liczba 21 jest rozwiązaniem równania.
Liczba 26 jest rozwiązaniem równania.

W kodzie przechodzimy w pętli po kolejnych elementach zbioru $\mathbb{Z}_{30}$ i sprawdzamy, czy zachodzi równość z treści zadania. Rachunki na elementach obiektu **Zmod(30)** domyślnie są wykonywane modulo 30. Następnie stosujemy składnię **f-stringa**, aby umieścić znalezione rozwiązania w tekście. Tę samą czynność możemy również wykonać za pomocą funkcji **solve_mod()**. Należy jedynie pamiętać o zadeklarowaniu zmiennej x jako symbolu, ponieważ po wyjściu z pętli **for** system **SageMath** traktuje ją jako ostatni element obiektu **Zmod()**:

```
var('x')  
solve_mod( x^5 - 3*x^3 == 2*x^2 + 6, 30 )  
Out: [(6,), (26,), (21,), (11,)]
```

2.2.3. Elementy teorii liczb

Poniżej przeanalizujemy wybrane zagadnienia z teorii liczb w pierścieniu całkowitym $\mathbb{Z}$:

1. Liczby pierwsze.

Definicja. W pierścieniu całkowitym $(\mathcal{R}, \circ, \star)$ element $n \in \mathcal{R}$ jest podzielny przez element $d \in \mathcal{R}$, jeżeli istnieje element $k \in \mathcal{R}$, taki że:

$$n = d \star k.$$

Mówimy wtedy, że d dzieli n , co zapisujemy symbolicznie jako $d \mid n$.

Definicja. W pierścieniu całkowitym $\mathcal{R}$ element $p \in \mathcal{R} \setminus \{-1, 0, 1\}$ nazywamy pierwszym, gdy dla wszystkich $x, y \in \mathcal{R}$ spełniony jest warunek:

$$p \mid x \star y \Rightarrow (p \mid x \vee p \mid y).$$

Innymi słowy – jeśli p dzieli iloczyn dwóch elementów, to dzieli przynajmniej jeden z nich.

Zgodnie z definicją ogólną elementami pierwszymi w pierścieniu całkowitym $\mathbb{Z}$ są liczby należące do zbioru $\{\pm 2, \pm 3, \pm 5, \dots\}$. Zatem definicja ta pokrywa się z klasyczną definicją liczby pierwszej z dokładnością do znaku. My jednak będziemy zakładać, że liczby pierwsze przyjmują tylko wartości dodatnie.

W tabeli 2.2 zebrane zostały funkcje dostępne w **Sage**, które służą do wykonywania rachunków na liczbach pierwszych.

Tabela 2.2. Funkcje w **Sage** do obliczeń na liczbach pierwszych

	Nazwa funkcji	Przykład	Wynik
N-ta liczba pierwsza	<code>nth_prime()</code>	<code>nth_prime(4)</code>	7
Poprzednia/kolejna liczba pierwsza	<code>previous_prime()</code> <code>next_prime()</code>	<code>next_prime(9)</code>	11
Zakres liczb pierwszych	<code>prime_range()</code>	<code>prime_range(3, 11)</code>	[3, 5, 7]
Testowanie liczb pierwszych	<code>is_prime()</code>	<code>is_prime(17)</code>	True
Lista czynników pierwszych	<code>prime_factors()</code>	<code>prime_factors(20)</code>	[2, 5]
Rozkład na czynniki pierwsze	<code>factor()</code>	<code>factor(45)</code>	$3^2 \cdot 5$
Funkcja Eulera ¹⁰	<code>euler_phi()</code>	<code>euler_phi(12)</code>	4

¹⁰ Funkcja Eulera φ dla danej liczby naturalnej dodatniej n zwraca liczbę elementów zbioru $\{1, 2, 3, \dots, n-1\}$, które są względnie pierwsze z n .

Przykład 2.17

Znajdziemy dwie najmniejsze kolejne liczby naturalne, które mają w rozkładzie na czynniki dokładnie pięć różnych czynników pierwszych.

Przeanalizujemy najpierw dostępne narzędzia dla funkcji **factor()** na przykładzie liczby 1000 – funkcja **len()** zwraca liczbę różnych czynników pierwszych, natomiast indeksowanie pozwala znaleźć liczbę wystąpień konkretnego czynnika:

```
print( 'Czynniki pierwsze:', factor(1000) )
print( 'Liczba czynników pierwszych:', len(factor(1000)) )
print( 'Pierwszy czynnik i liczba wystąpień:',
       factor(1000)[0] )
```

```
Out: Czynniki pierwsze: 2^3 * 5^3
      Liczba czynników pierwszych: 2
      Pierwszy czynnik i liczba wystąpień: (2, 3)
```

Zacniemy od najmniejszej liczby naturalnej, która w rozkładzie na czynniki ma dokładnie pięć różnych czynników pierwszych – jest to liczba:

```
n = 2*3*5*7*11
```

```
n
```

```
Out: 2310
```

Stosujemy pętlę **while** w celu znalezienia dwóch kolejnych liczb spełniających warunki zadania:

```
while True:
    if len( factor(n) ) == 5 and len( factor(n + 1) ) == 5:
        print(f'Szukane liczby to: {n} oraz {n+1}')
        break
    n += 1
```

```
Out: Szukane liczby to: 254540 oraz 254541
```

Pętla **while** wykonywana jest tak długo, dopóki spełniony jest warunek przed symbolem dwukropka. W naszym przypadku warunek ten ma postać **True**, więc pętla nigdy się nie kończy – jej działanie zatrzymuje jedynie słowo kluczowe **break**. Tutaj w każdej iteracji zwiększamy wartość zmiennej n o 1 i sprawdzamy za pomocą instrukcji warunkowej **if**, czy spełniony jest żądany warunek. Jeżeli tak, to znalezione liczby wypisujemy na ekranie, a następnie przerywamy działanie pętli. Na koniec, aby sprawdzić poprawność wyniku, przedstawiamy rozkłady znalezionych liczb na czynniki pierwsze:

```
print( f'Rozkład liczby {n}:', factor(n) )
print( f'Rozkład liczby {n+1}:', factor(n+1) )
Out: Rozkład liczby 254540: 2^2 * 5 * 11 * 13 * 89
      Rozkład liczby 254541: 3 * 7 * 17 * 23 * 31
```

2. Równania diofantyczne.

Celem równań diofantycznych jest znalezienie rozwiązań jedynie w liczbach całkowitych.

Definicja. Równanie diofantyczne liniowe ma postać:

$$ax + by = c, \quad \text{gdzie } a, b, c \in \mathbb{Z}.$$

Dla $d := \text{NWD}(a, b)$ równanie ma rozwiązanie całkowite tylko wtedy, gdy $d \mid c$.

Rozwiązanie szczególne (x_0, y_0) można znaleźć za pomocą algorytmu Euklidesa. Wówczas wszystkie rozwiązania równania są postaci:

$$x = x_0 + \frac{b}{d}t, \quad y = y_0 - \frac{a}{d}t, \quad \text{gdzie } t \in \mathbb{Z}.$$

Przykład 2.18

Rozwiążemy równanie diofantyczne:

$$5x + 7y = 2$$

oraz przedstawimy interpretację graficzną rozwiązań.

Otrzymujemy $d = \text{NWD}(5, 7) = 1$, więc rozwiązania równania istnieją, ponieważ $1 \mid 2$. Zaczynamy od wyznaczenia rozwiązania szczególnego (x_0, y_0) . W tym celu w pierścieniu $\mathbb{Z}_p$, gdzie $p = \max(a, b) = 7$, wykonujemy procedurę

analogiczną do szukania elementu odwrotnego do liczby $\min(a, b) = 5$. Stosujemy algorytm Euklidesa:

$$7 = 5 \cdot 1 + 2$$

$$5 = 2 \cdot 2 + 1.$$

Naszym celem jest wyznaczenie równości wiążącej jedynkę z liczbami a i b . Wykonujemy rekonstrukcję opisaną w przykładzie 2.10:

$$1 \stackrel{(1)}{=} 5 - 2 \cdot 2 \stackrel{(1)}{=} 5 - 2 \cdot (7 - 5) \stackrel{(2)}{=} -2 \cdot 7 + 3 \cdot 5.$$

Ponieważ $c = 2$, uzyskaną równość mnożymy przez 2. Otrzymujemy:

$$5 \cdot 6 + 7 \cdot (-4) = 2.$$

Porównując uzyskany wynik z początkowym równaniem, możemy odczytać rozwiązanie szczególne jako $x_0 = 6$ i $y_0 = -4$. Rozwiązanie ogólne tego równania ma zatem postać:

$$x = 6 + 7t, \quad y = -4 - 5t, \quad \text{gdzie } t \in \mathbb{Z}.$$

W **Sage** stosujemy funkcję **xgcd()**, która zwraca współczynniki równości uzyskanej za pomocą algorytmu Euklidesa:

```
a, b, c = 5, 7, 2
d, x0, y0 = xgcd(a, b)
d, x0, y0
Out: (1, 3, -2)
```

Następnie implementujemy wzory na rozwiązanie ogólne równania. Dodatkowo za pomocą instrukcji **if** sprawdzamy, czy spełniony jest warunek istnienia rozwiązań:

```
var('t')
if c % d == 0:
    x0, y0 = (c // d)*x0, (c // d)*y0
    x, y = x0 + (b // d)*t, y0 - (a // d)*t
    print(f'Rozwiązanie: {x = }, {y = }')
else:
    print('Brak rozwiązań')
Out: Rozwiązanie: x = 7*t + 6, y = -5*t - 4
```

Znak dzielenia całkowitego (//) w powyższym kodzie dla zmiennych x_0 i y_0 stosowany jest po to, aby program działał poprawnie dla wartości d różnych od 1. Aby rozwiązać takie równanie ręcznie, należałoby na początku zastosować dzielenie stronami przez d .

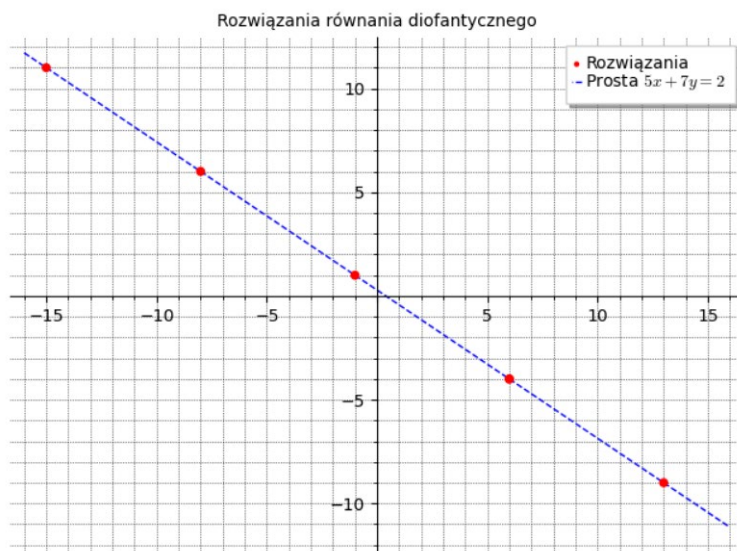
Na koniec przedstawiamy interpretację graficzną rozwiązań – są to punkty o obu współrzędnych całkowitych leżące na prostej o równaniu $5x + 7y = 2$, czyli tzw. punkty kratowe. Zaczynamy od umieszczenia punktów na rysunku – zaznaczamy rozwiązania uzyskane dla wybranych wartości parametru t :

```
rys = point( (x0, y0), legend_label='Rozwiązania',
            color='red', size=30, zorder=5 )
for t in [-3, -2, -1, 0, 1]:
    rys += point( (x(t=t), y(t=t)), color='red',
                 size=30, zorder=5, gridlines='minor' )
```

Pierwszy punkt (x_0, y_0) rysujemy oddzielnie, aby można było stosować operator += wewnątrz pętli **for** oraz w celu uniknięcia powtórzeń tekstu umieszczonego w legendzie. Następnie dodajemy do rysunku punkty uzyskane dla wartości parametru t ze zbioru $\{-3, -2, -1, 0, 1\}$. Wewnątrz funkcji **point()** należy zastosować zapis $x(t=t)$, ponieważ zmienne x i y zostały zdefiniowane jako wyrażenia, a nie funkcje. Argument $gridlines='minor'$ pozwala natomiast umieścić siatkę układu współrzędnych tylko w punktach o współrzędnych całkowitych. Aby ułatwić interpretację, do rysunku dodajemy wykres prostej $5x + 7y = 2$ uzyskany poprzez wyznaczenie zmiennej y z równania prostej:

```
var('x')
rys += plot( (c - a*x)/b, (x, -16, 16), ymin=-12, ymax=12,
            linestyle='--', legend_label='Prosta $5x+7y=2$',
            title='Rozwiązania równania diofantycznego' )
rys
```

Efekt końcowy przedstawiono na rysunku 2.1. Prostą $5x + 7y = 2$ zaznaczono kolorem niebieskim i linią przerywaną. Czerwone punkty kratowe reprezentują natomiast punkty, których współrzędne są rozwiązaniami równania diofantycznego.



Rysunek 2.1. Interpretacja graficzna rozwiązań równania diofantycznego

3. Układy kongruencji.

Przedstawimy tzw. chińskie twierdzenie o resztach (**Chinese Remainder Theorem**) na podstawie układu trzech kongruencji (czyli równań modularnych):

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ x \equiv a_3 \pmod{m_3}, \end{cases}$$

gdzie $a_1, a_2, a_3 \in \mathbb{Z}$ oraz m_1, m_2, m_3 są liczbami naturalnymi większymi od 1. Twierdzenie mówi, że jeżeli liczby m_i są parami względnie pierwsze, czyli $\text{NWD}(m_i, m_j) = 1$ dla $i \neq j$, to układ równań ma dokładnie jedno rozwiązanie modulo $M = m_1 \cdot m_2 \cdot m_3$.

Przykład 2.19

Rozwiążemy układ kongruencji:

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 3 \pmod{5} \\ x \equiv 5 \pmod{7}. \end{cases}$$

Zgodnie z oznaczeniami w twierdzeniu mamy $a_1 = 2$, $a_2 = 3$ i $a_3 = 5$. Liczby $m_1 = 3$, $m_2 = 5$ i $m_3 = 7$ są parami względnie pierwsze, zatem układ ma dokładnie jedno rozwiązanie modulo $M = 3 \cdot 5 \cdot 7 = 105$. Rozwiążemy ten układ w dwóch krokach:

1. Wyznaczenie elementów odwrotnych.

Definiujemy liczby postaci:

$$M_1 = \frac{M}{m_1}, \quad M_2 = \frac{M}{m_2}, \quad M_3 = \frac{M}{m_3}.$$

Mamy $M_1 = 35$, $M_2 = 21$ i $M_3 = 15$. Następnie znajdujemy elementy odwrotne:

$$y_i \equiv M_i^{-1} \pmod{m_i},$$

dla $i = 1, 2, 3$. Poniższe rachunki przeprowadzamy odpowiednio modulo 3, 5 i 7:

$$y_1 = 35^{-1} = 2^{-1} = 2$$

$$y_2 = 21^{-1} = 1^{-1} = 1$$

$$y_3 = 15^{-1} = 1^{-1} = 1.$$

2. Zastosowanie wzoru.

Rozwiązanie układu dane jest wzorem:

$$x \equiv a_1 y_1 M_1 + a_2 y_2 M_2 + a_3 y_3 M_3 \pmod{M}.$$

Wykonujemy rachunek modulo 105, wstawiając wyznaczone wartości:

$$x = 2 \cdot 2 \cdot 35 + 3 \cdot 1 \cdot 21 + 5 \cdot 1 \cdot 15 = 140 + 63 + 75 = 278 = 68.$$

W **Sage** układ kongruencji możemy rozwiązać za pomocą funkcji **crt()**. Pierwszym argumentem tej funkcji jest lista współczynników a_i , natomiast drugim – lista współczynników m_i :

```
crt( [2, 3, 5], [3, 5, 7] )
```

```
Out: 68
```

2.3. Permutacje

W tym rozdziale zajmiemy się pojęciem permutacji – jest to funkcja, która zmienia kolejność elementów w danym zbiorze. Zbiór wszystkich permutacji wraz z działaniem złożenia „o” tworzy grupę zwaną grupą symetryczną. Podobnie jak

w przypadku grupy bijekcji $(\mathcal{F}, \circ)$, o której wspominaliśmy wcześniej w tabeli 2.1, także tutaj działanie nie jest przemienne – kolejność stosowanych permutacji ma znaczenie i nie można jej dowolnie zmieniać.

Na początku omówimy podstawowe operacje w grupie permutacji – złożenie i potęgowanie. Poznamy również element neutralny, zwany permutacją identycznościową, a także sposób wyznaczania permutacji odwrotnych. Następnie wprowadzimy pojęcie parzystości permutacji oraz zobaczymy, w jaki sposób każdą z nich można rozłożyć na prostsze składniki – cykle. Ten rozkład na cykle będzie nam potrzebny do kolejnych zastosowań przedstawionych w dalszej części książki.

2.3.1. Działania na permutacjach

Definicja. Permutacją nazywamy funkcję, która odwzorowuje dany zbiór na siebie. Zatem dla zbioru n -elementowego $\{1, 2, 3, \dots, n\}$ permutacja σ jest bijekcją, taką że:

$$\sigma: \{1, 2, 3, \dots, n\} \rightarrow \{1, 2, 3, \dots, n\}.$$

Oznacza to, że zbiór $\{1, 2, 3, \dots, n\}$ jest dziedziną funkcji σ oraz jednocześnie jej zbiorem wartości.

Przykład 2.20

Zdefiniujemy permutację zbioru 7-elementowego:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 7 & 2 & 5 & 4 & 1 & 6 \end{pmatrix},$$

oraz wyznaczmy wartości funkcji σ dla wybranych elementów dziedziny $\{1, 2, 3, \dots, 7\}$.

W języku **Sage** permutację definiujemy za pomocą obiektu **Permutation()**. W liście przekazujemy kolejne elementy z dolnego wiersza:

```
s = Permutation( [3, 7, 2, 5, 4, 1, 6] )
```

```
s
```

```
Out: [3, 7, 2, 5, 4, 1, 6]
```

Liczby w pierwszym wierszu permutacji σ są zawsze uporządkowane rosnąco. W drugim wierszu znajdują się natomiast wartości funkcji σ dla odpowiednich argumentów z pierwszego wiersza. Możemy odczytać na przykład, że:

$$\sigma(1) = 3, \quad \sigma(2) = 7, \quad \sigma(3) = 2 \quad \text{itd.}$$

W kodzie dla permutacji s możemy stosować składnię, którą wykorzystywaliśmy dla funkcji, czyli:

```
s(1), s(2), s(3)
```

```
Out: (3, 7, 2)
```

Zbiór wszystkich permutacji zbioru $\{1, 2, 3, \dots, n\}$ oznaczamy przez S_n . Moc tego zbioru, czyli liczba jego elementów, jest równa:

$$|S_n| = n!$$

Przykład 2.21

Wypiszemy wszystkie elementy zbioru S_3 , który składa się z permutacji zbioru 3-elementowego.

Liczba elementów zbioru S_3 jest równa $3!$ – jest to iloczyn trzech kolejnych liczb naturalnych dodatnich:

$$|S_3| = 3! = 1 \cdot 2 \cdot 3 = 6.$$

Zapisujemy sześć elementów zbioru S_3 , przedstawiając wartości w dolnym wierszu:

$$S_3 = \left\{ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}, \right. \\ \left. \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \right\}.$$

Pierwszy element powyższego zbioru nazywamy permutacją identycznościową – jest to permutacja, która nie zmienia kolejności elementów.

W **Sage** wszystkie permutacje elementów przekazanej listy uzyskujemy za pomocą obiektu **Permutations()**. Funkcja **list()** przeprowadza jedynie rzutowanie uzyskanego wyniku na listę – dzięki temu możemy zobaczyć na ekranie uzyskany rezultat:

```
Permutations([1, 2, 3]).list()
```

```
Out: [[1, 2, 3], [1, 3, 2], [2, 1, 3],  
      [2, 3, 1], [3, 1, 2], [3, 2, 1]]
```

Ten sam wynik moglibyśmy uzyskać, korzystając z polecenia `Permutations(3).list()`. Aby otrzymać permutację identycznościową, zapisujemy:

```
Permutations(3).identity()
```

```
Out: [1, 2, 3]
```

W zbiorze permutacji S_n wprowadzamy działanie złożenia permutacji „ $\circ$ ”. Dla dwóch permutacji $\sigma, \tau \in S_n$ działanie to jest zdefiniowane tak samo jak złożenie funkcji, czyli:

$$(\sigma \circ \tau)(x) = \sigma(\tau(x)).$$

Złożenie permutacji nazywamy też iloczynem permutacji i zapisujemy krótko jako $\sigma\tau$. Warto zwrócić uwagę, że działanie to wykonujemy „od prawej do lewej strony” – najpierw liczymy wartość funkcji τ dla argumentu x , a następnie wartość funkcji σ dla uzyskanego wyniku $\tau(x)$.

Definicja. Para $(S_n, \circ)$ jest grupą, którą nazywamy grupą symetryczną. Elementem neutralnym w tej grupie jest permutacja identycznościowa oznaczana jako id – jest to permutacja, która odwzorowuje każdy element na siebie.

Grupa symetryczna jest nieprzemienne, ponieważ iloczyn permutacji nie jest przemienny. Zatem dla dowolnych permutacji $\sigma, \tau \in S_n$ na ogół mamy następującą zależność:

$$\sigma\tau \neq \tau\sigma.$$

Przykład 2.22

Dla dwóch permutacji:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix} \quad \text{oraz} \quad \tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 4 & 1 & 2 \end{pmatrix},$$

znajdziemy iloczyn $\sigma\tau$.

Iloczyn permutacji, ze względu na definicję złożenia, wyznaczamy „od prawej do lewej strony”. Na przykład dla liczby 1 najpierw znajdujemy wartość permutacji τ dla jedynki, a następnie wartość permutacji σ dla uzyskanego wyniku:

$$(\sigma \circ \tau)(1) = \sigma(\tau(1)) = \sigma(3) = 5.$$

Zatem, aby znaleźć iloczyn $\sigma\tau$, stosujemy dla każdej liczby schemat: „idziemy w dół” – dla permutacji τ , a następnie dla σ . Przykładowo dla liczby 2 otrzymujemy ciąg przejść:

$$2 \xrightarrow{\tau} 5 \xrightarrow{\sigma} 3.$$

Powtarzając ten proces dla wszystkich elementów zbioru $\{1, 2, 3, 4, 5\}$, otrzymujemy wynik:

$$\sigma\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 3 & 2 & 4 & 1 \end{pmatrix}.$$

W Sage iloczyn permutacji wykonujemy za pomocą symbolu gwiazdki (*). Należy zwrócić uwagę, że działanie $\sigma\tau$ zapisujemy w odwrotnej kolejności – podobnie jak przy ręcznych rachunkach:

```
s = Permutation( [4, 1, 5, 2, 3] )
t = Permutation( [3, 5, 4, 1, 2] )
t*s
Out: [5, 3, 2, 4, 1]
```

Przykład 2.23

Dana jest permutacja:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 7 & 1 & 5 & 8 & 2 & 6 & 3 \end{pmatrix}.$$

Znajdziemy wyniki działań:

a) σ^3 ; b) σ^{-1} ; c) σ^{-2} .

a) Aby znaleźć permutację σ^3 , trzykrotnie „idziemy w dół”. Na przykład dla liczby 1 mamy wynik $\sigma^3(1) = 8$, ponieważ otrzymujemy ciąg przejść: $1 \xrightarrow{\sigma} 4 \xrightarrow{\sigma} 5 \xrightarrow{\sigma} 8$. Ostatecznie:

$$\sigma^3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 2 & 5 & 3 & 1 & 6 & 7 & 4 \end{pmatrix}.$$

W Sage dla obiektu **Permutation()** możemy stosować jedynie symbol mnożenia (*). Niestety w poniższym kodzie nie zadziała zapis s^3 :

```
s = Permutation( [4, 7, 1, 5, 8, 2, 6, 3] )
s*s*s
Out: [8, 2, 5, 3, 1, 6, 7, 4]
```

Aby ominąć ten problem, stosujemy obiekt **PermutationGroupElement()**, na którym już działa symbol potęgowania (^). Wynik s^3 zwracany jest w postaci rozkładu permutacji na cykle, co zostanie opisane później. W związku z tym w celu uzyskania zrozumiałej odpowiedzi rzutujemy go z powrotem na obiekt **Permutation()**:

```
s = PermutationGroupElement( [4, 7, 1, 5, 8, 2, 6, 3] )
Permutation(s^3)
Out: [8, 2, 5, 3, 1, 6, 7, 4]
```

b) Permutacja σ^{-1} jest permutacją odwrotną do σ . Wyznaczamy ją, stosując schemat: „idziemy w górę”. Na przykład nad jedynką w dolnym wierszu znajduje się liczba 3, więc mamy $\sigma^{-1}(1) = 3$. Po wykonaniu tej operacji dla każdego argumentu otrzymujemy:

$$\sigma^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 6 & 8 & 1 & 4 & 7 & 2 & 5 \end{pmatrix}.$$

W Sage na obiekcie **PermutationGroupElement()** z punktu a) permutację odwrotną możemy wyznaczyć za pomocą symbolu potęgowania (^), natomiast dla obiektu **Permutation()** należy zastosować instrukcję `s.inverse()`:

```
Permutation(s^-1)
Out: [3, 6, 8, 1, 4, 7, 2, 5]
```

c) Aby znaleźć permutację σ^{-2} , dwukrotnie „idziemy w górę”. Na przykład dla liczby 2 uzyskujemy ciąg przejść: $2 \xrightarrow{\sigma^{-1}} 6 \xrightarrow{\sigma^{-1}} 7$. Ostatecznie otrzymujemy permutację:

$$\sigma^{-2} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 7 & 5 & 3 & 1 & 2 & 6 & 4 \end{pmatrix}.$$

W kodzie, podobnie jak w podpunkcie b), wykonujemy rachunek na obiekcie **PermutationGroupElement()**, a wynik rzutujemy na obiekt **Permutation()**:

```
Permutation(s^-2)
```

```
Out: [8, 7, 5, 3, 1, 2, 6, 4]
```

2.3.2. Parzystość permutacji

Definicja. Dla danej permutacji σ inwersją nazywamy parę liczb (i, j) , taką że:

$$i < j \quad \text{oraz} \quad \sigma(i) > \sigma(j).$$

Przykład 2.24

Wyznamy wszystkie inwersje permutacji:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 5 & 1 & 4 & 3 \end{pmatrix}.$$

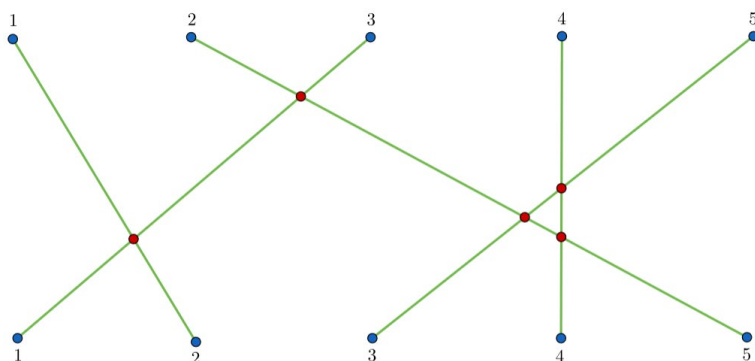
Na początku przyjmujemy $i = 1$ i sprawdzamy, czy warunek w definicji zachodzi dla liczb j spełniających nierówność $i < j$, czyli $j = 2, 3, 4, 5$. Uzyskujemy tylko jedną inwersję $(1, 3)$, ponieważ:

$$1 < 3 \quad \text{oraz} \quad 2 = \sigma(1) > \sigma(3) = 1.$$

Następnie ustawiamy $i = 2$ i badamy wartości $j = 3, 4, 5$. Otrzymujemy inwersje $(2, 3)$, $(2, 4)$ i $(2, 5)$. Postępując analogicznie dla $i = 3$ oraz $i = 4$, uzyskujemy zbiór inwersji:

$$I_\sigma = \{(1, 3), (2, 3), (2, 4), (2, 5), (4, 5)\}.$$

Inwersje można także interpretować geometrycznie – jako punkty przecięcia między odcinkami reprezentującymi permutację na diagramie, gdzie łączymy pozycje początkowe z odpowiadającymi im pozycjami końcowymi. Idea ta została zilustrowana na rysunku 2.2. Na diagramie widocznych jest pięć czerwonych punktów przecięcia, co odpowiada pięciu inwersjom uzyskanym w przykładzie.



Rysunek 2.2. Interpretacja geometryczna inwersji – zielone połączenia uzyskiwane są ze wzoru permutacji

W języku **Sage** inwersje oraz ich liczbę uzyskujemy odpowiednio za pomocą funkcji **inversions()** i **number_of_inversions()**:

```
s = Permutation( [2, 5, 1, 4, 3] )
print( 'Inwersje:', s.inversions() )
print( 'Liczba inwersji:', s.number_of_inversions() )
Out: Inwersje: [(1, 3), (2, 3), (2, 4), (2, 5), (4, 5)]
      Liczba inwersji: 5
```

Definicja. Na podstawie liczby inwersji można określić parzystość permutacji σ . Powiemy, że permutacja jest:

- parzysta, gdy liczba jej inwersji jest parzysta;
- nieparzysta, gdy liczba jej inwersji jest nieparzysta.

Znak permutacji σ definiujemy natomiast jako:

$$\text{sgn}(\sigma) = \begin{cases} 1, & \text{gdy } \sigma \text{ jest parzysta,} \\ -1, & \text{gdy } \sigma \text{ jest nieparzysta.} \end{cases}$$

Przykład 2.25

Określimy parzystość oraz znak permutacji σ z przykładu 2.24.

