МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ
УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Р. В. Сачок

МЕТОДИ КОМП'ЮТЕРНОГО РОЗРАХУНКУ

ТЕОРІЯ І ПРАКТИЧНІ ЗАВДАННЯ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для студентів,
які навчаються за спеціальністю 133 «Галузеве машинобудування»,
спеціалізації «Інжиніринг, комп'ютерне моделювання та проектування обладнання
хімічних і нафтопереробних виробництв»

Київ
КПІ ім. Ігоря Сікорського
2018

Рецензент: *Черьопкін Євгеній Сергійович,* канд. техн. наук, асистент

Відповідальний редактор *Корнієнко Ярослав Микитович*, д-р техн. наук, проф.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 10 від 21.06.2018 р.)
за поданням Вченої ради ІХФ (протокол № 5 від 29.05.2018 р.)

Електронне мережне навчальне видання

Сачок Роман Володимирович, канд. техн. наук.

МЕТОДИ КОМП'ЮТЕРНОГО РОЗРАХУНКУ
ТЕОРІЯ І ПРАКТИЧНІ ЗАВДАННЯ

Методи комп'ютерного розрахунку: теорія і практичні завдання [Електронний ресурс]: навч. посіб. для студ. спеціальності 133 «Галузеве машинобудування», спеціалізації «Інжиніринг, комп'ютерне моделювання та проектування обладнання хімічних і нафтопереробних виробництв» / Р.В. Сачок. – Електронні текстові данні (1 файл: 2,00 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 141 с.

Навчальний посібник присвячений застосуванню комп'ютерної техніки для розрахунків та моделювання. Розглянуті основні питання, пов'язані з розв'язком алгебраїчних й трансцендентних рівнянь та систем рівнянь, викладено основи складання алгоритмів та програмування у середовищах MathCad та MatLab. Також розглянуті питання застосування графіки у середовищах MathCad та MatLab для розв'язку алгебраїчних рівнянь та систем алгебраїчних рівнянь. Розглянуті методи розв'язку визначених і невизначених інтегралів у середовищі MathCad, розглянуті основні принципи застосування функцій середовища MatLab для використання їх при розв'язку диференціальних рівнянь першого, другого порядку та вищих порядків.

ЗМІСТ

ВСТУП.....4
1 ВИКОРИСТАННЯ СЕРЕДОВИЩА MATHCAD.....5
1.1 Загальні характеристики та робоче вікно MathCAD.....5
1.2 Елементи MathCAD.....10
1.3 Форматування чисел.....17
1.4 Робота з текстом.....18
1.5 Робота з графіками.....19
1.6 Способи розв’язку рівнянь в MathCAD.....28
1.7 Розв’язок систем рівнянь.....33
1.8 Програмування в пакеті MathCAD.....38
1.9 Індивідуальні завдання для розв’язку в середовищі MathCAD.....46
2 ВИКОРИСТАННЯ СЕРЕДОВИЩА MATLAB.....62
2.1 Команди, функції пакету MATLAB.....62
2.2 Програмування в пакеті MATLAB. Умовні переходи, цикли, перемикачі.....86
2.3 Спеціальні можливості при зверненні до функцій.....94
2.4 Графічні функції пакету MATLAB.....102
2.5 Числовий розв'язок задачі Коші для звичайних диференціальних рівнянь.....124
2.6 Індивідуальні завдання для розв’язку в середовищі MatLab.....135
Перелік посилань.....140

ВСТУП

Впровадження у виробництво наукових досліджень складається, у тому числі, й зі створення конструкторської документації. Розробка таких документів є творчим процесом,що вимагає від конструктора не тільки глибоких знань дисциплін, що викладаються у ВИШі, а й уміння використовувати їх при проектуванні обладнання та проведенні його розрахунків, що підтверджують працездатність та надійність устаткування.

Для розрахунків, які необхідні для проектування обладнання, в рівній мірі як і для розрахунків математичних моделей процесів хімічної технології існують сучасні середовища MathCAD і MatLab.

MathCAD є унікальною системою для проведення обрахунків за визначеними виразами, з текстовою інформацією та графіками, призначений для вирішення різноманітних задач. Також він є інструментом для проведення наукових і технічних розрахунків для вчених і фахівців у всьому світі.

Поява навчального посібника викликана, з одного боку, потребою в компактному викладі первинних відомостей про інтерфейс і мови програмування MatCAD і MATLAB. Ця допомога орієнтована на використання його студентами-механіками ОКР бакалавр на початкових стадіях своєї наукової діяльності.

Автор прагнув подати матеріал так, щоб посібник можна було використовувати як самовчитель. При цьому відзначено особливості мови, функції бібліотеки стандартних програм і використаних в ній обчислювальних методів, не очевидні для недосвідченого читача.

Від читача посібника, що намагається реалізувати отримані відомості в своїй практичній діяльності, не вимагається попереднього знання будь-яких мов програмування. Проте, автор припускав, що читач добре знайомий з традиційним дизайном вікон Windows і вміє ними користуватися.

Матеріал посібника викладено за авторами [1…23].

1 ВИКОРИСТАННЯ СЕРЕДОВИЩА MATHCAD

1.1 Загальні характеристики та робоче вікно MathCAD

Основні характеристики системи MathCAD

– MathCAD дозволяє формувати записи формул за доповмогою комп’ютерної техніки в звичному для користувача вигляді.

– Застосовувати символьні перетворення для обчислень, розраховувати похідні математичних й трансцендентних функцій, обраховувати визначені і невизначені інтеграли, розв’язувати рівняння і системи рівнянь, а також працювати х матрицями та векторами за правилами матричної алгебри.

– Середовище має можливість також програмування у ньому, реалізуючи конструкції умовних операторів, циклів, підпрограм тощо.

– Будувати двовимірні і тривимірні графіки для розв’язку рівнянь та систем рівнянь, а саме для виділення початкового наближення коренів.

Елементи вікна MathCAD

До елементів вікна MathCAD відносяться, рисунок 1.1 [1-4]:

– Рядок заголовка, який містить назву програми та ім'я відкритого документа. При збереженні документа середовища за замовчуванням до заданого імені файлу додає розширення **MCD**.

– Головне меню.

– Панелі інструментів.

– Вікно робочого аркуша, в якому й формується майбутній документ., лінійки прокручування.

– Рядок стану. На початку рядка зазначено призначення активного пункту меню.

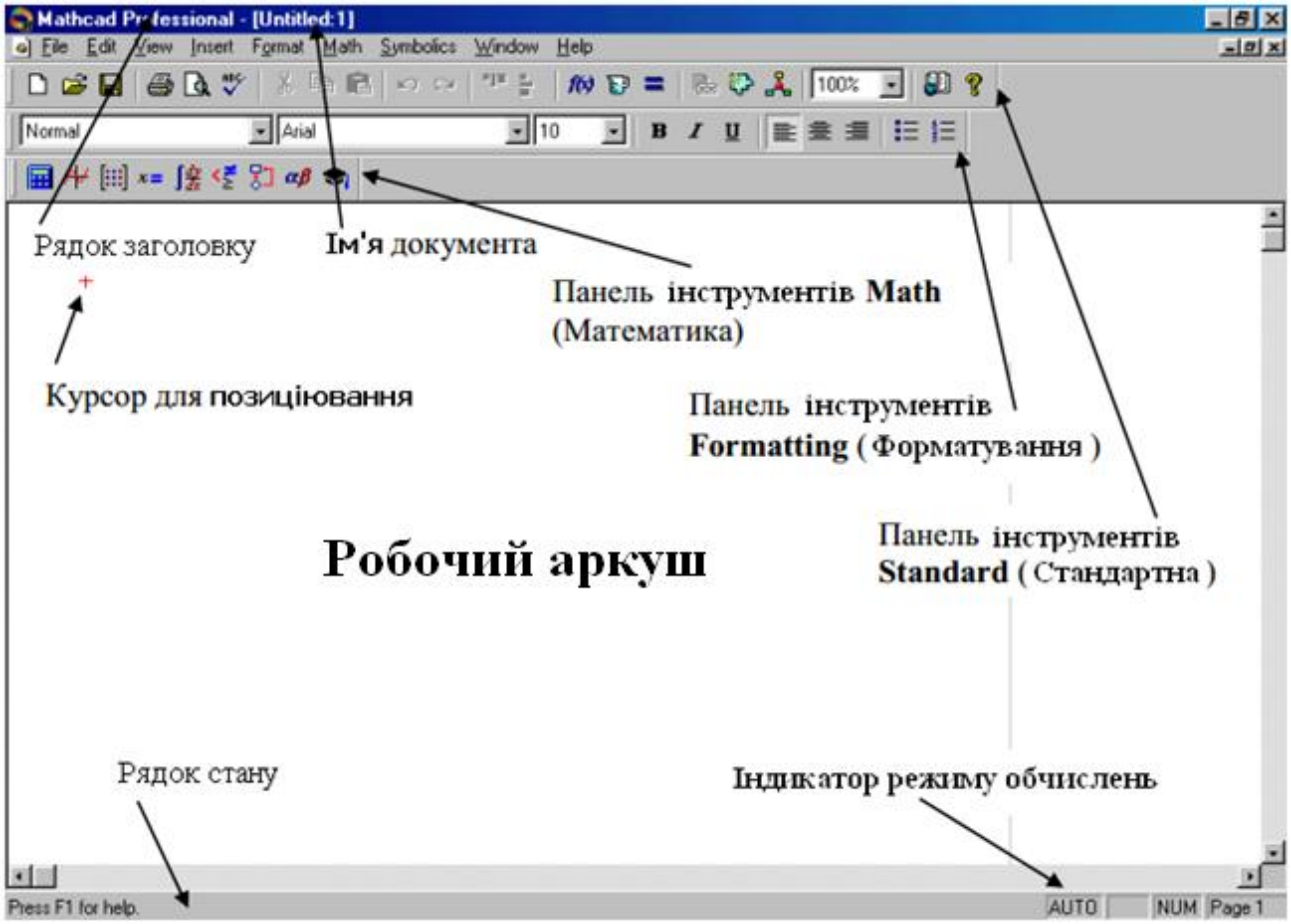


Рисунок 1.1 – Елементи вікна MathCAD

Головне меню: містить команди, які дають змогу створювати та редагувати вихідний документ, керувати обчисленнями та параметрами середовища, що можуть бути налаштовані

Призначення пунктів меню дається в таблиці 1.1.

Команди, об'єднані в перерахованих групах, можуть бути активізовані за допомогою натискання відповідних кнопок, розміщених на панелі інструментів.

Таблиця 1.1 – Структура головного меню

Пункт меню	Призначення
File (Файл)	Команди роботи з файлами
Edit (Правка)	Команди редагування документу
Format (Формат)	Команди, які дозволяють задавати різноманітні параметри, що визначають зовнішній вид чисел, формул, тексту, колонтитулів, тощо.
Insert (Вставка)	Команди вставки в MathCAD–документ: графіків, матриць, функцій та інших об'єктів.
View (Вид)	Команди управління елементами екрану (панель інструментів, рядок стану, та інші).
Math (Математика)	Команди керування режимом обчислень та зміни параметрів обчислень.
Symbolics (Символи)	Команди символьних розрахунків.
Window (Вікно)	Команди для роботи з вікнами.
Help (Допомога)	Команди, що забезпечують доступ до довідкової системи.

1.1.1 Панелі інструментів

До основних панелямів інструментів відноситься:

- панель **Standard (Стандартні)**;
- панель **Formatting (Форматування)**;
- панель **Math (Математика)**.

Режим зображення панелей інструментів може бути скорегований командою **View (Вид)** з подальшим вибором необхідного пункту меню [2].

Рекомендується завжди мати включеними всі три перераховані панелі інструментів, рисунок1.1.

На перших двох панелях розміщено кнопки, які призначені, як і в будь-якому Windows-додатку, рисунок 1.2, вказано призначення деяких кнопок, тому докладно буде розглянуто третю панель **Math**.

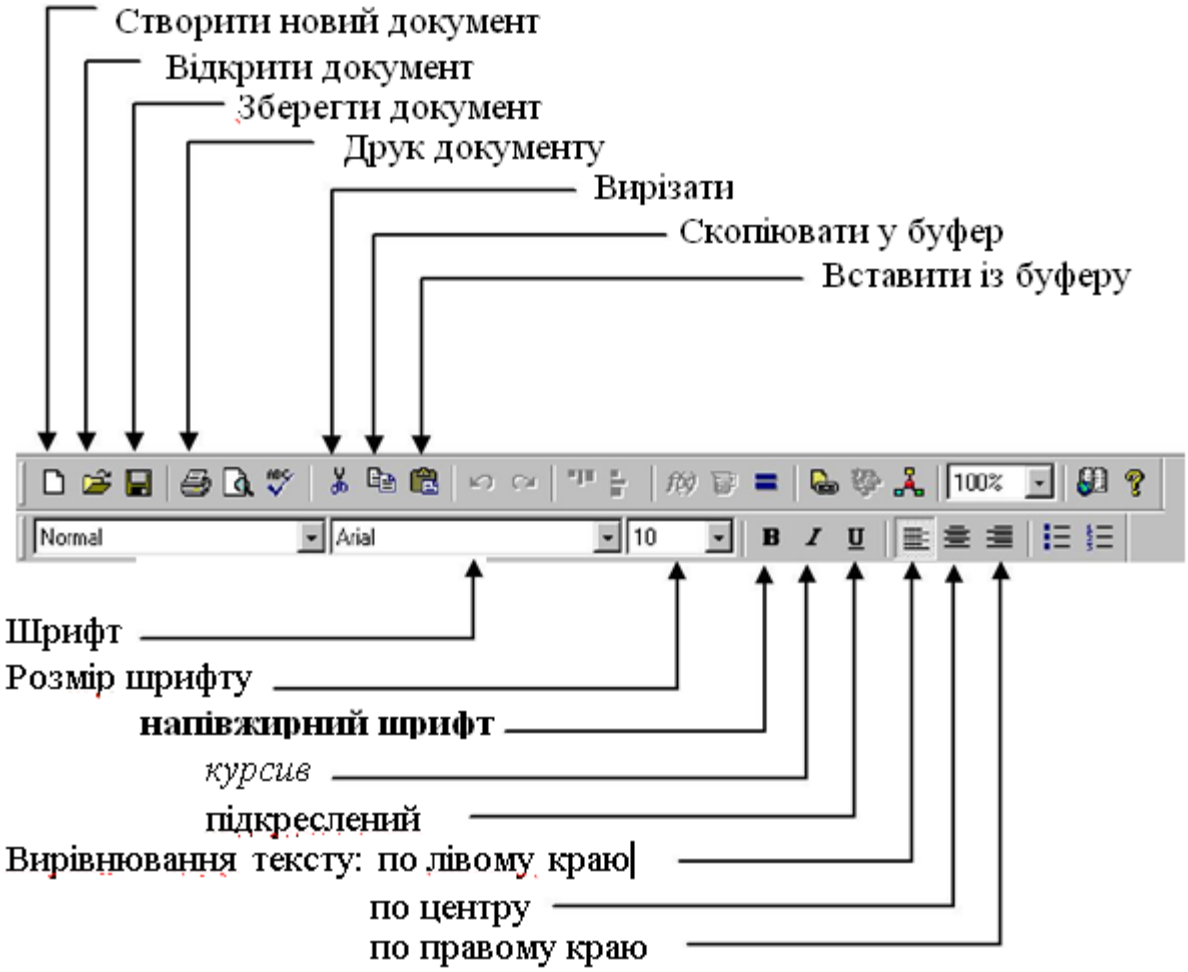
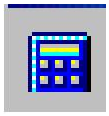
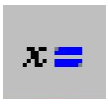
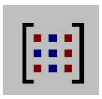


Рисунок 1.2 – Склад панелей **Standard** і **Formatting**

Панель інструментів Math (Математика)

Панель **Math** містить кнопки для відтворення панелей інструментів, які необхідні для роботи з формулами, графіками, виконання символьних операцій. Характеристики кнопок панелі – таблиця 1.2. [4]

Таблиця 1.2 – Панель інструментів **Math**

Піктограма	Назва панелі	Призначення панелі
	**Calculator (Калькулятор)**	Робота з арифметичними операціями, деякі часто використовувані функції, оператори присвоювання й виведення обрахованого результату.
	**Boolean (Булева, Логічна)**	Логічних оператори та оператори порівняння.
	**Evaluation (Визначення, Обчислення,)**	Оператори локального і глобального присвоювання.
	**Graph (Графіка)**	Робота з дво- і тривимірними графіками.
	**Matrix (Матриці)**	Робота з векторами, матрицями і операторaторами їх обробки.
	**Calculus (Обчислення)**	Обчислення інтегралів, похідних, сум, добутків і границь, тригонометричних функцій.
	**Greek (Грецький алфавіт)**	Грецькі літери.
	**Symbolic (Символи)**	Використання ключових слів для виконання символьних обрахунків.
	**Programming (Програмування)**	Оператори програмування – створення блоку програми у середовищі.

На рисунку 1.3 показані склад і можливий варіант розміщення перерахованих у таблиці 1.2 панелей.

Також, деякі панелі є можливість розміщувати в тих рядках вікна, де розміщеється основна кількість панелей інструментів, наприклад, рисунок 1.4.

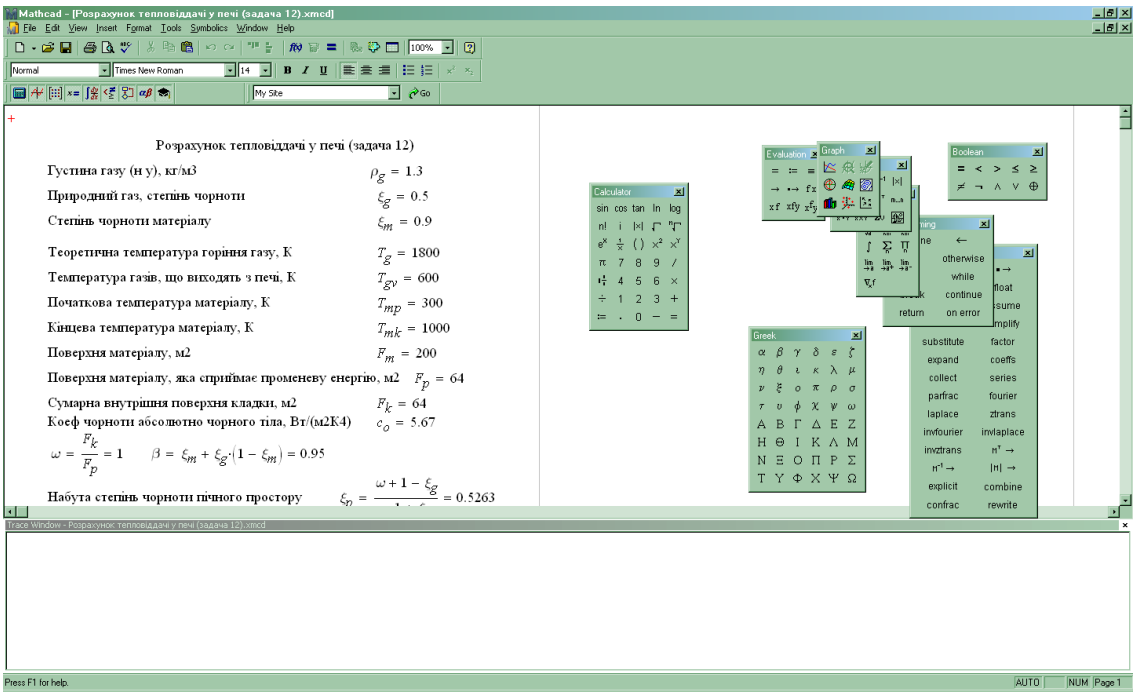


Рисунок 1.3 – Структура основних панелей інструментів **Math**

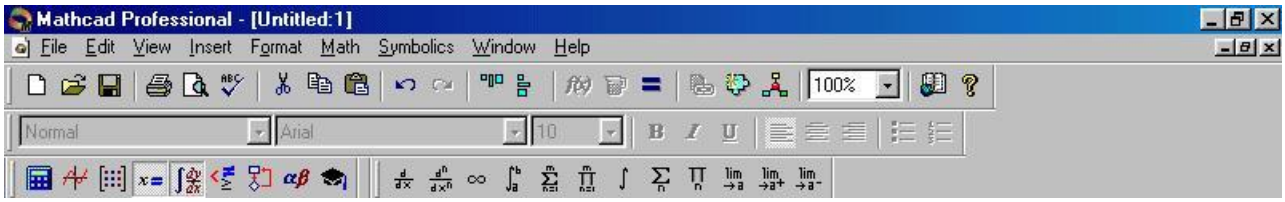


Рисунок 1.4 – Можливий варіант розміщення панелей **Math** і **Calculus**

1.2 Елементи MathCAD

MathCAD-документ складається з окремих областей різних типів, як то, графік, текст, формула, рисунок 1.3. Кожна область прямокутної форми може бути розташована в довільному місці на робочім аркуші. Для перемішення областей використовують курсор (червоного кольору) хрестоподібної форми, рисунок 1.1. Тобто, у області тексту курсор приймає форму вертикальної червоної лінії, тоді як в області формул й графіків – кута синього кольору. Переміщення курсору реалізується клавішами керування курсором або переміщенням покажчика миші з подальшим одноразовим натисканням лівої

кнопки. Активна або поточна область для створення, редагування виділяється прямокутною рамкою.

Основними операндами MathCAD є оператори, константи, змінні, масиви й функції.

1.2.1 Оператори

Операторами називаються елементи середовища, за допомогою яких формуються математичні вирази. До таких відносяться символи арифметичних операцій, знаки обчислення сумування, добутку, похідної, інтегралу, тощо. [1-4]

Запис оператора визначає: дію, яка буде виконана при наявності різних значень операндів; скільки і яких операндів буде введено для виконання оператору.

Операндом назичвається вираз, з яким працює оператор. Наприклад, у виразі 2!+7 числа 2! і 7 – операнди оператора плюс, а число 2 – операнд факторіалу.

Оператор в MathCAD вводиться двома способами:

- натисканням клавіши (комбінації клавіш) на клавіатурі;
- з використанням математичної панелі.

Для присвоювання змінній значення або виразу використовуються такі оператори:

▮ := ▮ – знак присвоювання(клавіша «**:**» на клавіатурі або натисканням кнопки панелі ***Калькулятор***);

Таке присвоювання називається *локальним*, тобто змінній буде присвоєно значення, встановлене в данному місці користувачем. До цього змінна не визначена й не може бути використана.

▪ = ▪ – оператор наближеної рівності. Використовується при розв'язанні рівнянь та систем рівнянь. Вводиться натисканням клавіші «**;**» на клавіатурі або натисканням кнопки на ***Булевій панелі.***

▪ ≡ ▪ – глобальний оператор присвоювання, може бути використано в будь–якому місці документу. Якщо змінна присвоєна таким чином значення в кінці документу, таке ж значення вона буде мати й на початку документу.

Найпростіші обчислення

Процес обчислення відбувається за допомогою таких панелей:

 – панель **Calculator** (**Калькулятор**);

 – панель **Calculus (Обчислення)**;

 – панель **Evaluation** (**Визначення**)

Якщо необхідно розділити весь вираз в чисельнику, його потрібно спочатку виділити, натиснувши пробіл на клавіатурі, або помістити в дужки.

1.2.2 Константи

Константи — іменовані об'єкти, що зберігають значення, що не змінюються.

Наприклад, π = 3.14.

Розмірні константи — це загальноприйняті одиниці вимірювання. Наприклад, метри, секунди, та т. ін.

Щоб записати таку константу, після числа треба ввести знак «*», вибрати в пункті меню **Вставка** підпункт **Юніт**. У вимірюваннях найбільше розповсюджені такі категорії:**Mass** — маса (гр, кг, т); **Time** — час (хв, сек, год), **Length** — довжина (м, км, см). [4,6,10]

1.2.3 Змінні

Змінними називаються ідентифікатори, що містять певні дані, які можуть змінюватися при виконанні програми. Змінні можуть бути числовими, символьними, текстовими, логічними тощо. Значення змінним задають за допомогою присвоювання **(: =)**.

Зауважимо, що в середовищі MathCAD заголовні та прописні літери сприймаються як різні ідентифікатори.

Системні змінні

В MathCAD існує невелика група особливих об'єктів, які неможна віднести ні до класу констант, ні до класу змінних, значення яких визначенні одразу після запуску програми [4]. Їх правильно вважати *системними змінними*. Наприклад, **ORIGIN[0]** – нижня межа значення індексу індексації векторів або матриць. При необхідності цим змінним можна задати інші величини.

Ранжирувані змінні

Ці змінні мають ряд фіксованих значень, які можуть змінюватися із певним кроком, починаючи від початкового значення, й закінчуючи кінцевим.

Для створення ранжируваної змінної необхідно записати:

< ім'я змінної >: =<початкове значення>**,** <початкове значення>+<крок>**..** кінцеве значення

Ранжируванні змінні зазвичай використовуються при побудові графіків. Наприклад, якщо для побудови графіку функції *f(x)* необхідно створити певний ряд значень змінної *x,* для цього вона повинна бути ранжируваною. Якщо в діапазоні змінної не вказувати крок, автоматично він буде дорівнювати одиниці.

Приклад. Змінна *x* змінюється від –5 до +5 з кроком 0.01.

Для щапису ранжируваної змінної необхідно ввести:

– назву змінної (*x*);

– знак присвоювання (:=);

– початкове значення діапазону (–5);

– кому;

– друге значення діапазону– суму першого значення і кроку, тобто –-5+0.1);

– знак "**..**" — задає діапазон змінної в визначених межах (вводиться натисканням крапки з комою англійської розкладки);

– останнє значення діапазону (5).

В результаті отримаємо: *x* := –5,–5+0.1..5.

В MathCad як десятковий роздільник використовується крапка "."

Таблиці виведення

Вираз, що містить ранжирувані змінні після знаку рівності ініціює таблицю виведення.

В таблиці виведення можна вставляти числові значення і корегувати їх.

Змінна з індексом

Змінною з індексом називають змінну, якій присвоєно набір не зв'язаних даних, кожне із яких має свій номер (індекс).

Введення індексу реалізується за допомогою натискання лівої квадратної дужки на клавіатурі або кнопки $\mathbf{x_n}$ на панелі **Калькулятор**.

В якості індексу можна використовувати константу, змінну, вираз. Таким чином може бути реалізовано введення елементів масиву, розділяючи їх комами.

Приклад. Введення індексних змінних.

i:= 1..3 – індекс змінюється від 1 до 3 (змінна буде містити 3 елементи).

$s_i :=$

–2
2
5.75

– введення числових значень в таблицю відбувається через кому;

$s_1 = -2$ – вивід значення нульового елементу вектору S;

$s_2 = 2$ – вивід значення першого елементу вектору S.

1.2.4 Масиви

Масивом називають сукупність даних, які мають унікальне ім’я та відрізняються порядковим номером – індексом.

В пакеті MathCAD використовуються масиви:

– одновимірні (вектори);

– двовимірні (матриці).

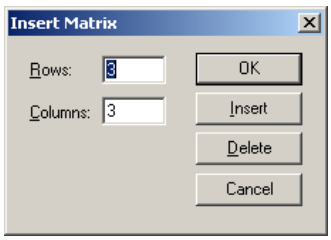
Вивести шаблон вектора або матриці можна таким чином:

– обрати пункт меню **Вставка - Матриця**;

– комбінацію клавіш **Ctrl + M**;

– кнопку  на **Панелі векторів і матриць.**

В результаті з’явиться діалогове вікно, в якому задається необхідна кількість рядків та стовпчиків:

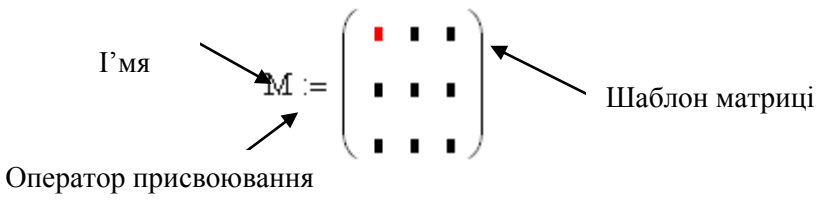


де **Rows** — число рядків матриці

Columns — число стовпців матриці

Якщо матриці або вектору необхідно присвоїти ім’я, на початку вводиться ім’я матриці або вектора, а потім – оператор присвоювання, а в кінці – шаблон матриці.

Наприклад:



Матриця — двовимірний масив $\mathbf{M_{n,m}}$, що містить **n** рядків та **m** стовпчиків.

З матрицями можна виконувати математичні операції, відповідно до правил матричної алгебри.

1.2.5 Функції

Функцією називається вираз, за яким проводяться обчислення з аргументом або аргументами й визначається їх значення. Прикладами функцій можуть **cos(x), tan(x)**, **ln(x)** тощо.

Функції в пакеті можуть бути вбудованими або користувальницькими – визначеними користувачем. Вставити функції можна у такий спосіб:

- Вибрати пункт меню **Вставка – Функція**.
- Натиснуть клавіши **Ctrl + E**.
- Обрати  . на панелі інструментів.
- Набрати ім’я функції на клавіатурі.

Користувальницькі функції зазвичай використовуються для багатократних обчислень одного виразу. Для того, щоб задати користувальництку функцію, необхідно:

- Ввести власне ім’я функції із записом аргумента в дужках, наприклад, *f(x).*
- Ввести оператор присвоювання **(:=)**.
- Ввести вираз для обчислення функції.

Приклад. f*(t)* := cos(2t).

1.3 Форматування чисел

В пакеті MathCAD є можливість змінювати формат виведення даних. Зазвичай обчислення проводяться з точністю до 20 знаків, але виводяться на екран не всі значущі цифри. Щоб змінити формат числа, необхідно два рази натиснути на вибраному результаті. З'явиться вікно форматування, яке відкрите на вкладці ***Number Format*** із наступними форматами:

– ***General*** (**Основний**) – за замовчуванням. Числа відображаються з порядком (наприклад, 1.22×10^5). Число знаків мантиси можна встановити в полі ***Exponential Threshold*** **(Поріг експоненціального представлення)**. При перевищенні цього порогу число відображається з порядком. Число знаків після десяткового знаку (крапки) змінюється в полі ***Number of decimal places***.

– ***Decimal*** (**Десятковий**) – представлення чисел з плаваючим десятковим знаком (наприклад, 12.3542).

– ***Scientific*** (**Науковий**) –відображення чисел тільки з порядком.

– ***Engineering*** (**Інженерний**) – числа відображені з порядком, кратним трьом (наприклад, 1.22×10^6).

Якщо після встановлення необхідного формату у вікні форматування чисел вибрати кнопку **Ок**, формат буде встановлено тільки для виділеного числа, але, якщо обрати кнопку **Set as Default**, формат буде застосований для всіх чисел в документі.

За замовчуванням числа округляються до нуля, якщо вони менше встановленого порогу, який встановлено для всього документу. Щоб змінити поріг округлення до нуля, треба вибрати пункт меню **Форматування – Результат** та у вкладці ***Tolerance***, і в полі, що відкрилося, ***Zero threshold*** ввести обране значення порогу обчислень.

1.4 Робота з текстом

Текстові фрагменти є частинами тексту, які користувач хоче додати до свого документу. Вони вставляються за допомогою пункту меню **Вставка – Текстовий регіон,** рисунок. 1.5.

Дозволяється виконувати форматування тексту, для цього необхідно його виділити і вибрати параметри на панелі шрифтів або в меню **Форматування – Текст**.