Liczba inwersji permutacji σ jest nieparzysta:

$$|I_\sigma| = 5.$$

Wobec tego σ jest permutacją nieparzystą. Sprawdzamy to, korzystając z funkcji `is_even()`, która stwierdza, czy dana permutacja jest parzysta:

```
s.is_even()  
Out: False
```

Znak permutacji σ odczytujemy natomiast z definicji jako $\text{sgn}(\sigma) = -1$. W **Sage** ten wynik uzyskujemy za pomocą funkcji `sign()`:

```
s.sign()  
Out: -1
```

Przykład 2.26

Wyznamy wszystkie wartości parametrów $k, \ell, m \in \mathbb{N}$, dla których permutacja

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & k & 2 & 6 & \ell & 3 & m \end{pmatrix}$$

jest parzysta. Dla znalezionych wartości parametrów przedstawimy wszystkie inwersje odpowiadających im permutacji oraz podamy ich liczbę.

W drugim wierszu permutacji σ brakuje liczb 1, 4 i 7, zatem k, ℓ i m są różnymi elementami ze zbioru $\{1, 4, 7\}$. Wszystkie możliwe permutacje tego zbioru uzyskujemy za pomocą funkcji **Permutations()**:

```
permutacje = Permutations( [1, 4, 7] )  
permutacje.list()  
Out: [[1, 4, 7], [1, 7, 4], [4, 1, 7],  
      [4, 7, 1], [7, 1, 4], [7, 4, 1]]
```

Teraz możemy iterować w pętli **for** po obiekcie `permutacje`, wykorzystując trzy zmienne jednocześnie. Parzystość permutacji sprawdzamy za pomocą funkcji `is_even()` i wypisujemy żądane wielkości na ekranie:

```
for k, l, m in permutacje:
```

```
    s = Permutation( [5, k, 2, 6, l, 3, m] )
```

```
    if s.is_even():
```

```
        print( 'Permutacja jest parzysta',
```

```
              f'dla {k = }, {l = } oraz {m = }.' )
```

```
        print( 'Inwersje:', s.inversions() )
```

```
        print( 'Liczba inwersji:', s.number_of_inversions() )
```

```
        print()
```

Out: Permutacja jest parzysta dla k = 1, l = 7 oraz m = 4.

Inwersje: [(1, 2), (1, 3), (1, 6), (1, 7),
(4, 6), (4, 7), (5, 6), (5, 7)]

Liczba inwersji: 8

Permutacja jest parzysta dla k = 4, l = 1 oraz m = 7.

Inwersje: [(1, 2), (1, 3), (1, 5), (1, 6), (2, 3),
(2, 5), (2, 6), (3, 5), (4, 5), (4, 6)]

Liczba inwersji: 10

Permutacja jest parzysta dla k = 7, l = 4 oraz m = 1.

Inwersje: [(1, 3), (1, 5), (1, 6), (1, 7),
(2, 3), (2, 4), (2, 5), (2, 6),
(2, 7), (3, 7), (4, 5), (4, 6),
(4, 7), (5, 6), (5, 7), (6, 7)]

Liczba inwersji: 16

2.3.3. Rozkład na cykle

Definicja. Cyklem nazywamy permutację, która przesuwa pewne elementy „w kółko”, a pozostałe zostawia bez zmian. Na przykład cykl $\mu = (2\ 3\ 5)$ ze zbioru S_8 to permutacja:

$$\mu = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 3 & 5 & 4 & 2 & 6 & 7 & 8 \end{pmatrix}.$$

Możemy zauważyć, że elementy 2, 3 i 5 występujące w zapisie cyklu są odwzorowywane kolejno zgodnie z ciągiem przejść: $2 \xrightarrow{\mu} 3 \xrightarrow{\mu} 5 \xrightarrow{\mu} 2$, który następnie powtarza się cyklicznie.

Twierdzenie. Każdą permutację można przedstawić w postaci iloczynu cykli rozłącznych, czyli takich cykli, które nie mają wspólnych elementów.

Ponieważ cykle rozłączne działają niezależnie od siebie, ich iloczyn jest przemienny – kolejność zapisu nie wpływa na wynik.

Przykład 2.27

Zapiszemy poniższą permutację w postaci iloczynu cykli rozłącznych:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 6 & 1 & 7 & 5 & 8 & 2 & 3 & 4 \end{pmatrix}.$$

Zaczynamy od liczby 1 i znajdujemy „ścieżkę” kolejnych wartości permutacji σ . Uzyskujemy ciąg przejść: $1 \xrightarrow{\sigma} 6 \xrightarrow{\sigma} 2 \xrightarrow{\sigma} 1$. Gdy wracamy do punktu początkowego, kończymy cykl. Zapisujemy to jako:

$$\sigma = (1 \ 6 \ 2) \dots$$

Następnie zaczynamy od kolejnej liczby, która jeszcze nie wystąpiła w powyższym zapisie, czyli na przykład 3. Otrzymujemy kolejny ciąg przejść: $3 \xrightarrow{\sigma} 7 \xrightarrow{\sigma} 3$. Stąd możemy zapisać:

$$\sigma = (1 \ 6 \ 2)(3 \ 7) \dots$$

Kontynuujemy ten proces, zaczynając od liczby 4. Otrzymujemy ostatnią „ścieżkę” postaci: $4 \xrightarrow{\sigma} 5 \xrightarrow{\sigma} 8 \xrightarrow{\sigma} 4$. Tym razem wyczerpaliśmy wszystkie elementy, zatem rozkład permutacji σ na iloczyn cykli rozłącznych to:

$$\sigma = (1 \ 6 \ 2)(3 \ 7)(4 \ 5 \ 8).$$

Każdy z trzech otrzymanych cykli jest osobną permutacją, natomiast pomiędzy nawiasami mamy działanie złożenia. Po wykonaniu tych operacji uzyskalibyśmy początkową permutację σ . Ponadto nawiasy w powyższym zapisie możemy umieścić w dowolnej kolejności, ponieważ mnożenie cykli rozłącznych jest przemienne.

W Sage z danej permutacji możemy wyodrębnić cykle rozłączne za pomocą funkcji `to_cycles()`:

```
s = Permutation( [6, 1, 7, 5, 8, 2, 3, 4] )
s.to_cycles()
Out: [(1, 6, 2), (3, 7), (4, 5, 8)]
```

Ponadto obiekt **PermutationGroupElement()** domyślnie zapisuje permutację jako iloczyn cykli rozłącznych:

```
PermutationGroupElement( [6, 1, 7, 5, 8, 2, 3, 4] )
Out: (1,6,2)(3,7)(4,5,8)
```

Jeżeli w permutacji σ pewien element jest odwzorowywany na siebie, to w rozkładzie na cykle odpowiada mu cykl jednoelementowy. Tego typu cykle zazwyczaj się pomija w zapisie. Ta sama zasada stosowana jest również przez system **SageMath** dla obiektu **PermutationGroupElement()** – cykle jednoelementowe nie są wyświetlane w wynikach.

2.4. Własności permutacji

W tej części zajmiemy się własnościami permutacji, zaczynając od szczególnego przypadku, jakim jest transpozycja, czyli cykl dwuelementowy. W poprzednim rozdziale zobaczyliśmy, że każdą permutację da się zapisać jako iloczyn cykli rozłącznych. Teraz pokażemy, że można ją również przedstawić jako iloczyn samych transpozycji. Oznacza to, że transpozycje stanowią najmniejsze „cegiełki”, z których budujemy wszystkie permutacje. Co więcej, mają one również wpływ na badanie parzystości – każda transpozycja jest nieparzysta, a mnożenie przez nią zawsze zmienia znak permutacji.

Następnie przejdziemy do pojęcia rzędu permutacji, określającego, do której potęgi należy podnieść daną permutację, aby uzyskać „pełny obieg”, czyli powrócić do permutacji identycznościowej. Rząd pozwala więc przewidzieć cykliczne zachowanie permutacji i w prosty sposób obliczać jej potęgi. W dalszej części rozdziału zobaczymy także, w jaki sposób te narzędzia mogą być wykorzystywane przy rozwiązywaniu równań liniowych w grupie permutacji.

2.4.1. Transpozycje

Definicja. Transpozycją nazywamy cykl dwuelementowy. Jest to permutacja, która zamienia miejscami dokładnie dwa elementy permutowanego zbioru, a wszystkie pozostałe zostawia bez zmian. Na przykład transpozycja $\tau = (3\ 7)$ w grupie S_8 wygląda następująco:

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 7 & 4 & 5 & 6 & 3 & 8 \end{pmatrix}.$$

Twierdzenie. Każdą permutację można zapisać w postaci iloczynu transpozycji.

Przykład 2.28

Zapiszemy permutację

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 5 & 7 & 9 & 4 & 6 & 3 & 8 & 2 & 1 \end{pmatrix}$$

jako iloczyn transpozycji.

Zaczynamy od rozkładu permutacji σ na iloczyn cykli rozłącznych. Stosujemy procedurę opisaną w przykładzie 2.27, pomijając cykl jednoelementowy złożony z liczby 4:

$$\sigma = (1\ 5\ 6\ 3\ 9)(2\ 7\ 8).$$

Następnie w obrębie każdego cyklu zaczynamy od pierwszego elementu i zapisujemy kolejne transpozycje, idąc od końca. Dla pierwszego cyklu otrzymujemy rozkład:

$$(1\ 5\ 6\ 3\ 9) = (1\ 9)(1\ 3)(1\ 6)(1\ 5),$$

zaś dla drugiego:

$$(2\ 7\ 8) = (2\ 8)(2\ 7).$$

Ostatecznie szukany rozkład na iloczyn transpozycji jest postaci:

$$\sigma = (1\ 9)(1\ 3)(1\ 6)(1\ 5)(2\ 8)(2\ 7).$$

W **Sage** definiujemy permutację σ i wykonujemy rozkład na iloczyn cykli rozłącznych za pomocą funkcji **to_cycles()**. Aby cykl jednoelementowy (4,) nie był wyświetlany w wyniku, możemy zastosować argument **singletons=False**:

```
s = Permutation( [5, 7, 9, 4, 6, 3, 8, 2, 1] )
s.to_cycles()
Out: [(1, 5, 6, 3, 9), (2, 7, 8), (4,)]
```

Teraz wykorzystujemy listę składaną, aby iterując po elementach **s.to_cycles()**, zamienić poszczególne cykle na transpozycje. Funkcja **reversed()** pozwala przechodzić po danym obiekcie od końca, natomiast zapis **c[1:]** oznacza wzięcie wszystkich elementów danego cyklu oprócz pierwszego:

```
[(c[0], x) for c in s.to_cycles() for x in reversed(c[1:])]
Out: [(1, 9), (1, 3), (1, 6), (1, 5), (2, 8), (2, 7)]
```

Każda transpozycja jest permutacją nieparzystą. Mnożenie dowolnej permutacji przez transpozycję zmienia jej znak, stąd dla każdej permutacji $\sigma \in S_n$ i transpozycji $\tau \in S_n$ zachodzi równość:

$$\operatorname{sgn}(\sigma\tau) = -\operatorname{sgn}(\sigma).$$

Wobec tego rozkład permutacji na iloczyn transpozycji można wykorzystać do określenia jej parzystości. Powiemy, że permutacja jest:

- parzysta, gdy można ją zapisać jako iloczyn parzystej liczby transpozycji;
- nieparzysta, gdy wymaga nieparzystej liczby transpozycji.

Ogólniej, dla dowolnych permutacji $\sigma_1, \sigma_2 \in S_n$ zachodzi własność:

$$\operatorname{sgn}(\sigma_1\sigma_2) = \operatorname{sgn}(\sigma_1) \cdot \operatorname{sgn}(\sigma_2).$$

Przykład 2.29

Dana jest permutacja:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 7 & 2 & 1 & 4 & 5 & 6 \end{pmatrix}.$$

Określimy parzystość permutacji σ^5 .

Zapisujemy permutację σ w postaci cyklu:

$$\sigma = (1\ 3\ 2\ 7\ 6\ 5\ 4).$$

Następnie, korzystając z algorytmu przedstawionego w przykładzie 2.28, przedstawiamy ją jako iloczyn transpozycji:

$$\sigma = (14)(15)(16)(17)(12)(13).$$

Uzyskaliśmy rozkład¹¹ na iloczyn 6 transpozycji, a więc σ jest permutacją parzystą. Stąd wynika, że także permutacja σ^5 jest parzysta – jej rozkład zawiera

¹¹ Rozkład permutacji na iloczyn transpozycji nie jest jednoznaczny, jednak w każdym przypadku zachowana jest parzystość liczby transpozycji.

$6 \cdot 5 = 30$ transpozycji. Ponadto dowolna potęga permutacji parzystej σ pozostaje permutacją parzystą.

W **Sage** możemy użyć funkcji **sign()** na obiekcie **PermutationGroupElement()**, który pozwala wykonać operację potęgowania:

```
s = PermutationGroupElement( [3, 7, 2, 1, 4, 5, 6] )
(s^5).sign()
Out: 1
```

Zbiór wszystkich permutacji parzystych zbioru n -elementowego $\{1, 2, 3, \dots, n\}$ oznaczamy przez A_n . Zbiór permutacji S_n zawiera dokładnie połowę permutacji parzystych, więc zachodzi równość:

$$|A_n| = \frac{n!}{2}.$$

Definicja. Para $(A_n, \circ)$ tworzy grupę, którą nazywamy grupą alternującą.

Przykład 2.30

Narysujemy tabelę działania w grupie symetrycznej S_3 oraz w grupie alternującej A_3 .

Aby narysować tabelę w grupie S_3 , korzystamy z funkcji **OperationTable()** oraz obiektu **SymmetricGroup()** generującego elementy grupy symetrycznej:

```
from sage.matrix.operation_table import OperationTable
R = SymmetricGroup(3)
OperationTable( R, operator.mul, names='elements' )
Out:  *      ()  (2,3)  (1,2)  (1,2,3)  (1,3,2)  (1,3)
      +-----+
      () |      ()  (2,3)  (1,2)  (1,2,3)  (1,3,2)  (1,3)
      (2,3) |      (2,3)      ()  (1,2,3)  (1,2)  (1,3)  (1,3,2)
      (1,2) |      (1,2)  (1,3,2)      ()  (1,3)  (2,3)  (1,2,3)
      (1,2,3) |      (1,2,3)  (1,3)  (2,3)  (1,3,2)      ()  (1,2)
      (1,3,2) |      (1,3,2)  (1,2)  (1,3)      ()  (1,2,3)  (2,3)
      (1,3) |      (1,3)  (1,2,3)  (1,3,2)  (2,3)  (1,2)      ()
```

Grupę alternującą natomiast otrzymujemy, używając obiektu **AlternatingGroup()**. Możemy zaobserwować, że zbiór A_3 zawiera połowę elementów zbioru S_3 :

```
R = AlternatingGroup(3)
OperationTable( R, operator.mul, names='elements' )
Out:      *      () (1,2,3) (1,3,2)
      +-----+
      () |      () (1,2,3) (1,3,2)
(1,2,3) | (1,2,3) (1,3,2)      ()
(1,3,2) | (1,3,2)      () (1,2,3)
```

2.4.2. Rząd permutacji

Definicja. Rzędem permutacji σ nazywamy najmniejszą liczbę naturalną dodatnią $k \in \mathbb{N}_+$, taką że:

$$\sigma^k = \text{id}.$$

Rząd $r(\sigma)$ możemy wyznaczyć ze wzoru:

$$r(\sigma) = \text{NWW}(d_1, \dots, d_n),$$

gdzie $d_1, \dots, d_n$ to długości cykli w rozkładzie permutacji σ na iloczyn cykli rozłącznych.

Przykład 2.31

Znajdziemy rząd permutacji:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 9 & 4 & 6 & 7 & 1 & 3 & 2 & 5 & 8 \end{pmatrix}.$$

Zapisujemy permutację σ w postaci iloczynu cykli rozłącznych oraz znajdujemy długości poszczególnych cykli:

$$\sigma = \underbrace{(1 \ 9 \ 8 \ 5)}_{d_1=4} \underbrace{(2 \ 4 \ 7)}_{d_2=3} \underbrace{(3 \ 6)}_{d_3=2}.$$

Szukany rząd to najmniejsza wspólna wielokrotność uzyskanych długości cykli, czyli:

$$r(\sigma) = \text{NWW}(d_1, d_2, d_3) = \text{NWW}(4, 3, 2) = 12.$$

Jest to najmniejsza liczba naturalna dodatnia, dla której permutacja σ podniesiona do tej potęgi daje permutację identycznościową. Stąd:

$$\sigma^{12} = \text{id}.$$

W **Sage** wykonujemy kolejno powyższe operacje – rozkładamy permutację na iloczyn cykli rozłącznych za pomocą funkcji **to_cycles()**, a następnie znajdujemy najmniejszą wspólną wielokrotność ich długości, korzystając z funkcji **lcm()**:

```
s = Permutation( [9, 4, 6, 7, 1, 3, 2, 5, 8] )
cykle = s.to_cycles()
dlugosci = [len(x) for x in cykle]
rząd = lcm(dlugosci)
rząd
Out: 12
```

Możemy również skorzystać z gotowej funkcji wbudowanej – **order()** – która zwraca rząd danej permutacji:

```
s.order()
Out: 12
```

Przykład 2.32

Dla danej permutacji:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 4 & 6 & 7 & 2 & 8 & 1 & 3 \end{pmatrix}$$

znajdziemy najmniejszą liczbę naturalną $m \in \mathbb{N}$, dla której zachodzi równość:

$$\sigma^{95} = \sigma^m.$$

Ponownie zaczynamy od zapisania permutacji σ w postaci iloczynu cykli rozłącznych oraz znalezienia ich długości:

$$\sigma = \underbrace{(1\ 5\ 2\ 4\ 7)}_{d_1=5} \underbrace{(3\ 6\ 8)}_{d_2=3}.$$

Wynika stąd, że rząd permutacji σ jest równy:

$$r(\sigma) = \text{NWW}(d_1, d_2) = \text{NWW}(5, 3) = 15.$$

Z definicji rzędu wynika, że w wykładniku permutacji σ możemy wykonywać działania modulo 15. Stąd otrzymujemy równość:

$$\sigma^{95} = \sigma^{95-6 \cdot 15} = \sigma^5,$$

a zatem szukaną najmniejszą liczbą naturalną jest liczba $m = 5$.

W **Sage** możemy w tym celu zastosować symbol operacji modulo %, tak jak to zrobiliśmy w przykładzie 2.18, lub wykorzystać funkcję **mod()**:

```
s = Permutation( [5, 4, 6, 7, 2, 8, 1, 3] )
mod( 95, s.order() )
Out: 5
```

2.4.3. Równania permutacyjne

Równania liniowe w grupie permutacji zawsze mają dokładnie jedno rozwiązanie, podobnie jak w grupie reszt modulo n . Trzeba jednak pamiętać, że w przeciwieństwie do grupy $\mathbb{Z}_n$, działanie składania permutacji „ $\circ$ ” nie jest przemienne, dlatego nie możemy dowolnie zmieniać kolejności permutacji występujących w równaniu.

Przykład 2.33

Znajdziemy permutację $\chi \in S_7$ spełniającą równanie:

$$\sigma^{44} \circ \chi \circ \tau^{-2} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 7 & 5 & 1 & 2 & 6 & 4 \end{pmatrix},$$

gdzie:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 5 & 4 & 2 & 7 & 3 & 6 \end{pmatrix} \quad \text{oraz} \quad \tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 2 & 6 & 4 & 3 & 1 & 7 & 5 \end{pmatrix}.$$

Wprowadzamy oznaczenie dla permutacji występującej po prawej stronie równania:

$$\mu = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 7 & 5 & 1 & 2 & 6 & 4 \end{pmatrix}.$$

Wtedy równanie przyjmuje postać:

$$\sigma^{44} \circ \chi \circ \tau^{-2} = \mu.$$

Teraz chcemy przekształcić równanie tak, aby uzyskać wzór na wyznaczenie permutacji χ . Najpierw mnożymy je stronami przez σ^{-44} z lewej strony:

$$\underbrace{\sigma^{-44} \circ \sigma^{44}}_{\text{id}} \circ \chi \circ \tau^{-2} = \sigma^{-44} \circ \mu.$$

Po lewej stronie uzyskujemy permutację identycznościową id , czyli element neutralny w grupie permutacji, który nie zmienia wyniku działania. Zatem równanie upraszcza się do:

$$\chi \circ \tau^{-2} = \sigma^{-44} \circ \mu.$$

Następnie wykonujemy analogiczną operację, tylko tym razem mnożymy równanie stronami przez τ^2 z prawej strony:

$$\chi \circ \underbrace{\tau^{-2} \circ \tau^2}_{\text{id}} = \sigma^{-44} \circ \mu \circ \tau^2.$$

Trzeba zwracać uwagę, z której strony dokonujemy mnożenia, ponieważ w grupie permutacji nie można zmieniać kolejności elementów. W rezultacie otrzymujemy wzór pozwalający obliczyć permutację χ :

$$\chi = \sigma^{-44} \circ \mu \circ \tau^2.$$

Znajdujemy teraz rząd permutacji σ , aby obniżyć wykładnik potęgi w zapisie σ^{-44} , tak jak robiliśmy to w przykładzie 2.32. Permutację tę można zapisać w postaci jednego cyklu $\sigma = (2\ 5\ 7\ 6\ 3\ 4)$ o długości $d_1 = 6$, zatem jej rząd jest równy:

$$r(\sigma) = 6.$$

Możemy więc w wykładniku permutacji σ wykonywać działania modulo 6:

$$\sigma^{-44} = \sigma^{-44+7 \cdot 6} = \sigma^{-2}.$$

Wartość tę można też zapisać jako σ^4 , jednak obecny wykładnik znajduje się bliżej zera. Aby wyznaczyć σ^{-2} , w permutacji σ dwukrotnie „idziemy w górę”:

$$\sigma^{-44} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 3 & 7 & 6 & 4 & 5 & 2 \end{pmatrix}.$$

Z kolei, aby znaleźć τ^2 , dwukrotnie „idziemy w dół” w permutacji τ :

$$\tau^2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 6 & 7 & 3 & 4 & 2 & 5 & 1 \end{pmatrix}.$$

Wracając do wzoru na permutację χ , wykonujemy działania „od prawej do lewej strony” zgodnie z kolejnością: $\tau^2 \rightarrow \mu \rightarrow \sigma^{-44}$. Otrzymujemy wynik:

$$\chi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 6 & 4 & 1 & 2 & 3 & 7 \end{pmatrix}.$$

W **Sage** permutacje definiujemy za pomocą obiektu **PermutationGroupElement()**, aby dostępna była operacja potęgowania. Następnie stosujemy wyprowadzony wzór, pamiętając o tym, że działania zapisujemy w odwrotnej kolejności:

```
s = PermutationGroupElement( [1, 5, 4, 2, 7, 3, 6] )
t = PermutationGroupElement( [2, 6, 4, 3, 1, 7, 5] )
m = PermutationGroupElement( [3, 7, 5, 1, 2, 6, 4] )

Permutation( t^2 * m * s^-44 )
Out: [5, 6, 4, 1, 2, 3, 7]
```

Rozwiązanie można również znaleźć metodą **Brute Force** – poprzez sprawdzenie w pętli **for** wszystkich elementów grupy symetrycznej S_7 , którą w **Sage** uzyskujemy za pomocą polecenia **SymmetricGroup(7)**:

```
for x in SymmetricGroup(7):
    if t^-2 * x * s^44 == m:
        print( 'Rozwiązanie:', Permutation(x) )
Out: Rozwiązanie: [5, 6, 4, 1, 2, 3, 7]
```

Warto zapoznać się z innymi możliwościami dotyczącymi grup permutacji w dokumentacji permutacji¹² w **Sage** [4]. Więcej informacji na temat struktur algebraicznych i ich zastosowań można znaleźć w pozycjach [2] i [3].

Dasz radę! 

¹² Sage, Permutations, <https://doc.sagemath.org/html/en/reference/combinat/sage/combinat/permutation.html> [dostęp: 20.04.2026].

3. Macierze

3.1. Operacje na macierzach

Macierz to prostokątna tablica, której elementami najczęściej są liczby, choć w bardziej ogólnym ujęciu mogą być nimi również inne obiekty, takie jak symbole czy wyrażenia algebraiczne. Struktura ta stanowi niezwykle ważne narzędzie w algebrze, informatyce i naukach technicznych, ponieważ pozwala w zwartej formie zapisywać dane oraz opisywać różnorodne przekształcenia. Intuicyjnie macierz można traktować jako uporządkowaną „tabelę liczb”, która nie tylko przechowuje informacje, lecz także umożliwia wykonywanie operacji algebraicznych.

Każda macierz ma określone wymiary, które zapisujemy, podając kolejno liczbę wierszy oraz liczbę kolumn. Okazuje się, że zbiór wszystkich macierzy o tych samych wymiarach wraz z działaniem dodawania tworzą grupę przemenną. W przypadku mnożenia sytuacja jest bardziej złożona: nie każdą parę macierzy można ze sobą pomnożyć, a samo mnożenie macierzy nie jest przemienne, podobnie jak złożenie permutacji. Ponadto jedynie wybrane macierze mają element odwrotny względem mnożenia, zwany macierzą odwrotną. To powoduje, że zbiór wszystkich macierzy nie tworzy ciała, lecz pierścienia.

W tym rozdziale skupimy się na wprowadzeniu podstawowych pojęć związanych z macierzami. Zaczniemy od omówienia sposobów zapisu macierzy w języku **Sage**. Następnie przedstawimy metody segmentacji, czyli wyodrębniania wybranych fragmentów macierzy oraz odwoływania się do jej elementów. Przeanalizujemy również podstawowe działania na macierzach oraz przedstawimy przykłady macierzy charakterystycznych oraz specjalnych typów operacji, które odgrywają istotną rolę w dalszych zastosowaniach.

3.1.1. Segmentacja macierzy

Definicja. Macierzą o m wierszach i n kolumnach nazywamy odwzorowanie:

$$A: \{1, \dots, m\} \times \{1, \dots, n\} \rightarrow \mathbb{K},$$

gdzie $\mathbb{K}$ jest ciałem. Wartość tego odwzorowania dla pary (i, j) oznaczamy przez a_{ij} i zapisujemy w postaci:

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}.$$

Mówimy, że macierz $\mathbf{A}$ ma wymiary $m \times n$ – zawsze najpierw podajemy liczbę wierszy, a następnie liczbę kolumn.

Przykład 3.1

Zdefiniujemy macierz o wymiarach 3×4 :

$$\mathbf{M} = \begin{bmatrix} 2 & -1 & 5 & 3 \\ -3 & 0 & 1 & 1 \\ -1 & 4 & 0 & -1 \end{bmatrix}.$$

Aby utworzyć macierz w **Sage**, korzystamy z funkcji **matrix()**. Poszczególne wiersze zapisujemy w oddzielnych nawiasach kwadratowych:

```
M = matrix( [[2, -1, 5, 3], [-3, 0, 1, 1], [-1, 4, 0, -1]] )
M
Out: [ 2 -1 5 3]
      [-3 0 1 1]
      [-1 4 0 -1]
```

Alternatywnie można przekazać kolejne elementy macierzy w jednej liście, określając jednocześnie liczbę wierszy (nrows) lub kolumn (ncols):

```
M = matrix( [2, -1, 5, 3, -3, 0, 1, 1, -1, 4, 0, -1],
             nrows=3 )
show(M)
Out: ( 2  -1  5  3 )
      (-3  0  1  1 )
      (-1  4  0 -1 )
```

Liczbę wierszy i kolumn uzyskujemy odpowiednio za pomocą funkcji **nrows()** i **ncols()**:

```
print( 'Liczba wierszy:', M.nrows() )
print( 'Liczba kolumn:', M.ncols() )
```

Out: Liczba wierszy: 3

Liczba kolumn: 4

W praktycznych zastosowaniach często zachodzi potrzeba odwołania się do konkretnego elementu macierzy lub wyodrębnienia jej wybranego fragmentu. Taką operację nazywamy segmentacją tablicy dwuwymiarowej.

Przykład 3.2

Zaprezentujemy metody segmentacji macierzy:

$$\mathbf{C} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}.$$

Macierz $\mathbf{C}$ można zdefiniować sprytnie, wykorzystując funkcję **range()**, która generuje listę kolejnych liczb całkowitych z przedziału lewostronnie domkniętego:

```
C = matrix( range(1, 10), nrows=3 )
```

C

Out: [1 2 3]

[4 5 6]

[7 8 9]

Do poszczególnych elementów c_{ij} macierzy $\mathbf{C}$ możemy odwoływać się za pomocą składni $C[i, j]$ lub $C[i][j]$. Poniżej wypiszemy wartości elementów:

- c_{11} znajdującego się w lewym górnym rogu macierzy $\mathbf{C}$;
- c_{23} położonego na przecięciu drugiego wiersza i trzeciej kolumny.