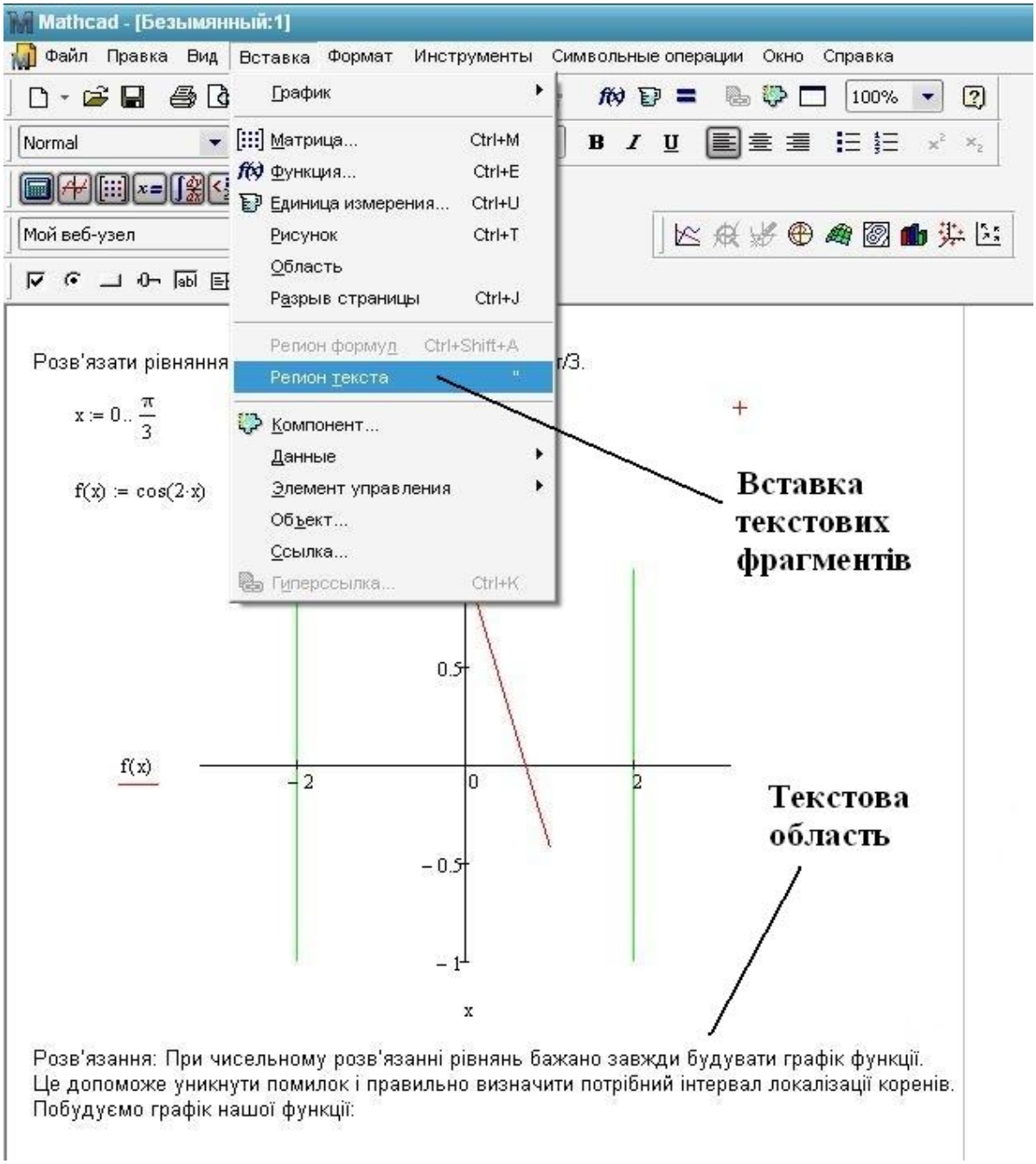


Рисунок 1.5 – Приклад текстових данних

1.5 Робота з графіками

При вирішенні ряду різноманітних задач, пов’язаних із дослідженням функцій, виникає необхідність побудови графіка функції.

В середовищі MathCAD існує можливість побудови різноманітних графіків: в Декартові та полярній системах координат, поверхонь тіл обертання, багатокутників, тривимірних, , просторових кривих, графіків векторного поля тощо.

1.5.1 Побудова двовимірних графіків

Для побудови двовимірного графіка потрібно:

– задати діапазон значень аргумента;

– визначити функцію;

– встановити курсор туди, де має бути побудовано графік, й на математичній панелі обрати кнопку **Graph (Графік)**, надалі на відкритій панелі кнопку **X-Y Plot (Двовимірний графік)**;

– в отриманому шаблоні графіка – пустий прямокутник із мітками даних, в центральну мітку даних вісі абсцис (вісь X) ввести ім’я змінної, а на місці центральної мітки даних по вісі ординат (вісь Y) ввести ім’я функції, рисунок 1.6;

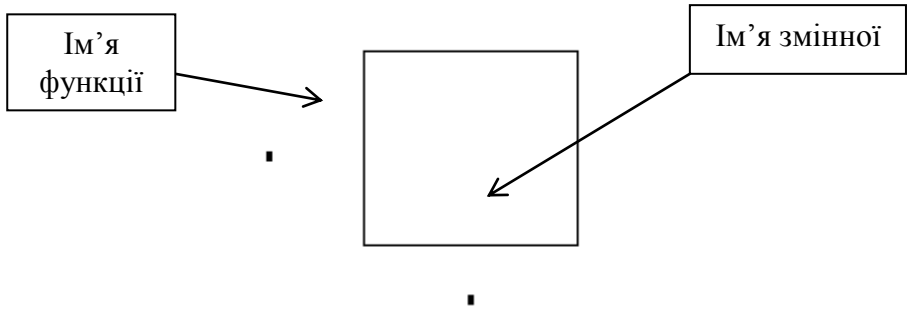


Рисунок 1.6 – Шаблон двовимірного графіку

– клацнути мишкою поза шаблоном графіку, і графік функції буде побудований.

Діапазон зміни аргументу складається з трьох значень: початкове, друге і кінцеве.

Приклад 1. Нехай необхідно побудувати графік функції на інтервалі [-3,3] з кроком 0.1. Значення змінної *t* задається у вигляді діапазону наступним чином:

t:= –3, –2.9 .. 3,

де: -3 – початкове значення діапазону;

–2.9 (–3 + 0.1) — друге значення діапазону (початкове значення плюс крок);

3 — кінцеве значення діапазону.

Двокрапка вводиться натисканням крапки з комою в англійській розкладці клавіатури.

Приклад 2. Побудова графіка функції $y = x^2$ на інтервалі [–5,5] з кроком 0.5. Розв'язок наведено на рисунку 1.7.

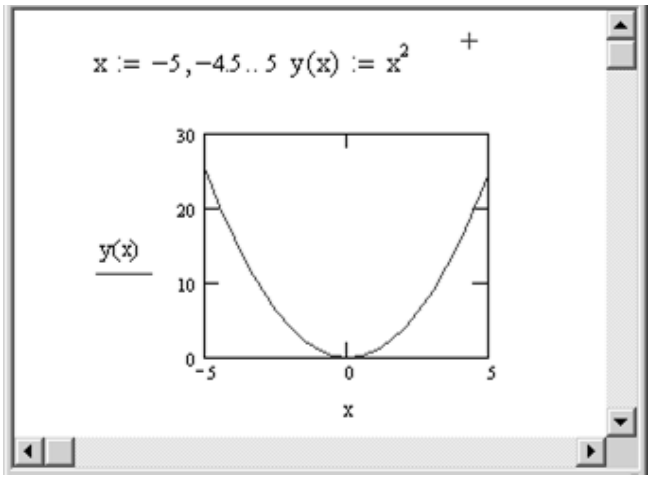


Рисунок 1.7 – Побудова графіку функції $y = x^2$

При побудові графіків необхідно враховувати такі зауваження:

– Якщо діапазон значень аргументу не задано, за замовчуванням графік буде побудовано в діапазоні [–10,10].

– При розміщенні в одному шаблоні декількох графіків імена функцій вказуються через кому.

– Якщо дві функції мають різні аргументи, наприклад *f1(x)* і *f2(y)*, то на вісі ординат (Y) через кому вказуються імена функцій, а по вісі абсцис (X) — імена обидвох змінних через кому.

– Крайні мітки даних на шаблоні графіку служать для визначення граничних значень вісі абсцис і ординат, вони задають масштаб графіка. Якщо залишити ці мітки пустими, масштаб буде встановлений за замовчуванням.

Якщо після побудови графік не приймає потрібний вигляд, можна:

– Зменшити крок заданого значення x.

– Змінити інтервал побудови.

– Зменшити граничні значення вісей абсцис та ординат.

Приклад. Побудова кола з центром в точці (2,3) и радіусом $R = 6$.

Рівняння кола з центром в точці з координатами (x_0,y_0) і радіусом R записується:

$$(x - x_0)^2 + (y - y_0)^2 = R^2$$

З цього рівняння y:

$$(y - y_0)^2 = R^2 - (x - x_0)^2$$
$$y - y_0 = \pm\sqrt{R^2 - (x - x_0)^2}$$
$$y = \pm\sqrt{R^2 - (x - x_0)^2} + y_0$$

Таким чином, для побудови графіка кола необхідно задати дві функції: а саме верхнього і нижнього півкола. Діапазон значень аргументу обчислюється таким чином:

– початкове значення діапазону $x_0 - R$;

– кінцеве значення діапазону = $x_0 + R$;

– крок краще буде взяти 0.1

Вирішення наведено на рисунку 1.8.

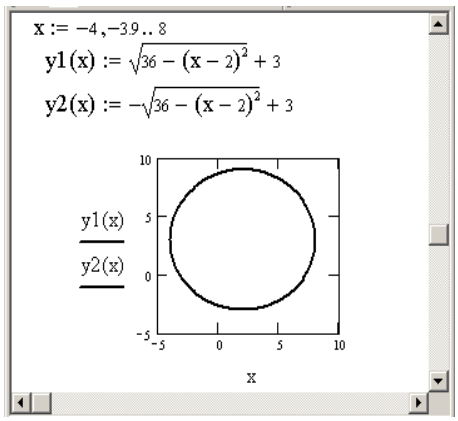


Рисунок 1.8 – Побудова кола

Параметричний графік функції

При побудові параметричного графіку по вісям абсцис і ординат вказуються імена функцій одного аргументу.

Приклад. Побудова кола з центром в точці (2,3) і радіусом 6. Для побудови використано параметричне рівняння кола:

$$x = x_0 + R\cos(t)\ y = y_0 + R\sin(t)$$

Вирішення наведено на рисунку 1.9.

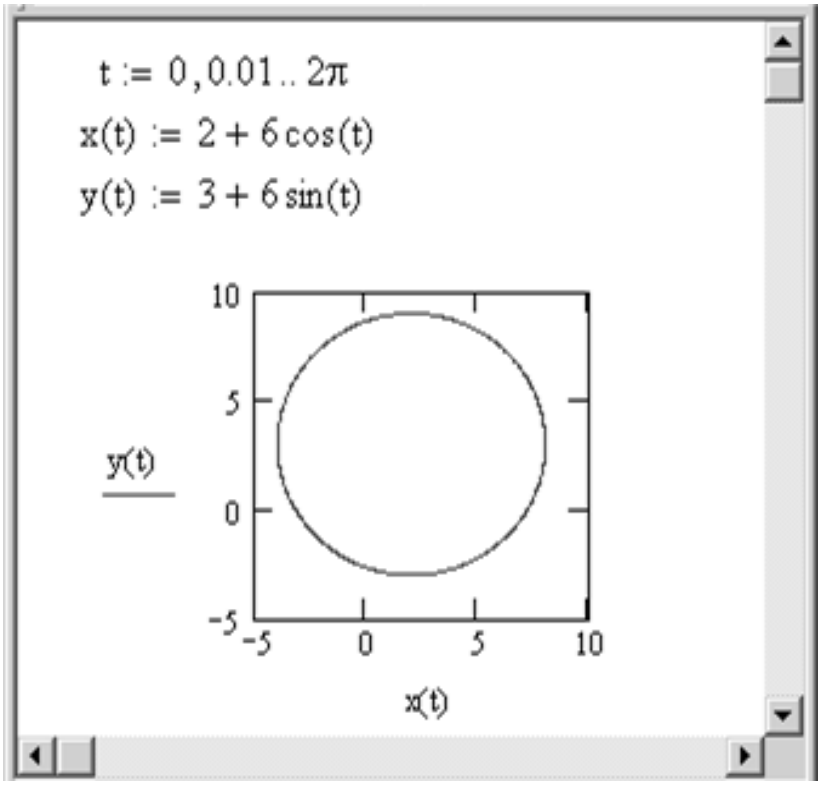


Рисунок 1.9 – Побудова кола за параметричним рівнянням

Форматування графіків

Для форматування графіку потрібно два рази натиснути в області графіку, відкривши діалогове вікно форматування графіку. Вкладки вікна форматування графіку:

– **X-Y Axes –** форматування вісей координат.

Встановивши прапорці, можна:

– **Log Scale –** представлення числових значень на вісях в логарифмічному масштабі.

– **Grid Lines –**сітка ліній.

– **Numbered –** розмістити числа по координатним вісям.

– **Auto Scale –** автоматичний вибір граничних числових значень по вісях (якщо прапорець знятий, граничними є максимальні обчислені значення).

– **Show Marker** – нанесення міток на графік у вигляді горизонтальних або вертикальних пунктирних ліній, що відповідають вказаному значенню на вісі, причому самі значення виводяться вкінці ліній (на кожній осі з'являються два місця введення, в які можна ввести числові значення).

– **Auto Grid** – автоматичний вибір числа ліній сітки (якщо прапорець знятий, необхідно задати число ліній в полі **Number of Grids**).

– **Crossed –** вісь абсцис проходить через нуль ординати.

– **Boxed** – вісь абсцис проходить по нижньому краю графіка.

– **Trace** – форматування ліній графіків функцій. Для кожного графіка окремо можна змінити:

– символ (**Symbol**) на графіку для вузлових точок (кружок, хрестик, прямокутник, ромб);

– вигляд ліній (**Dot** – пунктир, **Dash** – штрихи, **Dadot** – штрих–пунктир, **Solid** – суцільна);

– колір лінії (**Color**);

– тип (**Type**) графіку (**Lines** – лінія, **Points** – точки, **Bar** або **Solidbar** – стовпчики, **Step** – ступінчастий графік, та т.ін.);

– товщину ліній (**Weight**).

– **Label** – заголовок в області графіка. В поле ***Title*** (**Заголовок**) можна записати текст заголовка, вибрати його положення – зверху або знизу графіка (***Above*** – зверху, ***Below*** – знизу). Можна вписати, якщо треба, назву аргументу і функції (***Axis Labels*****).**

– **Defaults** – за допомогою цієї вкладки можна вернутися до виду графіку, прийнятому за замовчуванням (**Change to default**), або зроблені вами зміни на графіку використати за замовчуванням для всіх графіків даного документу (**Use for Defaults**).

1.5.2 Побудова графіків в полярних координатах

Для побудови графіка функції необхідно:

– задати діапазон значень аргументу;

– задати функцію;

– встановити курсор туди, де буде побудовано графік, вибрати на математичній панелі іконку **Graph (Графік)** і, у панелі, що відкриється, обрати кнопку **Polar Plot (Полярний графік)**;

– у місті виведення шаблону необхідно ввести аргумент функції (внизу) та ім’я функції (зліва).

Приклад. Побудова лемніскати Бернуллі: $\rho = \sqrt{2\cos(2\varphi)}$. Розв’язок наведено на рисунку 1.10.

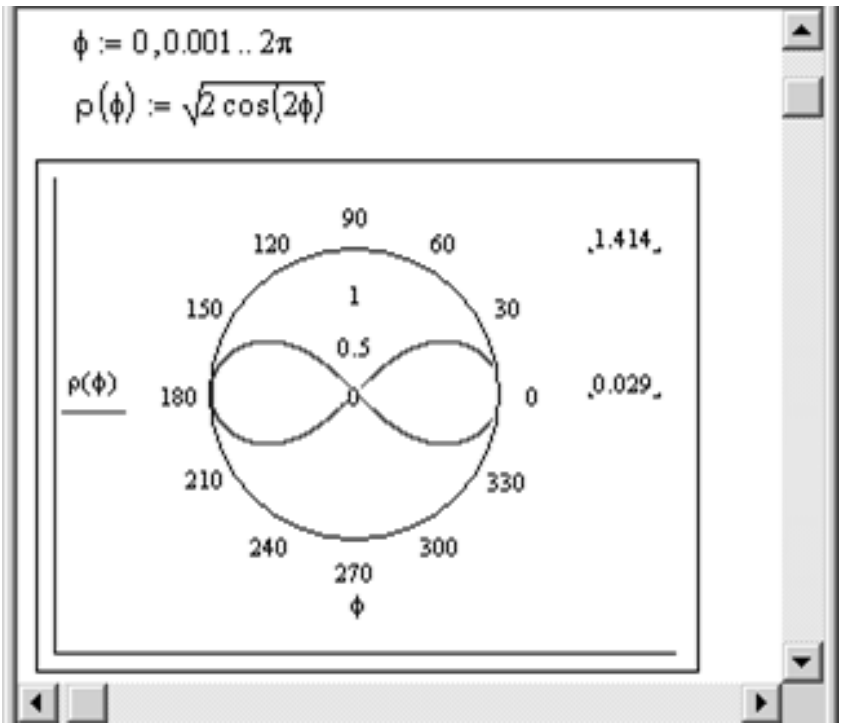


Рисунок 1.10 – Приклад побудови графіка у полярних координатах

1.5.3 Побудова графіків поверхонь (тривимірних або 3D-графіки)

Для побудови тривимірних графіків застосовують панель **Graph** (**Графік**) на математичній панелі. Тривимірний графік може бути побудуваний у декілька способів:

– за допомогою майстра, викликаного з головного меню;

– із застосуванням матриці значень функції двох змінних;

– застосовуючи швидкий метод побудови;

– за допомогою спеціальних функцій **Create Mech** і **Create Space**, що призначені для створення одночасно масиву значень функції і для побудови графіку.

Швидка побудова графіку

Швидка побудова тривимірного графіку може бути реалізована у такий спосіб:

– описати функцію;

– встановити курсор туди, де буде побудовано графік, на математичній панелі обрати іконку **Graph (Графік)** та у відкритій панелі обрати іконку  (**Поверхневий графік**);

– в одне місце шаблону ввести ім’я функції (без указання змінних);

– клацнути лівою клавішею миші поза шаблоном.

Приклад. Побудова графіку функції $z(x,y) = x^2 + y^2 - 30$. Реалізацію наведено на рисунку 1.11.

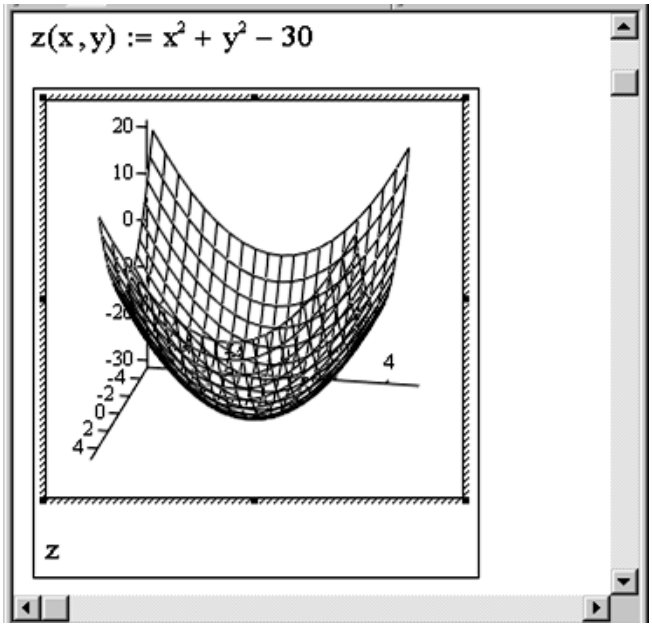


Рисунок 1.11 – Приклад «швидкої» побудови графіка поверхні

Є можливість керувати побудованим графіком:

– обертання графіку реалізується, якщо навести на нього покажчик миші і затиснути ліву клавішу;

– масштабування реалізується, якщо навести на графік покажчик миші при одночасному утриманні її лівої кнопки та клавіші **Ctrl** (якщо рухати мишею, графік буде наближатися або віддалятися);

– анімація графіка може бути реалізована аналогічним чином, але при додатковому натисканні клавіші **Shift**. Для зупинки обертання необхідно клацнути ліву кнопку миші всередині області графіка.

Також можливо побудувати декілька поверхонь у одному рисунку. Для цього потрібно описати дві функції, і через кому вказати імена функцій на шаблоні графіка.

При швидкій побудові графіка за замовчуванням вибираються значення обох аргументів в межах від -5 до +5 і число контурних ліній, дорівнює 20. Для зміни цих значень необхідно:

– два рази натиснути по графіку лівою клавішею миші;

– у відкритому вікні вибрати вкладку **Quick Plot Data**;

– ввести нові значення в області вікна **Range1** — для першого аргументу і **Range2** — для другого аргументу (start — початкове значення, end — кінцеве значення);

– в полі **# of Grids** змінити число ліній сітки, що покривають поверхню;

– натиснути кнопку **Ок**.

Приклад. Побудова графіку функції $z(x,y) = -\sin(x^2 + y^2)$. Розв'язок наведено на рисунку 1.12.

При побудові цього графіку границі зміни значень обох аргументів краще вибрати від –2 до +2.

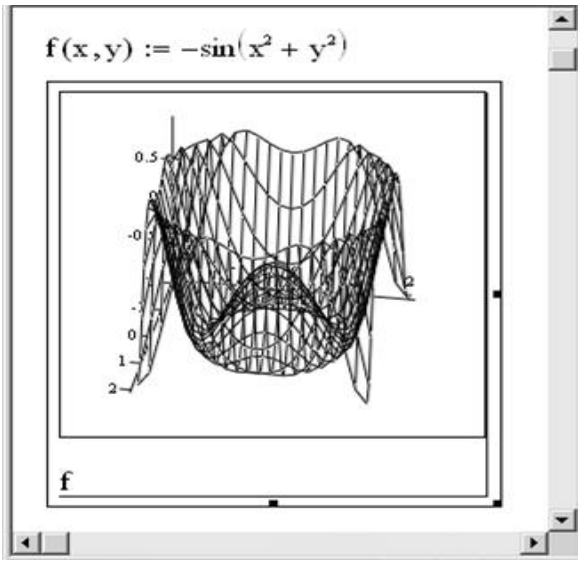


Рисунок 1.12 – Приклад побудови графіку функції $z(x,y) = -\sin(x^2 + y^2)$

Форматування тривимірних графіків

Для форматування графіку треба двічи клацнути лівою кнопкою миші в області побудови – тоді отримаємо вікно форматування із декількома вкладками.

Вкладка **Appearance** дозволяє змінювати зовнішній вигляд графіка. Поле ***Fill Options*** відповідає параметрам заливання, поле ***Line Option*** — параметрам ліній, ***Point Options*** — параметрам точок відповідно.

На вкладці **General (Загальні**) в групі **View** є можливість обрати кут повороту зображення навколо всіх трьох вісей; в групі **Display as** змінюють тип графіку.

За допомогою вкладки **Lighting** (**Освітлення**) керують освітленням рисунку, встановлюючи прапорець **Enable Lighting** (**Увімкнути освітлення**) та перемикач **On**. Одна із шести можливих схем обирається зі списку **Lighting scheme** (**Схема освітлення**).

1.6 Способи розв’язку рівнянь в MathCAD

Даний розділ присвячено розв’язку найпростіших рівнянь в системі MathCAD. Як відомо, рівняння в загальному вигляді має вигляд *f(x)=0*.

Розв’язати рівняння аналітично – значить знайти всі його корені, тобто значення шуканої змінної, при підстановці яких у вихідне рівняння перетворюється на тотожність.

Розв’язати рівняння графічно, відповідно, це знайти точки перетину графіка функції *f(x)* з віссю OX.

1.6.1 Розв’язок рівнянь з використанням функції root(f(x),x)

Для розв’язку рівняння з одним невідомим вигляду *f(x)=0* в пакеті існує спеціальна функція: **root(f(x),x),**

де *f*(*x*) —вираз, що дорівнює нулю (функція);

x — аргумент.

Функція **root(f(x),x)** повертає з заданою точністю значення змінної, при якому значення виразу *f(x)*буде рівним *0*. Якщо права частина рівняння *не дорівнює нулю*, необхідно привести його до нормального вигляду (перенести все в ліву частину).

Перед використанням функції **root** аргументу *x* задається початкове наближення (обов'язково). Якщо коренів декілька, для пошуку кожного кореня потрібно буде задавати окреме початкове наближення.

Перед розв'язком бажано побудувати графік функції для перевірки і виокремлення коренів рівняння. Початкове наближення обирають за графіком в околі точки перетину.

Приклад. Розв'язати рівняння $x^3 = 15x$ з використанням команди **root**.

Розв'язок наведено на рисунку 1.13. Попередньо привести рівняння до нормального вигляду: все перемістити в ліву частину. Рівняння набуде вигляду: $x^3 - 15x = 0$.

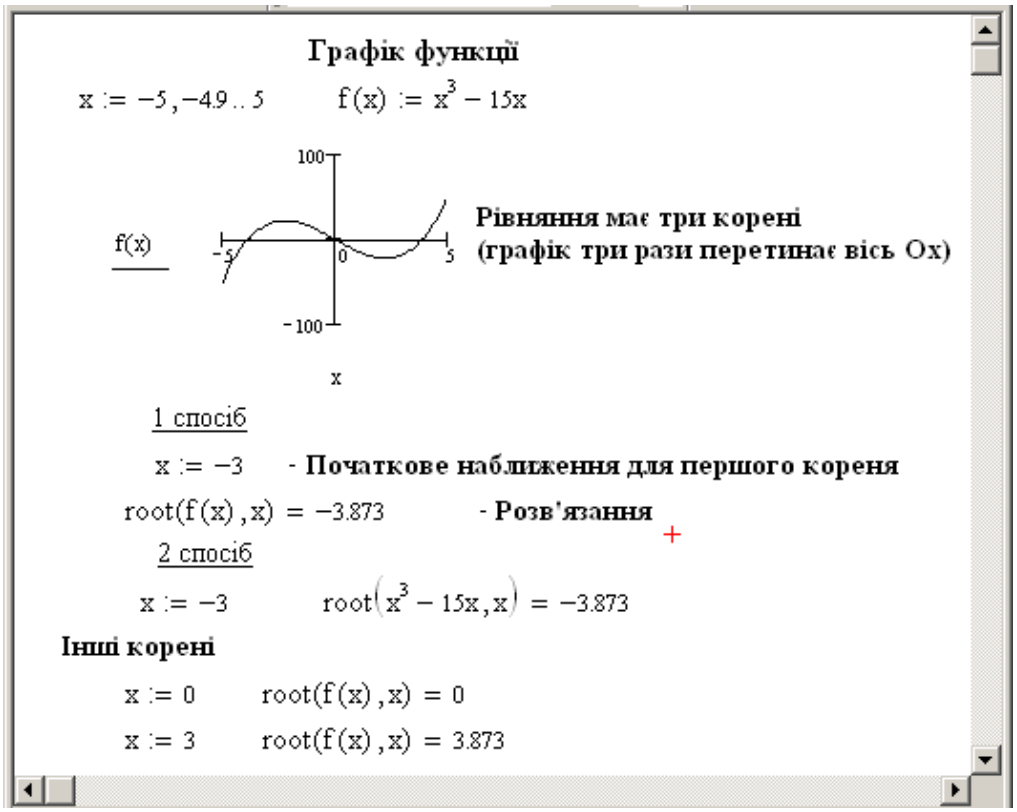


Рисунок 1.13 – Розв'язок рівняння за допомогою функції root

1.6.2 Розв’язок рівнянь за допомогою функції Polyroots(v)

Для знаходження всіх коренів поліноміального рівняння у середовищі є можливість використання функції **Polyroots(v),** де v — вектор коефіцієнтів поліноміального рівняння від вільного члену до найвищого ступіня невідомої**.** Нульові коефіцієнти пропускаються. На відміну від функції **root, polyroots** не вимагає задавати початкове наближення.

Приклад. Розв’язати поліноміальне рівняння $0.75 \cdot x^3 - 8 \cdot x + 5 = 0$ з використанням функції **polyroots.** Розв’язок зображено на рисунку 1.14.

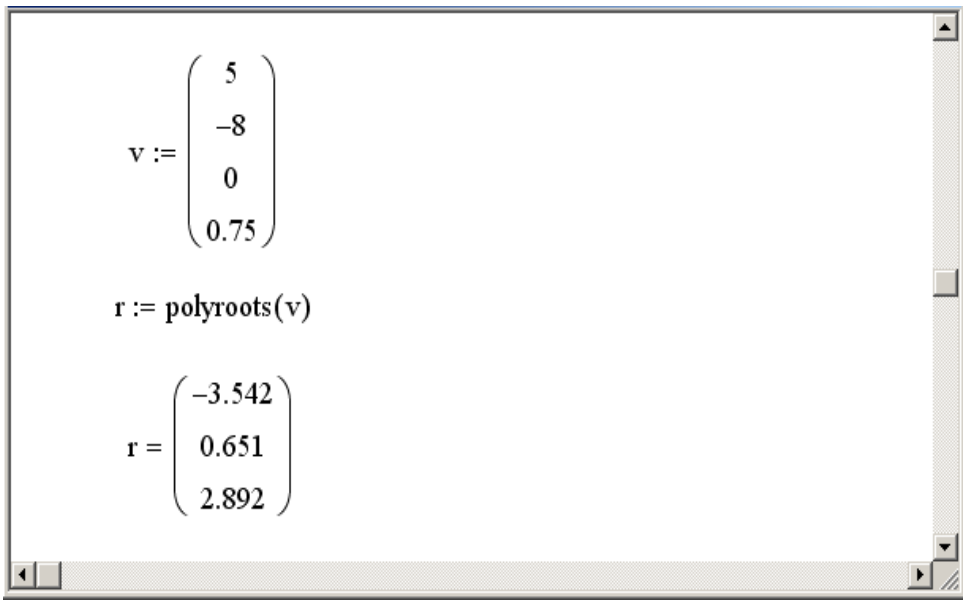


Рисунок 1.14 – Розв’язок поліноміального рівняння з використанням функції polyroots

1.6.3 Розв’язок рівнянь з використанням функції Find (x)

В пакеті функція **Find** завжди має використовуватись в разом з ключовим словом **Given**. Конструкція **Given - Find** використовує методику наближеного розрахунку, яка базується на пошуку кореня в околі точки початкового наближення, що задана користувачем.

Якщо задано рівняння f (x) = 0, то його можна розв’язати з застосуванням блоку **Given - Find** таким чином:

– задати початкове наближення: $x := x_0$

– ввести службове слово **Given**

– записати рівняння, використовуючи знак наближеної рівності: $f(x) \approx 0$

– записати функцію **Find** з невідомою змінною як параметр:

find(x)=

В результаті після знаку дорівнює буде виведено шуканий корінь.

Якщо існує декілька коренів, їх можна знайти, змінюючи початкове наближення x_0 на найбільш близьке значення до шуканого кореня.

Приклад. Розв'язок рівняння $x^2 + 8 = e^x$ з використанням функції **find.** Розв'язок зображено на рисунку 1.15.

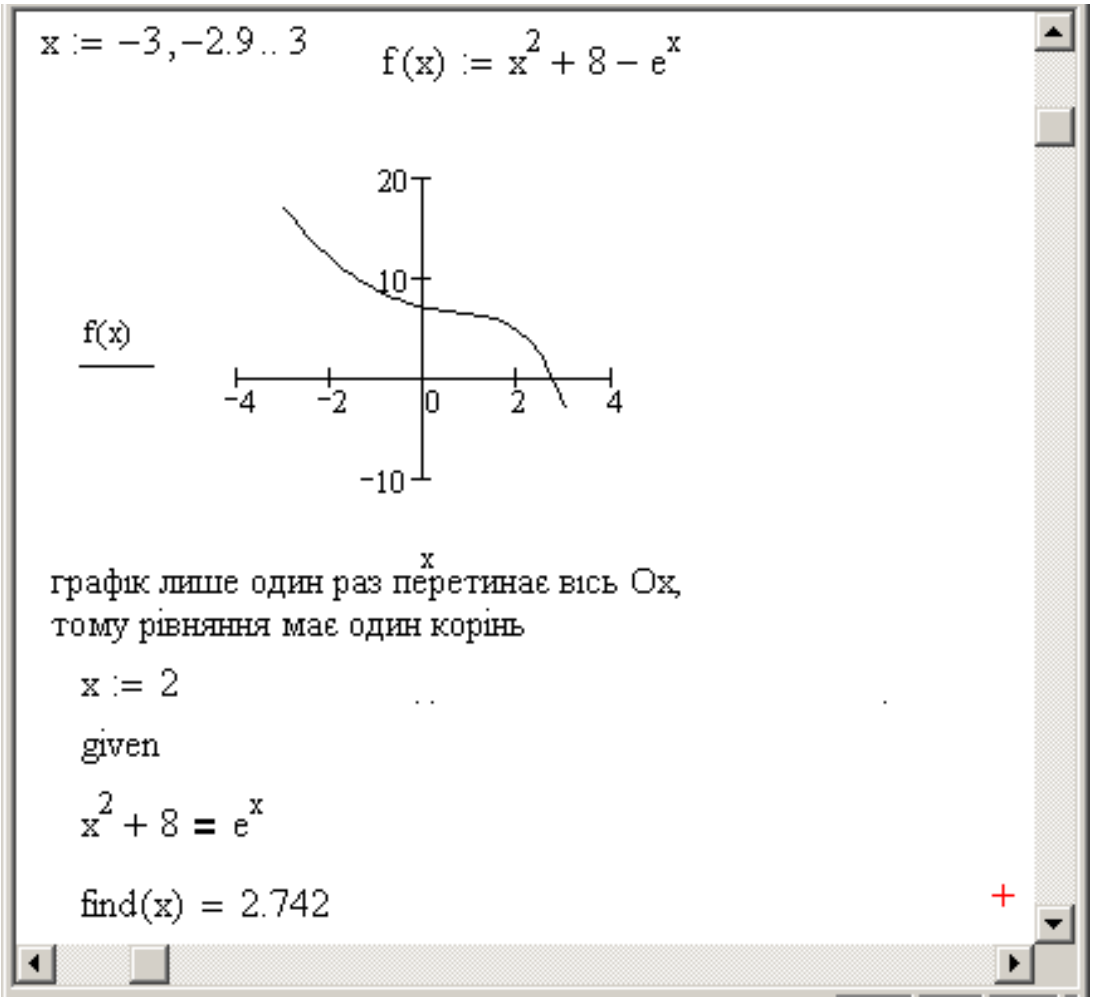


Рисунок 1.15 – Розв'язок рівняння з використанням функції **Find**

Для того, щоб знайти точки перетину графіка функції з віссю Ox), треба:

– вказати значення x заданої точки (по вісі Ox) і значення функції в цій точці (по осі Oy);

– двічі клацнути по графіку і у вікні форматування на вкладці **Traces** і для відповідної лінії обрати тип графіка – points, товщину лінії – 2 або 3.