W języku **Python** obowiązuje indeksowanie od zera, więc dla każdego indeksu podajemy wartość mniejszą o 1:

```
print( 'Element c11:', C[0, 0] )
```

```
print( 'Element c23:', C[1][2] )
```

Out: Element c11: 1

Element c23: 6

Drugi sposób zapisu, czyli $C[i][j]$, jest bardziej restrykcyjny – nie pozwala na nadpisywanie elementów macierzy. Przykładowo, jeżeli chcemy w ten sposób zmienić wartość elementu c_{33} w prawym dolnym rogu, to otrzymamy błąd:

```
C[2][2] = 12
```

```
Out: ValueError: vector is immutable;  
      please change a copy instead (use copy())
```

Pierwsza konwencja zapisu, czyli $C[i, j]$, pozwala już natomiast poprawnie wykonać tę operację nadpisania:

```
C[2, 2] = 12
```

```
C
```

```
Out: [ 1  2  3]  
      [ 4  5  6]  
      [ 7  8 12]
```

W **Pythonie** zapis `:2` oznacza pobranie wszystkich elementów od początku listy do elementu o indeksie 2, ale nie włącznie. W praktyce daje to więc dwa pierwsze elementy. Z kolei zapis `2:` zwraca wszystkie elementy od indeksu 2 włącznie aż do końca listy. Dla macierzy obowiązuje ta sama zasada, przy czym pierwszy indeks odnosi się do numerów wierszy, a drugi – do kolumn. Poniżej wyodrębniamy macierze o wymiarach 2×2 , znajdujące się odpowiednio w lewym górnym oraz prawym dolnym rogu macierzy **C**:

```
print('Macierz 2x2 w lewym górnym rogu:')  
show( C[:2, :2] )  
print('Macierz 2x2 w prawym dolnym rogu:')  
show( C[1:, 1:] )
```

```
Out: Macierz 2x2 w lewym górnym rogu:
```

```
( 1  2 )  
( 4  5 )
```

```
Macierz 2x2 w prawym dolnym rogu:
```

```
( 5  6 )  
( 8 12 )
```

Gdy chcemy wziąć wszystkie wiersze (lub kolumny), stosujemy w odpowiednim miejscu symbol dwukropka. Weźmiemy drugą kolumnę macierzy **C**:

```
C[:, 1]
Out: [2]
      [5]
      [8]
```

Aby otrzymać drugi wiersz tej macierzy, wystarczy użyć zapisu `C[1, :]`. W tym przypadku możliwy jest także zapis bez dwukropka, jednak wtedy wynikiem będzie wektor, a nie macierz. Jest on wyświetlany poziomo w nawiasach okrągłych:

```
C[1]
Out: (4, 5, 6)
```

W języku **Python** dostępne są również indeksy ujemne, które oznaczają odliczanie od końca. Na przykład poniższy zapis wybiera dwie ostatnie kolumny macierzy **C**:

```
C[:, -2:]
Out: [ 2  3]
      [ 5  6]
      [ 8 12]
```

Poszczególne wiersze i kolumny możemy również powielać. Wówczas przekazujemy w liście ich indeksy na odpowiedniej pozycji. Weźmiemy pierwszy wiersz oraz dwa drugie wiersze:

```
C[[0, 1, 1]]
Out: [1 2 3]
      [4 5 6]
      [4 5 6]
```

Poniższy kod zwraca z kolei kolumny w przestawionej kolejności:

```
C[:, [2, 0, 1]]  
Out: [ 3  1  2]  
      [ 6  4  5]  
      [12  7  8]
```

Należy zwrócić uwagę, że wszystkie te operacje, z wyjątkiem modyfikacji wartości elementu, jedynie wypisują wynik na ekranie, a nie zmieniają oryginalnej macierzy.

3.1.2. Podstawowe działania na macierzach

Zbiór wszystkich macierzy o wymiarach $m \times n$ i elementach należących do ciała $\mathbb{K}$ oznaczamy symbolem $\mathbf{M}_{m \times n}(\mathbb{K})$.

Przykład 3.3

Przeanalizujemy dostępne operacje, które można wykonać na poniższych dwóch macierzach należących do zbioru $\mathbf{M}_{2 \times 2}(\mathbb{R})$:

$$\mathbf{A} = \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix}.$$

1. Dodawanie i odejmowanie.

Dwie macierze możemy dodawać i odejmować tylko wtedy, gdy mają takie same wymiary. Wówczas wykonujemy te działania dla odpowiednich elementów:

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} = \begin{bmatrix} 1+2 & 3+3 \\ 0+4 & 2+1 \end{bmatrix} = \begin{bmatrix} 3 & 6 \\ 4 & 3 \end{bmatrix}$$

$$\mathbf{A} - \mathbf{B} = \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix} - \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} = \begin{bmatrix} 1-2 & 3-3 \\ 0-4 & 2-1 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ -4 & 1 \end{bmatrix}.$$

Ponadto dodawanie macierzy jest działaniem przemiennym. W **Sage** definiujemy macierze **A** i **B**, a następnie wykorzystujemy operator dodawania (+) lub odejmowania (-):

```
A = matrix( [[1, 3], [0, 2]] )
B = matrix( [[2, 3], [4, 1]] )
A+B, A-B
Out: (
  [3 6] [-1 0]
  [4 3], [-4 1]
)
```

2. Mnożenie przez skalar.

Skalarem nazywamy element ciała $\mathbb{K}$. W przypadku macierzy najczęściej wykorzystujemy ciała liczb rzeczywistych ($\mathbb{R}$) oraz zespolonych ($\mathbb{C}$). Mnożenie macierzy przez skalar jest przemienne, czyli dla dowolnego skalaru $k \in \mathbb{K}$ i dowolnej macierzy $\mathbf{A} \in \mathbf{M}_{m \times n}(\mathbb{K})$ mamy:

$$k \cdot \mathbf{A} = \mathbf{A} \cdot k.$$

Przykładowo poniżej pomnożymy macierz $\mathbf{A}$ przez liczbę 2:

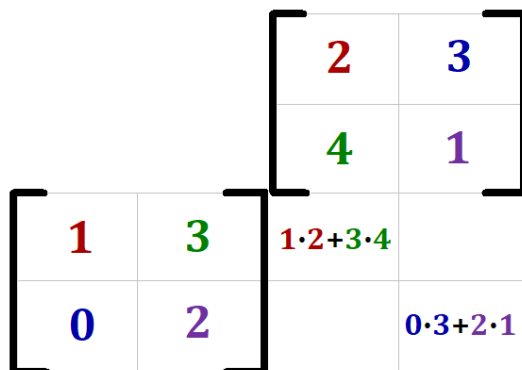
$$2\mathbf{A} = 2 \cdot \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 2 \cdot 1 & 2 \cdot 3 \\ 2 \cdot 0 & 2 \cdot 2 \end{bmatrix} = \begin{bmatrix} 2 & 6 \\ 0 & 4 \end{bmatrix}.$$

W **Sage** stosujemy operator mnożenia (*):

```
2*A
Out: [2 6]
     [0 4]
```

3. Mnożenie macierzy.

Dwie macierze można pomnożyć tylko wtedy, gdy ich środkowe wymiary są takie same. Oznacza to, że działanie mnożenia jest możliwe dla macierzy o wymiarach $m \times k$ oraz $k \times n$. W wyniku otrzymujemy wówczas macierz o wymiarach $m \times n$. Na rysunku 3.1 przedstawiono schemat ułatwiający zapamiętanie sposobu mnożenia macierzy – łączymy je wierzchołkami i znajdujemy sumę iloczynów odpowiadających sobie elementów.



Rysunek 3.1. Schemat mnożenia macierzy

Mnożenie macierzy nie jest przemienne. Możemy zapisać to schematycznie jako:

$$\mathbf{AB} \neq \mathbf{BA}.$$

Co więcej, może się zdarzyć, że jednego z tych działań w ogóle nie można wykonać – na przykład dla macierzy **A** i **B** o wymiarach odpowiednio 2×3 i 3×4 . Aby przeprowadzić mnożenie macierzy w **Sage**, również korzystamy z operatora *****:

A*B

Out: [14 6]

[8 2]

4. Potęgowanie.

Potęgowanie macierzy jest skróconym zapisem jej mnożenia przez siebie samą, czyli:

$$\mathbf{A}^n = \underbrace{\mathbf{A} \cdot \dots \cdot \mathbf{A}}_n$$

Obliczymy na przykład drugą potęgę macierzy **B**:

$$\mathbf{B}^2 = \mathbf{B} \cdot \mathbf{B} = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} = \begin{bmatrix} 2 \cdot 2 + 3 \cdot 4 & 2 \cdot 3 + 3 \cdot 1 \\ 4 \cdot 2 + 1 \cdot 4 & 4 \cdot 3 + 1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 16 & 9 \\ 12 & 13 \end{bmatrix}.$$

Potęgować możemy tylko macierz kwadratową, czyli macierz o takiej samej liczbie wierszy i kolumn. Aby wykonać to działanie w **Sage**, stosujemy operator potęgowania (^):

```
B^2
```

```
Out: [16 9]
      [12 13]
```

5. Transpozycja.

Transpozycja macierzy polega na zamianie jej wierszy na kolumny. Macierz transponowana do macierzy o wymiarach $m \times n$ ma wymiary $n \times m$. Poniżej transponujemy macierz **A**:

$$\mathbf{A}^T = \begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 3 & 2 \end{bmatrix}.$$

W **Sage** operację tę wykonujemy za pomocą funkcji **transpose()** lub krócej jako:

```
A.T
```

```
Out: [1 0]
      [3 2]
```

6. Łączenie macierzy.

W języku **Sage** macierze można połączyć:

- wzdłuż wierszy, korzystając z funkcji **stack()**;
- wzdłuż kolumn za pomocą funkcji **augment()**.

Poniżej przedstawiamy przykład zastosowania drugiej funkcji:

```
A.augment(B)
```

```
Out: [1 3 2 3]
      [0 2 4 1]
```

Przykład 3.4

Dane są macierze:

$$\mathbf{A} = \begin{bmatrix} 0 & 3 \\ 1 & 4 \\ 2 & 5 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} -3 & -2 & -1 \\ 0 & 1 & 2 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0 & 2 \\ 4 & 6 \end{bmatrix}.$$

Wykonamy poniższe działania lub uzasadnimy, dlaczego nie jest to możliwe:

- a) $\mathbf{AC}^T\mathbf{B}$; b) $(\mathbf{CB})^T\mathbf{A}$.

Definiujemy macierze **A**, **B** i **C**. Możemy to zrobić sprytnie w następujący sposób:

```
A = matrix( range(6), nrows=2 ).T
B = matrix( range(-3, 3), nrows=2 )
C = matrix( range(0, 7, 2), nrows=2 )
```

```
print('Macierz A:', A, sep='\n')
print('Macierz B:', B, sep='\n')
print('Macierz C:', C, sep='\n')
```

Out: Macierz A:

[0 3]

[1 4]

[2 5]

Macierz B:

[-3 -2 -1]

[0 1 2]

Macierz C:

[0 2]

[4 6]

- a) Przeprowadzamy rachunki na wymiarach macierzy występujących w działaniu $\mathbf{AC}^T\mathbf{B}$:

$$3 \times 2 \cdot (2 \times 2)^T \cdot 2 \times 3 = 3 \times \underbrace{2 \cdot 2}_{=} \times \underbrace{2 \cdot 2}_{=} \times 3 = 3 \times 3.$$

Możemy wykonać to działanie, ponieważ środkowe wymiary występujące w iloczynach macierzy są równe:

```
A * C.T * B
```

Out: [-18 6 30]

[-24 12 48]

[-30 18 66]

b) Ponownie zaczynamy od rachunków na wymiarach dla działania $(CB)^T A$:

$$\left(2 \times \underbrace{2 \cdot 2}_{=} \times 3\right)^T \cdot 3 \times 2 = (2 \times 3)^T \cdot 3 \times 2 = 3 \times \underbrace{2 \cdot 3}_{\neq} \times 2.$$

Tym razem nie jesteśmy w stanie wykonać tego działania, ponieważ w jednym z iloczynów środkowe wymiary macierzy nie są równe. Po zastosowaniu tej operacji w **Sage** pojawia się komunikat o błędzie – nie można pomnożyć dwóch macierzy o wymiarach 3×2 :

```
(C*B).T * A
```

```
Out: TypeError: unsupported operand parent(s) for *:
```

```
'Full MatrixSpace of 3 by 2 dense
```

```
matrices over Integer Ring' and
```

```
'Full MatrixSpace of 3 by 2 dense
```

```
matrices over Integer Ring'
```

3.1.3. Macierze charakterystyczne

W tej części przyjrzymy się kilku szczególnym rodzajom macierzy i zobaczymy, w jaki sposób można je utworzyć w języku **Sage**.

1. Macierz zerowa.

Macierzą zerową nazywamy macierz złożoną wyłącznie z zer. Jest ona elementem neutralnym dodawania w grupie macierzy o wymiarach $m \times n$, czyli $(\mathbf{M}_{m \times n}(\mathbb{R}), +)$. Oznaczamy ją symbolem $\mathbf{0}$ i ma ona tę własność, że dla każdej macierzy $\mathbf{A} \in \mathbf{M}_{m \times n}(\mathbb{R})$ zachodzi:

$$\mathbf{A} + \mathbf{0} = \mathbf{0} + \mathbf{A} = \mathbf{A}.$$

W **Sage** macierz zerową można utworzyć za pomocą funkcji `zero_matrix()`. Jako argumenty podajemy kolejno liczbę wierszy i kolumn docelowej macierzy:

```
zero_matrix(3, 2)
```

```
Out: [0 0]
```

```
      [0 0]
```

```
      [0 0]
```

2. Macierz jednostkowa.

Macierz jednostkowa, oznaczana symbolem **I**, pełni funkcję elementu neutralnego względem mnożenia macierzy. Jest to macierz kwadratowa, która na tzw. przekątnej głównej zawiera jedynki, a w pozostałych miejscach zera. Dla dowolnej macierzy kwadratowej **A** zachodzi równość:

$$\mathbf{A} \cdot \mathbf{I} = \mathbf{I} \cdot \mathbf{A} = \mathbf{A}.$$

Aby zdefiniować ją w **Sage**, korzystamy z funkcji **identity_matrix()**:

```
identity_matrix(3)
```

```
Out: [1 0 0]
```

```
      [0 1 0]
```

```
      [0 0 1]
```

3. Macierz diagonalna.

Macierz diagonalna to macierz kwadratowa, w której wszystkie elementy poza przekątną główną są równe zero. Do jej utworzenia w **Sage** służy funkcja **diagonal_matrix()**, której przekazujemy listę elementów umieszczanych na przekątnej głównej:

```
diagonal_matrix( [1, 2, 4, 3] )
```

```
Out: [1 0 0 0]
```

```
      [0 2 0 0]
```

```
      [0 0 4 0]
```

```
      [0 0 0 3]
```

4. Macierz trójkątna dolna i trójkątna górna.

Macierz trójkątna dolna to macierz kwadratowa, w której wszystkie elementy znajdujące się nad przekątną główną są równe zero. Analogicznie macierz trójkątna górna ma zera pod przekątną główną. Tego typu macierze odgrywają ważną rolę w algorytmach numerycznych oraz w dekompozycjach macierzy, czyli procesach ich rozkładu na iloczyn innych macierzy o specyficznych właściwościach. Przykładem jest dekompozycja LU (**Lower-Upper Decomposition**), która pozwala zapisać daną macierz **A** w postaci:

$$\mathbf{P} \cdot \mathbf{A} = \mathbf{L} \cdot \mathbf{U},$$

gdzie **L** jest macierzą trójkątną dolną, **U** – trójkątną górną, a **P** jest tzw. macierzą permutacji.

Macierze występujące w dekompozycji LU dla zadanej macierzy **A** można uzyskać w **Sage** za pomocą funkcji **LU()**, która zwraca trójkę macierzy (**P**, **L**, **U**):

```
A = matrix( [[2, 3], [4, 1]] )
A.LU()
Out: (
  [0 1] [ 1 0] [ 4 1]
  [1 0], [1/2 1], [ 0 5/2]
)
```

5. Macierz blokowa.

Macierz blokowa to macierz, którą traktujemy jako podzieloną na prostokątne fragmenty, zwane blokami. Na przykład macierz 4×4 można zapisać jako macierz blokową postaci:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix},$$

gdzie **A**, **B**, **C** i **D** są macierzami o wymiarach 2×2 . Aby utworzyć ją w **Sage**, sklejamy mniejsze macierze za pomocą funkcji **block_matrix()**:

```
A = matrix( [[1, 2], [3, 4]] )
B = matrix( [[5, 6], [7, 8]] )
block_matrix( [[A, B], [B, A]] )
Out: [1 2|5 6]
      [3 4|7 8]
      [---+---]
      [5 6|1 2]
      [7 8|3 4]
```

3.2. Układy równań liniowych

Rozwiązywanie układów równań liniowych w szkole średniej zazwyczaj polegało na wyznaczaniu kolejnych zmiennych z układu i ich podstawianiu do pozostałych

równań, aż do uzyskania ostatecznego rozwiązania. Na studiach spotykamy tę samą ideę, ale przedstawioną w bardziej ogólnej i uporządkowanej formie, ukrytą pod nazwą „metoda eliminacji Gaussa”. Algorytm ten wykorzystuje zapis macierzowy, dzięki czemu cały układ można traktować jako jeden obiekt matematyczny, a kolejne przekształcenia sprowadzają się do prostych operacji na macierzach.

W tym rozdziale zobaczymy, jak zapisywać układ równań w postaci macierzowej oraz jakie operacje – zwane elementarnymi – można na nim wykonywać. Odpowiadają one przekształceniom stosowanym w zwykłych układach równań, lecz zapisanych w formie macierzowej. Narzędzia te będą miały kluczowe znaczenie w dalszych zastosowaniach algebry. Szczególnie istotne jest twierdzenie Kroneckera-Capellego, które pozwala określić liczbę rozwiązań układu równań oraz, co ważne, od ilu parametrów zależą te rozwiązania. Umożliwi nam to później dokładniejsze zrozumienie struktury całej przestrzeni rozwiązań.

3.2.1. Operacje elementarne

Dla uproszczenia teorii dotyczącej układów równań liniowych przedstawimy na konkretnym przykładzie:

$$\begin{cases} -x + 2y - 3z = 4 \\ x + y + z = 0 \\ 3x - 5y + 4z = 1. \end{cases}$$

1. **Macierzą układu** nazywamy macierz:

$$\mathbf{A} = \begin{bmatrix} -1 & 2 & -3 \\ 1 & 1 & 1 \\ 3 & -5 & 4 \end{bmatrix},$$

złożoną ze współczynników stojących przy niewiadomych. Liczba jej wierszy odpowiada liczbie równań, a liczba kolumn – liczbie zmiennych w układzie.

2. **Wektorem wyrazów wolnych** nazywamy wektor:

$$\mathbf{b} = \begin{bmatrix} 4 \\ 0 \\ 1 \end{bmatrix},$$

który składa się z liczb znajdujących się po prawej stronie równań.

3. **Kolumna niewiadomych** to wektor:

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix},$$

zawierający wszystkie zmienne występujące w układzie. Dzięki tym oznaczeniom cały układ równań możemy zapisać w tzw. postaci macierzowej, czyli:

$$\mathbf{Ax} = \mathbf{b}.$$

Pozwala nam to uzyskać równoważny zapis układu w języku macierzy. Rzeczywiście, po wykonaniu mnożenia macierzy otrzymujemy układ równoważny pierwotnemu, zapisany w postaci równości wektorów kolumnowych:

$$\begin{bmatrix} -1 & 2 & -3 \\ 1 & 1 & 1 \\ 3 & -5 & 4 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} -x + 2y - 3z \\ x + y + z \\ 3x - 5y + 4z \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 1 \end{bmatrix}.$$

4. **Macierz uzupełnioną** otrzymujemy poprzez połączenie macierzy układu $\mathbf{A}$ i kolumny wyrazów wolnych $\mathbf{b}$:

$$\mathbf{U} = \left[\begin{array}{ccc|c} -1 & 2 & -3 & 4 \\ 1 & 1 & 1 & 0 \\ 3 & -5 & 4 & 1 \end{array} \right].$$

Taki zapis pozwala bezpośrednio powiązać układ równań z odpowiadającą mu macierzą.

Operacje elementarne na wierszach macierzy to przekształcenia, które odpowiadają dozwolonym operacjom na związanym z nią układzie równań. Pozwalają one uzyskać układ równoważny, który ma dokładnie te same rozwiązania. Wyróżniamy następujące rodzaje operacji elementarnych:

1. Zamiana dwóch wierszy.

Jeżeli w układzie równań zamienimy miejscami pierwsze i drugie równanie, uzyskamy w wyniku ten sam układ:

$$\begin{cases} -x + 2y - 3z = 4 \\ x + y + z = 0 \\ 3x - 5y + 4z = 1 \end{cases} \rightarrow \begin{cases} x + y + z = 0 \\ -x + 2y - 3z = 4 \\ 3x - 5y + 4z = 1 \end{cases}$$

W związku z tym jedną z operacji elementarnych jest zamiana miejscami pierwszego i drugiego wiersza macierzy:

$$\left[\begin{array}{ccc|c} -1 & 2 & -3 & 4 \\ 1 & 1 & 1 & 0 \\ 3 & -5 & 4 & 1 \end{array} \right] \xrightarrow{w_1 \leftrightarrow w_2} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ -1 & 2 & -3 & 4 \\ 3 & -5 & 4 & 1 \end{array} \right].$$

W **Sage** rozpoczynamy od zdefiniowania macierzy uzupełnionej **U**. Aby zamienić miejscami dwa pierwsze wiersze, możemy zastosować zapis `U[[1, 0, 2]]` lub skorzystać z funkcji `swap_rows()`. Drugi sposób bezpośrednio modyfikuje macierz **U**:

```
U = matrix([[-1, 2, -3, 4], [1, 1, 1, 0], [3, -5, 4, 1]])
U.swap_rows(0, 1)
U
Out: [ 1 1 1 0]
      [-1 2 -3 4]
      [ 3 -5 4 1]
```

2. Mnożenie wiersza przez liczbę różną od zera.

W układzie jedną z dozwolonych operacji jest pomnożenie drugiego równania przez 2:

$$\begin{cases} x + y + z = 0 \\ -x + 2y - 3z = 4 \\ 3x - 5y + 4z = 1 \end{cases} \rightarrow \begin{cases} x + y + z = 0 \\ -2x + 4y - 6z = 8 \\ 3x - 5y + 4z = 1 \end{cases}$$

dlatego operacją elementarną jest pomnożenie drugiego wiersza macierzy przez 2:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ -1 & 2 & -3 & 4 \\ 3 & -5 & 4 & 1 \end{array} \right] \xrightarrow{w_2 \cdot 2} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ -2 & 4 & -6 & 8 \\ 3 & -5 & 4 & 1 \end{array} \right].$$

Aby wykonać tę czynność w **Sage**, stosujemy operator `*=`:

```
U[1] *= 2
U
Out: [ 1 1 1 0]
      [-2 4 -6 8]
      [ 3 -5 4 1]
```

3. Dodanie do wiersza innego wiersza pomnożonego przez liczbę.

Równania w układzie możemy dodawać i odejmować stronami – tak działa na przykład metoda przeciwnych współczynników. Zatem można do trzeciego równania dodać równanie drugie:

$$\begin{cases} x + y + z = 0 \\ -2x + 4y - 6z = 8 \\ 3x - 5y + 4z = 1 \end{cases} \rightarrow \begin{cases} x + y + z = 0 \\ -2x + 4y - 6z = 8 \\ x - y - 2z = 9 \end{cases}$$

Analogicznie operacją elementarną jest dodanie drugiego wiersza do wiersza trzeciego:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ -2 & 4 & -6 & 8 \\ 3 & -5 & 4 & 1 \end{array} \right] \xrightarrow{w_3+w_2} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ -2 & 4 & -6 & 8 \\ 1 & -1 & -2 & 9 \end{array} \right]$$

W **Sage** możemy zastosować operator `+=`:

```
U[2] += U[1]
U
Out: [ 1  1  1  0]
     [-2  4 -6  8]
     [ 1 -1 -2  9]
```

3.2.2. Metoda eliminacji Gaussa

Rozwiązywanie układu równań metodą eliminacji Gaussa polega na sprowadzeniu macierzy uzupełnionej układu **U** za pomocą operacji elementarnych do postaci schodkowej. Na rysunku 3.2 przedstawiono przykładową postać schodkową macierzy o wymiarach 4×6 , w której:

- kolorem czerwonym zaznaczone są tzw. schodki – występują one w kolumnach 1, 2 oraz 4 i opadają zawsze tylko o jeden poziom, czyli nie mogą być „dwupiętrowe”;
- kolor zielony reprezentuje elementy wiodące (pivoty) – są to pierwsze niezerowe elementy w każdym wierszu macierzy schodkowej;
- pod schodkami znajdują się same zera, natomiast na pozostałych pozycjach mogą występować dowolne liczby.

Twierdzenie. Każdą macierz można sprowadzić do postaci schodkowej za pomocą operacji elementarnych na wierszach.

●					
0	●				
0	0	0	●		
0	0	0	0	0	0

Rysunek 3.2. Przykład postaci schodkowej macierzy o wymiarach 4×6

Przykład 3.5

Sprowadzimy do postaci schodkowej macierz uzupełnioną układu:

$$\begin{cases} -2x + y + 5z = -1 \\ 2x - 4y - 2z = 4 \\ 3x - 4y - 7z = 3. \end{cases}$$

Najpierw zapisujemy macierz uzupełnioną układu:

$$U = \left[\begin{array}{ccc|c} -2 & 1 & 5 & -1 \\ 2 & -4 & -2 & 4 \\ 3 & -4 & -7 & 3 \end{array} \right].$$

Poniżej przedstawiamy algorytm sprowadzania dowolnej macierzy do postaci schodkowej:

1. Jeżeli to możliwe, dzielimy wiersz przez pewną liczbę.

Możemy podzielić drugi wiersz przez 2, ponieważ wszystkie jego elementy są parzyste. W ten sposób upraszczamy rachunki, zmniejszając wartości elementów macierzy:

$$\left[\begin{array}{ccc|c} -2 & 1 & 5 & -1 \\ 2 & -4 & -2 & 4 \\ 3 & -4 & -7 & 3 \end{array} \right] \xrightarrow{w_2 \cdot \frac{1}{2}} \left[\begin{array}{ccc|c} -2 & 1 & 5 & -1 \\ 1 & -2 & -1 & 2 \\ 3 & -4 & -7 & 3 \end{array} \right].$$

2. Ustawiamy w lewym górnym rogu liczbę 1 lub -1 .

Aby pierwszy element wiodący był równy 1, zamieniamy miejscami pierwszy i drugi wiersz:

$$\left[\begin{array}{ccc|c} -2 & 1 & 5 & -1 \\ 1 & -2 & -1 & 2 \\ 3 & -4 & -7 & 3 \end{array} \right] \xrightarrow{w_1 \leftrightarrow w_2} \left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ -2 & 1 & 5 & -1 \\ 3 & -4 & -7 & 3 \end{array} \right].$$

3. Wykorzystujemy element wiodący do wyzerowania wartości pod spodem.

Jest to powód, dla którego warto mieć w lewym górnym rogu liczbę 1 lub -1 . Możemy ją wykorzystać do zerowania elementów znajdujących się poniżej:

$$\left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ -2 & 1 & 5 & -1 \\ 3 & -4 & -7 & 3 \end{array} \right] \xrightarrow[w_3 - 3w_1]{w_2 + 2w_1} \left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -3 & 3 & 3 \\ 0 & 2 & -4 & -3 \end{array} \right].$$

Następnie powtarzamy algorytm dla kolejnego elementu wiodącego, pomijając pierwszy wiersz macierzy. Najpierw dzielimy drugi wiersz przez 3:

$$\left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -3 & 3 & 3 \\ 0 & 2 & -4 & -3 \end{array} \right] \xrightarrow{w_2 \cdot \frac{1}{3}} \left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 2 & -4 & -3 \end{array} \right].$$

Drugi element wiodący jest już równy -1 , więc wykorzystujemy go do wyzerowania liczby pod spodem. W ten sposób uzyskujemy postać schodkową macierzy **U**:

$$\left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 2 & -4 & -3 \end{array} \right] \xrightarrow{w_3 + 2w_2} \left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 0 & -2 & -1 \end{array} \right].$$

Przeprowadzamy kolejno wszystkie powyższe operacje elementarne w **Sage**:

```
U = matrix( [[-2, 1, 5, -1],
              [2, -4, -2, 4],
              [3, -4, -7, 3]] )
```

```
U[1] /= 2
```

```
U.swap_rows(0, 1)
```

```
U[1] += 2*U[0]
```

```
U[2] -= 3*U[0]
```

```
U[1] /= 3
```

```
U[2] += 2*U[1]
```

```
U
```

```
Out: [ 1 -2 -1 2]
```

```
     [ 0 -1 1 1]
```

```
     [ 0 0 -2 -1]
```

Macierz można sprowadzić do postaci schodkowej bezpośrednio za pomocą jednej z funkcji wbudowanych: **echelon_form()**, **rref()** lub **echelonize()**. Pierwsza z nich wypisuje postać schodkową na ekranie, druga dodatkowo zeruje elementy nad pivotami, natomiast ostatnia modyfikuje oryginalną macierz. Poniżej ponownie definiujemy macierz uzupełnioną **U**, tym razem łącząc w poziomie macierz układu **A** z wektorem wyrazów wolnych **b**:

```
A = matrix( [[-2, 1, 5], [2, -4, -2], [3, -4, -7]] )
b = vector( [-1, 4, 3] )
U = A.augment(b, subdivide=True)
U
Out: [-2  1  5|-1]
      [ 2 -4 -2| 4]
      [ 3 -4 -7| 3]
```

Przedstawiamy efekt działania funkcji **echelon_form()** oraz **rref()**:

```
U.echelon_form(), U.rref()
Out: (
  [ 1  0  7  5] [  1  0  0| 3/2]
  [ 0  1  7  3] [  0  1  0|-1/2]
  [ 0  0 12  6], [  0  0  1| 1/2]
)
```

Uzyskujemy inne wyniki niż przy obliczeniach ręcznych – jest to normalne, ponieważ każda macierz ma nieskończenie wiele różnych postaci schodkowych.

Przykład 3.6

Rozwiążemy układ równań z przykładu 3.5 metodą eliminacji Gaussa.

W poprzednim przykładzie sprowadziliśmy macierz uzupełnioną układu **U** do postaci schodkowej:

$$\mathbf{U} = \left[\begin{array}{ccc|c} -2 & 1 & 5 & -1 \\ 2 & -4 & -2 & 4 \\ 3 & -4 & -7 & 3 \end{array} \right] \sim \left[\begin{array}{ccc|c} 1 & -2 & -1 & 2 \\ 0 & -1 & 1 & 1 \\ 0 & 0 & -2 & -1 \end{array} \right].$$

Jest to najważniejszy etap metody eliminacji Gaussa. Aby jednak uzyskać pełne rozwiązanie układu równań, należy jeszcze wyznaczyć wartości niewiadomych. W tym celu zapisujemy kolejne wiersze macierzy schodkowej w postaci równań – od dołu do góry – a następnie rozwiązujemy je ręcznie. Zaczynamy od ostatniego wiersza:

$$-2z = -1 \rightarrow z = \frac{1}{2}.$$

W równaniu wynikającym z drugiego wiersza możemy już wykorzystać wyznaczoną wartość zmiennej z :

$$-y + z = 1 \rightarrow -y + \frac{1}{2} = 1 \rightarrow y = -\frac{1}{2}.$$

Na koniec wykonujemy analogiczne obliczenia dla wiersza pierwszego:

$$x - 2y - z = 2 \rightarrow x - 2 \cdot \left(-\frac{1}{2}\right) - \frac{1}{2} = 2 \rightarrow x = \frac{3}{2}.$$

W **Sage** możemy wykorzystać macierze **A** i **b** zdefiniowane w przykładzie 3.5, aby rozwiązać ten układ za pomocą funkcji **solve_right()**:

```
A.solve_right(b)
```

```
Out: (3/2, -1/2, 1/2)
```

Ten sposób działa jednak poprawnie tylko wtedy, gdy układ ma dokładnie jedno rozwiązanie. W przypadku ogólnym najlepiej skorzystać z funkcji **solve()**:

```
var('x y z')
```

```
solve([ -2*x+y+5*z== -1, 2*x-4*y-2*z==4, 3*x-4*y-7*z==3],  
      [x, y, z] )
```

```
Out: [[x == (3/2), y == (-1/2), z == (1/2)]]
```

3.2.3. Twierdzenie Kroneckera-Capellego

Definicja. Rzędem macierzy $\mathbf{A}$ nazywamy liczbę niezerowych wierszy uzyskanych w jej postaci schodkowej (czyli liczbę schodków). Oznaczamy go symbolem $r(\mathbf{A})$.

Przykład 3.7

Znajdziemy rząd macierzy:

$$\mathbf{A} = \begin{bmatrix} -1 & 3 & 2 & -4 \\ 2 & -2 & -1 & 0 \\ 3 & -1 & 0 & 2 \\ -1 & 7 & 5 & 0 \end{bmatrix}.$$

Sprowadzamy macierz $\mathbf{A}$ do postaci schodkowej:

```
A = matrix( [[-1, 3, 2, -4],  
             [2, -2, -1, 0],  
             [3, -1, 0, 2],  
             [-1, 7, 5, 0]] )
```

```
A.echelon_form()
```

```
Out: [1 1 1 2]  
     [0 4 3 4]  
     [0 0 0 6]  
     [0 0 0 0]
```

Uzyskujemy trzy schodki znajdujące się w kolumnach 1, 2 i 4. Rząd macierzy $\mathbf{A}$ jest zatem równy 3, co zapisujemy jako:

$$r(\mathbf{A}) = 3.$$

W Sage rząd wyznaczamy za pomocą funkcji `rank()`:

```
rank(A)  
Out: 3
```

Uwaga. Rząd macierzy spełnia własność:

$$r(\mathbf{A}) = r(\mathbf{A}^T).$$

Oznacza to, że operacje elementarne można wykonywać nie tylko na wierszach, ale również na kolumnach macierzy. Dla macierzy układu nie stosujemy jednak operacji elementarnych na kolumnach, ponieważ prowadziłyby to do zmiany kolejności zmiennych w układzie.

Układ równań liniowych może mieć tylko trzy możliwe liczby rozwiązań: 0, 1 albo nieskończenie wiele. Mówimy, że układ jest:

- sprzeczny, gdy nie ma rozwiązań;
- oznaczony, gdy ma dokładnie jedno rozwiązanie;
- nieoznaczony, gdy ma nieskończenie wiele rozwiązań.

Przykład 3.8

Ustalimy liczbę rozwiązań układu równań:

$$\begin{cases} x + 3y - 2z = 1 \\ 2x \quad \quad - z = 2 \\ -x + y \quad \quad = 3. \end{cases}$$

Zapisujemy macierz uzupełnioną układu:

```
U = matrix( [[1, 3, -2, 1], [2, 0, -1, 2], [-1, 1, 0, 3]] )
```

```
U
```

```
Out: [ 1  3 -2  1]
```

```
      [ 2  0 -1  2]
```

```
      [-1  1  0  3]
```

Sprowadzamy ją do postaci schodkowej:

```
U.echelon_form()
```

```
Out: [ 1  1 -1  5]
```

```
      [ 0  2 -1  8]
```

```
      [ 0  0  0 12]
```

Po zamianie ostatniego wiersza na równanie otrzymujemy równanie sprzeczne:

$$0 = 12,$$

a stąd wynika, że cały układ jest sprzeczny, czyli nie ma rozwiązań.

Zauważmy, że macierz uzyskana po usunięciu ostatniej kolumny ma dwa schodki. Zatem rząd macierzy układu $\mathbf{A}$ jest równy:

$$r(\mathbf{A}) = 2,$$

natomiast macierz uzupełniona $\mathbf{U}$ ma trzy schodki, stąd:

$$r(\mathbf{U}) = 3.$$

Wniosek. Jeżeli zachodzi nierówność $r(\mathbf{A}) < r(\mathbf{U})$, ostatnie równanie jest sprzeczne, stąd cały układ jest sprzeczny.

Widzimy zatem, że liczbę rozwiązań układu można ustalić na podstawie porównania rzędów macierzy układu $\mathbf{A}$ oraz macierzy uzupełnionej $\mathbf{U}$. Okazuje się, że zależność tę dokładnie określa poniższe twierdzenie:

Twierdzenie (Kroneckera-Capellego). Jeżeli $r(\mathbf{A}) = r(\mathbf{U})$, to układ $\mathbf{Ax} = \mathbf{b}$ ma rozwiązanie.

Załóżmy teraz, że rozwiązanie układu istnieje. Oznaczmy wspólną wartość rzędu macierzy układu $\mathbf{A}$ i macierzy uzupełnionej $\mathbf{U}$ przez r , czyli:

$$r := r(\mathbf{A}) = r(\mathbf{U}).$$

Ponadto niech n oznacza liczbę zmiennych w układzie (jest to również liczba kolumn macierzy układu $\mathbf{A}$). Wówczas, jeżeli zachodzi warunek:

- $r = n$, to układ ma dokładnie jedno rozwiązanie;
- $r < n$, to układ ma nieskończenie wiele rozwiązań zależnych od $n - r$ parametrów.

Przypadek $r > n$ nie może natomiast wystąpić, ponieważ macierz układu $\mathbf{A}$, która jest częścią macierzy $\mathbf{U}$, nie może mieć większego rzędu niż cała macierz uzupełniona.

Przykład 3.9

Rozwiążemy układ równań:

$$\begin{cases} x - y + z + 2t = -1 \\ x \quad \quad + z + 3t = 0 \\ -x + 2y - z - t = 2 \\ -x + 3y - z \quad = 3. \end{cases}$$

Pierwszy etap metody eliminacji Gaussa wykonamy w systemie **SageMath**. Najpierw zapisujemy macierz uzupełnioną układu:

```
U = matrix( [[1, -1, 1, 2, -1], [1, 0, 1, 3, 0],  
             [-1, 2, -1, -1, 2], [-1, 3, -1, 0, 3]] )  
U  
Out: [ 1 -1 1 2 -1]  
      [ 1 0 1 3 0]  
      [-1 2 -1 -1 2]  
      [-1 3 -1 0 3]
```

Następnie sprowadzamy ją do postaci schodkowej:

```
U.echelon_form()  
Out: [1 0 1 3 0]  
      [0 1 0 1 1]  
      [0 0 0 0 0]  
      [0 0 0 0 0]
```

Uzyskana macierz ma dwa niezerowe wiersze, zatem zachodzi równość $r(\mathbf{A}) = r(\mathbf{U}) = 2$. Wobec tego:

$$r = 2.$$

Układ ma cztery zmienne: x , y , z i t . Stąd:

$$n = 4.$$

Otrzymujemy zależność $r < n$, czyli układ ma nieskończenie wiele rozwiązań zależnych od $\underbrace{n-r}_{4-2} = 2$ parametrów.

Gdy rozwiązujemy taki układ równań ręcznie, warto stosować następującą zasadę: parametry wybieramy w tych kolumnach, gdzie nie ma schodków. W macierzy uzyskanej za pomocą funkcji **echelon_form()** brakuje schodków w kolumnach 3 i 4, dlatego jako parametry przyjmujemy zmienne z i t . Następnie zapisujemy kolejne wiersze macierzy w postaci równań, idąc „od dołu do góry”. Z drugiego wiersza otrzymujemy:

$$y + t = 1 \rightarrow y = 1 - t,$$

natomiast z pierwszego wiersza mamy zależność:

$$x + z + 3t = 0 \rightarrow x = -z - 3t.$$

Wyznaczyliśmy pozostałe zmienne x i y za pomocą parametrów $z, t \in \mathbb{R}$, więc możemy zapisać końcowe rozwiązanie układu w postaci:

$$\begin{cases} x = -z - 3t \\ y = 1 - t \\ z = z \\ t = t \end{cases}, \text{ gdzie } z, t \in \mathbb{R}.$$

W **Sage** układ ten można rozwiązać za pomocą funkcji **solve()**:

```
var('x y z t')
solve([x-y+z+2*t==-1, x+z+3*t==0,
      -x+2*y-z-t==2, -x+3*y-z==3], [x, y, z, t])
Out: [[x == r1, y == 1/3*r1 + 1/3*r2 + 1,
      z == r2, t == -1/3*r1 - 1/3*r2]]
```

Przykład 3.10

Korzystając z twierdzenia Kroneckera-Capellego, określmy liczbę rozwiązań układu:

$$\begin{cases} 3x - 2y + z - 2t + 3s = 0 \\ -x + 3y - 2z + 3t - s = 1 \\ 2x - y + 3z - t + 2s = 2 \\ x + 2y + z + 2t + s = 3. \end{cases}$$

Definiujemy w **Sage** macierz układu **A** i łączymy ją z wektorem wyrazów wolnych **b**, aby uzyskać macierz uzupełnioną **U**. Macierz **A** ma wymiary 4×5 , ponieważ układ składa się z 4 równań i 5 niewiadomych:

```
A = matrix( [[3, -2, 1, -2, 3],
             [-1, 3, -2, 3, -1],
             [2, -1, 3, -1, 2],
             [1, 2, 1, 2, 1]] )
b = vector( [0, 1, 2, 3] )
U = A.augment(b, subdivide=True)
```

```
print('Macierz układu:')
show( LatexExpr(r'\textbf{A}='), A )
print('\nMacierz uzupełniona:')
show( LatexExpr(r'\textbf{U}='), U )
```

Out: Macierz układu:

$$\mathbf{A} = \begin{pmatrix} 3 & -2 & 1 & -2 & 3 \\ -1 & 3 & -2 & 3 & -1 \\ 2 & -1 & 3 & -1 & 2 \\ 1 & 2 & 1 & 2 & 1 \end{pmatrix}$$

Macierz uzupełniona:

$$\mathbf{U} = \left(\begin{array}{ccccc|c} 3 & -2 & 1 & -2 & 3 & 0 \\ -1 & 3 & -2 & 3 & -1 & 1 \\ 2 & -1 & 3 & -1 & 2 & 2 \\ 1 & 2 & 1 & 2 & 1 & 3 \end{array} \right)$$

Obliczamy rzędy macierzy **A** oraz **U**:

```
print( 'Rząd macierzy A:', rank(A) )
print( 'Rząd macierzy U:', rank(U) )
```

Out: Rząd macierzy A: 3

Rząd macierzy U: 3

Ponieważ rzędy są równe, na podstawie twierdzenia Kroneckera-Capellego układ ma rozwiązanie. Wspólną wartość rzędu tych macierzy oznaczamy przez r , a liczbę zmiennych w układzie przez n :

```
r = rank(A)
n = A.ncols()
r, n
Out: (3, 5)
```

Tutaj spełniony jest warunek $r < n$, więc układ ma nieskończenie wiele rozwiązań. Poniższy kod zwraca ten wynik, wykorzystując instrukcję warunkową:

```
if r == n:
    print('Układ ma 1 rozwiązanie')
else:
    print('Układ ma nieskończenie wiele rozwiązań',
          f'zależnych od {n-r} parametrów')
Out: Układ ma nieskończenie wiele rozwiązań zależnych od 2
parametrów
```

3.3. Wyznacznik macierzy

Wyznacznik to liczba przypisana do macierzy kwadratowej, która pozwala analizować jej podstawowe własności. Intuicyjnie można go traktować jako wskaźnik określający, czy dana macierz $\mathbf{A}$ jest „zwyczajna”, czy też prowadzi do sytuacji szczególnych:

- jeśli wyznacznik jest różny od zera, macierz $\mathbf{A}$ nazywamy nieosobliwą – oznacza to, że jest ona odwracalna, czyli istnieje taka macierz $\mathbf{A}^{-1}$, która pozwala „cofnąć” działanie macierzy $\mathbf{A}$;
- jeśli wyznacznik jest równy zeru, mówimy, że macierz $\mathbf{A}$ jest osobliwa – w tym przypadku macierz odwrotna nie istnieje, a operacja reprezentowana przez macierz $\mathbf{A}$ jest nieodwracalna.

Dla każdej macierzy o wyznaczniku różnym od zera istnieje tzw. macierz odwrotna, która pełni funkcję analogiczną do elementu odwrotnego przy mnożeniu

liczb. Co więcej, zbiór wszystkich macierzy nieosobliwych z działaniem mnożenia tworzy grupę, w której elementem neutralnym jest macierz jednostkowa $\mathbf{I}$.

Mimo że wyznacznik definiowany jest wyłącznie dla macierzy kwadratowych, stanowi on niezwykle przydatne narzędzie. Pozwala m.in. wyznaczać rząd macierzy, a co za tym idzie – analizować liczbę rozwiązań układów równań liniowych. Odgrywa także istotną rolę w geometrii analitycznej, ponieważ pozwala obliczać pola figur i objętości brył oraz badać wzajemne położenie wektorów.

W niniejszym rozdziale poznamy metody obliczania wyznacznika, omówimy jego podstawowe własności oraz przedstawimy wybrane zastosowania – w szczególności związek z macierzą odwrotną i rozwiązywaniem układów równań liniowych. Warto dodać, że pojęcie wyznacznika będzie się jeszcze wielokrotnie pojawiać w bardziej zaawansowanych zagadnieniach – nie tylko w algebrze, lecz także w analizie i innych działach matematyki.

3.3.1. Obliczanie wyznacznika

Wyznacznik jest pewną liczbą związaną z macierzą kwadratową (zob. definicję na s. 104). Dla macierzy

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

oznaczamy go symbolem:

$$\begin{vmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{vmatrix} \quad \text{lub} \quad \det \mathbf{A}.$$

W najprostszym przypadku, dla macierzy stopnia pierwszego $\mathbf{A} = [a_{11}]$, wyznacznik jest równy jej jednemu elementowi, czyli:

$$\det \mathbf{A} = a_{11}.$$

Dla macierzy 2×2 postaci

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

wyznacznik obliczamy według wzoru:

$$\det \mathbf{A} = ad - bc. \quad (3.1)$$

Przykład 3.11

Znajdziemy wyznacznik macierzy 2×2 :

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}.$$

Korzystając ze wzoru (3.1), otrzymujemy:

$$\det \mathbf{A} = 1 \cdot 4 - 2 \cdot 3 = -2.$$

W Sage wyznacznik liczymy za pomocą funkcji **det()**:

```
A = matrix( range(1, 5), nrows=2 )
det(A)
Out: -2
```

Dla macierzy 3×3 postaci

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

wyznacznik można obliczyć, korzystając z tzw. reguły Sarrusa. Polega ona na tym, że:

- najpierw dodajemy iloczyny elementów leżących wzdłuż przekątnych z lewej górnej strony macierzy (tak, jak zaczynamy czytać):

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{21}a_{32}a_{13} + \dots;$$

- następnie odejmujemy iloczyny elementów leżących na przekątnych biegnących z prawej górnej strony macierzy:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{21}a_{32}a_{13} - (a_{13}a_{22}a_{31} + a_{12}a_{21}a_{33} + a_{23}a_{32}a_{11}). \quad (3.2)$$

Przykład 3.12

Obliczymy wyznacznik macierzy 3×3 :

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}.$$

Stosujemy regułę Sarrusa (3.2) dla macierzy $\mathbf{A}$:

$$\det \mathbf{A} = (1 \cdot 5 \cdot 9 + 2 \cdot 6 \cdot 7 + 4 \cdot 8 \cdot 3) - (3 \cdot 5 \cdot 7 + 2 \cdot 4 \cdot 9 + 6 \cdot 8 \cdot 1) = 0.$$

W **Sage** funkcja **det()** pozwala znaleźć wyznacznik macierzy kwadratowej dowolnego rozmiaru:

```
A = matrix( range(1, 10), nrows=3 )
det(A)
Out: 0
```

Definicja. Minorem stopnia k macierzy $\mathbf{A}$ nazywamy wyznacznik macierzy kwadratowej stopnia k (czyli macierzy o wymiarach $k \times k$), powstałej przez wykreślenie z macierzy $\mathbf{A}$ odpowiedniej liczby wierszy i kolumn.

Przykład 3.13

Znajdziemy przykładowy minor stopnia 2 macierzy o wymiarach 3×4 :

$$\mathbf{A} = \begin{bmatrix} -4 & -3 & -2 & -1 \\ 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \end{bmatrix}.$$

Chcemy uzyskać tzw. podmacierz 2×2 macierzy $\mathbf{A}$, czyli wybrany fragment tej macierzy. W tym celu usuwamy jeden wiersz i dwie kolumny macierzy $\mathbf{A}$. Przykładowo możemy wykreślić drugi wiersz oraz pierwszą i trzecią kolumnę, co prowadzi do szablonu:

$$\mathbf{A} = \begin{bmatrix} \bullet & -3 & \bullet & -1 \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & 5 & \bullet & 7 \end{bmatrix}.$$

Wobec tego jednym z szukanych minorów jest wyznacznik:

$$\begin{vmatrix} -3 & -1 \\ 5 & 7 \end{vmatrix} = -3 \cdot 7 - (-1) \cdot 5 = -16.$$

Przy okazji możemy zauważyć, że jeden wiersz można wykreślić na $\binom{3}{1}$ sposobów, natomiast dwie kolumny – na $\binom{4}{2}$ sposobów. Zatem każda macierz o wymiarach 3×4 ma

$$\binom{3}{1} \cdot \binom{4}{2} = 3 \cdot 6 = 18$$

różnych minorów stopnia 2.

W **Sage** podmacierz kwadratową możemy otrzymać za pomocą segmentacji:

```
A = matrix( range(-4, 8), nrows=3 )
A[[0, 2], [1, 3]]
Out: [-3 -1]
     [ 5  7]
```

W języku minorów zazwyczaj mówimy jednak o wykreślaniu wierszy i kolumn, dlatego wygodniej jest używać funkcji **delete_rows()** oraz **delete_columns()**. Funkcje te nie są inwazyjne, czyli nie modyfikują oryginalnej macierzy, a jedynie zwracają wynik. Dlatego najpierw przypisujemy go do zmiennej, a następnie wyznaczamy interesujący nas minor:

```
podmacierz = A.delete_rows( [1] ).delete_columns( [0, 2] )
minor = det(podmacierz)
minor
Out: -16
```

Aby obliczyć wyznacznik stopnia wyższego niż 3, należy najpierw zmniejszyć jego stopień. W tym celu stosuje się tzw. rozwinięcie Laplace’a. Polega ono na rozwinięciu wyznacznika względem dowolnego wiersza lub kolumny. Wyznacznik zapisujemy wówczas jako sumę składników postaci

$$(-1)^{i+j} \cdot a_{ij} \cdot \mathbf{M}_{ij},$$

gdzie:

- a_{ij} jest elementem macierzy w i -tym wierszu oraz j -tej kolumnie;
- M_{ij} oznacza minor elementu a_{ij} , czyli wyznacznik macierzy powstałej po wykreśleniu i -tego wiersza oraz j -tej kolumny.

Przykład 3.14

Obliczymy wyznacznik stopnia 4:

$$\begin{vmatrix} 1 & -1 & 2 & 3 \\ -2 & 0 & 1 & 4 \\ -4 & 3 & -1 & 2 \\ 3 & 2 & 1 & 0 \end{vmatrix}.$$

Stosujemy rozwinięcie Laplace'a względem czwartego wiersza. Otrzymujemy wówczas sumę:

$$\begin{aligned} \begin{vmatrix} 1 & -1 & 2 & 3 \\ -2 & 0 & 1 & 4 \\ -4 & 3 & -1 & 2 \\ \color{red}{3} & \color{green}{2} & \color{blue}{1} & \color{violet}{0} \end{vmatrix} &= (-1)^{4+1} \cdot \color{red}{3} \cdot \begin{vmatrix} -1 & 2 & 3 \\ 0 & 1 & 4 \\ 3 & -1 & 2 \end{vmatrix} \\ &+ (-1)^{4+2} \cdot \color{green}{2} \cdot \begin{vmatrix} 1 & 2 & 3 \\ -2 & 1 & 4 \\ -4 & -1 & 2 \end{vmatrix} \\ &+ (-1)^{4+3} \cdot \color{blue}{1} \cdot \begin{vmatrix} 1 & -1 & 3 \\ -2 & 0 & 4 \\ -4 & 3 & 2 \end{vmatrix} \\ &+ (-1)^{4+4} \cdot \color{violet}{0} \cdot \begin{vmatrix} 1 & -1 & 2 \\ -2 & 0 & 1 \\ -4 & 3 & -1 \end{vmatrix}. \end{aligned}$$

Następnie liczymy wyznaczniki stopnia 3, stosując regułę Sarrusa:

$$\begin{vmatrix} -1 & 2 & 3 \\ 0 & 1 & 4 \\ 3 & -1 & 2 \end{vmatrix} = 9, \quad \begin{vmatrix} 1 & 2 & 3 \\ -2 & 1 & 4 \\ -4 & -1 & 2 \end{vmatrix} = 0, \quad \begin{vmatrix} 1 & -1 & 3 \\ -2 & 0 & 4 \\ -4 & 3 & 2 \end{vmatrix} = -18.$$

Czwartego wyznacznika nie musimy już obliczać, ponieważ ostatni składnik jest równy 0. Po podstawieniu otrzymanych wartości do rozwinięcia Laplace’a otrzymujemy:

$$\begin{vmatrix} 1 & -1 & 2 & 3 \\ -2 & 0 & 1 & 4 \\ -4 & 3 & -1 & -2 \\ 3 & 2 & 1 & 0 \end{vmatrix} = (-1) \cdot 3 \cdot 9 + 1 \cdot 2 \cdot 0 + (-1) \cdot 1 \cdot (-18) = -9.$$

Wynik możemy sprawdzić w **Sage**, korzystając z funkcji **det()**:

```
A = matrix( [[1, -1, 2, 3], [-2, 0, 1, 4],
              [-4, 3, -1, 2], [3, 2, 1, 0]] )
det(A)
Out: -9
```

Warto zwrócić uwagę, że gdyby w wybranym wierszu (lub kolumnie) występowało więcej zer, rachunki byłyby znacznie prostsze. W dalszej części rozdziału pokażemy, jak można doprowadzić do takiej sytuacji, korzystając z własności wyznacznika.

3.3.2. Własności wyznacznika

Na początku przeanalizujemy wpływ poszczególnych operacji elementarnych na wartość wyznacznika. Przedstawimy to na przykładzie macierzy:

$$A = \begin{bmatrix} 1 & -1 & 1 \\ 3 & 1 & -4 \\ -2 & 0 & 2 \end{bmatrix}.$$

Zaczynamy od obliczenia wyznacznika w **Sage**:

```
A = matrix( [[1, -1, 1], [3, 1, -4], [-2, 0, 2]] )
det(A)
Out: 2
```

W przypadku wyznaczników operacje elementarne można wykonywać zarówno na wierszach, jak i na kolumnach. Obowiązują wtedy następujące reguły:

1. Zamiana dwóch wierszy (lub kolumn).

Zamiana dowolnych dwóch wierszy (lub kolumn) zmienia znak wyznacznika na przeciwny. Na przykład po zamianie drugiego i trzeciego wiersza macierzy **A** otrzymujemy:

$$\det \mathbf{A} = \begin{vmatrix} 1 & -1 & 1 \\ 3 & 1 & -4 \\ -2 & 0 & 2 \end{vmatrix} \xrightarrow{w_2 \leftrightarrow w_3} \begin{vmatrix} 1 & -1 & 1 \\ -2 & 0 & 2 \\ 3 & 1 & -4 \end{vmatrix} = -\det \mathbf{A}.$$

W **Sage** możemy to sprawdzić, stosując segmentację, aby nie modyfikować oryginalnej macierzy:

```
det( A[[0, 2, 1]] )
Out: -2
```

2. Mnożenie wiersza (lub kolumny) przez liczbę różną od zera.

Mnożenie wiersza (lub kolumny) przez liczbę $k \neq 0$ zwiększa wartość wyznacznika k razy. Przykładowo pomnożymy pierwszy wiersz macierzy **A** przez 2:

$$\det \mathbf{A} = \begin{vmatrix} 1 & -1 & 1 \\ 3 & 1 & -4 \\ -2 & 0 & 2 \end{vmatrix} \xrightarrow{w_1 \cdot 2} \begin{vmatrix} 2 & -2 & 2 \\ 3 & 1 & -4 \\ -2 & 0 & 2 \end{vmatrix} = 2 \cdot \det \mathbf{A}.$$

W **Sage** stosujemy funkcję **copy()**, aby zachować pierwotną macierz:

```
B = copy(A)
B[0] *= 2
det(B)
Out: 4
```

Wniosek. Jeżeli wiersz (lub kolumna) macierzy składa się z samych zer, to jej wyznacznik jest równy 0.

Ponadto, jeżeli pomnożymy całą macierz przez 2 (czyli każdy z trzech wierszy), wartość wyznacznika będzie $2^3 = 8$ razy większa:

$$\det(2\mathbf{A}) = \begin{vmatrix} 2 \cdot 1 & 2 \cdot (-1) & 2 \cdot 1 \\ 2 \cdot 3 & 2 \cdot 1 & 2 \cdot (-4) \\ 2 \cdot (-2) & 2 \cdot 0 & 2 \cdot 2 \end{vmatrix} = 2 \cdot 2 \cdot 2 \cdot \det \mathbf{A} = 8 \cdot \det \mathbf{A}.$$

Sprawdzamy tę obserwację w **Sage**:

```
det(2*A)
```

```
Out: 16
```

Wniosek. Dla dowolnej macierzy kwadratowej $\mathbf{A} \in \mathbf{M}_{n \times n}(\mathbb{R})$ i stałej $\alpha \in \mathbb{R}$ zachodzi wzór:

$$\det(\alpha \cdot \mathbf{A}) = \alpha^n \cdot \det \mathbf{A}. \quad (3.3)$$

3. Dodanie do wiersza (lub kolumny) innego wiersza pomnożonego przez liczbę.

Jest to jedyna operacja elementarna, która nie zmienia wartości wyznacznika. W tej sytuacji, w przeciwieństwie do macierzy, możemy stosować znak równości między wyznacznikami:

$$\det \mathbf{A} = \begin{vmatrix} 1 & -1 & 1 \\ 3 & 1 & -4 \\ -2 & 0 & 2 \end{vmatrix} \stackrel{w_2 - 3w_1}{=} \begin{vmatrix} 1 & -1 & 1 \\ 0 & 4 & -7 \\ -2 & 0 & 2 \end{vmatrix} = \det \mathbf{A}.$$

Sprawdzamy w **Sage**, że operacja ta rzeczywiście zwraca początkową wartość wyznacznika macierzy $\mathbf{A}$:

```
A[1] -= 3*A[0]
```

```
det(A)
```

```
Out: 2
```

Przykład 3.15

Obliczymy wyznacznik z przykładu 3.14, wykorzystując operacje elementarne.