Приклад. На графіку відмічено точка перетину функції $f(x)=x^2+8-e^x$ із віссю Ox. Координата x цієї точки була знайдена в попередньому прикладі:

$x=2.742$ (корінь рівняння). Розв'язок наведено на рисунку 1.16.

У вікні форматування графіка у вкладці **Traces** для trace2 було змінено: тип графіка – points, товщина лінії – 3, колір – чорний.

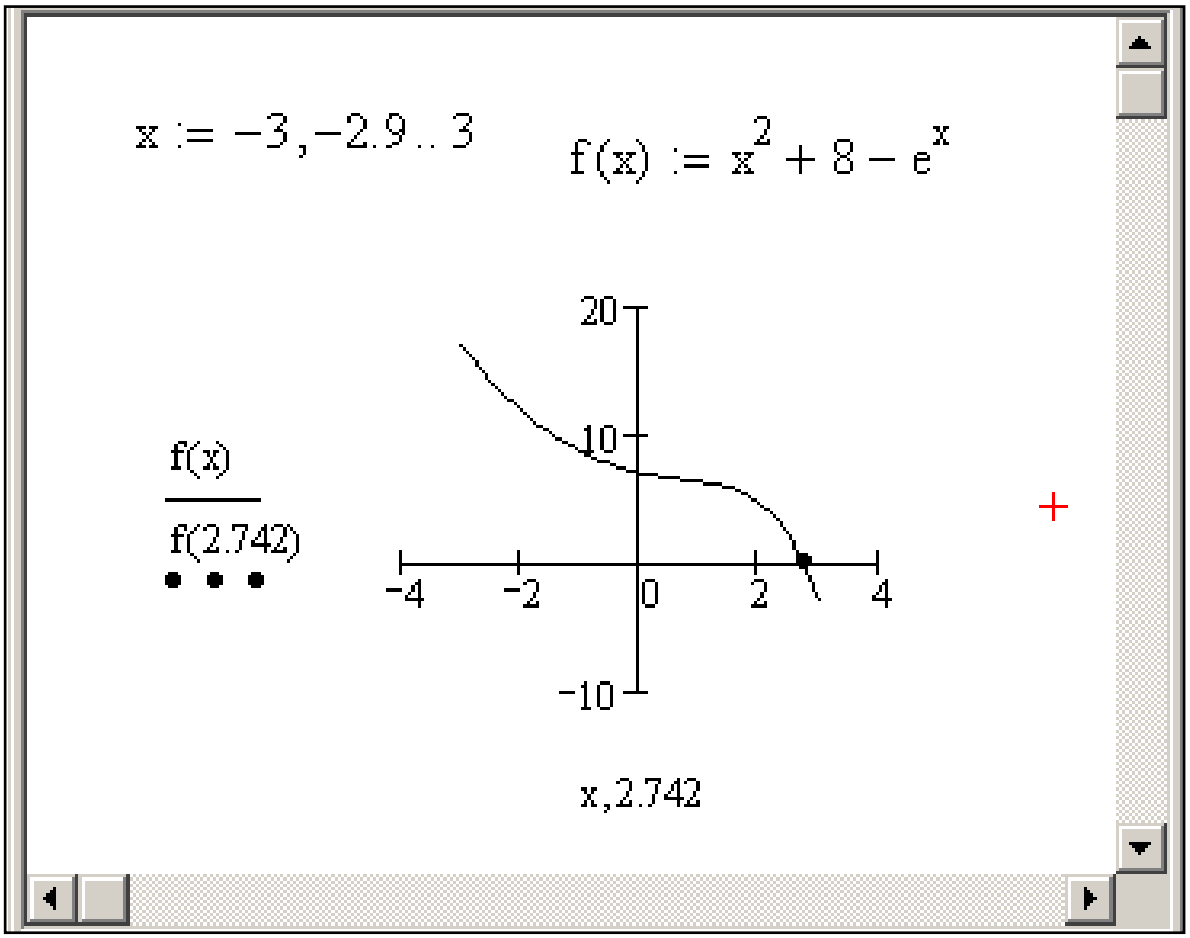


Рисунок 1.16 – Графік функції $f(x)=x^2+8-e^x$ з позначеною точкою перетину з віссю OX

1.7 Розв’язок систем рівнянь

1.7.1 Розв’язок систем лінійних рівнянь

В пакеті **MatLab** є можливість розв’язати систему лінійних рівнянь (СЛР) ***матричним методом*** або з використанням зворотньої матриці, або з використанням функції **lsolve** (A, B)) для двох функцій: **Find** і **Minerr**.

Матричний метод

Приклад. Дано СЛР:

$$\begin{cases} 3x1 + 4x2 = 5 \\ 5x1 + 4x2 = 3 \end{cases}.$$

Розв’язок системи матричним методом зображено нарисунку 1.17.

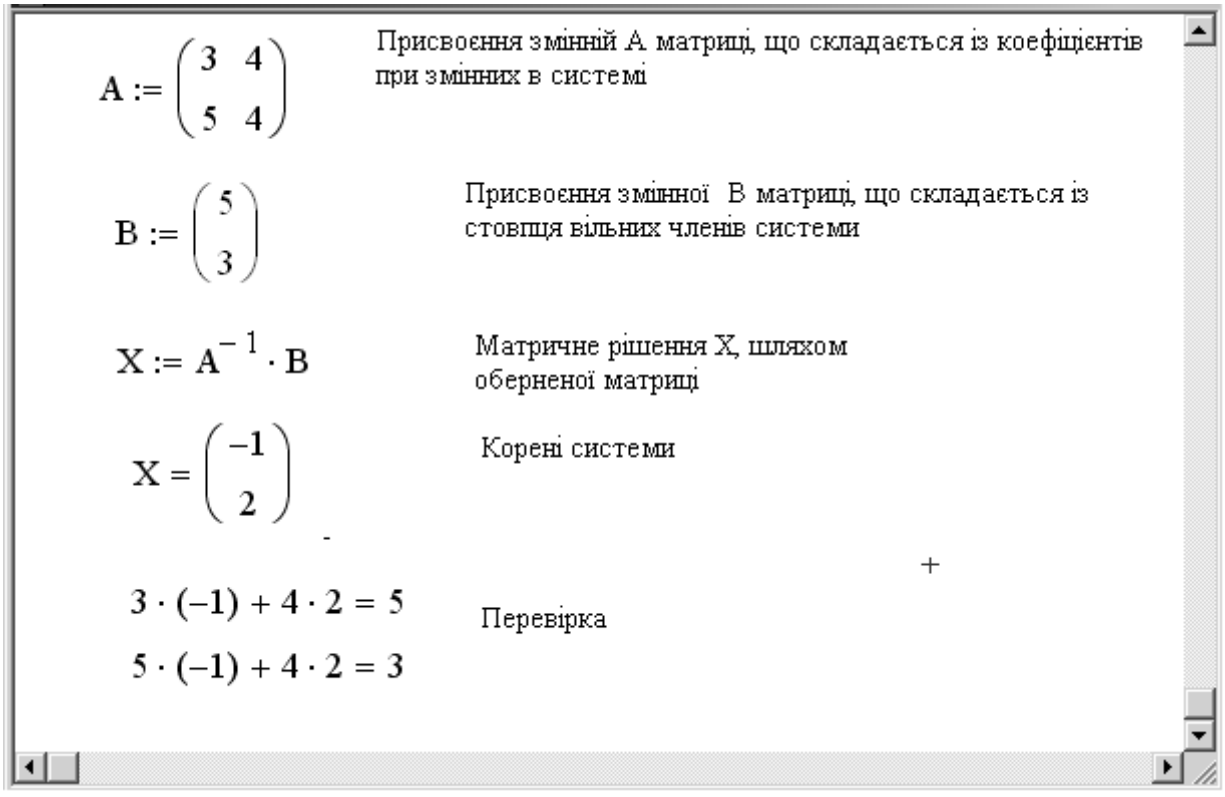


Рисунок 1.17 – Розв’язок СЛР

Використання функції lsolve(A,B)

Lsolve (A, B) є вбудованою функцією, що повертає вектор X заданої системи лінійних рівнянь, для якої сформована матриця коефіцієнтів A та вектор вільних членів B.

Приклад. Дано систему рівнянь:

$$\begin{cases} 1.2357\,x1 + 2.1742\,x2 - 5.4834\,x3 = 1 \\ 6.0696\,x1 - 6.2163\,x2 - 4.6921\,x3 = 1 \\ 3.4873\,x1 + 6.1365\,x2 - 4.7483\,x3 = 1 \end{cases}.$$

Метод вирішення СЛР функцією **lsolve** (A, B) наведено на рисунку 1.18.

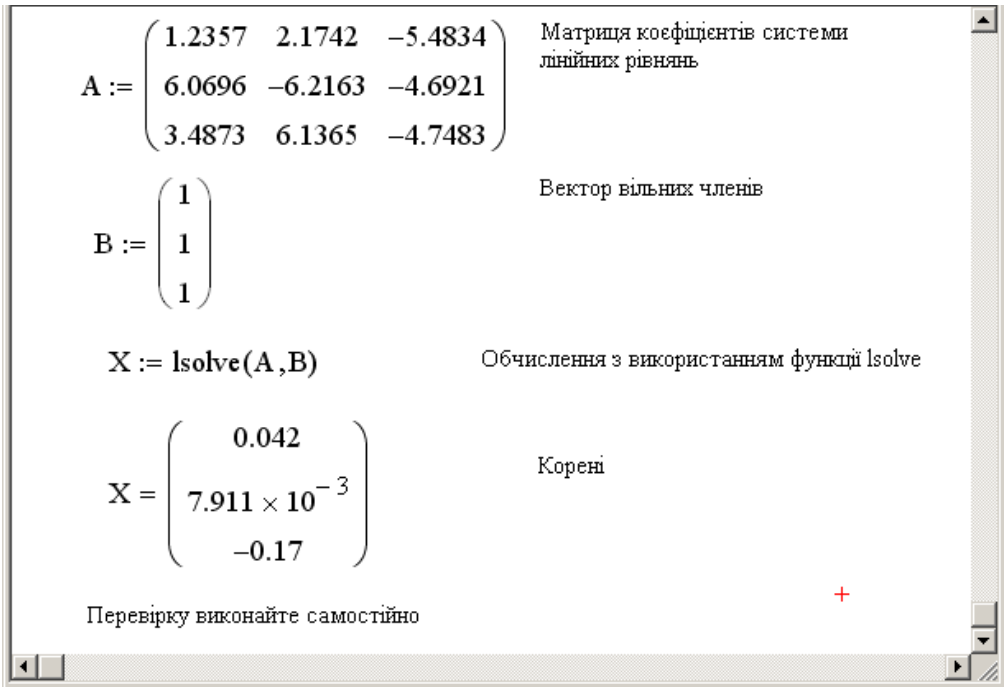


Рисунок 1.18 – Розв’язок СЛР функцією **lsolve**

Розв’язок СЛР функцією Find

При цьому методі є можливість ввести рівняння вводяться, “as is”, попередньо вказавши початкові наближення невідомих: бути будь-які числа з області визначення. Часто за початкові наближення беруть стовпець вільних членів.

Для розв’язку СЛР блоком **Given - Find**, потрібно:

– вказати початкові наближення всіх змінних;

– ввести **Given**;

– записати СЛР з використанням знаку **жирне дорівнює** (⩵);

– записати **Find,** перелічивши невідомі, як параметри функції.

В результаті розрахунків буде виведено вектор розв’язку системи.

Приклад. Дано СЛР:

$$\begin{cases} 3x1 + 4x2 = 5 \\ 5x1 + 4x2 = 3 \end{cases}$$

Розв’язок системи обчислювальним блоком **Given - Find** наведено на рисунку 1.19.

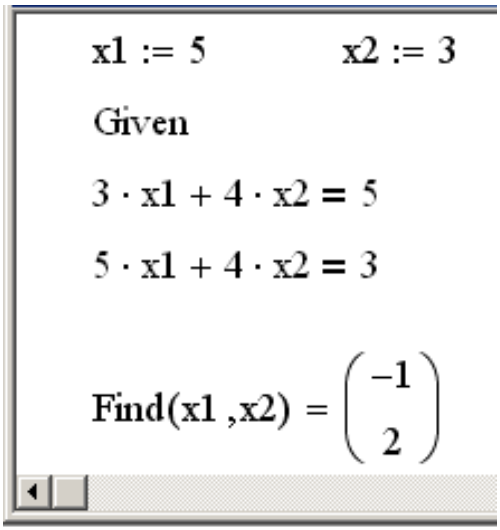


Рисунок 1.19 – Розв’язок СЛР функцією **Find**

Наближений розв’язок СЛР

Розв’язок СЛР функцією **Minerr** аналогічний до розв’язку за допомогою функції **Find** (використовується тей самий алгоритм), але функція **Find** дає точний розв’язок, а **Minerr** - наближений. Якщо в результаті пошуку не буде отримано подальше уточнення поточного наближення до розв'язку, **Minerr** повертає це наближення, а функція **Find** – повідомлення про помилку.

Загальні рекомендації з розв’язку рівнянь і систем рівнянь

У випадку, коли MathCAD не може самостійно знайти розв’язок можна:

– підібрати інше початкове наближення.

– збільшити або зменшити точність розрахунків. Для цього в меню **Math ► Options** (**Математика – Опції**), вкладку **Built-In Variables** (**Вбудовані змінні**). У вкладці зменшити допустиму похибку обчислень (**Convergence Tolerance (TOL)**), яка за замовчуванням дорівнює **0.001**.

При матричному методі розв'язку потрібно розставляти коефіцієнти згідно зростанню невідомих *x1, x2, x3, x4*.

1.7.2 Розв'язок систем нелінійних рівнянь

Системи нелінійних рівнянь (СНР) в середовищі MathCAD розв'язують обчислювальним блоком **Given - Find**.

Для розв'язку системи рівнянь допомогою блоком **Given - Find** потрібно:

– задати початкові наближення всіх змінних;

– ввести **Given;**

– записати СНР з використанням знаку **жирне дорівнює** (⩵);

– записати функцію **Find,** перелічивши невідомі, як параметри функції.

В результаті розрахунків виведеться розв'язання системи у векторному вигляді.

Якщо система має кілька розв'язків, алгоритм повторити з іншими заданими початковими наближеннями.

Якщо розв'зують система двох рівнянь з двома невідомими, перед розв'язком бажано побудувати графіки функцій для пошуку ізольованих коренів.