Chcemy uzyskać sytuację, w której w danym wierszu (lub kolumnie) wszystkie elementy poza jednym są równe zero. W tym celu stosujemy trzecią operację

elementarną na kolumnach. Wykorzystujemy jedynkę w ostatnim wierszu, aby wyzerować jego pozostałe elementy:

$$\left| \begin{array}{cccc} 1 & -1 & 2 & 3 \\ -2 & 0 & 1 & 4 \\ -4 & 3 & -1 & 2 \\ 3 & 2 & 1 & 0 \end{array} \right| \begin{array}{l} k_1 - 3k_3 \\ \underline{k_2 - 2k_3} \\ \end{array} \left| \begin{array}{cccc} -5 & -5 & 2 & 3 \\ -5 & -2 & 1 & 4 \\ -1 & 5 & -1 & 2 \\ 0 & 0 & 1 & 0 \end{array} \right|.$$

Następnie stosujemy rozwinięcie Laplace'a względem czwartego wiersza. Tym razem otrzymujemy tylko jeden niezerowy składnik rozwinięcia:

$$\dots = \left| \begin{array}{cccc} -5 & -5 & 2 & 3 \\ -5 & -2 & 1 & 4 \\ -1 & 5 & -1 & 2 \\ 0 & 0 & 1 & 0 \end{array} \right| = 0 + 0 + (-1)^{4+3} \cdot 1 \cdot \left| \begin{array}{ccc} -5 & -5 & 3 \\ -5 & -2 & 4 \\ -1 & 5 & 2 \end{array} \right| + 0.$$

Uzyskany wyznacznik stopnia 3 możemy już obliczyć za pomocą reguły Sarrusa (3.2):

$$\dots = (-1) \cdot [-5 \cdot (-2) \cdot 2 + (-5) \cdot 4 \cdot (-1) + (-5) \cdot 5 \cdot 3 - (3 \cdot (-2) \cdot (-1) + (-5) \cdot (-5) \cdot 2 + 4 \cdot 5 \cdot (-5))] = -9.$$

Sprawdzamy rachunki w **Sage** – najpierw wykonujemy operacje elementarne na kolumnach macierzy **A**:

```
A = matrix( [[1, -1, 2, 3], [-2, 0, 1, 4],
              [-4, 3, -1, 2], [3, 2, 1, 0]] )
A[:, 0] -= 3*A[:, 2]
A[:, 1] -= 2*A[:, 2]
A
Out: [-5 -5 2 3]
      [-5 -2 1 4]
      [-1 5 -1 2]
      [ 0 0 1 0]
```

Następnie obliczamy wyznacznik zmodyfikowanej macierzy:

```
det(A)
Out: -9
```

Podamy teraz równoważną definicję rzędu macierzy, która wykorzystuje pojęcie wyznacznika:

Definicja. Rzędem macierzy **A** nazywamy stopień największego niezerowego minora tej macierzy.

Przykład 3.16

Wyznamy rząd macierzy

$$\mathbf{A} = \begin{bmatrix} 1 & -3 & -2 & -1 \\ -2 & 2 & -1 & 3 \\ 3 & -5 & -1 & 0 \\ 2 & -2 & 1 & -7 \end{bmatrix}$$

poprzez znalezienie jej największego niezerowego minora.

Największy możliwy stopień minora to 4 – jest nim wyznacznik całej macierzy, czyli $\det \mathbf{A}$ (nie wykreślamy żadnych wierszy ani kolumn). Obliczamy go w **Sage**:

```
A = matrix( [[1, -3, -2, -1], [-2, 2, -1, 3],
              [3, -5, -1, 0], [2, -2, 1, -7]] )
det(A)
Out: 0
```

Otrzymany minor jest równy 0 i jest to jedyny minor stopnia 4. Wnioskujemy stąd, że rząd macierzy jest mniejszy niż 4. Następnie sprawdzamy minor stopnia 3 w lewym górnym rogu, czyli tzw. wiodący minor główny macierzy **A** (usuwamy ostatni wiersz i ostatnią kolumnę):

$$\mathbf{M}_1 = \begin{vmatrix} 1 & -3 & -2 \\ -2 & 2 & -1 \\ 3 & -5 & -1 \end{vmatrix}.$$

W **Sage** wykorzystujemy segmentację macierzy **A**:

```
M1 = A[:3, :3]
det(M1)
Out: 0
```

Ten minor również jest równy 0. Nie oznacza to jednak, że rząd macierzy **A** jest mniejszy niż 3, ponieważ nie sprawdziliśmy jeszcze wszystkich minorów stopnia 3, a jest ich:

$$\binom{4}{1} \cdot \binom{4}{1} = 4 \cdot 4 = 16.$$

Wobec tego bierzemy kolejny minor stopnia 3 – tym razem w prawym dolnym rogu (wykreślamy pierwszy wiersz i pierwszą kolumnę):

$$\mathbf{M}_2 = \begin{vmatrix} 2 & -1 & 3 \\ -5 & -1 & 0 \\ -2 & 1 & -7 \end{vmatrix}.$$

W **Sage** ponownie stosujemy segmentację macierzy **A**:

```
M2 = A[1:, 1:]
det(M2)
Out: 28
```

Otrzymaliśmy niezerowy minor stopnia 3. Wcześniej sprawdziliśmy, że nie ma niezerowych minorów wyższego stopnia. Zatem rząd macierzy **A** jest równy 3.

Wniosek. Rząd macierzy $\mathbf{A} \in \mathbf{M}_{m \times n}(\mathbb{R})$ nie może być większy niż mniejszy z wymiarów tej macierzy, czyli:

$$r(\mathbf{A}) \leq \min\{m, n\}.$$

Przykład 3.17

Określimy maksymalny rząd macierzy **A** o wymiarach 7×4 .

Na podstawie wniosku maksymalny możliwy rząd macierzy o wymiarach 7×4 to:

$$\min\{7, 4\} = 4.$$

Stąd mamy pewność, że zachodzi ograniczenie:

$$r(\mathbf{A}) \leq 4.$$

Przetestujemy tę obserwację w **Sage** poprzez wylosowanie macierzy o wymiarach 7×4 za pomocą funkcji **random_matrix()**. Jako pierwszy argument przekazujemy pierścień, do którego będą należeć elementy – w tym przypadku jest to obiekt QQ reprezentujący ciało liczb wymiernych:

```
random_matrix(QQ, 7, 4)
Out: [ 1  -1  -1  0]
      [-2  -1  1  1]
      [ 1   1  2 -1]
      [-1 -1/2  1  0]
      [-1 -1/2 -2  0]
      [1/2  -1  0 -1/2]
      [-1   0 -2  1]
```

Ponieważ macierz generowana jest losowo, za każdym razem otrzymamy inne liczby, jednak niezależnie od wylosowanych elementów rząd macierzy nigdy nie przekroczy 4. Sprawdzamy to doświadczalnie, korzystając z funkcji **rank()**:

```
rank(A)
Out: 4
```

Wniosek. Z ostatniej definicji wynika ciekawa zależność dla macierzy kwadratowych. Niech $\mathbf{A} \in \mathbf{M}_{n \times n}(\mathbb{R})$. Wówczas, jeżeli:

- $\det \mathbf{A} \neq 0$, to rząd macierzy $\mathbf{A}$ jest maksymalny, czyli $r(\mathbf{A}) = n$;
- $\det \mathbf{A} = 0$, to rząd macierzy $\mathbf{A}$ jest mniejszy od jej stopnia, tzn. $r(\mathbf{A}) < n$.

Przykład 3.18

Zbadamy rząd poniższej macierzy, obliczając jej wyznacznik:

$$\mathbf{A} = \begin{bmatrix} 2 & 0 & -1 \\ 1 & 3 & -2 \\ -1 & -1 & 2 \end{bmatrix}.$$

Obliczamy wyznacznik macierzy **A**:

```
A = matrix( [[2, 0, -1], [1, 3, -2], [-1, -1, 2]] )  
det(A)  
Out: 6
```

Wyznacznik tej macierzy jest różny od zera, zatem na podstawie wcześniejszego wniosku rząd macierzy **A** jest maksymalny, czyli równy 3. Sprawdzamy to w **Sage**:

```
rank(A)  
Out: 3
```

Przykład 3.19

Wyznamy rząd macierzy w zależności od wartości parametru $a \in \mathbb{R}$:

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & 2 \\ 2 & a & a \\ 3 & 1 & 2 \end{bmatrix}.$$

Rząd macierzy 3×3 określamy na podstawie zależności:

- $r(\mathbf{A}) = 3$ tylko wtedy, gdy $\det \mathbf{A} \neq 0$;
- $r(\mathbf{A}) < 3$ tylko wtedy, gdy $\det \mathbf{A} = 0$.

Zaczynamy zatem od obliczenia wyznacznika macierzy **A**, korzystając z reguły Sarrusa:

$$\det \mathbf{A} = (2a + 9a + 4) - (6a + 12 + a) = 4a - 8.$$

Stąd rząd macierzy **A** jest równy 3 tylko wtedy, gdy:

$$\det \mathbf{A} \neq 0 \quad \rightarrow \quad 4a - 8 \neq 0 \quad \rightarrow \quad a \neq 2.$$

W **Sage** możemy to sprawdzić za pomocą poleceń:

```
var('a')  
A = matrix( [[1, 3, 2], [2, a, a], [3, 1, 2]] )  
dA = det(A)
```

```
solve(dA, a)
```

```
Out: [a == 2]
```

Dla $a = 2$ wiemy obecnie natomiast jedynie, że $\det \mathbf{A} = 0$, a więc rząd macierzy $\mathbf{A}$ jest mniejszy niż 3. Aby ustalić jego dokładną wartość, podstawiamy $a = 2$ do macierzy i sprowadzamy ją do postaci schodkowej:

```
A(a=2).echelon_form()
```

```
Out: [ 1  0 1/2]
```

```
      [ 0  1 1/2]
```

```
      [ 0  0  0]
```

Macierz ma dwa niezerowe wiersze, stąd:

$$r(\mathbf{A}) = 2, \text{ dla } a = 2.$$

Widzimy zatem, że macierz kwadratowa $\mathbf{A}$, której wyznacznik jest równy 0, posiada inne własności niż macierze o wyznaczniku niezerowym. Okazuje się, że to rozróżnienie jest istotne w wielu zastosowaniach, dlatego stosujemy następujące nazewnictwo:

- jeżeli $\det \mathbf{A} \neq 0$, to mówimy, że macierz $\mathbf{A}$ jest odwracalna lub nieosobliwa;
- jeżeli $\det \mathbf{A} = 0$, to macierz $\mathbf{A}$ nazywamy nieodwracalną lub osobliwą.

Pierwsza nazwa wiąże się bezpośrednio z istnieniem tzw. macierzy odwrotnej.

Definicja. Macierzą odwrotną do macierzy $\mathbf{A} \in \mathbf{M}_{n \times n}(\mathbb{R})$, oznaczaną symbolem $\mathbf{A}^{-1}$, nazywamy macierz spełniającą warunek:

$$\mathbf{A} \cdot \mathbf{A}^{-1} = \mathbf{A}^{-1} \cdot \mathbf{A} = \mathbf{I},$$

gdzie $\mathbf{I}$ jest macierzą jednostkową stopnia n . Macierz odwrotna $\mathbf{A}^{-1}$ istnieje tylko wtedy, gdy macierz $\mathbf{A}$ jest kwadratowa i odwracalna, czyli gdy jej wyznacznik jest różny od zera.

Przykład 3.20

Zbadamy, czy istnieje macierz odwrotna do macierzy:

$$\mathbf{A} = \begin{bmatrix} 6 & 3 \\ -4 & -2 \end{bmatrix}.$$

Na podstawie wzoru (3.1) wyznacznik macierzy $\mathbf{A}$ jest równy:

$$\det \mathbf{A} = 6 \cdot (-2) - 3 \cdot (-4) = 0,$$

zatem macierz odwrotna do $\mathbf{A}$ nie istnieje.

Aby znaleźć w **Sage** macierz odwrotną, stosujemy zapis $\mathbf{A}^{-1}$ lub $\sim \mathbf{A}$. Gdy próbujemy wykonać tę operację dla macierzy $\mathbf{A}$, otrzymujemy komunikat o błędzie, informujący, że macierz musi być nieosobliwa:

```
A = matrix( [[6, 3], [-4, -2]] )
A^-1
Out: ZeroDivisionError: matrix must be nonsingular
```

Dla macierzy 2×2 postaci:

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

macierz odwrotną $\mathbf{A}^{-1}$ możemy obliczyć ze wzoru:

$$\mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}} \cdot \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}. \quad (3.4)$$

Przykład 3.21

Znajdziemy macierz odwrotną do macierzy 2×2 :

$$\mathbf{A} = \begin{bmatrix} 4 & 6 \\ 1 & 2 \end{bmatrix}.$$

Najpierw sprawdzamy, czy wyznacznik macierzy $\mathbf{A}$ jest różny od zera, korzystając ze wzoru (3.1):

$$\det \mathbf{A} = 4 \cdot 2 - 6 \cdot 1 = 2.$$

Następnie stosujemy wzór (3.4), aby obliczyć macierz odwrotną:

$$\mathbf{A}^{-1} = \frac{1}{2} \cdot \begin{bmatrix} 2 & -6 \\ -1 & 4 \end{bmatrix} = \begin{bmatrix} 1 & -3 \\ -1/2 & 2 \end{bmatrix}.$$

Sprawdzamy rachunki w **Sage**:

```
A = matrix( [[4, 6], [1, 2]] )
A^-1
Out: [ 1 -3]
      [-1/2 2]
```

Zobaczmy teraz, w jaki sposób można znaleźć macierz odwrotną do macierzy stopnia większego niż 2. W tym celu wprowadzamy pojęcie dopełnienia algebraicznego.

Definicja. Dopełnieniem algebraicznym elementu a_{ij} macierzy

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

nazywamy liczbę:

$$d_{ij} = (-1)^{i+j} \cdot \mathbf{M}_{ij}, \quad (3.5)$$

gdzie $\mathbf{M}_{ij}$ oznacza minor elementu a_{ij} , czyli wyznacznik macierzy powstałej po wykreśleniu i -tego wiersza oraz j -tej kolumny.

Warto zauważyć, że podobne wyrażenie pojawiło się już przy rozwinięciu Laplace'a. Na podstawie dopełnień algebraicznych tworzymy tzw. macierz dopełnień:

$$\mathbf{D} = \begin{bmatrix} d_{11} & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \cdots & d_{nn} \end{bmatrix}.$$

Wówczas macierz odwrotna do macierzy $\mathbf{A}$ dana jest wzorem:

$$\mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}} \cdot \mathbf{D}^T. \quad (3.6)$$

Przykład 3.22

Znajdziemy macierz odwrotną do macierzy 3×3 :

$$\mathbf{A} = \begin{bmatrix} -1 & 2 & 1 \\ 1 & -2 & 0 \\ 2 & 0 & -1 \end{bmatrix}.$$

Najpierw obliczamy wyznacznik macierzy $\mathbf{A}$ za pomocą reguły Sarrusa:

$$\det \mathbf{A} = (-2 + 0 + 0) - (-4 - 2 + 0) = 4.$$

Ponieważ zachodzi warunek $\det \mathbf{A} \neq 0$, macierz odwrotna $\mathbf{A}^{-1}$ istnieje. Korzystając ze wzoru (3.5), znajdujemy dopełnienie algebraiczne elementu a_{11} :

$$d_{11} = (-1)^{1+1} \cdot \begin{vmatrix} -2 & 0 \\ 0 & -1 \end{vmatrix} = 2.$$

Postępując analogicznie, wyznaczamy pozostałe elementy macierzy dopełnień:

$$\mathbf{D} = \begin{bmatrix} (-1)^{1+1} \cdot \begin{vmatrix} -2 & 0 \\ 0 & -1 \end{vmatrix} & (-1)^{1+2} \cdot \begin{vmatrix} 1 & 0 \\ 2 & -1 \end{vmatrix} & (-1)^{1+3} \cdot \begin{vmatrix} 1 & -2 \\ 2 & 0 \end{vmatrix} \\ (-1)^{2+1} \cdot \begin{vmatrix} 2 & 1 \\ 0 & -1 \end{vmatrix} & (-1)^{2+2} \cdot \begin{vmatrix} -1 & 1 \\ 2 & -1 \end{vmatrix} & (-1)^{2+3} \cdot \begin{vmatrix} -1 & 2 \\ 2 & 0 \end{vmatrix} \\ (-1)^{3+1} \cdot \begin{vmatrix} 2 & 1 \\ -2 & 0 \end{vmatrix} & (-1)^{3+2} \cdot \begin{vmatrix} -1 & 1 \\ 1 & 0 \end{vmatrix} & (-1)^{3+3} \cdot \begin{vmatrix} -1 & 2 \\ 1 & -2 \end{vmatrix} \end{bmatrix}.$$

Po wykonaniu obliczeń uzyskujemy wynik:

$$\mathbf{D} = \begin{bmatrix} 2 & 1 & 4 \\ 2 & -1 & 4 \\ 2 & 1 & 0 \end{bmatrix}.$$

Następnie obliczamy transpozycję macierzy $\mathbf{D}$, zapisując jej kolejne wiersze jako kolumny:

$$\mathbf{D}^T = \begin{bmatrix} 2 & 2 & 2 \\ 1 & -1 & 1 \\ 4 & 4 & 0 \end{bmatrix}.$$

Na koniec stosujemy wzór (3.6), aby wyznaczyć macierz odwrotną:

$$\mathbf{A}^{-1} = \frac{1}{4} \cdot \begin{bmatrix} 2 & 2 & 2 \\ 1 & -1 & 1 \\ 4 & 4 & 0 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 1/2 \\ 1/4 & -1/4 & 1/4 \\ 1 & 1 & 0 \end{bmatrix}.$$

Sprawdzamy poprawność rachunków w **Sage**:

```
A = matrix( [[-1, 2, 1], [1, -2, 0], [2, 0, -1]] )
```

```
~A
```

```
Out: [ 1/2  1/2  1/2]
```

```
      [ 1/4 -1/4  1/4]
```

```
      [  1    1    0]
```

Przykład 3.23

Dana jest macierz:

$$\mathbf{A} = \begin{bmatrix} a & 3 & 2 \\ 4 & 2 & 1 \\ a & 1 & 3 \end{bmatrix}.$$

Znajdziemy wartości parametru $a \in \mathbb{R}$, dla których suma wszystkich elementów pierwszego wiersza macierzy odwrotnej $\mathbf{A}^{-1}$ jest równa 3.

Stosujemy regułę Sarrusa, aby obliczyć wyznacznik macierzy $\mathbf{A}$:

$$\det \mathbf{A} = (6a + 3a + 8) - (4a + 36 + a) = 4a - 28.$$

Analizując wzór (3.6), dochodzimy do wniosku, że elementy pierwszego wiersza macierzy odwrotnej $\mathbf{A}^{-1}$ odpowiadają elementom pierwszej kolumny macierzy dopełnień $\mathbf{D}$, ponieważ we wzorze występuje transpozycja. W związku z tym obliczamy te dopełnienia algebraiczne zgodnie ze wzorem (3.5):

$$d_{11} = (-1)^{1+1} \cdot \begin{vmatrix} 2 & 1 \\ 1 & 3 \end{vmatrix} = 5$$

$$d_{21} = (-1)^{2+1} \cdot \begin{vmatrix} 3 & 2 \\ 1 & 3 \end{vmatrix} = -7$$

$$d_{31} = (-1)^{3+1} \cdot \begin{vmatrix} 3 & 2 \\ 2 & 1 \end{vmatrix} = -1.$$

Otrzymujemy szablon macierzy dopełnień:

$$\mathbf{D} = \begin{bmatrix} 5 & \bullet & \bullet \\ -7 & \bullet & \bullet \\ -1 & \bullet & \bullet \end{bmatrix},$$

zatem transpozycja macierzy dopełnień jest postaci:

$$\mathbf{D}^T = \begin{bmatrix} 5 & -7 & -1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}.$$

Na podstawie wzoru (3.6) macierz odwrotna to:

$$\mathbf{A}^{-1} = \frac{1}{4a-28} \cdot \begin{bmatrix} 5 & -7 & -1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix},$$

w związku z czym suma elementów pierwszego wiersza macierzy odwrotnej wynosi:

$$S = \frac{1}{4a-28} \cdot (5 + (-7) + (-1)) = \frac{-3}{4a-28}.$$

Rozwiązując poniższe równanie, znajdujemy wartość parametru a , dla której ta suma jest równa 3:

$$\frac{-3}{4a-28} = 3 \quad \rightarrow \quad -3 = 3 \cdot (4a-28) \quad \rightarrow \quad 4a-28 = -1 \quad \rightarrow \quad a = \frac{27}{4}.$$

Wykonujemy powyższe kroki w **Sage**. Najpierw obliczamy wyznacznik macierzy **A**:

```
var('a')
A = matrix( [[a, 3, 2], [4, 2, 1], [a, 1, 3]] )
dA = det(A)
dA
Out: 4*a - 28
```

Następnie znajdujemy dopełnienia algebraiczne elementów z pierwszej kolumny:

```
M11 = A.delete_rows( [0] ).delete_columns( [0] ).det()
d11 = (-1)^(1+1) * M11
print('Dopełnienie elementu a11:', d11)
M21 = A.delete_rows( [1] ).delete_columns( [0] ).det()
d21 = (-1)^(2+1) * M21
print('Dopełnienie elementu a21:', d21)
```

```
M31 = A.delete_rows( [2] ).delete_columns( [0] ).det()
d31 = (-1)^(3+1) * M31
print('Dopełnienie elementu a31:', d31)
Out: Dopełnienie elementu a11: 5
      Dopełnienie elementu a21: -7
      Dopełnienie elementu a31: -1
```

Zapisujemy sumę i wyznaczamy szukane wartości parametru:

```
S = (d11 + d21 + d31) / dA
solve( S==3, a )
Out: [a == (27/4)]
```

Zadanie możemy też rozwiązać bezpośrednio, obliczając macierz odwrotną $\mathbf{A}^{-1}$:

```
B = A^-1
solve( B[0, 0] + B[0, 1] + B[0, 2] == 3, a )
Out: [a == (27/4)]
```

Na koniec przedstawimy tzw. permutacyjną definicję wyznacznika, która wykorzystuje pojęcie permutacji.

Definicja. Wyznacznikiem macierzy kwadratowej $\mathbf{A} \in \mathbf{M}_{n \times n}(\mathbb{R})$ nazywamy liczbę:

$$\det \mathbf{A} = \sum_{\sigma \in S_n} (\operatorname{sgn}(\sigma) \cdot a_{1\sigma(1)} \cdot \dots \cdot a_{n\sigma(n)}).$$

Symbol Σ w definicji oznacza sumę wszystkich składników z zadanego zbioru (lub zakresu) – u nas jest to zbiór S_n permutacji zbioru n -elementowego $\{1, 2, 3, \dots, n\}$. Oznacza to, że dodajemy wyrażenia znajdujące się w nawiasie, przy czym w każdej „iteracji” permutacja σ jest innym elementem zbioru S_n .

Przykład 3.24

Obliczymy wyznacznik macierzy 3×3 , korzystając z definicji permutacyjnej:

$$\mathbf{A} = \begin{bmatrix} 0 & 2 & 3 \\ 0 & 1 & 2 \\ 1 & 4 & 5 \end{bmatrix}.$$

W definicji sumujemy po wszystkich permutacjach zbioru S_3 , czyli występuje w niej $3! = 6$ składników. Ponieważ permutacja jest bijekcją, możemy ją utożsamiać z pewnym układem wież na szachownicy 3×3 , tak aby żadna wieża nie była pod biciem. Przedstawimy to schematycznie, zapisując iloczyny $a_{1\sigma(1)} \cdot \dots \cdot a_{n\sigma(n)}$ uzyskane dla poszczególnych permutacji:

1. Permutacje parzyste.

$$\sigma_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \rightarrow \begin{bmatrix} \cdot & & \\ & \cdot & \\ & & \cdot \end{bmatrix} \rightarrow a_{11}a_{22}a_{33}$$

$$\sigma_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} \rightarrow \begin{bmatrix} & \cdot & \\ \cdot & & \\ & & \cdot \end{bmatrix} \rightarrow a_{12}a_{23}a_{31}$$

$$\sigma_3 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix} \rightarrow \begin{bmatrix} & & \cdot \\ \cdot & & \\ & \cdot & \end{bmatrix} \rightarrow a_{13}a_{21}a_{32}.$$

2. Permutacje nieparzyste.

$$\sigma_4 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \rightarrow \begin{bmatrix} & & \cdot \\ & \cdot & \\ \cdot & & \end{bmatrix} \rightarrow a_{13}a_{22}a_{31}$$

$$\sigma_5 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix} \rightarrow \begin{bmatrix} & \cdot & \\ \cdot & & \\ & & \cdot \end{bmatrix} \rightarrow a_{12}a_{21}a_{33}$$

$$\sigma_6 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix} \rightarrow \begin{bmatrix} \cdot & & \\ & \cdot & \\ & & \cdot \end{bmatrix} \rightarrow a_{11}a_{23}a_{32}.$$

Znak $\text{sgn}(\sigma)$ jest równy 1 dla permutacji parzystych i -1 dla permutacji nieparzystych. Okazuje się, że po zsumowaniu wszystkich składników otrzymujemy regułę Sarrusa (3.2):

$$\det \mathbf{A} = \sum_{\sigma \in S_3} (\text{sgn}(\sigma) \cdot a_{1\sigma(1)} \cdot a_{2\sigma(2)} \cdot a_{3\sigma(3)})$$

$$= a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{12}a_{21}a_{33} - a_{11}a_{23}a_{32}.$$

Wobec tego wyznacznik macierzy $\mathbf{A}$ jest równy:

$$\det \mathbf{A} = 0 + 4 + 0 - 3 - 0 - 0 = 1.$$

Sprawdzamy uzyskany wynik w **Sage**:

```
A = matrix( [[0, 2, 3], [0, 1, 2], [1, 4, 5]] )
det(A)
Out: 1
```

3.3.3. Zastosowanie wyznacznika w rozwiązywaniu układów równań

Definicja. Układem Cramera nazywamy układ równań liniowych, w którym macierz układu $\mathbf{A}$ jest kwadratowa i odwracalna.

Poniższe twierdzenie pozwala określić liczbę rozwiązań takiego układu:

Twierdzenie (Cramera). Jeżeli $\det \mathbf{A} \neq 0$, to układ $\mathbf{Ax} = \mathbf{b}$ jest oznaczony.

Przykład 3.25

Zbadamy liczbę rozwiązań układu:

$$\begin{cases} x + 3y + 2z = 0 \\ -2x + y + 3z = 0 \\ 3x - 2y + z = 0. \end{cases}$$

Macierz układu $\mathbf{A}$ ma postać:

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & 2 \\ -2 & 1 & 3 \\ 3 & -2 & 1 \end{bmatrix}.$$

Ponieważ macierz **A** jest kwadratowa, możemy zastosować twierdzenie Cramera. Sprawdzamy wyznacznik macierzy **A**:

```
A = matrix( [[1, 3, 2], [-2, 1, 3], [3, -2, 1]] )
det(A)
Out: 42
```

Ponieważ wyznacznik macierzy **A** jest różny od zera, układ ma dokładnie jedno rozwiązanie. Dodatkowo układ, w którym wektor wyrazów wolnych **b** jest wektorem zerowym, czyli:

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

nazywamy układem jednorodnym. Taki układ zawsze ma rozwiązanie zerowe:
 $x = y = z = 0$.

W naszym przykładzie jest to jedyne rozwiązanie.

Układ Cramera można rozwiązać za pomocą macierzy odwrotnej, ponieważ dla macierzy odwracalnej **A** zachodzi:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \quad \rightarrow \quad \mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}.$$

W Sage zapisujemy to działanie w następujący sposób:

```
b = vector( [0, 0, 0] )
A^-1 * b
Out: (0, 0, 0)
```

Weźmy teraz dla uproszczenia układ Cramera składający się z trzech równań. Rozwiązanie takiego układu dane jest tzw. wzorami Cramera:

$$x = \frac{W_x}{W}, \quad y = \frac{W_y}{W}, \quad z = \frac{W_z}{W},$$

gdzie $W = \det A$, natomiast W_x , W_y i W_z to wyznaczniki macierzy powstałych z macierzy **A** poprzez zastąpienie kolumny odpowiadającej danej zmiennej wektorem wyrazów wolnych.

Przykład 3.26

Korzystając ze wzorów Cramera, rozwiążemy układ równań:

$$\begin{cases} 7x + 5y - 3z = 2 \\ -4x - 9y + 8z = 0 \\ -5x - 11y + 2z = 1. \end{cases}$$

Najpierw obliczamy wyznacznik macierzy układu:

$$\mathbf{W} = \begin{vmatrix} 7 & 5 & -3 \\ -4 & -9 & 8 \\ -5 & -11 & 2 \end{vmatrix} = (-126 - 200 - 132) - (-135 - 616 - 40) = 333.$$

Mamy $\mathbf{W} \neq 0$, więc układ jest oznaczony. Następne liczymy wyznaczniki $\mathbf{W}_x$, $\mathbf{W}_y$ i $\mathbf{W}_z$, zastępując kolejne kolumny macierzy wektorem wyrazów wolnych $\mathbf{b} = [2, 0, 1]^T$:

$$\mathbf{W}_x = \begin{vmatrix} 2 & 5 & -3 \\ 0 & -9 & 8 \\ 1 & -11 & 2 \end{vmatrix} = (-36 + 40 + 0) - (27 + 0 - 176) = 153$$

$$\mathbf{W}_y = \begin{vmatrix} 7 & 2 & -3 \\ -4 & 0 & 8 \\ -5 & 1 & 2 \end{vmatrix} = (0 - 80 + 12) - (0 - 16 + 56) = -108$$

$$\mathbf{W}_z = \begin{vmatrix} 7 & 5 & 2 \\ -4 & -9 & 0 \\ -5 & -11 & 1 \end{vmatrix} = (-63 + 0 + 88) - (90 - 20 + 0) = -45.$$

Stąd na podstawie wzorów Cramera rozwiązaniem układu jest trójka liczb:

$$x = \frac{153}{333} = \frac{17}{37}, \quad y = \frac{-108}{333} = -\frac{12}{37}, \quad z = \frac{-45}{333} = -\frac{5}{37}.$$

W Sage tworzymy trzy kopie macierzy $\mathbf{A}$, zastępujemy kolejne kolumny wektorem wyrazów wolnych $\mathbf{b}$, a następnie stosujemy wzory Cramera:

```
A = matrix( [[7, 5, -3], [-4, -9, 8], [-5, -11, 2]] )
```

```
b = vector( [2, 0, 1] )
```

```
A1 = copy(A)
```

```
A1[:, 0] = b
```

```
A2 = copy(A)
```

```

A2[:, 1] = b
A3 = copy(A)
A3[:, 2] = b

W = det(A)
Wx, Wy, Wz = det(A1), det(A2), det(A3)
x, y, z = Wx/W, Wy/W, Wz/W

print(f'Rozwiązanie układu: {x = }, {y = }, {z = }')
Out: Rozwiązanie układu: x = 17/37, y = -12/37, z = -5/37

```

Przykład 3.27

Określmy liczbę rozwiązań poniższego układu równań w zależności od parametru $a \in \mathbb{R}$:

$$\begin{cases} ax + y + 2z = 1 \\ 5x + ay + 4z = 1 \\ 7x + ay + 5z = a. \end{cases}$$

Macierz układu $\mathbf{A}$ jest kwadratowa, więc możemy obliczyć jej wyznacznik. Stosujemy regułę Sarrusa:

$$\det \mathbf{A} = \begin{vmatrix} a & 1 & 2 \\ 5 & a & 4 \\ 7 & a & 5 \end{vmatrix} = (5a^2 + 28 + 10a) - (14a + 25 + 4a^2) = a^2 - 4a + 3.$$

Na podstawie twierdzenia Cramera układ ma jedno rozwiązanie tylko wtedy, gdy:

$$\begin{aligned} \det \mathbf{A} \neq 0 &\rightarrow a^2 - 4a + 3 = 0 \rightarrow (a - 1)(a - 3) \neq 0 \\ &\rightarrow a \neq 1 \wedge a \neq 3. \end{aligned}$$

Aby rozwiązać to równanie w **Sage**, możemy wykorzystać funkcję **solve()**:

```

var('a')
A = matrix( [[a, 1, 2], [5, a, 4], [7, a, 5]] )
dA = det(A)
solve(dA, a)
Out: [a == 3, a == 1]

```

Przypadki $a = 1$ i $a = 3$ rozpatrujemy osobno. Zapisujemy macierz uzupełnioną układu:

```
b = vector( [1, 1, a] )
U = A.augment(b, subdivide=True)
U
Out: [a 1 2|1]
      [5 a 4|1]
      [7 a 5|a]
```

Podstawiamy wartość $a = 1$ i sprowadzamy macierz uzupełnioną $\mathbf{U}$ do postaci schodkowej:

```
U(a=1).echelon_form()
Out: [ 1  0 1/2  0]
      [ 0  1 3/2  1]
      [ 0  0  0  0]
```

Otrzymujemy:

$$r = r(\mathbf{A}) = r(\mathbf{U}) = 2 \quad \text{oraz} \quad n = 3,$$

czyli na podstawie twierdzenia Kroneckera-Capellego układ ma dla $a = 1$ nieskończenie wiele rozwiązań zależnych od $\underbrace{n - r}_{3-2} = 1$ parametru.

Dla $a = 3$ mamy natomiast następującą postać schodkową:

```
U(a=3).echelon_form()
Out: [ 1  0 1/2  0]
      [ 0  1 1/2  0]
      [ 0  0  0  1]
```

Wobec tego otrzymujemy zależność:

$$2 = r(\mathbf{A}) < r(\mathbf{U}) = 3,$$

czyli dla $a = 3$ układ jest sprzeczny.

3.4. Równania macierzowe

Równania macierzowe rozwiązuje się w sposób podobny do równań permutacyjnych. Wynika to z faktu, że zarówno mnożenie macierzy, jak i składanie permutacji nie są działaniami przemiennymi. W równaniach macierzowych pojawia się jednak dodatkowy element – oprócz mnożenia występują także dodawanie i odejmowanie. To sprawia, że podczas ich rozwiązywania możemy postępować podobnie jak przy zwykłych równaniach liniowych w liczbach rzeczywistych, ale dodatkowo musimy zwracać uwagę na kolejność wykonywanych działań.

Rozwiązywanie równań macierzowych można przeprowadzać dwoma sposobami. Wykonujemy wówczas jedną z poniższych czynności:

- wyznaczamy szukaną macierz z równania i przeprowadzamy odpowiednie rachunki, podobnie jak w przypadku równań permutacyjnych;
- stosujemy bezpośrednie podstawienie, czyli przewidujemy wymiar rozwiązania i rozpisujemy równanie na zwykłe układy równań dla elementów macierzy.

W tym rozdziale omówimy obie metody, a następnie przejdziemy do równań, w których pojawiają się wyznaczniki. Przedstawimy również twierdzenie Cauchy’ego, które pozwala w prosty sposób obliczać wyznacznik iloczynu dwóch macierzy.

3.4.1. Wyznaczanie macierzy z równania

Podczas przekształcania równań macierzowych stosujemy techniki znane z rozwiązywania zwykłych równań w liczbach rzeczywistych. W szczególności w równaniach liniowych przenosimy wszystkie niewiadome na jedną stronę, a macierze znane – na drugą.

Przykład 3.28

Dla macierzy

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & -1 \\ 1 & 3 & -2 \\ 0 & 2 & 3 \end{bmatrix}$$

rozwiążemy równanie:

$$3\mathbf{X} + 2\mathbf{A} = \mathbf{I} + \mathbf{X}.$$

Jest to równanie liniowe, ponieważ wszystkie niewiadome $\mathbf{X}$ występują w pierwszej potęgze. Wobec tego przenosimy wszystkie składniki zawierające szukaną macierz $\mathbf{X}$ na lewą stronę równania, a pozostałe – na drugą:

$$3\mathbf{X} - \mathbf{X} = \mathbf{I} - 2\mathbf{A}.$$

Wykonujemy odejmowanie po lewej stronie równania:

$$2\mathbf{X} = \mathbf{I} - 2\mathbf{A},$$

a następnie dzielimy równanie stronami przez 2:

$$\mathbf{X} = \frac{1}{2} \cdot (\mathbf{I} - 2\mathbf{A}).$$

Aby wyznaczyć macierz $\mathbf{X}$, wykonujemy rachunki otrzymane po prawej stronie. Ponieważ $\mathbf{A}$ jest macierzą 3×3 , macierz jednostkowa $\mathbf{I}$ również będzie miała ten sam wymiar, aby można było wykonać odejmowanie:

$$\begin{aligned} \mathbf{X} &= \frac{1}{2} \cdot \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - 2 \cdot \begin{bmatrix} 2 & -1 & -1 \\ 1 & 3 & -2 \\ 0 & 2 & 3 \end{bmatrix} \right) \\ &= \frac{1}{2} \cdot \begin{bmatrix} -3 & 2 & 2 \\ -2 & -5 & 4 \\ 0 & -4 & -5 \end{bmatrix} = \begin{bmatrix} -3/2 & 1 & 1 \\ -1 & -5/2 & 2 \\ 0 & -2 & -5/2 \end{bmatrix}. \end{aligned}$$

Sprawdzamy ten rachunek w **Sage** – macierz jednostkową oznaczamy jako `Id`, aby nie nadpisywać jednostki urojonej `I`:

```
A = matrix( [[2, -1, -1], [1, 3, -2], [0, 2, 3]] )
Id = identity_matrix(3)
X = (Id - 2*A) / 2
X
Out: [-3/2  1  1]
      [-1 -5/2  2]
      [ 0 -2 -5/2]
```

Macierze w równaniach można również wyłączać przed nawias. Trzeba jednak pamiętać, aby robić to po właściwej stronie, ponieważ mnożenie dwóch macierzy nie jest przemienne.

Przykład 3.29

Dane są macierze:

$$\mathbf{A} = \begin{bmatrix} -1 & 3 \\ 0 & 1 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} 2 & 1 \\ -1 & -1 \end{bmatrix}.$$

Rozwiążemy równanie:

$$\mathbf{AX} - \mathbf{A} = \mathbf{BX}.$$

Przenosimy wszystkie wyrazy z niewiadomą $\mathbf{X}$ na jedną stronę, a dane – na drugą. Otrzymujemy:

$$\mathbf{AX} - \mathbf{BX} = \mathbf{A}.$$

Wyłączamy szukaną macierz $\mathbf{X}$ przed nawias. Możemy to zrobić, ponieważ $\mathbf{X}$ znajduje się po prawej stronie w obu składnikach ($\mathbf{AX}$ oraz $\mathbf{BX}$). Kolejność działań jest tutaj istotna, dlatego $\mathbf{X}$ po wyłączeniu również pozostaje po prawej stronie:

$$(\mathbf{A} - \mathbf{B}) \cdot \mathbf{X} = \mathbf{A}.$$

Teraz mnożymy obustronnie przez macierz odwrotną do $(\mathbf{A} - \mathbf{B})$. Ponieważ mnożenie macierzy nie jest przemienne, należy koniecznie wykonać tę operację z lewej strony:

$$\underbrace{(\mathbf{A} - \mathbf{B})^{-1} \cdot (\mathbf{A} - \mathbf{B})}_{\mathbf{I}} \cdot \mathbf{X} = (\mathbf{A} - \mathbf{B})^{-1} \cdot \mathbf{A}.$$

Iloczyn $(\mathbf{A} - \mathbf{B})^{-1} \cdot (\mathbf{A} - \mathbf{B})$ daje macierz jednostkową $\mathbf{I}$, a mnożenie przez $\mathbf{I}$ (czyli $\mathbf{I} \cdot \mathbf{X}$) nie zmienia wyniku. Ostatecznie otrzymujemy:

$$\mathbf{X} = (\mathbf{A} - \mathbf{B})^{-1} \cdot \mathbf{A}.$$

Obliczenia przeprowadzamy krok po kroku. Na początku znajdujemy macierz $\mathbf{A} - \mathbf{B}$, czyli:

$$\mathbf{A} - \mathbf{B} = \begin{bmatrix} -1 & 3 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 2 & 1 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} -3 & 2 \\ 1 & 2 \end{bmatrix}.$$

Następnie wyznaczamy macierz odwrotną do macierzy $(\mathbf{A} - \mathbf{B})$. W tym celu liczymy jej wyznacznik:

$$\det(\mathbf{A} - \mathbf{B}) = -3 \cdot 2 - 2 \cdot 1 = -8,$$

a następnie korzystamy ze wzoru (3.4) na macierz odwrotną do macierzy 2×2 :

$$(\mathbf{A} - \mathbf{B})^{-1} = \frac{1}{-8} \cdot \begin{bmatrix} 2 & -2 \\ -1 & -3 \end{bmatrix} = \begin{bmatrix} -1/4 & 1/4 \\ 1/8 & 3/8 \end{bmatrix}.$$

Na koniec wykonujemy mnożenie za pomocą schematu przedstawionego w przykładzie 3.3:

$$\mathbf{X} = (\mathbf{A} - \mathbf{B})^{-1} \cdot \mathbf{A} = \begin{bmatrix} -1/4 & 1/4 \\ 1/8 & 3/8 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1/4 & -1/2 \\ -1/8 & 3/4 \end{bmatrix}.$$

Sprawdzamy poprawność obliczeń, korzystając z **Sage**:

```
A = matrix( [[-1, 3], [0, 1]] )
```

```
B = matrix( [[2, 1], [-1, -1]] )
```

```
X = (A - B)^-1 * A
```

```
X
```

```
Out: [ 1/4 -1/2]
```

```
[-1/8 3/4]
```

Wyjątkiem jest mnożenie macierzy przez skalar – w tym przypadku kolejność wykonywania działań nie ma znaczenia. Podobną własność obserwowaliśmy wcześniej przy składaniu cykli rozłącznych – to działanie również było przemienne.

Przykład 3.30

Niech:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ -1 & 2 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} -1 & 1 \\ 0 & 2 \end{bmatrix}.$$

Rozwiążemy równanie:

$$\mathbf{XA} + 2\mathbf{X} = \mathbf{I} + 3\mathbf{B}.$$

Na początku nie możemy jeszcze wyłączyć macierzy $\mathbf{X}$ przed nawias, ponieważ w składniku $\mathbf{XA}$ stoi ona po lewej stronie, a w $2\mathbf{X}$ – po prawej. Korzystając jednak z faktu, że mnożenie macierzy przez liczbę jest przemienne, zapisujemy:

$$\mathbf{X} \cdot \mathbf{A} + \mathbf{X} \cdot 2 = \mathbf{I} + 3\mathbf{B}.$$

Chcemy teraz wyłączyć $\mathbf{X}$ z lewej strony, jednak po wykonaniu tej czynności otrzymalibyśmy w nawiasie wyrażenie $(\mathbf{A} + 2)$, które nie ma sensu, ponieważ nie można dodawać liczby do macierzy. Dlatego, aby operacja ta była możliwa,

sztucznie domyślamy liczbę 2 przez macierz jednostkową $\mathbf{I}$ (operacja ta nie zmienia wyniku działania):

$$\mathbf{X} \cdot \mathbf{A} + \mathbf{X} \cdot 2\mathbf{I} = \mathbf{I} + 3\mathbf{B}.$$

Teraz możemy wyłączyć $\mathbf{X}$ z lewej strony:

$$\mathbf{X} \cdot (\mathbf{A} + 2\mathbf{I}) = \mathbf{I} + 3\mathbf{B}.$$

Dalej postępujemy podobnie jak w przykładzie 3.29 – mnożymy równanie obustronnie przez macierz odwrotną do $(\mathbf{A} + 2\mathbf{I})$. Tym razem jednak wykonujemy tę operację z prawej strony:

$$\mathbf{X} = (\mathbf{I} + 3\mathbf{B}) \cdot (\mathbf{A} + 2\mathbf{I})^{-1}.$$

Tym razem obliczenia przeprowadzimy jedynie w **SageMath**. Tutaj macierz jednostkowa ma wymiary 2×2 , ponieważ macierze $\mathbf{A}$ i $\mathbf{B}$ również mają wymiary 2×2 :

```
A = matrix( [[1, 0], [-1, 2]] )
B = matrix( [[-1, 1], [0, 2]] )
Id = identity_matrix(2)

X = (Id + 3*B) * (A + 2*Id)^-1
X
Out: [-5/12  3/4]
      [ 7/12  7/4]
```

3.4.2. Podstawienie bezpośrednie

Może się zdarzyć sytuacja, w której nie będziemy w stanie podać przepisu na szukaną macierz $\mathbf{X}$. Wówczas postępujemy według następującego schematu:

1. Określamy wymiar niewiadomej macierzy $\mathbf{X}$.
2. Wprowadzamy oznaczenia dla jej elementów i wykonujemy odpowiednie obliczenia na tych symbolach.
3. Korzystamy z własności: dwie macierze są równe tylko wtedy, gdy równe są wszystkie ich odpowiadające elementy.

Przykład 3.31

Weźmy macierze:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & -1 & -1 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} -1 & 1 \\ 0 & 3 \end{bmatrix}.$$

Rozwiążemy równanie:

$$\mathbf{AX} = \mathbf{B}.$$

W tym przypadku nie możemy pomnożyć równania stronami przez macierz odwrotną, ponieważ macierz $\mathbf{A}$ nie jest kwadratowa. Dlatego stosujemy opisany wcześniej algorytm:

1. Określenie wymiaru.

Macierz $\mathbf{A}$ ma wymiary 2×3 . Aby iloczyn $\mathbf{AX}$ był określony, pierwszy wymiar macierzy $\mathbf{X}$ powinien być równy 3. Ponadto wynik mnożenia (czyli $\mathbf{B}$) jest macierzą 2×2 , więc macierz $\mathbf{X}$ musi mieć wymiary 3×2 . Wtedy uzyskujemy poprawną równość dla wymiarów:

$$2 \times \underbrace{3 \cdot 3}_{=} \times 2 = 2 \times 2.$$

2. Wprowadzenie oznaczeń.

Na podstawie kroku pierwszego szukaną macierz $\mathbf{X}$ możemy zapisać w postaci:

$$\mathbf{X} = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix},$$

gdzie a, b, c, d, e i f są pewnymi liczbami rzeczywistymi (lub ewentualnie – zespolonymi). Wykonujemy iloczyn macierzy po lewej stronie równania:

$$\mathbf{AX} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & -1 & -1 \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} = \begin{bmatrix} a + 2e & b + 2f \\ -c - e & -d - f \end{bmatrix}.$$

Równanie macierzowe można zatem zapisać w postaci równości macierzy 2×2 :

$$\begin{bmatrix} a + 2e & b + 2f \\ -c - e & -d - f \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & 3 \end{bmatrix}.$$

3. Rozwiązanie układu równań.

Dwie macierze są równe, gdy ich elementy na odpowiadających sobie pozycjach są identyczne. Porównując te elementy, zapisujemy równanie macierzowe równoważnie w postaci układu czterech równań:

$$\begin{cases} a + 2e = -1 \\ b + 2f = 1 \\ -c - e = 0 \\ -d - f = 3. \end{cases}$$

Rozwiązujemy ten układ za pomocą funkcji `solve()`:

```
var('a b c d e f')
solve([a+2*e==-1, b+2*f==1, -c-e==0, -d-f==3],
      [a, b, c, d, e, f] )
Out: [[a == -2*r2 - 1, b == -2*r1 + 1,
      c == -r2, d == -r1 - 3,
      e == r2, f == r1]]
```

W **Sage** symbole `r1` i `r2` oznaczają parametry. Przyjmując $e = r_2$ oraz $f = r_1$, macierz $\mathbf{X}$ możemy zapisać w postaci:

$$\mathbf{X} = \begin{bmatrix} -2e - 1 & -2f + 1 \\ -e & -f - 3 \\ e & f \end{bmatrix}, \text{ gdzie } e, f \in \mathbb{R}.$$

Otrzymaliśmy zatem nieskończenie wiele rozwiązań, które zależą od dwóch parametrów. Oznacza to, że po wstawieniu w miejsce tych parametrów dowolnych liczb rzeczywistych uzyskujemy macierz spełniającą początkowe równanie.

Przykład 3.32

Rozwiążemy równanie:

$$\mathbf{X}^2 = \begin{bmatrix} 1 & 0 \\ 8 & 1 \end{bmatrix}.$$

Potęgowanie jest określone tylko dla macierzy kwadratowych. Stąd wynika, że $\mathbf{X}$ musi być macierzą 2×2 . Wprowadzamy oznaczenia elementów tej macierzy:

$$\mathbf{X} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Wykonujemy potęgowanie po lewej stronie równania, zapisując to działanie jako iloczyn:

$$\mathbf{X}^2 = \mathbf{X} \cdot \mathbf{X} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a^2 + bc & ab + bd \\ ac + cd & bc + d^2 \end{bmatrix}.$$

Otrzymujemy równanie macierzowe:

$$\begin{bmatrix} a^2 + bc & ab + bd \\ ac + cd & bc + d^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 8 & 1 \end{bmatrix},$$

które zapisujemy równoważnie w postaci układu równań:

$$\begin{cases} a^2 + bc = 1 \\ ab + bd = 0 \\ ac + cd = 8 \\ bc + d^2 = 1. \end{cases}$$

Teraz możemy rozwiązać ten układ za pomocą funkcji `solve()`, podobnie jak to zostało przedstawione w przykładzie 3.31:

```
solve([a^2+b*c==1, a*b+b*d==0, a*c+c*d==8, b*c+d^2==1],
      [a, b, c, d])
Out: [[a == -1, b == 0, c == -4, d == -1],
      [a == 1, b == 0, c == 4, d == 1]]
```

Stąd uzyskujemy dwa rozwiązania równania:

$$\mathbf{X} = \begin{bmatrix} -1 & 0 \\ -4 & -1 \end{bmatrix} \quad \text{lub} \quad \mathbf{X} = \begin{bmatrix} 1 & 0 \\ 4 & 1 \end{bmatrix}.$$

Okazuje się jednak, że w **Sage** istnieje sposób na rozwiązanie dowolnego równania macierzowego bez wykonywania ręcznych obliczeń. Wystarczy określić wymiar szukanej macierzy $\mathbf{X}$, zapisać obie strony równania oraz wykorzystać kod:

```
X = matrix([ [a, b], [c, d] ])

L = X^2
P = matrix([ [1, 0], [8, 1] ])

solve((L-P).list(), [a, b, c, d])
Out: [[a == -1, b == 0, c == -4, d == -1],
      [a == 1, b == 0, c == 4, d == 1]]
```

Przeanalizujemy działanie ostatniej instrukcji. Zapis L-P przenosi wszystkie elementy równania na jedną stronę. Jest to zatem lewa strona równania postaci:

$$\mathbf{X}^2 - \begin{bmatrix} 1 & 0 \\ 8 & 1 \end{bmatrix} = \mathbf{0},$$

gdzie $\mathbf{0}$ oznacza macierz zerową. Po wstawieniu wyliczonej macierzy $\mathbf{X}$ i wykonaniu odejmowania otrzymujemy równoważną postać równania:

$$\begin{bmatrix} a^2 + bc - 1 & ab + bd \\ ac + cd - 8 & bc + d^2 - 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Macierz L-P po lewej stronie tej równości zamieniamy następnie na listę za pomocą funkcji `list()`. W efekcie uzyskujemy zwykłą listę z języka **Python**:

```
(L-P).list()
```

```
Out: [a^2 + b*c - 1, a*b + b*d, a*c + c*d - 8, b*c + d^2 - 1]
```

Listę tę przekazujemy jako pierwszy argument funkcji `solve()`. Ponieważ `solve()` domyślnie zakłada, że prawa strona każdej równości jest równa 0, uzyskujemy w ten sposób ten sam układ równań, który wcześniej wprowadziliśmy ręcznie.

3.4.3. Twierdzenie Cauchy'ego

Zacniemy od przedstawienia sposobu rozwiązywania równań, w których występują wyznaczniki. Ponieważ wyznacznik macierzy kwadratowej jest liczbą, takie równania rozwiązujemy podobnie jak zwykle równania w liczbach rzeczywistych lub zespolonych.

Przykład 3.33

Rozwiążemy równanie:

$$\begin{vmatrix} 1 & 2 & 0 \\ 2 & x & -1 \\ 0 & -1 & x^2 \end{vmatrix} = 2x - 9.$$

Najpierw obliczamy wyznacznik macierzy 3×3 po lewej stronie, korzystając z reguły Sarrusa:

$$\begin{vmatrix} 1 & 2 & 0 \\ 2 & x & -1 \\ 0 & -1 & x^2 \end{vmatrix} = (x^3 + 0 + 0) - (0 + 4x^2 + 1) = x^3 - 4x^2 - 1.$$

Po podstawieniu wyniku do równania otrzymujemy:

$$x^3 - 4x^2 - 1 = 2x - 9.$$

Jest to równanie wielomianowe trzeciego stopnia, zatem przenosimy wszystkie wyrazy na jedną stronę i przyrównujemy do zera:

$$x^3 - 4x^2 - 2x + 8 = 0.$$

Iloczyn współczynników przy wyrazach skrajnych (x^3 i 8) jest równy iloczynowi współczynników przy wyrazach środkowych ($-4x^2$ i $-2x$), zatem możemy zastosować metodę grupowania:

$$x^2(x - 4) - 2(x - 4) = 0 \rightarrow (x - 4)(x^2 - 2) = 0.$$

Stąd uzyskujemy trzy rozwiązania:

$$x - 4 = 0 \vee x^2 - 2 = 0 \rightarrow x = 4 \vee x = -\sqrt{2} \vee x = \sqrt{2}.$$

W **Sage** równanie standardowo rozwiązujemy za pomocą funkcji **solve()**:

```
A = matrix( [[1, 2, 0], [2, x, -1], [0, -1, x^2]] )
dA = det(A)
solve( dA==2*x-9, x )
Out: [x == -sqrt(2), x == sqrt(2), x == 4]
```

Twierdzenie (Cauchy’ego). Dla macierzy kwadratowych $\mathbf{A}, \mathbf{B} \in \mathbf{M}_{n \times n}(\mathbb{R})$ zachodzi wzór:

$$\det(\mathbf{A} \cdot \mathbf{B}) = \det \mathbf{A} \cdot \det \mathbf{B}. \quad (3.7)$$

Przykład 3.34

Dana jest macierz:

$$\mathbf{A} = \begin{bmatrix} a & 3 & 2 & 3 \\ 2 & 1 & 2 & 2 \\ 2 & 2 & 2 & 1 \\ 3 & 2 & 3 & 2 \end{bmatrix}.$$

Wyznamy takie wartości parametru $a \in \mathbb{R}$, że wyznacznik macierzy $3\mathbf{A}^6$ jest równy $\frac{1}{9}$.

Zaczynamy od obliczenia wyznacznika macierzy $\mathbf{A}$ w zależności od parametru a . Stosujemy w tym celu rozwinięcie Laplace'a, ponieważ stopień wyznacznika jest większy niż 3. Najpierw upraszczamy obliczenia poprzez wyzerowanie trzech elementów, tak jak zostało to pokazane w przykładzie 3.15. Na wyznacznikach możemy wykonywać tylko operacje elementarne, polegające na dodawaniu wierszy lub kolumn przemnożonych przez liczbę:

$$\det \mathbf{A} = \begin{vmatrix} a & 3 & 2 & 3 \\ 2 & 1 & 2 & 2 \\ 2 & 2 & 2 & 1 \\ 3 & 2 & 3 & 2 \end{vmatrix} \stackrel{w_3 - w_2}{=} \begin{vmatrix} a & 3 & 2 & 3 \\ 2 & 1 & 2 & 2 \\ 0 & 1 & 0 & -1 \\ 3 & 2 & 3 & 2 \end{vmatrix}.$$

Możemy w jednym rachunku wykonywać operacje zarówno na wierszach, jak i na kolumnach, ponieważ oba te przekształcenia nie zmieniają wartości wyznacznika:

$$\dots = \begin{vmatrix} a & 3 & 2 & 3 \\ 2 & 1 & 2 & 2 \\ 0 & 1 & 0 & -1 \\ 3 & 2 & 3 & 2 \end{vmatrix} \stackrel{k_4 + k_2}{=} \begin{vmatrix} a & 3 & 2 & 6 \\ 2 & 1 & 2 & 3 \\ 0 & 1 & 0 & 0 \\ 3 & 2 & 3 & 4 \end{vmatrix}.$$

Następnie stosujemy rozwinięcie Laplace'a względem trzeciego wiersza:

$$\dots = \begin{vmatrix} a & 3 & 2 & 6 \\ 2 & 1 & 2 & 3 \\ 0 & \color{red}{1} & 0 & 0 \\ 3 & 2 & 3 & 4 \end{vmatrix} = (-1)^{3+2} \cdot \color{red}{1} \cdot \begin{vmatrix} a & 2 & 6 \\ 2 & 2 & 3 \\ 3 & 3 & 4 \end{vmatrix}.$$

Uzyskany wyznacznik stopnia 3 liczymy za pomocą reguły Sarrusa:

$$\det \mathbf{A} = -1 \cdot [(8a + 18 + 36) - (36 + 16 + 9a)] = a - 2.$$

Aby znaleźć wyznacznik macierzy $\mathbf{A}^6$, możemy zastosować twierdzenie Cauchy'ego:

$$\det(\mathbf{A}^6) = \det\left(\underbrace{\mathbf{A} \cdot \dots \cdot \mathbf{A}}_6\right) = \underbrace{\det \mathbf{A} \cdot \dots \cdot \det \mathbf{A}}_6 = (\det \mathbf{A})^6.$$

Zajmiemy się teraz macierzą $3\mathbf{A}^6$. Z własności wyznacznika wiemy, że przemnożenie jednego wiersza przez 3 zwiększa jego wartość trzykrotnie. Tutaj jednak mnożymy przez 3 całą macierz 4×4 , zatem wykonujemy tę operację dla każdego z czterech wierszy. W efekcie, na podstawie wzoru (3.3), wartość wyznacznika zwiększa się $3^4 = 81$ razy. Ostatecznie uzyskujemy równość:

$$\det(3\mathbf{A}^6) = 3^4 \cdot \det(\mathbf{A}^6) = 81 \cdot \det(\mathbf{A}^6) = 81 \cdot (\det \mathbf{A})^6 = 81 \cdot (a - 2)^6.$$

Mamy zatem do rozwiązania równanie:

$$81 \cdot (a - 2)^6 = \frac{1}{9}.$$

Dzielimy je obustronnie przez 81 i upraszczamy jego prawą stronę:

$$\begin{aligned} (a - 2)^6 &= \frac{1}{9 \cdot 81} \rightarrow (a - 2)^6 = \frac{1}{3^2 \cdot 3^4} \rightarrow (a - 2)^6 = \frac{1}{3^6} \\ &\rightarrow (a - 2)^6 = \left(\frac{1}{3}\right)^6. \end{aligned}$$

Ponieważ wykładnik przy niewiadomej jest parzysty, równanie ma dwa rozwiązania:

$$a - 2 = -\frac{1}{3} \quad \vee \quad a - 2 = \frac{1}{3},$$

a stąd:

$$a = \frac{5}{3} \quad \vee \quad a = \frac{7}{3}.$$

W **Sage** za pomocą funkcji **assume()** możemy założyć, że parametr a jest liczbą rzeczywistą. Następnie rozwiązujemy równanie, korzystając z funkcji **solve()**:

```
var('a')
assume(a, 'real')

A = matrix( [[a, 3, 2, 3], [2, 1, 2, 2],
              [2, 2, 2, 1], [3, 2, 3, 2]] )
```

```
solve( det(3*A^6) == 1/9, a )
```

```
Out: [a == (5/3), a == (7/3)]
```

Więcej szczegółów dotyczących macierzy i układów równań w **Sage** można odnaleźć w pozycji [1].