Приклад. Дано СНР:

$$\begin{cases} y = x^2 + 14 \\ y = 7x + 45 \end{cases}.$$

Перед розв’язком цієї системи необхідно побудувати графіки функцій: параболічної (перше рівняння) і прямої (друге рівняння). Побудова графіків в одній системі координат наведена на рисунку 1.20.

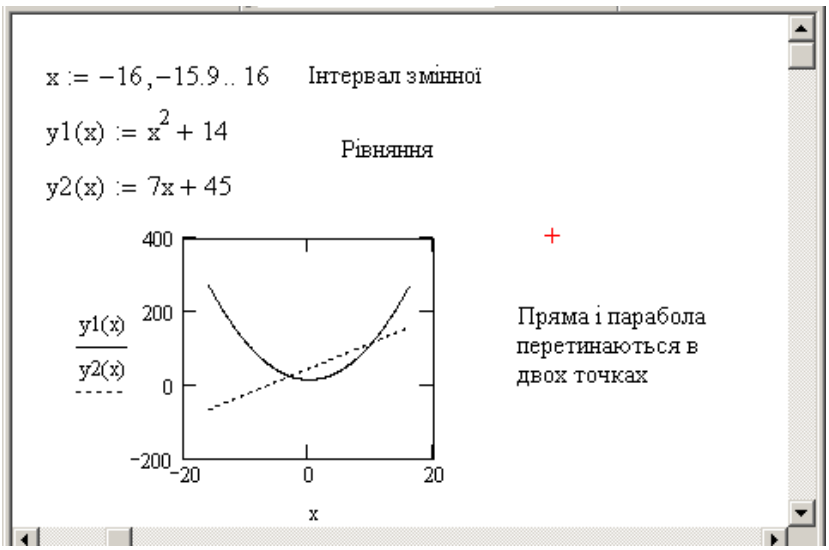


Рисунок 1.20 – Побудова графіка двох функцій в одній системі координат

Пряма і парабола перетинаються в двох точках, значить, система має два кореня. За графіком вибираємо початкові наближення невідомих *x* і *y* для кожного рішення. Знаходження коренів СНР наведено на рисунку 1.21.

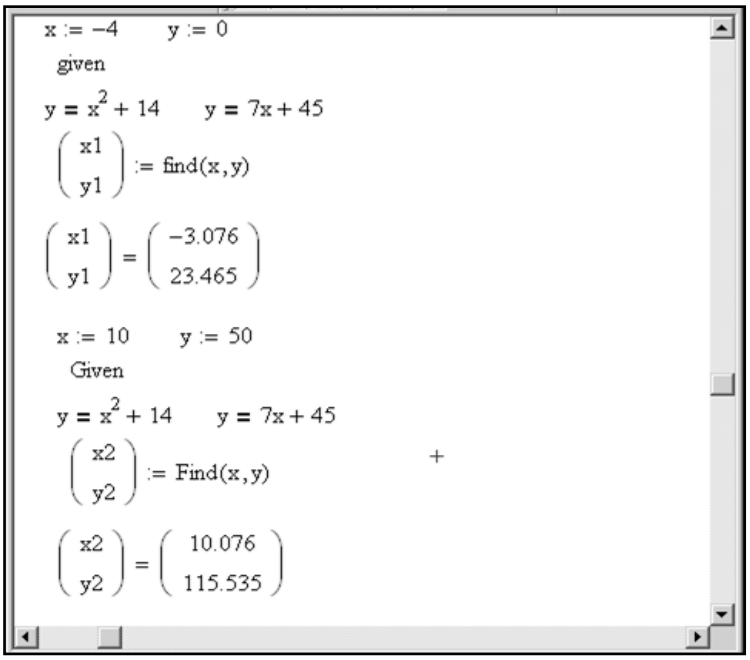
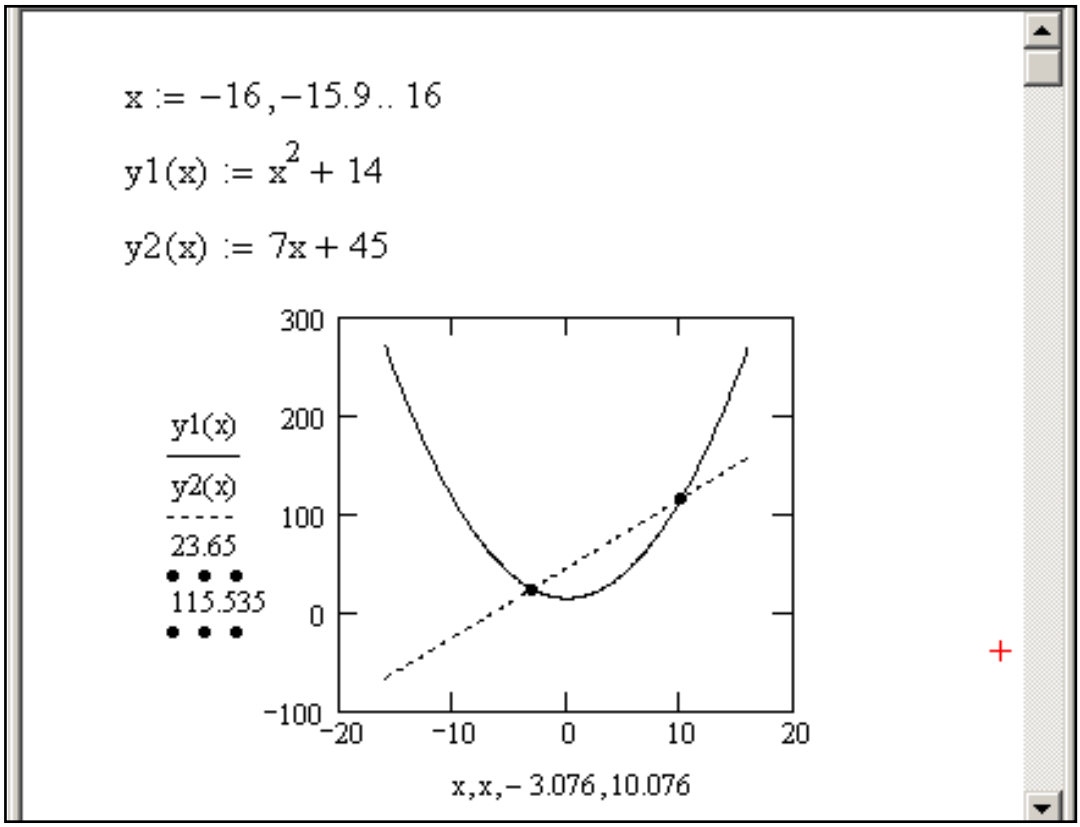


Рисунок 1.21 – Знаходження коренів системи нелінійних рівнянь

Для того, щоб відзначити на графіку координати точок перетину параболи і прямої, знайдені при розв'язок системи, введемо через кому значення *x* (по осі Ox) та значення *y (*по осі Oy). У вікні форматування графіка у вкладці **Traces** для **trace3** і **trace4** змінимо: тип графіка – **points**, товщина лінії – 3, колір – чорний, рисунок 1.22.



Рисинок 1.22 – Графіки функцій з позначеними точками перетину

1.8 Програмування в пакеті MathCAD

Програмування – це створення блоку, в якому поєднано різнманітні операції. За допомогою програмних блоків можуть бути створені програми. Функції з ранжованими змінними і операції **if** надають можливість використовувати умовні вирази та організовувати циклічні конструкції, проте їх можливості обмежені.

Блоки програмування створюють за допомогою команд зі списку **Інструменти програмування** (**Programming Toolbar**) панелі інструментів **Математика (Math)**:

– **Add Line** – створити, або, при необхідності, продовжити жирну вертикальну лінію –знак обмеження програмного блоку;

– ← – оператор присвоювання в програмному блоці;

– **if** –умовний оператор;

– **while** – оператор циклу з передумовою;

– **otherwise** – оператор іншого вибору;

– **break** – припинення розрахунків;

– **continue** – продовження розрахунків;

– **return** – повернення на початок циклу;

– **on error** – обробка помилок.

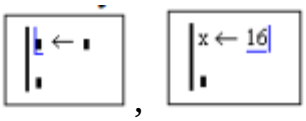
Оператор **Add Line.**

Визначає межі програмного блоку (програми або підпрограми), тобто списку команд, що виконуються послідовно. Програмний блок може бути функцією без імені, що буде розраховувати результат.

Результат може бути отримано у декілька кроків:

1. Нажати на панелі програмування клавішу “ **]** ” або **Add Line на** панелі інструментів **Математика (Math)**. В робочому полі, на місці розташування курсору буде виведено шаблон блоку з двома місцями введення інформації

2. У верхнє місце введення необхідно поставити курсор, після чого натиснути клавішу “ ← ” і чекати на появу шаблону верхнього рядка. Шаблон потрібно буде заповнити ідентефікатором змінної *x* та константою *16* (значення, що буде присвоєно):



3. У нижнє місце ввести шаблон квадратного кореня, аргументу x і натиснути клавішу “**Пробіл**” – буде створено режим, коли курсор підкреслює увесь вираз.

4. Натиснути клавішу “=” і одержати результат.

Приклад. Знайти функцію x, якщо $\sqrt{x}=16$:

$$\begin{vmatrix} x \leftarrow 16 \\ \sqrt{x} \end{vmatrix} = 4$$

Слід відзначити, що змінні у блоці та змінні поза блоком можуть мати однакові ідентифікатори, але це змінні є різними й не залежать одна від одної. Іноді їх буде доцільно пов’язати процедурою передачі параметрів.

Приклад. Виконати обчислення функції $B=\dfrac{x+y+z}{x+y\cdot z}$ при значеннях вхідних змінних $x=1,\ y=2,\ z=3$ та $x=5,\ y=7,\ z=2$.

$$B(x,y,z):=\frac{x+y+z}{x+y\cdot z}$$

Для цього необхідно задати значення формальним параметрам *x, y, z* і отримати результат:

$$B(1,2,3)=0.857$$

$$B(5,7,2)=0.737$$

Приклад. Виконати обчислення функції, заданої у попередньому прикладі для двох значень змінних $x=1,\ y=2,\ z=3$ та $x=5,\ y=7,\ z=2$, з використанням програмного блоку.

$$B(x,y,z):=\begin{vmatrix} a \leftarrow x+y\cdot z \\ \dfrac{x+y+z}{a} \end{vmatrix}$$

$$B(1,2,3)=0.857$$

$$B(5,7,2)=0.737$$

Програмному блоку з ім'ям ***B*** присвоєно результат, а ***x, y, z*** –значення формальних параметрів, що будуть передавати у блок фактичні, відповідно.

Оператор if та otherwise

Всі команди у програмному блоці мають шаблони з місцями для введення даних у вигляді ідентифікаторів, арифметичних, логічних виразів тощо. Команди **if** та **otherwise** мають такі шаблони:

■ if ■ ■ otherwise .

Їх використання у програмному блоці буде продемонстровоно програмою визначення модуля змінної за таким алгоритмом: якщо задане значення менше *0*, вивести його зі знаком мінус, в іншому випадку – зі знаком плюс, тобто без змін.

$$\text{abs}(x) := \begin{cases} -x & \text{if } x < 0 \\ x & \text{otherwise} \end{cases} \qquad \text{abs}(-15) = 15 \qquad \text{abs}(15) = 15.$$

Приклад. Знайти значення числа Нусельта. Задано значення числа Рейносьдса $\text{Re} = 2800$. При $\text{Re} \geq 10000$ значення критерій Нусельта знаходитьься з залежності $Nu = 0{,}021 \cdot \text{Re}^{0,6}$, при $\text{Re} < 10000$ –з залежністі $Nu = 0{,}08 \cdot \text{Re}^{0,8}$.

Розв'язок наведено на рисунку 1.23:

– записати задачу і задати фукцію Nu:= ;

– використовуючу оператор **Add Line** задати програмний блок:

вказати у верхньому рядку $0.021 \cdot \text{Re}^{0.6}$ if $\text{Re} \geq 1000$,

вказати у нижньому рядку $0.08 \cdot \text{Re}^{0.9}$ та обрати умову (команду умовного оператора програмного блоку) otherwise – інакше;

– записати фукцію Nu= і отримати бажаний результат.

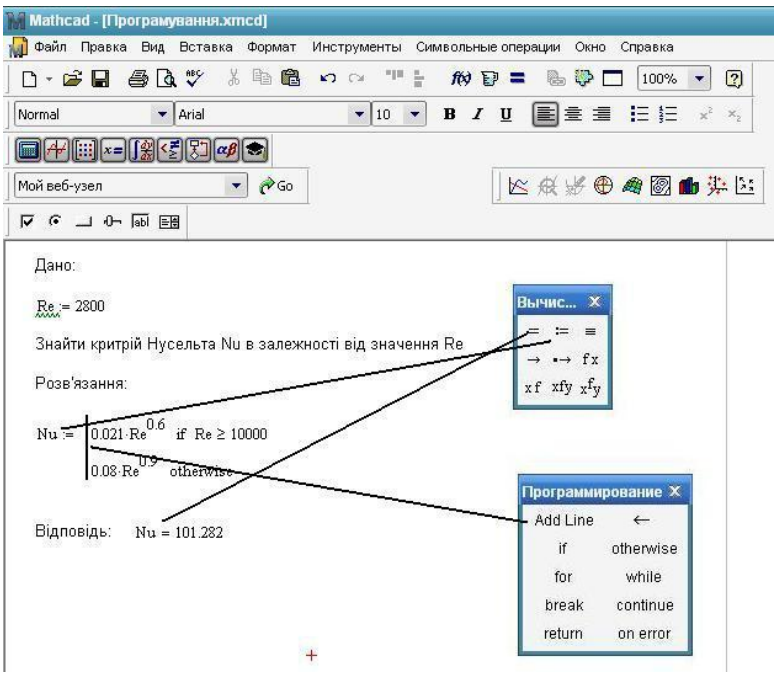
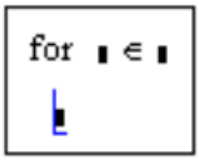


Рисунок 1.23 – Приклад виконання програми з використанням умовного оператора **if**

Оператори циклу for, while та break

Синтаксис оператора циклу з параметром **for**:



Приклад програми:

$$s(n) := \left| \begin{array}{l} s \leftarrow 0 \\ \text{for } i \in 1..n \\ \quad s \leftarrow s + i \end{array} \right. \qquad \begin{array}{l} s(5) = 15 \\ \\ s(10) = 55 \end{array}$$

Вищеописана програма складена з трьох операторів: перший оператор присвоює змінній ***s*** початкове значення – нуль. Другий оператор задає цикл з розрахунками, в якому параметр циклу ***i*** приймає значення від ***1*** до ***n,*** крок – ***1***, а третій оператор буде підсумовувати попереднє значення ***s*** для поточного ***i***. Формальний параметр ***n*** призначений для передачі програмі конкретне значення.

Приклад. Обрахувати значення функції $y = \sin(x)$ при заданих значеннях змінної $x = 2{,}37;\ 2.38,\ \ldots,\ 5{,}52$ **.**

Приклади розв’язку наведено на рисунках 1.24 і 1.25:

– записати умовау й задати фукцію y(a,b):=; (рисунок 1.24)

– використовуючи оператор **Add Line** задати програмні блоки, рис. 1.25;

– записати межі, в яких змінюється функція – (2.37,2.52), натиснути на клавіатурі знак дорівнює, або обрати з панелі **Evaluation** та отримати результат, рисунок 1.25

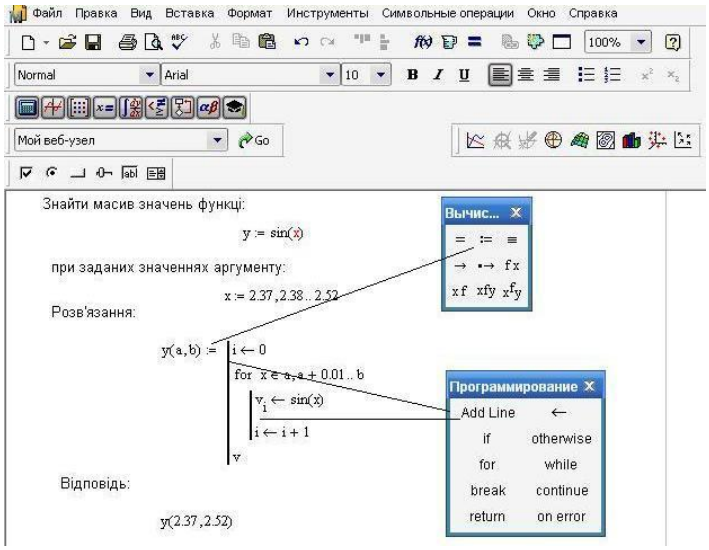


Рисунок 1.24 – Приклад циклу **for**

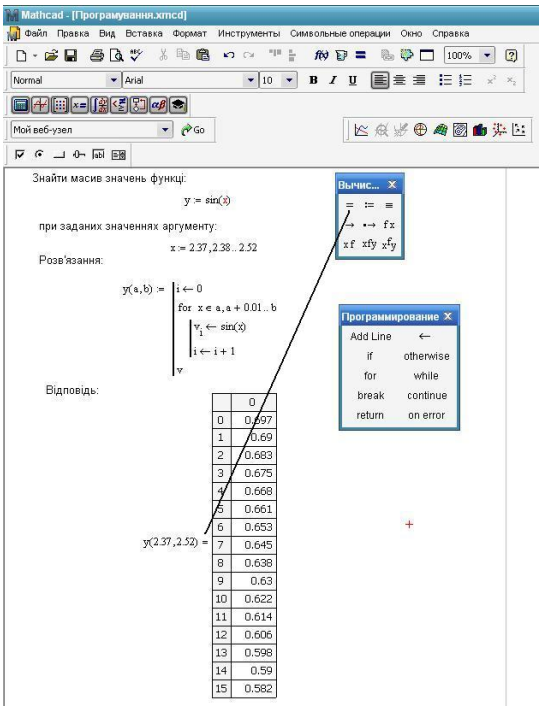


Рисунок 1.25 –Результати роботи циклу **for**

Наступні приклади – варіанти розрахунків факторіалу цілого числа з використанням програмного блока, застосовуючи:

а) оператор циклу **for:**

```
pr(n) := | p ← 1                    pr(3) = 6
         | for i ∈ 1..n
         |   p ← p · i              pr(5) = 120 ;
```

б) оператор циклу **while:**

```
fc(n) := | f ← 1
         | while n ← n − 1
         |   f ← f · (n + 1)
         | f                        fc(5) = 120 ;
```

в) операторів циклу **while** та **break:**

```
fct(n) := | f ← n
          | while 1
          |   | f ← f · (n − 1)
          |   | n ← n − 1
          |   | break if n = 1
          | f                      fct(5) = 120
```

Оператор error.

Оператор **error** діє тоді, коли результати в інших рядках програмного блоку не визначені, разом з цим оператором в дужках позначають вміст спеціального повідомлення, яке потрібно вивести.

Приклад. Функція ***R(i)*** буде виводити на екран текстові дані, якщо параметру ***i*** присвоєні значення ***1*** або ***2,*** інакше функція буде не визначена, і курсор залишиться в межах оператора виведення. На екрані робочого полябуде виведено повідомлення (тут виводиться текст **no!no!**), рисунок 1.26.

$$R(i) := \begin{cases} \text{"One"} & \text{if } i = 1 \\ \text{"Two"} & \text{if } i = 2 \\ \text{error("no!no!")} & \text{otherwise} \end{cases}$$

$R(1) = \text{"One"}$

$R(2) = \text{"Two"}$

$R(3) = \blacksquare\blacksquare$

no!no!

Рисунок 1.26 – Приклад програми з використанням оператора **error**

Оператор on error.

Синтаксичний запис оператора:

$$\blacksquare \quad \text{on error} \quad \blacksquare$$

Зліва необхідно ввести функцію або текстовий рядок з повідомленням, який буде активізовано, якщо неможливо розрахувати функцію.

Приклад. Програма повинна вивести на екран повідомлення з результатами розрахунку функції, якщо його немає можливості обрахувати, як у прикладі, рисунок 1.27, тоді оператор **on error** виведе результат обчислення вказаної зліва оператора функції – t(0)=0.

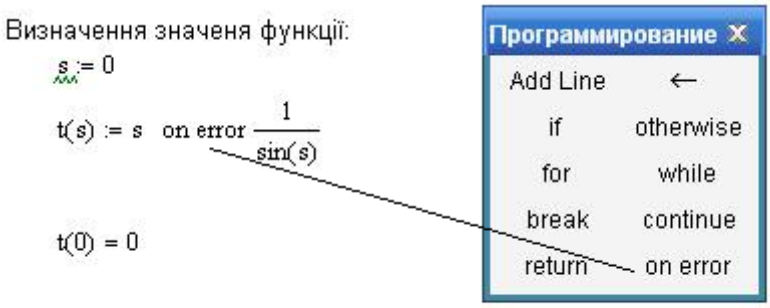


Рис. 1.27 – Приклад програми оператором **error**

1.9 Індивідуальні завдання для розв'язку в середовищі MathCAD

Варіант 1

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n^2+2}-\sqrt{n^2-2}\right)$$

$$\lim_{n\to\infty}\left(\sqrt{(n^2+2)(n^2-3)}-\sqrt{n^4-8}\right)$$

$$\lim_{n\to\infty}\left(\frac{3n^2+2}{3n^2+1}\right)^{n^2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{1-\cos 5x}{e^{x^2}-1}$$

$$\lim_{x\to 0}\frac{e^{\sin 2x}-e^{\sin 3x}}{\operatorname{tg} 2x}$$

$$\lim_{x\to 1}\left(\frac{2-x}{x}\right)^{1/\ln(3-2x)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{4x^2+1}\right)+x\left(x^2+3\sqrt{4x^2+1}\right)$$

$$y=\frac{x\arccos 3x}{\sqrt{1-9x^2}}+\ln\sqrt{1-9x^2}$$

$$y=2\ln\frac{x}{1+\sqrt{1-x^2}}-\frac{\sqrt{1-x^2}}{2}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+11x^2+16x+10}{(x+1)^2(x^2+x+2)}dx$$

$$\int(x^2-px+1)\cdot e^{2x}dx$$

$$\int x^2\cos 2x dx$$

5. Обчислити визначений інтеграл

$$\int_1^2 x^2\sqrt[5]{2-x}dx$$

$$\int_{-2}^{1}\frac{xdx}{x^3+1}$$

$$\int_{\pi/2}^{\pi} x\sin^3 x dx$$

Варіант 2

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n(n-1)}-\sqrt{n^2+2}\right)$$

$$\lim_{n\to\infty} \frac{\sqrt{n^5-6}-n\sqrt{n(n^2+4)}}{\sqrt{n}}$$

$$\lim_{n\to\infty}\left(\frac{5n+3}{5n+1}\right)^{n+2}$$

2. Знайти границю функції

$$\lim_{x\to 0} \frac{\ln(1-2\sin x)}{\sin 3x}$$

$$\lim_{x\to -1} \frac{x^2-1}{\sin 2\pi x}$$

$$\lim_{x\to 4}\left(\frac{2x-3}{x+1}\right)^{1/(\sqrt{x}-2)}$$

3. Знайти похідну функції

$$y=\frac{x^3}{\arcsin x}+\frac{x^2+1}{x}\sqrt{1-x^2}$$

$$y=\ln\left(2x+\sqrt{1+4x^2}\right)-\sqrt{1+4x^2}\,\operatorname{arcctg} 2x$$

$$y=3\arccos\frac{1}{2x+3}-2\sqrt{x^2+3x+2}\ ,\ 2x+3>0$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3-2x^2+x+1}{(x-1)^2(x^2+x+1)}dx$$

$$\int(x+2)^2\cdot e^{-3x}dx$$

$$\int x\sin^2 2xdx$$

5. Обчислити визначений інтеграл

$$\int_{\pi/4}^{\pi/2} x^2\cos 3xdx$$

$$\int_{e/2}^{1} x^2\ln 2xdx$$

$$\int_0^{1/2}\arccos\sqrt{1-2x^2}\,dx$$

Варіант 3

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(n-\sqrt[3]{n^3-4}\right)\cdot n\sqrt{n}$$

$$\lim_{n\to\infty}\left(n+\sqrt[3]{3-n^3}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n-1}{2n+3}\right)^{n-2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\cos 3x-\cos 5x}{\cos x-1}$$

$$\lim_{x\to 0}\frac{2^x+2^{-x}-2}{1-\cos 3x}$$

$$\lim_{x\to 1}\left(\frac{2x-1}{x}\right)^{1/(\sqrt{x}-1)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{1+9x^2}\right)-x\left(x^2+3\sqrt{1+9x^2}\right)$$

$$y=\ln\left(x-\sqrt{1+4x^2}\right)+\frac{\sqrt{1+4x^2}}{x}$$

$$y=\sqrt{4-10x-3x^2}+\frac{1}{2\sqrt{3}}\arcsin\frac{3x+5}{3\sqrt{3}}$$

4. Знайти невизначений інтеграл

$$\int\frac{2x^3+2x^2+6x+1}{\left(x^2+4\right)\left(x^2+x+1\right)}dx$$

$$\int\left(x^2+1\right)\ln x\,dx$$

$$\int\left(x^2+4\right)\operatorname{arctg}x\,dx$$

5. Обчислити визначений інтеграл

$$\int_1^3\left(x-\left|2-x^2\right|\right)dx$$

$$\int_1^2\frac{\ln^2 x}{x\sqrt[3]{1+\ln x}}dx$$

$$\int_{2e}^{3e}\frac{dx}{x\ln x\ln\ln x}$$

Варіант 4

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(\sqrt{n^2-5n+6}-n\right)$$

$$\lim_{n\to\infty}\left(\sqrt{n(n+3)}-\sqrt{n^2-3n+2}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n^2-n+3}{2n^2+2n+1}\right)^{-n+1}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\arcsin 2x}{x^2+\pi x}$$

$$\lim_{x\to 0}\frac{\sqrt{1+\sin 2x}-\sqrt{1+\operatorname{tg} 2x}}{x^3}$$

$$\lim_{x\to\pi}\left(\cos 2x\right)^{3/\sin^2 x}$$

3. Знайти похідну функції

$$y=\sqrt{(6+x)(2-x)}+2\ln\left(\sqrt{5+x}+\sqrt{2-x}\right)$$

$$y=\operatorname{arctg}\frac{3x+1}{\sqrt{5}}+\ln\frac{\sqrt{3x^2+2x+2}}{x}$$

$$y=\ln\left(\sqrt[3]{\frac{2x-1}{2x+1}}\right)-\arcsin\sqrt{\frac{2x-1}{2x+1}}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+10x^2+14x+5}{(x+1)^2(x^2+3x+4)}dx$$

$$\int\ln^3 x\,dx$$

$$\int(x-1)^2\arcsin 3x\,dx$$

5. Обчислити визначений інтеграл

$$\int_0^1\frac{dx}{e^{-x}+e^{-3x}}$$

$$\int_0^1 x^2\operatorname{arctg}\pi x\,dx$$

$$\int_{1/e}^{e}\left(-|\ln x|\right)^2 dx$$

Варіант 5

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n^2+2}-\sqrt{n^2-2}\right)$$

$$\lim_{n\to\infty}\left(\sqrt{(n^2+2)(n^2-3)}-\sqrt{n^4-8}\right)$$

$$\lim_{n\to\infty}\left(\frac{3n^2+2}{3n^2+1}\right)^{n^2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{1-\cos 5x}{e^{x^2}-1}$$

$$\lim_{x\to 0}\frac{e^{\sin 2x}-e^{\sin 3x}}{\operatorname{tg} 2x}$$

$$\lim_{x\to 1}\left(\frac{2-x}{x}\right)^{1/\ln(3-2x)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{4x^2+1}\right)+x\left(x^2+3\sqrt{4x^2+1}\right)$$

$$y=\frac{x\arccos 3x}{\sqrt{1-9x^2}}+\ln\sqrt{1-9x^2}$$

$$y=2\ln\frac{x}{1+\sqrt{1-x^2}}-\frac{\sqrt{1-x^2}}{2}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+11x^2+16x+10}{(x+1)^2(x^2+x+2)}dx$$

5. Обчислити визначений інтеграл

$$\blacksquare := \blacksquare$$

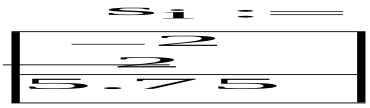
$$\blacksquare = \blacksquare$$

$$\blacksquare \equiv \blacksquare$$

Варіант 6

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n(n-1)}-\sqrt{n^2+2}\right)$$



$$\lim_{n\to\infty}\left(\frac{5n+3}{5n+1}\right)^{n+2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\ln(1-2\sin x)}{\sin 3x}$$

$$\lim_{x\to -1}\frac{x^2-1}{\sin 2\pi x}$$

$$\lim_{x\to 4}\left(\frac{2x-3}{x+1}\right)^{1/(\sqrt{x}-2)}$$

3. Знайти похідну функції

$$y=\frac{x^3}{\arcsin x}+\frac{x^2+1}{x}\sqrt{1-x^2}$$

$$y=\ln\left(2x+\sqrt{1+4x^2}\right)-\sqrt{1+4x^2}\,\operatorname{arcctg}2x$$

$$y=3\arccos\frac{1}{2x+3}-2\sqrt{x^2+3x+2}\ ,\ 2x+3>0$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3-2x^2+x+1}{(x-1)^2(x^2+x+1)}dx$$

$$\int(x+2)^2\cdot e^{-3x}dx$$

$$\int x\sin^2 2x dx$$

5. Обчислити визначений інтеграл

$$\int_{\pi/4}^{\pi/2} x^2\cos 3x dx$$

$$\int_{e/2}^{1} x^2\ln 2x dx$$

$$\int_0^{1/2}\arccos\sqrt{1-2x^2}\,dx$$

Варіант 7

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(n-\sqrt[3]{n^3-4}\right)\cdot n\sqrt{n}$$

$$\lim_{n\to\infty}\left(n+\sqrt[3]{3-n^3}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n-1}{2n+3}\right)^{n-2}$$

2 Знайти границю функції

$$\lim_{x\to 0}\frac{\cos 3x-\cos 5x}{\cos x-1}$$

$$\lim_{x\to 0}\frac{2^x+2^{-x}-2}{1-\cos 3x}$$

$$\lim_{x\to 1}\left(\frac{2x-1}{x}\right)^{1/(\sqrt{x}-1)}$$

3 Знайти похідну функції

$$y=\ln\left(x+\sqrt{1+9x^2}\right)-x\left(x^2+3\sqrt{1+9x^2}\right)$$

$$y=\ln\left(x-\sqrt{1+4x^2}\right)+\frac{\sqrt{1+4x^2}}{x}$$

$$y=\sqrt{4-10x-3x^2}+\frac{1}{2\sqrt{3}}\arcsin\frac{3x+5}{3\sqrt{3}}$$

4 Знайти невизначений інтеграл

$$\int\frac{2x^3+2x^2+6x+1}{\left(x^2+4\right)\left(x^2+x+1\right)}dx$$

$$\int\left(x^2+1\right)\ln x\,dx$$

$$\int\left(x^2+4\right)\operatorname{arctg}x\,dx$$

5 Обчислити визначений інтеграл

$$\int_1^3\left(x-\left|2-x^2\right|\right)dx$$

$$\int_1^2\frac{\ln^2 x}{x\sqrt[3]{1+\ln x}}dx$$

$$\int_{2e}^{3e}\frac{dx}{x\ln x\ln\ln x}$$

Варіант 8

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(\sqrt{n^2-5n+6}-n\right)$$

$$\lim_{n\to\infty}\left(\sqrt{n(n+3)}-\sqrt{n^2-3n+2}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n^2-n+3}{2n^2+2n+1}\right)^{-n+1}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\arcsin 2x}{x^2+\pi x}$$

$$\lim_{x\to 0}\frac{\sqrt{1+\sin 2x}-\sqrt{1+\operatorname{tg} 2x}}{x^3}$$

$$\lim_{x\to\pi}(\cos 2x)^{3/\sin^2 x}$$

3. Знайти похідну функції

$$y=\sqrt{(6+x)(2-x)}+2\ln\left(\sqrt{5+x}+\sqrt{2-x}\right)$$

$$y=\operatorname{arctg}\frac{3x+1}{\sqrt{5}}+\ln\frac{\sqrt{3x^2+2x+2}}{x}$$

$$y=\ln\left(\sqrt[3]{\frac{2x-1}{2x+1}}\right)-\arcsin\sqrt{\frac{2x-1}{2x+1}}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+10x^2+14x+5}{(x+1)^2(x^2+3x+4)}dx$$