Wszystko się uda! 😊

4. Zadania

4.1. Zadania z rozwiązaniami

Zadanie 4.1

Zbiór $X = \mathbb{Q} \setminus \left\{-\frac{7}{3}\right\}$ jest grupą z działaniem:

$$a \odot b \stackrel{\text{def}}{=} 3ab + 7a + 7b + 14, \quad \text{dla } a, b \in X,$$

gdzie po prawej stronie występuje standardowe dodawanie i mnożenie liczb wymiernych. Oblicz:

$$2 \odot 3^{-1} \odot 4.$$

Zaczynamy od wyznaczenia elementu neutralnego grupy $(X, \odot)$. Jest to jednoznacznie określona liczba wymierna $e \in X$, która dla każdego elementu $a \in X$ spełnia zależność:

$$a \odot e = a.$$

Korzystamy z definicji działania „ $\odot$ ” i uzyskujemy równanie, w którym niewiadomą jest e :

$$3ae + 7a + 7e + 14 = a.$$

Rozwiązaniem jest szukany element neutralny:

$$3ae + 7e = -6a - 14 \rightarrow e(3a + 7) = -2(3a + 7) \rightarrow e = -2.$$

W **Sage** definiujemy działanie d oraz korzystamy z funkcji **solve()**:

```
d(a, b) = 3*a*b + 7*a + 7*b + 14
e = solve( d(a, e) == a, e )[0].rhs()
print('Element neutralny:', e)
Out: Element neutralny: -2
```

Następnie wyznaczamy element odwrotny $a^{-1} \in X$ do dowolnego elementu $a \in X$. Mamy zależność:

$$a \odot a^{-1} = e,$$

gdzie $e = -2$ jest elementem neutralnym. Ponownie korzystamy z definicji działania „ $\odot$ ”, aby uzyskać równanie:

$$3aa^{-1} + 7a + 7a^{-1} + 14 = -2.$$

Po wyznaczeniu niewiadomej a^{-1} z tego równania otrzymujemy wzór pozwalający obliczyć element odwrotny do dowolnego elementu $a \in X$:

$$\begin{aligned} 3aa^{-1} + 7a^{-1} &= -7a - 16 \quad \rightarrow \quad a^{-1}(3a + 7) = -7a - 16 \\ \rightarrow \quad a^{-1} &= \frac{-7a - 16}{3a + 7}. \end{aligned}$$

W **Sage** element odwrotny a^{-1} oznaczamy literą b . Otrzymany wzór przypisujemy do funkcji f :

```
f(a) = solve( d(a, b) == e, b )[0].rhs()
```

```
print('Wzór na element odwrotny:', f)
```

```
Out: Wzór na element odwrotny: a |--> -(7*a + 16)/(3*a + 7)
```

Teraz możemy obliczyć wynik działania $2 \odot 3^{-1} \odot 4$. Zaczynamy od wyznaczenia elementu odwrotnego 3^{-1} na podstawie powyższego wzoru:

$$3^{-1} = \frac{-7 \cdot 3 - 16}{3 \cdot 3 + 7} = -\frac{37}{16}.$$

Obliczenia wykonujemy krok po kroku – najpierw znajdujemy wynik pierwszego działania:

$$2 \odot 3^{-1} = 2 \odot \left(-\frac{37}{16}\right) \stackrel{\text{def}}{=} 3 \cdot 2 \cdot \left(-\frac{37}{16}\right) + 7 \cdot 2 + 7 \cdot \left(-\frac{37}{16}\right) + 14 = -\frac{33}{16}.$$

Wykorzystujemy uzyskaną wartość, aby podać końcową odpowiedź:

$$(2 \odot 3^{-1}) \odot 4 = \left(-\frac{33}{16}\right) \odot 4 \stackrel{\text{def}}{=} 3 \cdot \left(-\frac{33}{16}\right) \cdot 4 + 7 \cdot \left(-\frac{33}{16}\right) + 7 \cdot 4 + 14 = \frac{45}{16}.$$

W Sage wynik tego działania możemy uzyskać w następujący sposób:

```
wynik = d( d(2, f(3)), 4 )  
print('Wynik działania:', wynik)  
Out: Wynik działania: 45/16
```

Zadanie 4.2

W ciele $\mathbb{Z}_{23}$ oblicz:

$$3^{95} + 5 \cdot 17^{-1} + 14^{-2}.$$

Policzymy osobno każdy z trzech składników występujących w tym działaniu:

1. Liczbę 3^{95} znajdujemy na podstawie małego twierdzenia Fermata. Dla $p = 23$ w wykładniku możemy dodawać i odejmować dowolną wielokrotność liczby $p - 1 = 22$:

$$3^{95} = 3^{95-4 \cdot 22} = 3^{95-88} = 3^7.$$

Wynik rozkładamy na iloczyn czynników znajdujących się w pobliżu pewnej wielokrotności liczby 23 – w ciele $\mathbb{Z}_{23}$ możemy równoważnie dodawać lub odejmować taką wartość:

$$\dots = 3^3 \cdot 3^3 \cdot 3^1 = 27 \cdot 27 \cdot 3 = 4 \cdot 4 \cdot 3 = 48 = 48 - 2 \cdot 23 = 2.$$

2. Szukamy elementu odwrotnego 17^{-1} . Jest to liczba spełniająca równość:

$$17 \cdot \square = 1.$$

Nie jesteśmy w stanie jednak łatwo odgadnąć, co powinno się znaleźć w brakującym polu. W związku z tym należy zastosować inne podejście. Zaprezentujemy tutaj dwie sprytne metody.

Pierwszy sposób polega na zapisaniu elementu odwrotnego w postaci ułamka:

$$17^{-1} = \frac{1}{17}.$$

Próbujemy dodać do licznika pewną wielokrotność liczby 23, aby uzyskać w wyniku liczbę całkowitą. Możemy kombinować w następujący sposób: przy dzieleniu 23

przez 17 mamy resztę 6, natomiast jedynka w liczniku daje resztę 1. Zatem możemy odjąć wartość $3 \cdot 23$, ponieważ wtedy reszta wyniesie $1 - 3 \cdot 6 = -17$, co daje liczbę podzielną przez 17:

$$\dots = \frac{1 - 3 \cdot 23}{17} = \frac{-68}{17} = -4.$$

Wynik można tymczasowo pozostawić jako liczbę ujemną w celu uproszczenia kolejnych rachunków. Należy jedynie pamiętać, aby końcowa odpowiedź należała do zbioru $\mathbb{Z}_{23}$.

Przykład ten rozwiążemy jeszcze drugim sposobem, który dobrze się sprawdza, gdy szukamy elementu odwrotnego do liczby pierwszej. Najpierw odejmujemy 23 i korzystamy z własności działań na potęgach:

$$17^{-1} = (-6)^{-1} = (-1)^{-1} \cdot 6^{-1}.$$

Okazuje się, że uzyskane elementy odwrotne możemy łatwo odgadnąć, ponieważ mamy:

$$(-1) \cdot \boxed{(-1)} = 1 \quad \text{oraz} \quad 6 \cdot \boxed{4} = 1.$$

Wobec tego szukany element odwrotny jest równy:

$$\dots = (-1) \cdot 4 = -4.$$

Ostatecznie drugi składnik przyjmuje wartość:

$$5 \cdot 17^{-1} = 5 \cdot (-4) = -20 = 3.$$

3. Działanie 14^{-2} możemy zapisać w postaci:

$$14^{-2} = (14^{-1})^2 \quad \text{lub} \quad 14^{-2} = (14^2)^{-1}.$$

Zacniemy od pierwszej możliwości – w tej sytuacji najpierw musimy znaleźć element odwrotny 14^{-1} . Wykorzystamy drugi sposób z punktu drugiego:

$$14^{-1} = (-9)^{-1} = (-1)^{-1} \cdot 9^{-1} = (-1) \cdot (3^{-1})^2 = (-1) \cdot 8^2 = -64 = 5,$$

ponieważ element odwrotny do liczby 3 otrzymujemy na podstawie równości $3 \cdot \boxed{8} = 1$.

Następnie podnosimy uzyskany wynik do kwadratu:

$$(14^{-1})^2 = 5^2 = 25 = 2.$$

W drugim przypadku zaczynamy natomiast od potęgowania, a następnie wyznaczamy element odwrotny. Otrzymujemy:

$$14^2 = (-9)^2 = 81 = 81 - 3 \cdot 23 = 12.$$

Wówczas, korzystając z równości $12 \cdot \boxed{2} = 1$, możemy odgadnąć końcowy wynik:

$$(14^2)^{-1} = 12^{-1} = 2.$$

Powyższe przykłady można było rozwiązać za pomocą algorytmu Euklidesa – tutaj zostały przedstawione sprytnie sposoby przeprowadzania tego typu rachunków. Ostatecznie wynik początkowego działania jest równy:

$$3^{95} + 5 \cdot 17^{-1} + 14^{-2} = 2 + 3 + 2 = 7.$$

Sprawdzamy uzyskany rezultat w **Sage**:

```
R = Zmod(23)
R( 3^95 + 5*17^-1 + 14^-2 )
Out: 7
```

Zadanie 4.3

Napisz program, który wypisuje kolejne etapy algorytmu Euklidesa wyznaczającego element odwrotny do liczby a w ciele reszt modulo N .

Zanim przejdziemy do pisania kodu zapiszemy pomocniczo algorytm Euklidesa znajdujący element odwrotny do liczby 17 w ciele reszt modulo 23. Postępując analogicznie jak w przykładzie 2.10, otrzymujemy w pierwszym kroku równości:

$$23 = 17 \cdot 1 + 6$$

$$17 = 6 \cdot 2 + 5$$

$$6 = 5 \cdot 1 + 1.$$

Drugi krok natomiast, czyli rekonstrukcja, wygląda następująco:

$$\begin{aligned} 1 &= 6 - 1 \cdot 5 = 6 - 1 \cdot (17 - 2 \cdot 6) = -1 \cdot 17 + 3 \cdot 6 \\ &= -1 \cdot 17 + 3 \cdot (23 - 17) = 3 \cdot 23 - 4 \cdot 17 = -4 \cdot 17. \end{aligned}$$

Zaczynamy od umieszczenia tych wartości w liście o nazwie `liczby`:

```
N, a = 23, 17
n, d = N, a
```

```

liczby = []
while d != 1:
    k, r = divmod(n, d)
    liczby.append( (n, d, k, r) )
    n, d = d, n%d
liczby
Out: [(23, 17, 1, 6), (17, 6, 2, 5), (6, 5, 1, 1)]

```

Funkcja **divmod()** zwraca iloraz oraz resztę z dzielenia liczby n przez d . Następnie dodajemy czwórkę (n, d, k, r) do listy `liczby` za pomocą funkcji **append()**. Czynności te powtarzamy wewnątrz pętli **while**, która jest wykonywana tak długo, dopóki reszta z dzielenia jest różna od 1 ($d \neq 1$). Teraz, korzystając z listy `liczby`, możemy wypisać równości występujące w pierwszym etapie algorytmu Euklidesa za pomocą składni f-stringa z języka **Python**:

```

for n, d, k, r in liczby:
    print(f'{n} = {d} * {k} + {r}')
Out: 23 = 17 * 1 + 6
     17 = 6 * 2 + 5
     6 = 5 * 1 + 1

```

Trudniejszym wyzwaniem będzie rekonstrukcja, ponieważ musimy przechowywać nie tylko wartości z listy `liczby`, ale również poszczególne współczynniki. Zaczynamy od przypisania do zmiennej `wynik` pierwszego przejścia, czyli $1 = 6 - 1 \cdot 5$. Jako początkowe współczynniki (występujące odpowiednio przy liczbach 6 i 5) ustawiamy 1 i -k:

```

n, d, k, r = liczby[-1]
w1, w2 = 1, -k
wynik = f'{r} = {n} - {k}*{d}'
wynik
Out: '1 = 6 - 1*5'

```

Teraz należy przeanalizować, w jaki sposób współczynniki zmieniają się w kolejnych krokach. W tym celu zapisujemy ogólnie proces rekonstrukcji:

$$1 = w_1 \cdot n_1 + w_2 \cdot d_1,$$

gdzie $w_1 = 1$, $w_2 = -k_1$, natomiast liczby n_i , d_i , k_i oraz r_i przyjmują dla $i = 1, 2, 3$ wartości współczynników występujących w kolejnych równościach z pierwszego kroku.

Następnie uwzględniamy zależności $d_1 = r_2$ i $n_1 = d_2$, otrzymując:

$$\dots = w_1 \cdot d_2 + w_2 \cdot (n_2 - k_2 \cdot d_2) = w_2 \cdot n_2 + (w_1 - w_2 k_2) \cdot d_2.$$

Wnioskujemy stąd, że nowe współczynniki przyjmują wartości:

$$w'_1 = w_2 \quad \text{oraz} \quad w'_2 = w_1 - w_2 k_2.$$

Wykorzystujemy tę obserwację w poniższym kodzie:

```
for n, d, k, r in reversed(liczby[:-1]):
    wynik += f' = {w1}*{d} + {w2}*({n} - {k}*{d}) = '
    w1, w2 = w2, w1-w2*k
    wynik += f'{w1}*{n} + {w2}*{d}'
wynik += f' = {w2}*{d}'
wynik
Out: '1 = 6 - 1*5 = 1*6 + -1*(17 - 2*6) = -1*17 + 3*6
     = -1*17 + 3*(23 - 1*17) = 3*23 + -4*17 = -4*17'
```

W pętli **for** przechodzimy po liście liczby od końca (z pominięciem ostatniego elementu), co odpowiada „zaczynaniu od dołu” w drugim kroku algorytmu Euklidesa. Pierwsza linijka w pętli dodaje do wyniku wyznaczenie reszty, natomiast ostatnia – upraszcza wynik za pomocą nowych współczynników w'_1 i w'_2 . Po wyjściu z pętli uwzględniamy to, że liczba $3 \cdot 23$ w ciele $\mathbb{Z}_{23}$ jest równa 0, więc otrzymujemy w wyniku $\{w_2\} \cdot \{d\}$.

Na koniec wystarczy odczytać element odwrotny – jest to współczynnik $w_2 = -4$, jednak bierzemy resztę z dzielenia przez N , aby wynik należał do zbioru $\mathbb{Z}_{23}$:

```
print(f'{a}^-1 = {w2 % N}')
Out: 17^-1 = 19
```

Program działa dla dowolnej liczby pierwszej N oraz dla liczby naturalnej a różnej od zera i mniejszej niż N . Aby otrzymać inne wyniki, wystarczy podmienić wartości tych zmiennych.

Zadanie 4.4

Dane są permutacje:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 9 & 4 & 5 & 8 & 6 & 3 & 7 & 2 & 1 \end{pmatrix},$$

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 9 & 5 & 7 & 4 & 1 & 3 & 6 & 8 \end{pmatrix}.$$

a) Znajdź najmniejszą liczbę naturalną $n \in \mathbb{N}$, dla której zachodzi równość:

$$\sigma^{80} = \sigma^n.$$

b) Wyznacz permutację $\chi \in S_9$ spełniającą równanie:

$$\sigma^{-80} \circ \tau^{-1} \circ \chi \circ \tau^2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 7 & 5 & 6 & 3 & 9 & 4 & 8 & 1 & 2 \end{pmatrix}.$$

a) Zapisujemy permutację σ w postaci iloczynu cykli rozłącznych:

$$\sigma = (1\ 9)(2\ 4\ 8)(3\ 5\ 6).$$

Rząd permutacji σ to najmniejsza wspólna wielokrotność długości tych cykli, czyli:

$$r(\sigma) = \text{NWW}(2, 3, 3) = 6.$$

Teraz w wykładniku możemy wykonywać działania modulo rząd. Otrzymujemy:

$$\sigma^{80} = \sigma^{80-13 \cdot 6} = \sigma^2,$$

więc szukaną najmniejszą liczbą naturalną jest $n = 2$. W **Sage** stosujemy funkcję **order()** oraz operację modulo (%):

```
s = Permutation( [9, 4, 5, 8, 6, 3, 7, 2, 1] )
n = 80 % s.order()
print('Najmniejsza liczba naturalna to:', n)
Out: Najmniejsza liczba naturalna to: 2
```

b) Oznaczamy przez μ permutację występującą po prawej stronie równania. Otrzymujemy:

$$\sigma^{-80} \circ \tau^{-1} \circ \chi \circ \tau^2 = \mu.$$

Mnożymy równanie obustronnie przez element odwrotny do permutacji σ^{-80} , czyli σ^{80} . Należy pamiętać, że składanie permutacji nie jest przemienne, więc działanie to należy koniecznie wykonać z lewej strony:

$$\tau^{-1} \circ \chi \circ \tau^2 = \sigma^{80} \circ \mu.$$

Następnie przeprowadzamy analogiczną operację – mnożymy równanie stronami przez permutacje τ i τ^{-2} odpowiednio z lewej i prawej strony. W ten sposób eliminujemy czynniki znajdujące się na zewnątrz, aby otrzymać permutację χ występującą pośrodku:

$$\chi = \tau \circ \sigma^{80} \circ \mu \circ \tau^{-2}.$$

Z podpunktu a) wiemy, że zachodzi równość $\sigma^{80} = \sigma^2$, więc wystarczy wykonać działanie:

$$\chi = \tau \circ \sigma^2 \circ \mu \circ \tau^{-2}.$$

Aby wyznaczyć permutację σ^2 , dwukrotnie „idziemy w dół”, otrzymując:

$$\sigma^2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 8 & 6 & 2 & 3 & 5 & 7 & 4 & 9 \end{pmatrix},$$

natomiast dla permutacji τ^{-2} dwukrotnie „idziemy w górę”:

$$\tau^{-2} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 8 & 6 & 4 & 3 & 7 & 9 & 5 & 2 & 1 \end{pmatrix}.$$

Na koniec wykonujemy działania od prawej do lewej strony, czyli w następującej kolejności: $\tau^{-2} \rightarrow \mu \rightarrow \sigma^2 \rightarrow \tau$. Na przykład dla liczby 5 mamy ciąg przejść: $5 \xrightarrow{\tau^{-2}} 7 \xrightarrow{\mu} 8 \xrightarrow{\sigma^2} 4 \xrightarrow{\tau} 7$. Ostatecznie uzyskujemy rozwiązanie równania:

$$\chi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 9 & 1 & 4 & 7 & 6 & 8 & 5 & 3 \end{pmatrix}.$$

W Sage permutacje definiujemy za pomocą obiektu **PermutationGroupElement()**, aby można było korzystać z symbolu potęgowania (^). Następnie zapisujemy działania od prawej do lewej strony:

```
s = PermutationGroupElement( [9, 4, 5, 8, 6, 3, 7, 2, 1] )
t = PermutationGroupElement( [2, 9, 5, 7, 4, 1, 3, 6, 8] )
m = PermutationGroupElement( [7, 5, 6, 3, 9, 4, 8, 1, 2] )
```

```
x = t^-2 * m * s^2 * t
Permutation(x)
Out: [2, 9, 1, 4, 7, 6, 8, 5, 3]
```

Ten sam wynik możemy również uzyskać metodą **Brute Force** poprzez sprawdzenie wszystkich elementów zbioru S_9 :

```
for x in SymmetricGroup(9):
    if t^2 * x * t^-1 * s^-80 == m:
        print( 'Rozwiązanie równania:', Permutation(x) )
        break
Out: Rozwiązanie równania: [2, 9, 1, 4, 7, 6, 8, 5, 3]
```

Zadanie 4.5

Korzystając z twierdzenia Kroneckera-Capellego, określ liczbę rozwiązań układu równań:

$$\begin{cases} x - 2y - z = 0 \\ -2x + y + 3z = 1 \\ -x - y + 2z = 1. \end{cases}$$

Przedstaw uzyskane rozwiązanie w przestrzeni trójwymiarowej.

Zapisujemy macierz układu **A**, wektor wyrazów wolnych **b** oraz macierz uzupełnioną **U**:

$$\mathbf{A} = \begin{bmatrix} 1 & -2 & -1 \\ -2 & 1 & 3 \\ -1 & -1 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{U} = \left[\begin{array}{ccc|c} 1 & -2 & -1 & 0 \\ -2 & 1 & 3 & 1 \\ -1 & -1 & 2 & 1 \end{array} \right].$$

Sprawdzamy w **Sage** rzędy macierzy **A** i **U**. Jeżeli zachodzi równość $r(\mathbf{A}) = r(\mathbf{U})$, to na podstawie twierdzenia Kroneckera-Capellego rozwiązanie układu istnieje (jedno lub nieskończenie wiele):

```
A = matrix( [[1, -2, -1], [-2, 1, 3], [-1, -1, 2]] )
b = vector( [0, 1, 1] )
U = A.augment(b)
```

```
print( 'Rząd macierzy A:', rank(A) )
```

```
print( 'Rząd macierzy U:', rank(U) )
```

Out: Rząd macierzy A: 2

Rząd macierzy U: 2

Wprowadzamy oznaczenia:

- r – wspólna wartość rzędu (u nas $r = 2$);
- n – liczba zmiennych w układzie (tutaj $n = 3$).

Rozpatrujemy wszystkie przypadki za pomocą instrukcji warunkowych **if-else**:

```
if rank(A) == rank(U):
```

```
    r = rank(A)
```

```
    n = A.ncols()
```

```
    if r == n: print('Układ ma dokładnie jedno rozwiązanie.')
```

```
    else: print('Układ ma oo wiele rozwiązań',  
              f'zależnych od {n-r} parametrów.')
```

```
else:
```

```
    print('Układ jest sprzeczny.')
```

Out: Układ ma oo wiele rozwiązań zależnych od 1 parametrów.

Aby przedstawić rozwiązania tego układu równań na rysunku, rozwiązujemy go za pomocą funkcji **solve()**:

```
var('x y z')
```

```
rozv = solve( [x-2*y-z==0, -2*x+y+3*z==1, -x-y+2*z==1],  
              [x, y, z] )
```

rozv

Out: $[[x == 5/3*r_1 - 2/3, y == 1/3*r_1 - 1/3, z == r_1]]$

Następnie wyodrębniamy wartości zmiennych x , y i z z listy `rozv` za pomocą segmentacji i funkcji **rhs()** oraz przypisujemy rezultat do zmiennej parametryzacja:

```
parametryzacja = [r.rhs() for r in rozv[0]]
```

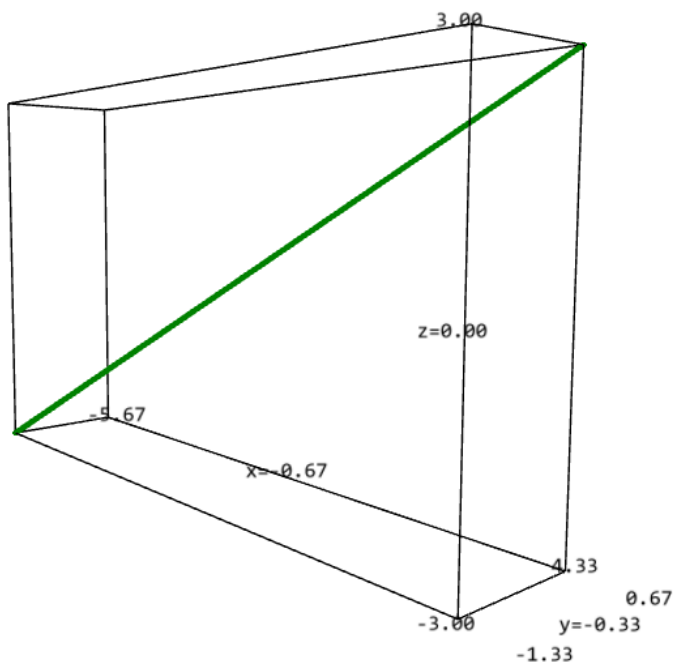
```
parametryzacja
```

```
Out: [5/3*r1 - 2/3, 1/3*r1 - 1/3, r1]
```

Do narysowania rozwiązań wykorzystujemy funkcję **parametric_plot()**:

```
p = parametryzacja[-1]
parametric_plot( parametryzacja, (p, -3, 3),
                 color='green', thickness=3 )
```

Uzyskany wynik został przedstawiony na rysunku 4.1. Do zmiennej *p* przypisujemy parametr *r1* będący ostatnim elementem listy *parametryzacja*. Następnie lista ta jest przekazywana jako pierwszy argument funkcji **parametric_plot()**, która sama rozpoznaje na podstawie liczby współrzędnych, czy należy wygenerować wykres dwu-, czy trójwymiarowy. W rezultacie otrzymujemy prostą, ponieważ rozwiązania zależą od jednego parametru. W notatniku **Jupyter** uzyskany rysunek można dodatkowo obracać kursorem myszy oraz przybliżać i oddalać.



Rysunek 4.1. Rozwiązania układu równań w przestrzeni trójwymiarowej

Zadanie 4.6

Wyznacz macierz odwrotną do macierzy:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 2 \\ -1 & 2 & 1 \\ 1 & -1 & 0 \end{bmatrix}.$$

a) Za pomocą operacji elementarnych.

b) Za pomocą macierzy dopełnień.

a) Macierz odwrotną $\mathbf{A}^{-1}$ można uzyskać w następujący sposób:

- do macierzy $\mathbf{A}$ dołączamy z prawej strony macierz jednostkową $\mathbf{I}$, tworząc macierz rozszerzoną $[\mathbf{A} | \mathbf{I}]$;
- za pomocą operacji elementarnych na wierszach dążymy do uzyskania macierzy jednostkowej $\mathbf{I}$ po lewej stronie – wówczas po prawej stronie otrzymamy macierz odwrotną $\mathbf{A}^{-1}$.

Schematycznie można to zapisać w postaci:

$$[\mathbf{A} | \mathbf{I}] \xrightarrow{\text{operacje elementarne}} [\mathbf{I} | \mathbf{A}^{-1}].$$

Wykonujemy poniższy ciąg operacji elementarnych dla macierzy $[\mathbf{A} | \mathbf{I}]$:

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 & 1 \end{array} \right] \xrightarrow[\begin{array}{l} w_2 + w_1 \\ w_3 - w_1 \end{array}]{\begin{array}{l} w_3 \cdot (-1) \\ w_2 \leftrightarrow w_3 \end{array}} \left[\begin{array}{ccc|ccc} 1 & 0 & 2 & 1 & 0 & 0 \\ 0 & 2 & 3 & 1 & 1 & 0 \\ 0 & -1 & -2 & -1 & 0 & 1 \end{array} \right]$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & -1 \\ 0 & 2 & 3 & 1 & 1 & 0 \end{array} \right] \xrightarrow{w_3 - 2w_2} \left[\begin{array}{ccc|ccc} 1 & 0 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & -1 \\ 0 & 0 & -1 & -1 & 1 & 2 \end{array} \right] \xrightarrow[\begin{array}{l} w_1 + 2w_3 \\ w_2 + 2w_3 \end{array}]{}$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & 2 & 4 \\ 0 & 1 & 0 & -1 & 2 & 3 \\ 0 & 0 & -1 & -1 & 1 & 2 \end{array} \right] \xrightarrow{w_3 \cdot (-1)} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & 2 & 4 \\ 0 & 1 & 0 & -1 & 2 & 3 \\ 0 & 0 & 1 & 1 & -1 & -2 \end{array} \right].$$

Macierz uzyskana po prawej stronie jest szukaną macierzą odwrotną:

$$\mathbf{A}^{-1} = \begin{bmatrix} -1 & 2 & 4 \\ -1 & 2 & 3 \\ 1 & -1 & -2 \end{bmatrix}.$$

W **Sage** korzystamy z funkcji **augment()**, aby połączyć macierze w poziomie. Następnie sprowadzamy uzyskaną macierz do postaci schodkowej:

```
A = matrix( [[1, 0, 2], [-1, 2, 1], [1, -1, 0]] )
Id = identity_matrix(3)
```

```
AI = A.augment(Id, subdivide=True)
AI.rref()
```

```
Out: [ 1  0  0|-1  2  4]
      [ 0  1  0|-1  2  3]
      [ 0  0  1| 1 -1 -2]
```

b) Obliczamy dopełnienia algebraiczne poszczególnych elementów macierzy **A**, korzystając ze wzoru (3.5):

$$d_{11} = \begin{vmatrix} 2 & 1 \\ -1 & 0 \end{vmatrix} = 1, \quad d_{12} = (-1) \cdot \begin{vmatrix} -1 & 1 \\ 1 & 0 \end{vmatrix} = 1, \quad d_{13} = \begin{vmatrix} -1 & 2 \\ 1 & -1 \end{vmatrix} = -1$$

$$d_{21} = (-1) \cdot \begin{vmatrix} 0 & 2 \\ -1 & 0 \end{vmatrix} = -2, \quad d_{22} = \begin{vmatrix} 1 & 2 \\ 1 & 0 \end{vmatrix} = -2, \quad d_{23} = (-1) \cdot \begin{vmatrix} 1 & 0 \\ 1 & -1 \end{vmatrix} = 1$$

$$d_{31} = \begin{vmatrix} 0 & 2 \\ 2 & 1 \end{vmatrix} = -4, \quad d_{32} = (-1) \cdot \begin{vmatrix} 1 & 2 \\ -1 & 1 \end{vmatrix} = -3, \quad d_{33} = \begin{vmatrix} 1 & 0 \\ -1 & 2 \end{vmatrix} = 2.$$

Otrzymujemy macierz dopełnień:

$$\mathbf{D} = \begin{bmatrix} 1 & 1 & -1 \\ -2 & -2 & 1 \\ -4 & -3 & 2 \end{bmatrix}.$$

Liczymy wyznacznik macierzy **A**, stosując regułę Sarrusa:

$$\det \mathbf{A} = \begin{vmatrix} 1 & 0 & 2 \\ -1 & 2 & 1 \\ 1 & -1 & 0 \end{vmatrix} = (0 + 0 + 2) - (4 + 0 - 1) = -1,$$

a następnie korzystamy ze wzoru (3.6), aby wyznaczyć macierz odwrotną:

$$\mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}} \cdot \mathbf{D}^T = \frac{1}{-1} \cdot \begin{bmatrix} 1 & -2 & -4 \\ 1 & -2 & -3 \\ -1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} -1 & 2 & 4 \\ -1 & 2 & 3 \\ 1 & -1 & -2 \end{bmatrix}.$$

W **Sage** zaczynamy od znalezienia macierzy dopełnień. Macierz D definiujemy jako macierz zerową stopnia 3. Następnie w pętli **for** obliczamy dopełnienia algebraiczne elementów macierzy A oraz nadpisujemy poszczególne elementy macierzy D :

```
D = zero_matrix(3)
n = A.nrows()

for i in range(n):
    for j in range(n):
        minor = A.delete_rows( [i] ).delete_columns( [j] )
        D[i, j] = (-1)^(i+j) * det(minor)

D
Out: [ 1  1 -1]
     [-2 -2  1]
     [-4 -3  2]
```

Następnie stosujemy wzór pozwalający wyznaczyć macierz odwrotną:

```
1/det(A) * D.T
Out: [-1  2  4]
     [-1  2  3]
     [ 1 -1 -2]
```

Zadanie 4.7

Określ w zależności od wartości parametru $a \in \mathbb{R}$ liczbę rozwiązań układu równań:

$$\begin{cases} ax + 3y + z = 2 \\ 2x + 2y + z = 1 \\ 3x + y + z = 0 \\ x + 2y + 3z = a. \end{cases}$$

Zapisujemy macierz układu **A** oraz macierz uzupełnioną **U**:

$$\mathbf{A} = \begin{bmatrix} a & 3 & 1 \\ 2 & 2 & 1 \\ 3 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} a & 3 & 1 & 2 \\ 2 & 2 & 1 & 1 \\ 3 & 1 & 1 & 0 \\ 1 & 2 & 3 & a \end{bmatrix}.$$

Macierz **U** jest macierzą kwadratową, więc możemy obliczyć jej wyznacznik. Na podstawie jego wartości wnioskujemy o rzędzie tej macierzy:

- jeżeli $\det \mathbf{U} \neq 0$, to $r(\mathbf{U}) = 4$;
- jeżeli $\det \mathbf{U} = 0$, to $r(\mathbf{U}) < 4$.

Najpierw próbujemy wyzerować jak najwięcej elementów wyznacznika macierzy **U**. W tym celu możemy użyć jednej z operacji elementarnych, która nie zmienia wartości wyznacznika – dodawania do wiersza (lub kolumny) innego wiersza (lub kolumny):

$$\det \mathbf{U} = \begin{vmatrix} a & 3 & 1 & 2 \\ 2 & 2 & 1 & 1 \\ 3 & 1 & 1 & 0 \\ 1 & 2 & 3 & a \end{vmatrix} \xrightarrow[k_1 - 3k_3]{k_2 - k_3} \begin{vmatrix} a - 3 & 2 & 1 & 2 \\ -1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ -8 & -1 & 3 & a \end{vmatrix}.$$

Następnie stosujemy rozwinięcie Laplace’a względem trzeciego wiersza:

$$\dots = 1 \cdot (-1)^{3+3} \cdot \begin{vmatrix} a - 3 & 2 & 2 \\ -1 & 1 & 1 \\ -8 & -1 & a \end{vmatrix}.$$

Uzyskany wyznacznik stopnia 3 możemy już obliczyć, korzystając z reguły Sarrusa:

$$\dots = (a(a - 3) - 16 + 2) - (-16 - 2a - (a - 3)) = a^2 - 1.$$

Sprawdzamy poprawność rachunków w systemie **SageMath**:

```
var('a')
U = matrix( [[a, 3, 1, 2], [2, 2, 1, 1],
              [3, 1, 1, 0], [1, 2, 3, a]] )
det(U)
Out: a^2 - 1
```

Wyznacznik macierzy $\mathbf{U}$ przyjmuje wartość 0 dla $a \in \{-1, 1\}$. Stąd otrzymujemy wniosek:

- $r(\mathbf{U}) = 4$ dla $a \in \mathbb{R} \setminus \{-1, 1\}$;
- $r(\mathbf{U}) < 4$ dla $a \in \{-1, 1\}$.

Macierz $\mathbf{A}$ ma wymiary 4×3 , więc jej rząd może być co najwyżej równy:

$$r(\mathbf{A}) \leq \min\{4, 3\} = 3.$$

Wynika stąd, że dla wartości parametrów $a \in \mathbb{R} \setminus \{-1, 1\}$ mamy:

$$r(\mathbf{A}) \leq 3 < 4 = r(\mathbf{U}),$$

co oznacza, że wówczas układ jest sprzeczny. Wobec tego pozostaje nam sprawdzić osobno liczbę rozwiązań układu dla $a \in \{-1, 1\}$.

Zapisujemy macierz uzupełnioną $\mathbf{U}$ w postaci schodkowej dla $a = 1$:

```
U(a=1).echelon_form()
```

```
Out: [ 1  0  0  0 -2/9]
```

```
      [ 0  1  0  0  7/9]
```

```
      [ 0  0  1  0 -1/9]
```

```
      [ 0  0  0  0  0]
```

Mamy $r(\mathbf{A}) = r(\mathbf{U}) = n = 3$, gdzie n jest liczbą zmiennych, więc dla $a = 1$ układ ma dokładnie jedno rozwiązanie. Podobnie dla $a = -1$ otrzymujemy:

```
U(a=-1).echelon_form()
```

```
Out: [ 1  0  0  0]
```

```
      [ 0  1  0  1]
```

```
      [ 0  0  1 -1]
```

```
      [ 0  0  0  0]
```

Zatem z tego samego powodu układ ma dokładnie jedno rozwiązanie również dla $a = -1$.

Zadanie 4.8

Dane są macierze:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix}.$$

Znajdź macierz $\mathbf{X}$ spełniającą równanie:

$$\mathbf{A}^T + \mathbf{A}(\mathbf{B}\mathbf{X})^T = \mathbf{A}(\mathbf{X}^T + 3\mathbf{I})\mathbf{B}.$$

Dla dowolnych macierzy $\mathbf{A} \in \mathbf{M}_{m \times k}(\mathbb{R})$ i $\mathbf{B} \in \mathbf{M}_{k \times n}(\mathbb{R})$ prawdziwy jest wzór:

$$(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T.$$

Stosując go po lewej stronie równania, otrzymujemy:

$$\mathbf{A}^T + \mathbf{A}\mathbf{X}^T \mathbf{B}^T = \mathbf{A}(\mathbf{X}^T + 3\mathbf{I})\mathbf{B}.$$

Następnie wykonujemy mnożenie po prawej stronie, pamiętając o konieczności zachowania kolejności działań, ponieważ mnożenie macierzy nie jest przemienne:

$$\mathbf{A}^T + \mathbf{A}\mathbf{X}^T \mathbf{B}^T = \mathbf{A}\mathbf{X}^T \mathbf{B} + \mathbf{A} \cdot 3\mathbf{I} \cdot \mathbf{B}.$$

Mnożenie przez macierz jednostkową $\mathbf{I}$ nie zmienia wyniku działania, natomiast mnożenie macierzy przez skalar jest przemienne, zatem możemy zapisać:

$$\mathbf{A}^T + \mathbf{A}\mathbf{X}^T \mathbf{B}^T = \mathbf{A}\mathbf{X}^T \mathbf{B} + 3\mathbf{AB}.$$

Przenosimy wszystkie wyrazy zawierające niewiadomą $\mathbf{X}$ na lewą stronę równania, a dane – na prawą:

$$\mathbf{A}\mathbf{X}^T \mathbf{B}^T - \mathbf{A}\mathbf{X}^T \mathbf{B} = 3\mathbf{AB} - \mathbf{A}^T.$$

Wyłączamy macierz $\mathbf{A}\mathbf{X}^T$ przed nawias z lewej strony:

$$\mathbf{A}\mathbf{X}^T (\mathbf{B}^T - \mathbf{B}) = 3\mathbf{AB} - \mathbf{A}^T.$$

Mnożymy równanie obustronnie przez macierz $\mathbf{A}^{-1}$ z lewej strony oraz przez macierz $(\mathbf{B}^T - \mathbf{B})^{-1}$ z prawej strony:

$$\mathbf{X}^T = \mathbf{A}^{-1} \cdot (3\mathbf{AB} - \mathbf{A}^T) \cdot (\mathbf{B}^T - \mathbf{B})^{-1}.$$

Korzystając ze wzoru $(\mathbf{X}^T)^T = \mathbf{X}$, otrzymujemy rozwiązanie równania:

$$\mathbf{X} = \left[\mathbf{A}^{-1} \cdot (3\mathbf{AB} - \mathbf{A}^T) \cdot (\mathbf{B}^T - \mathbf{B})^{-1} \right]^T.$$

Obliczenia wykonujemy w systemie **SageMath**:

```
A = matrix( [[2, 1], [0, 3]] )
B = matrix( [[0, 1], [-1, -1]] )
Id = identity_matrix(2)
X = ( A^-1 * (3*A*B - A.T) * (B.T - B)^-1 ).T
X
Out: [ -7/4   2]
     [-5/12 -5/3]
```

Możemy również skorzystać bezpośrednio z funkcji **solve()** dla początkowego równania:

```
var('a b c d')
X = matrix( [[a, b], [c, d]] )

L = A.T + A*(B*X).T
P = A * (X.T + 3*Id) * B
solve( (L-P).list(), [a, b, c, d] )
Out: [[a == (-7/4), b == 2, c == (-5/12), d == (-5/3)]]
```

Zadanie 4.9

Dana jest macierz:

$$\mathbf{A} = \begin{bmatrix} 3 & a & 2 \\ -1 & 2 & -2 \\ 1 & -1 & 1 \end{bmatrix}.$$

Wyznacz wszystkie wartości parametru $a \in \mathbb{R}$, dla których zachodzi równość:

$$\det \mathbf{A}^6 = 4 \cdot \det \mathbf{A}^2.$$

Obliczamy wyznacznik macierzy $\mathbf{A}$, korzystając z reguły Sarrusa:

$$\det \mathbf{A} = (6 - 2a + 2) - (4 - a + 6) = -a - 2.$$

Na podstawie wzoru (3.7), występującego w twierdzeniu Cauchy'ego, otrzymujemy:

$$\det \mathbf{A}^2 = \det(\mathbf{A} \cdot \mathbf{A}) = \det \mathbf{A} \cdot \det \mathbf{A} = (\det \mathbf{A})^2.$$

Analogicznie możemy pokazać, że zachodzi równość:

$$\det \mathbf{A}^6 = (\det \mathbf{A})^6.$$

Wobec tego należy rozwiązać równanie:

$$(\det \mathbf{A})^6 = 4 \cdot (\det \mathbf{A})^2 \rightarrow (-a - 2)^6 = 4 \cdot (-a - 2)^2.$$

Wyrażenia w nawiasach są podniesione do potęgi parzystej, zatem możemy zmienić znak:

$$(a + 2)^6 = 4 \cdot (a + 2)^2.$$

Przenosimy wszystkie wyrazy na jedną stronę i wyłączamy czynnik $(a + 2)^2$ przed nawias:

$$(a + 2)^2 \cdot [(a + 2)^4 - 4] = 0,$$

a stąd otrzymujemy alternatywę dwóch równań:

$$(a + 2)^2 = 0 \quad \vee \quad (a + 2)^4 - 4 = 0.$$

Pierwsze równanie ma jedno rozwiązanie, $a = -2$, natomiast drugie zapisujemy w postaci:

$$(a + 2)^4 = 4 \rightarrow (a + 2)^2 = 2.$$

Uzyskane równanie kwadratowe ma dwa rozwiązania:

$$a + 2 = \sqrt{2} \quad \vee \quad a + 2 = -\sqrt{2}.$$

Ostatecznie zapisujemy szukane wartości parametru:

$$a \in \{-2 - \sqrt{2}, -2, -2 + \sqrt{2}\}.$$

W **Sage** wystarczy obliczyć wyznacznik macierzy **A** oraz zastosować funkcję **solve()**. Ponadto wewnątrz funkcji **var()** określamy za pomocą argumentu **domain**, że parametr a może przyjmować tylko wartości rzeczywiste:

```
var('a', domain='real')
A = matrix( [[3, a, 2], [-1, 2, -2], [1, -1, 1]] )

dA = det(A)
solve( dA^6 == 4*dA^2, a )
Out: [a == -sqrt(2) - 2, a == sqrt(2) - 2, a == -2]
```

4.2. Zadania do samodzielnego rozwiązania

Zadanie 4.10

Zbiór $X = \mathbb{R} \setminus \{-1\}$ jest grupą z działaniem:

$$x \circ y \stackrel{\text{def}}{=} 2xy + 2x + 2y + 1, \quad \text{dla } x, y \in X,$$

gdzie po prawej stronie występuje standardowe dodawanie i mnożenie liczb rzeczywistych. Oblicz:

$$2^{-1} \circ (-3) \circ \frac{1}{4}.$$

Zadanie 4.11

Wykonaj poniższe polecenia:

a) Oblicz wartość wyrażenia w ciele $\mathbb{Z}_{137}$:

$$62^3 + 17 \cdot 5^{-4} + 41^{-1}.$$

b) Rozwiąż równanie w ciele $\mathbb{Z}_{89}$:

$$39x + 57 = 83.$$

c) Podaj liczbę reszt kwadratowych modulo 71.

d) Narysuj tabelę działań addytywnego i multiplikatywnego w ciele $\mathbb{Z}_{23}$.

Zadanie 4.12

W pierścieniu $\mathbb{Z}_{20}$:

a) Wyznacz wszystkie rozwiązania równania kwadratowego:

$$x^2 + 3x = 10.$$

b) Rozwiąż równanie wielomianowe czwartego stopnia:

$$2x^4 + 3x^2 = 5x - x^3 - 6.$$

c) Znajdź wszystkie dzielniki zera.

d) Wyznacz wszystkie elementy odwracalne, czyli takie liczby $x \in \mathbb{Z}_{20}$, dla których istnieje element $y \in \mathbb{Z}_{20}$ spełniający równość $xy = 1 \pmod{20}$.

Zadanie 4.13

Wyznacz takie wartości parametrów $k, \ell, m \in \mathbb{N}$, dla których permutacja

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ k & 8 & 1 & \ell & 9 & 2 & m & 6 & 5 \end{pmatrix}$$

ma 19 inwersji. Następnie znajdź permutację $\chi \in S_9$ spełniającą równanie:

$$\sigma^{-20} \circ \chi \circ \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 7 & 3 & 5 & 8 & 1 & 9 & 4 & 6 \end{pmatrix}.$$

Zadanie 4.14

Dane są macierze:

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & 7 \\ -1 & -2 & -1 \\ 5 & -3 & 4 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} -2 & 3 & 0 \\ 1 & -5 & -1 \\ -1 & 0 & 2 \end{bmatrix}.$$

- Wypisz drugi wiersz macierzy $\mathbf{A}$ oraz macierz uzyskaną po wykreśleniu pierwszej kolumny macierzy $\mathbf{B}$.
- Wykonaj działania $\mathbf{A}^T \mathbf{B}^{-1}$ oraz $\mathbf{BA} + 4\mathbf{I}$, gdzie $\mathbf{I}$ jest macierzą jednostkową.
- Połącz macierz $\mathbf{A}$ z macierzą $\mathbf{B}$ w pionie i oblicz rząd uzyskanej macierzy.
- Zamień dwa pierwsze wiersze macierzy $\mathbf{A}$ i pomnóż trzeci wiersz tej macierzy przez 2.

Zadanie 4.15

Dany jest układ równań:

$$\begin{cases} -x + y = 1 \\ 2x - y + z = -3 \\ -x + 2y + z = 0. \end{cases}$$

- Zapisz macierz $\mathbf{A}$ tego układu, wektor wyrazów wolnych $\mathbf{b}$ i macierz uzupełnioną $\mathbf{U}$.
- Korzystając z twierdzenia Kroneckera-Capellego, określ liczbę rozwiązań układu oraz liczbę parametrów, od których rozwiązania zależą.
- Znajdź wszystkie rozwiązania tego układu równań.
- Przedstaw zbiór rozwiązań układu w przestrzeni trójwymiarowej.

Zadanie 4.16

Określ liczbę rozwiązań układu równań w zależności od wartości parametru $a \in \mathbb{R}$:

$$\begin{cases} ax + ay + 3z = 2a \\ -2x + y + 2z = -2a \\ x + ay - z = a. \end{cases}$$

Zadanie 4.17

Rozwiąż układ równań liniowych:

$$\begin{cases} -6x + 8y - z + 7t = 1 \\ 4x - 2y - 9z + 3t = 2 \\ -5x + 2y + 3z - t = -1 \\ 5x + 6y - 2z + 5t = 3, \end{cases}$$

korzystając ze wzorów Cramera. Następnie sprawdź poprawność otrzymanego rozwiązania za pomocą funkcji `solve_right()`.

Zadanie 4.18

a) Dane są macierze:

$$\mathbf{A} = \begin{bmatrix} -2 & 1 \\ -5 & -1 \end{bmatrix} \quad \text{oraz} \quad \mathbf{B} = \begin{bmatrix} 1 & -8 \\ -3 & -4 \end{bmatrix}.$$

Rozwiąż równanie macierzowe poprzez wyznaczenie niewiadomej $\mathbf{X}$ z równania:

$$\mathbf{I} - 3\mathbf{A}\mathbf{X} = 2\mathbf{B}\mathbf{X} + \mathbf{A}^T.$$

b) Znajdź wszystkie macierze $\mathbf{X}$ spełniające równanie:

$$\begin{bmatrix} 4 & -1 \\ 2 & 3 \end{bmatrix} \cdot \mathbf{X} \cdot \begin{bmatrix} 2 & -1 & -1 \\ 0 & 3 & 1 \end{bmatrix} = \begin{bmatrix} 16 & -29 & -15 \\ 8 & 17 & 3 \end{bmatrix}.$$

c) Rozwiąż równanie zawierające wyznacznik:

$$\begin{vmatrix} 0 & -1 & 0 & 2 \\ -1 & 1 & -1 & 0 \\ 0 & -1 & x & 1 \\ 2 & 0 & 1 & x^2 \end{vmatrix} = x^2 + 5.$$

d) Dana jest macierz blokowa:

$$\mathbf{M} = \begin{bmatrix} \mathbf{X} & \mathbf{A} \\ \mathbf{B} & \mathbf{C} \end{bmatrix},$$

gdzie:

$$\mathbf{A} = \begin{bmatrix} 0 & 2 \\ -1 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} -1 & 0 \\ 2 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 3 & 2 \\ 4 & -1 \end{bmatrix}.$$

Znajdź wszystkie macierze rzeczywiste $\mathbf{X}$, dla których spełniona jest równość:

$$\mathbf{M} \cdot \begin{bmatrix} 1 \\ 3 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ 0 \\ -1 \end{bmatrix}.$$

4.3. Odpowiedzi do zadań

Zadanie 4.10

$$2^{-1} \circ (-3) \circ \frac{1}{4} = -\frac{11}{6}.$$

Zadanie 4.11

a) 77; b) $x = 60$; c) 36.

Zadanie 4.12

a) $x \in \{2, 7, 10, 15\}$; b) $x = 14$;
c) $\{2, 4, 5, 6, 8, 10, 12, 14, 15, 16, 18\}$; d) $\{1, 3, 7, 9, 11, 13, 17, 19\}$.

Zadanie 4.13

$$k = 7, \ell = 4, m = 3; \chi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 7 & 3 & 9 & 5 & 8 & 4 & 6 & 1 & 2 \end{pmatrix}.$$

Zadanie 4.14

a) $\begin{bmatrix} -1 & -2 & -1 \end{bmatrix}, \begin{bmatrix} 3 & 0 \\ -5 & -1 \\ 0 & 2 \end{bmatrix};$

b) $\mathbf{A}^T \mathbf{B}^{-1} = \begin{bmatrix} -\frac{54}{17} & -\frac{29}{17} & \frac{28}{17} \\ -\frac{3}{17} & \frac{5}{17} & -\frac{23}{17} \\ -\frac{89}{17} & -\frac{50}{17} & \frac{9}{17} \end{bmatrix}, \mathbf{BA} + 4\mathbf{I} = \begin{bmatrix} -5 & -10 & -17 \\ 3 & 19 & 8 \\ 7 & -8 & 5 \end{bmatrix};$

c) $\mathbf{C} = \begin{bmatrix} 3 & 2 & 7 \\ -1 & -2 & -1 \\ 5 & -3 & 4 \\ -2 & 3 & 0 \\ 1 & -5 & -1 \\ -1 & 0 & 2 \end{bmatrix}, r(\mathbf{C}) = 3;$

d) $\begin{bmatrix} -1 & -2 & -1 \\ 3 & 2 & 7 \\ 10 & -6 & 8 \end{bmatrix}.$

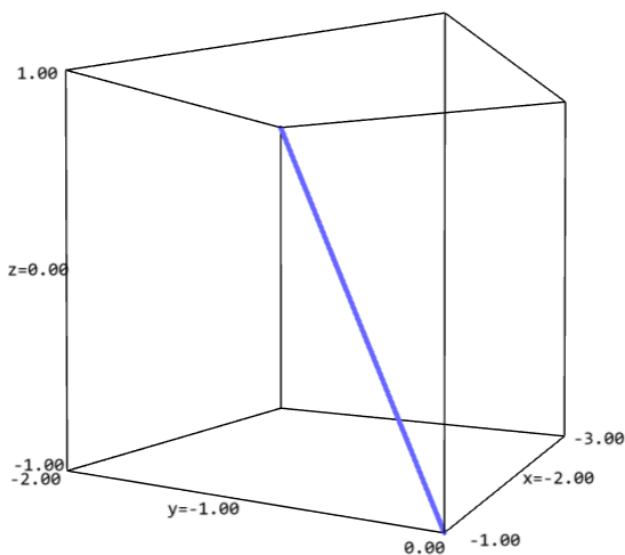
Zadanie 4.15

a) $\mathbf{A} = \begin{bmatrix} -1 & 1 & 0 \\ 2 & -1 & 1 \\ -1 & 2 & 1 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 1 \\ -3 \\ 0 \end{bmatrix}$, $\mathbf{U} = \left[\begin{array}{ccc|c} -1 & 1 & 0 & 1 \\ 2 & -1 & 1 & -3 \\ -1 & 2 & 1 & 0 \end{array} \right]$;

b) układ ma nieskończenie wiele rozwiązań zależnych od 1 parametru;

c)
$$\begin{cases} x = -2 - t \\ y = -1 - t \\ z = t, \end{cases} \quad t \in \mathbb{R};$$

d)



Zadanie 4.16

Układ ma dokładnie jedno rozwiązanie dla $a \in \mathbb{R} \setminus \left\{ -3, -\frac{1}{2} \right\}$.

Układ jest sprzeczny dla $a = -3$.

Układ ma nieskończenie wiele rozwiązań zależnych od 1 parametru dla $a = -\frac{1}{2}$.

Zadanie 4.17

$$\text{Rozwiązanie: } \begin{cases} x = \frac{31}{151} \\ y = \frac{95}{302} \\ z = -\frac{34}{151} \\ t = -\frac{11}{151}. \end{cases}$$

Zadanie 4.18

$$\text{a) } \mathbf{X} = \begin{bmatrix} \frac{46}{229} & \frac{29}{229} \\ -\frac{67}{229} & -\frac{97}{229} \end{bmatrix}; \quad \text{b) } \mathbf{X} = \begin{bmatrix} 2 & -1 \\ 0 & 3 \end{bmatrix};$$

$$\text{c) } x \in \{-1, -2i, 2i\}; \quad \text{d) } \mathbf{X} = \begin{bmatrix} -3t & t \\ -5-3s & s \end{bmatrix}, \quad t, s \in \mathbb{R}.$$

Dookoła jest pięknie! 😊

Dodatek A.

Spis funkcji występujących w książce

Nazwa funkcji	Opis
AlternatingGroup()	grupa alternująca (permutacji parzystych)
append()	dodaje element do listy
assume()	ustawia założenie o zmiennej
augment()	łączy macierze w poziomie
block_matrix()	macierz blokowa
copy()	tworzy kopię obiektu
crt()	rozwiązuje układ kongruencji
delete_columns()	usuwa kolumny o podanych indeksach
delete_rows()	usuwa wiersze o podanych indeksach
det()	wyznacznik macierzy
diagonal_matrix()	macierz diagonalna
divmod()	zwraca iloraz i resztę z dzielenia
echelon_form()	postać schodkowa macierzy (funkcja nieinwazyjna)
echelonize()	postać schodkowa macierzy (funkcja inwazyjna)
euler_phi()	funkcja Eulera
factor()	rozkład liczby na czynniki pierwsze
gcd()	największy wspólny dzielnik (NWD)
identity()	permutacja identycznościowa
identity_matrix()	macierz jednostkowa
Integers()	pierścień liczb całkowitych modulo n
inverse()	permutacja odwrotna
inversions()	inwersje permutacji
is_even()	sprawdza, czy permutacja jest parzysta
is_prime()	sprawdza, czy liczba jest pierwsza
LatexExpr()	konwertuje tekst na zapis w języku LaTeX
lcm()	najmniejsza wspólna wielokrotność (NWW)
len()	długość listy
list()	rzutowanie na listę

LU()	dekompozycja LU macierzy
matrix()	tworzy macierz
mod()	operacja modulo
ncols()	liczba kolumn macierzy
next_prime()	następna liczba pierwsza
number_of_inversions()	liczba inwersji permutacji
OperationTable()	tabela działania w grupie
order()	rzęd permutacji
parametric_plot()	rysuje wykres parametryczny
Permutation()	tworzy permutację
PermutationGroupElement()	tworzy permutację i umożliwia potęgowanie
Permutations()	wszystkie permutacje danej listy
plot()	rysuje wykres funkcji
point()	rysuje pojedynczy punkt
previous_prime()	poprzednia liczba pierwsza
prime_factors()	lista wszystkich czynników pierwszych
prime_range()	liczby pierwsze z danego zakresu
print()	wypisuje na ekran
random_matrix()	losowa macierz
range()	zakres liczb całkowitych
rank()	rzęd macierzy
reversed()	odwraca kolejność elementów listy
rhs()	prawa strona równości
rref()	zredukowana postać schodkowa macierzy
show()	wyświetla obiekt
sign()	znak permutacji
solve()	rozwiązuje równanie, nierówność lub układ równań
solve_mod()	rozwiązuje równanie modulo n
solve_right()	rozwiązuje układ równań w postaci macierzowej
sqrt()	pierwiastek kwadratowy
stack()	łączy macierze w pionie
swap_rows()	zamienia wiersze
SymmetricGroup()	grupa symetryczna (permutacji)
to_cycles()	cykle rozłączne permutacji
var()	deklaruje zmienne symboliczne
vector()	tworzy wektor
xgcd()	rozszerzony algorytm Euklidesa
zero_matrix()	macierz zerowa
Zmod()	pierścień $\mathbb{Z}_n$ (alias funkcji Integers())

Literatura

1. Bard G.V., *Sage for undergraduates*, American Mathematical Society, Providence 2015.
2. Jurlewicz T., Skoczylas Z., *Algebra i geometria analityczna. Definicje, twierdzenia, wzory*, Oficyna Wydawnicza GiS, Wrocław 2024.
3. Topp J., *Algebra liniowa*, Wydawnictwo Uniwersytetu Gdańskiego, Gdańsk 2015.
4. Sage Documentation, strona internetowa, <https://doc.sagemath.org> [dostęp: 20.04.2026].
5. Sierpiński W., *Teoria liczb*, Państwowe Wydawnictwo Naukowe, Warszawa 1959.

Spis tabel

Tabela 2.1. Przykłady grup nieskończonych.....	14
Tabela 2.2. Funkcje w Sage do obliczeń na liczbach pierwszych.....	31

Spis rysunków

Rysunek 2.1. Interpretacja graficzna rozwiązań równania diofantycznego	36
Rysunek 2.2. Interpretacja geometryczna inwersji – zielone połączenia uzyskiwane są ze wzoru permutacji.....	44
Rysunek 3.1. Schemat mnożenia macierzy	64
Rysunek 3.2. Przykład postaci schodkowej macierzy o wymiarach 4×6	74
Rysunek 4.1. Rozwiązania układu równań w przestrzeni trójwymiarowej.....	135

Streszczenie

Książka stanowi drugą część serii poświęconej algebrze w systemie **SageMath** i obejmuje zagadnienia dotyczące teorii grup, permutacji oraz macierzy. Na początku przedstawiono definicje działania i grupy, a także sposoby wyznaczania elementu neutralnego i elementów odwrotnych. Następnie omówiono pojęcia ciała i pierścienia oraz przedstawiono najważniejsze własności wyróżniające te struktury algebraiczne. Uzupełnieniem tych treści są przykłady typowych zagadnień pojawiających się w teorii liczb. Wszystkie omawiane pojęcia i techniki zilustrowano licznymi przykładami, które umożliwiają stopniowe przyswajanie materiału i samodzielne tworzenie nowych rozwiązań.

W kolejnych rozdziałach przedstawiono podstawy teorii macierzy i układów równań liniowych. Omówiono pojęcie wyznacznika, jego własności oraz zastosowania – m.in. do wyznaczania rzędu macierzy, znajdowania macierzy odwrotnej i badania liczby rozwiązań układów liniowych. Zaprezentowano także metody rozwiązywania równań macierzowych oraz twierdzenia opisujące zależności między strukturą macierzy a własnościami odpowiadających im układów. Materiał przedstawiono za pomocą praktycznych przykładów, które prezentują zarówno ręczne rozwiązanie konkretnych zadań, jak i ich implementację w języku **Sage**.

Abstract

This book is the second part of a series on algebra in **SageMath** and focuses on topics related to group theory, permutations and matrices. It begins with definitions related to groups and operations, as well as methods for determining the identity element and inverse elements. It then introduces the concepts of a field and a ring, presenting the key properties that distinguish these algebraic structures. These topics are complemented by examples of typical problems arising in number theory. All concepts and techniques are illustrated with numerous examples, enabling gradual learning and independent development of new solutions.

The following chapters presents the basics of matrix theory and systems of linear equations. They discuss the concept of the determinant, its properties, and its applications – including determining the rank of a matrix, finding the inverse matrix, and exploring the number of solutions of linear systems. Methods for solving matrix equations are also presented, along with theorems describing relationships between the structure of a matrix and the properties of the corresponding systems. The material is presented through practical examples that demonstrate both manual solutions to specific problems and their implementation in **Sage**.



Politechnika
Białostocka