$$\int\ln^3 x\,dx$$

$$\int(x-1)^2\arcsin 3x\,dx$$

5. Обчислити визначений інтеграл

$$\int_0^1\frac{dx}{e^{-x}+e^{-3x}}$$

$$\int_0^1 x^2\operatorname{arctg}\pi x\,dx$$

$$\int_{1/e}^{e}(-|\ln x|)^2 dx$$

Варіант 9

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n^2+2}-\sqrt{n^2-2}\right)$$

$$\lim_{n\to\infty}\left(\sqrt{(n^2+2)(n^2-3)}-\sqrt{n^4-8}\right)$$

$$\lim_{n\to\infty}\left(\frac{3n^2+2}{3n^2+1}\right)^{n^2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{1-\cos 5x}{e^{x^2}-1}$$

$$\lim_{x\to 0}\frac{e^{\sin 2x}-e^{\sin 3x}}{\operatorname{tg} 2x}$$

$$\lim_{x\to 1}\left(\frac{2-x}{x}\right)^{1/\ln(3-2x)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{4x^2+1}\right)+x\left(x^2+3\sqrt{4x^2+1}\right)$$

$$y=\frac{x\arccos 3x}{\sqrt{1-9x^2}}+\ln\sqrt{1-9x^2}$$

$$y=2\ln\frac{x}{1+\sqrt{1-x^2}}-\frac{\sqrt{1-x^2}}{2}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+11x^2+16x+10}{(x+1)^2\left(x^2+x+2\right)}dx$$

$$\int\left(x^2-px+1\right)\cdot e^{2x}dx$$

$$\int x^2\cos 2x dx$$

5. Обчислити визначений інтеграл

$$\int_1^2 x^2\sqrt[5]{2-x}dx$$

$$\int_{-2}^{1}\frac{xdx}{x^3+1}$$

$$\int_{\pi/2}^{\pi} x\sin^3 xdx$$

Варіант 10

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n(n-1)}-\sqrt{n^2+2}\right)$$

$$\lim_{n\to\infty} \frac{\sqrt{n^5-6}\ -n\sqrt{n(n^2+4)}}{\sqrt{n}}$$

$$\lim_{n\to\infty}\left(\frac{5n+3}{5n+1}\right)^{n+2}$$

2. Знайти границю функції

$$\lim_{x\to 0} \frac{\ln(1-2\sin x)}{\sin 3x}$$

$$\lim_{x\to -1} \frac{x^2-1}{\sin 2\pi x}$$

$$\lim_{x\to 4}\left(\frac{2x-3}{x+1}\right)^{1/(\sqrt{x}-2)}$$

3. Знайти похідну функції

$$y=\frac{x^3}{\arcsin x}+\frac{x^2+1}{x}\sqrt{1-x^2}$$

$$y=\ln\left(x+\sqrt{1+4x^2}\right)-\sqrt{1+4x^2}\ \operatorname{arcctg} 2x$$

$$y=3\arccos\frac{1}{2x+3}-2\sqrt{x^2+3x+2}\ ,\ 2x+3>0$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3-2x^2+x+1}{(x-1)^2(x^2+x+1)}dx$$

$$\int(x+2)^2\cdot e^{-3x}dx$$

$$\int x\sin^2 2xdx$$

5. Обчислити визначений інтеграл

$$\int_{\pi/4}^{\pi/2} x^2\cos 3xdx$$

$$\int_{e/2}^{1} x^2\ln 2xdx$$

$$\int_0^{1/2}\arccos\sqrt{1-2x^2}\,dx$$

Варіант 11

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(n-\sqrt[3]{n^3-4}\right)\cdot n\sqrt{n}$$

$$\lim_{n\to\infty}\left(n+\sqrt[3]{3-n^3}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n-1}{2n+3}\right)^{n-2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\cos 3x-\cos 5x}{\cos x-1}$$

$$\lim_{x\to 0}\frac{2^x+2^{-x}-2}{1-\cos 3x}$$

$$\lim_{x\to 1}\left(\frac{2x-1}{x}\right)^{1/(\sqrt{x}-1)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{1+9x^2}\right)-x\left(x^2+3\sqrt{1+9x^2}\right)$$

$$y=\ln\left(x-\sqrt{1+4x^2}\right)+\frac{\sqrt{1+4x^2}}{x}$$

$$y=\sqrt{4-10x-3x^2}+\frac{1}{2\sqrt{3}}\arcsin\frac{3x+5}{3\sqrt{3}}$$

4. Знайти невизначений інтеграл

$$\int\frac{2x^3+2x^2+6x+1}{\left(x^2+4\right)\left(x^2+x+1\right)}dx$$

$$\int\left(x^2+1\right)\ln x\,dx$$

$$\int\left(x^2+4\right)\operatorname{arctg}x\,dx$$

5. Обчислити визначений інтеграл

$$\int_1^3\left(x-\left|2-x^2\right|\right)^3dx$$

$$\int_1^2\frac{\ln^2 x}{x\sqrt[3]{1+\ln x}}dx$$

$$\int_{2e}^{3e}\frac{dx}{x\ln x\ln\ln x}$$

Варіант 12

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(\sqrt{n^2-5n+6}-n\right)$$

$$\lim_{n\to\infty}\left(\sqrt{n(n+3)}-\sqrt{n^2-3n+2}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n^2-n+3}{2n^2+2n+1}\right)^{-n+1}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\arcsin 2x}{x^2+\pi x}$$

$$\lim_{x\to 0}\frac{\sqrt{1+\sin 2x}-\sqrt{1+\operatorname{tg} 2x}}{x^3}$$

$$\lim_{x\to\pi}(\cos 2x)^{3/\sin^2 x}$$

3. Знайти похідну функції

$$y=\sqrt{(6+x)(2-x)}+2\ln\left(\sqrt{5+x}+\sqrt{2-x}\right)$$

$$y=\operatorname{arctg}\frac{3x+1}{\sqrt{5}}+\ln\frac{\sqrt{3x^2+2x+2}}{x}$$

$$y=\ln\left(\sqrt[3]{\frac{2x-1}{2x+1}}\right)-\arcsin\sqrt{\frac{2x-1}{2x+1}}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+10x^2+14x+5}{(x+1)^2(x^2+3x+4)}dx$$

$$\int\ln^3 x\,dx$$

$$\int(x-1)^2\arcsin 3x\,dx$$

5. Обчислити визначений інтеграл

$$\int_0^1\frac{dx}{e^{-x}+e^{-3x}}$$

$$\int_0^1 x^2\operatorname{arctg}\pi x\,dx$$

$$\int_{1/e}^{e}(-|\ln x|)^2 dx$$

Варіант 13

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n^2+2}-\sqrt{n^2-2}\right)$$

$$\lim_{n\to\infty}\left(\sqrt{(n^2+2)(n^2-3)}-\sqrt{n^4-8}\right)$$

$$\lim_{n\to\infty}\left(\frac{3n^2+2}{3n^2+1}\right)^{n^2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{1-\cos 5x}{e^{x^2}-1}$$

$$\lim_{x\to 0}\frac{e^{\sin 2x}-e^{\sin 3x}}{\operatorname{tg} 2x}$$

$$\lim_{x\to 1}\left(\frac{2-x}{x}\right)^{1/\ln(3-2x)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{4x^2+1}\right)+x\left(x^2+3\sqrt{4x^2+1}\right)$$

$$y=\frac{x\arccos 3x}{\sqrt{1-9x^2}}+\ln\sqrt{1-9x^2}$$

$$y=2\ln\frac{x}{1+\sqrt{1-x^2}}-\frac{\sqrt{1-x^2}}{2}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+11x^2+16x+10}{(x+1)^2(x^2+x+2)}dx$$

$$\int(x^2-px+1)\cdot e^{2x}dx$$

$$\int x^2\cos 2x dx$$

5. Обчислити визначений інтеграл

$$\int_1^2 x^2\sqrt[5]{2-x}dx$$

$$\int_{-2}^{1}\frac{xdx}{x^3+1}$$

$$\int_{\pi/2}^{\pi} x\sin^3 x dx$$

Варіант 14

1. Знайти границю числової послідовності

$$\lim_{n\to\infty} n\left(\sqrt{n(n-1)}-\sqrt{n^2+2}\right)$$

$$\lim_{n\to\infty} \frac{\sqrt{n^5-6}\ -n\sqrt{n(n^2+4)}}{\sqrt{n}}$$

$$\lim_{n\to\infty}\left(\frac{5n+3}{5n+1}\right)^{n+2}$$

2. Знайти границю функції

$$\lim_{x\to 0} \frac{\ln(1-2\sin x)}{\sin 3x}$$

$$\lim_{x\to -1} \frac{x^2-1}{\sin 2\pi x}$$

$$\lim_{x\to 4}\left(\frac{2x-3}{x+1}\right)^{1/(\sqrt{x}-2)}$$

3. Знайти похідну функції

$$y=\frac{x^3}{\arcsin x}+\frac{x^2+1}{x}\sqrt{1-x^2}$$

$$y=\ln\left(x+\sqrt{1+4x^2}\right)-\sqrt{1+4x^2}\,\operatorname{arcctg} 2x$$

$$y=3\arccos\frac{1}{2x+3}-2\sqrt{x^2+3x+2}\ ,\ 2x+3>0$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3-2x^2+x+1}{(x-1)^2(x^2+x+1)}dx$$

$$\int(x+2)^2\cdot e^{-3x}dx$$

$$\int x\sin^2 2xdx$$

5. Обчислити визначений інтеграл

$$\int_{\pi/4}^{\pi/2} x^2\cos 3xdx$$

$$\int_{e/2}^{1} x^2\ln 2xdx$$

$$\int_0^{1/2}\arccos\sqrt{1-2x^2}\,dx$$

Варіант 15

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(n-\sqrt[3]{n^3-4}\right)\cdot n\sqrt{n}$$

$$\lim_{n\to\infty}\left(n+\sqrt[3]{3-n^3}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n-1}{2n+3}\right)^{n-2}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\cos 3x-\cos 5x}{\cos x-1}$$

$$\lim_{x\to 0}\frac{2^x+2^{-x}-2}{1-\cos 3x}$$

$$\lim_{x\to 1}\left(\frac{2x-1}{x}\right)^{1/(\sqrt{x}-1)}$$

3. Знайти похідну функції

$$y=\ln\left(x+\sqrt{1+9x^2}\right)-x\left(x^2+3\sqrt{1+9x^2}\right)$$

$$y=\ln\left(x-\sqrt{1+4x^2}\right)+\frac{\sqrt{1+4x^2}}{x}$$

$$y=\sqrt{4-10x-3x^2}+\frac{1}{2\sqrt{3}}\arcsin\frac{3x+5}{3\sqrt{3}}$$

4. Знайти невизначений інтеграл

$$\int\frac{2x^3+2x^2+6x+1}{\left(x^2+4\right)\left(x^2+x+1\right)}dx$$

$$\int\left(x^2+1\right)\ln x\,dx$$

$$\int\left(x^2+4\right)\operatorname{arctg} x\,dx$$

5. Обчислити визначений інтеграл

$$\int_1^3\left(x-\left|2-x^2\right|\right)dx$$

$$\int_1^2\frac{\ln^2 x}{x\sqrt[3]{1+\ln x}}dx$$

$$\int_{2e}^{3e}\frac{dx}{x\ln x\ln\ln x}$$

Варіант 16

1. Знайти границю числової послідовності

$$\lim_{n\to\infty}\left(\sqrt{n^2-5n+6}-n\right)$$

$$\lim_{n\to\infty}\left(\sqrt{n(n+3)}-\sqrt{n^2-3n+2}\right)$$

$$\lim_{n\to\infty}\left(\frac{2n^2-n+3}{2n^2+2n+1}\right)^{-n+1}$$

2. Знайти границю функції

$$\lim_{x\to 0}\frac{\arcsin 2x}{x^2+\pi x}$$

$$\lim_{x\to 0}\frac{\sqrt{1+\sin 2x}-\sqrt{1+\operatorname{tg} 2x}}{x^3}$$

$$\lim_{x\to\pi}\left(\cos 2x\right)^{\frac{3}{\sin^2 x}}$$

3. Знайти похідну функції

$$y=\sqrt{(6+x)(2-x)}+2\ln\left(\sqrt{5+x}+\sqrt{2-x}\right)$$

$$y=\operatorname{arctg}\frac{3x+1}{\sqrt{5}}+\ln\frac{\sqrt{3x^2+2x+2}}{x}$$

$$y=\ln\left(\sqrt[3]{\frac{2x-1}{2x+1}}\right)-\arcsin\sqrt{\frac{2x-1}{2x+1}}$$

4. Знайти невизначений інтеграл

$$\int\frac{3x^3+10x^2+14x+5}{(x+1)^2(x^2+3x+4)}dx$$

$$\int\ln^3 x\,dx$$

$$\int(x-1)^2\arcsin 3x\,dx$$

5. Обчислити визначений інтеграл

$$\int_0^1\frac{dx}{e^{-x}+e^{-3x}}$$

$$\int_0^1 x^2\operatorname{arctg}\pi x\,dx$$

$$\int_{\frac{1}{e}}^{e}\left(-|\ln x|\right)^2 dx$$

2 ВИКОРИСТАННЯ СЕРЕДОВИЩА MATLAB

2.1 Команди, функції середовища MATLAB

Після запуску MATLAB-у на екрані з'являється головне вікно, яке містить меню та інструментальну лінійку з кнопками та клієнтську частину вікна із запрошенням «>>», яку називають командним вікном.

У командне вікно з клавіатури є можливітсть вводити команди, які складаються з літер, цифр та знаків операцій. Натискання клавіші "Enter "буде для транслятора середовища сигналом для виконання відповідної команди.

2.1.1 Корисні команди

Знайомство з MATLAB корисно починати з команд операторів загального призначення, які призначені для полегшення роботи користувача:

• *help* – виводить список TOOLBOX'ів, що підключено до роботи.

• *help help* - виводить на екран інформацію про роботу довідника з роботи у середовищі help.

• *help «ім'я функції»* - виводить допомогу про функцію, тобото, текст поміщений в початкові рядки коментарів М-файлу з текстом функції, помічених символом '%' (коментарі). Для вбудованих функцій середовища в папках TOOLBOX записаний М-файл, що містить тільки рядки коментарів, чиє ім’я співпадає з назвою функції.

• *help «ім'я файлу»* - виводить інформацію вмісту М-файлу аналогічно інформації про функцію.

• *lookfor «набір символів»* - виводить список М-файлів, що доступні для звернення, в яких у рядках коментарів зустрічається заданий набір символів.

• *demo* - список демонстраційних прикладів.

• *casesen « on/off»* - перемикання розрізнення регістрів: при відсутності опцій,

on буде встановлено режим розрізнення регістрів, *off* – відмінено відповідно.

• *diary «ім'я файла/on/off »* - керує режимом паралельного запису в файл. Якщо ім'я файлу вказане, встановлюється режим, згідно з яким все, що виводиться на екран, одночасно буде записано у файл з цим ім'ям. Опція *off* відключає цей режим, опція *on* повертає знову. Існуючий файл можна відкривати тільки на дозапис.

• *format «flag»* - керування форматом виведення результатів у командне вікно. Всі обчислення в MATLAB відбуваються з точністю 32-розрядної арифметики, але виведення значень у командне вікно відбувається з меншою точністю. При таких значеннях параметра *«flag»* встановлюються наступні види формату виведення:

—*short* - формат з фіксованою крапкою та з п’ятьма значущими цифрами (використовується за замовчуванням), встановлюється при записі *format* без вказівки опції *flag.*

—*long* - формат з фіксованою крапкою з 15 значущими символами.

—*short e* - формат з плаваючою крапкою з 5 значущими символами.

—*long e* - формат з плаваючою крапкою з 15 значущими символами.

—*short g*- вибирається кращий з двох попередніх форматів з 5 значущими символами.

—*long g*- вибирається кращий з двох попередніх форматів з 15 значущими символами.

—*hex* – шістнадцятковий формат.

—+ – відображується тільки знак.

—*rat* – вираз результату у вигляді відношення цілих чисел.

Такі ж формати виведення є можливість встановити у вікні *General* при виклику пункту підміню *Preferences* у пункті *File* головного меню.

• *home* - встановлює курсор з поточним командним рядком у верхній лівий кут вікна.

• *clc* - очищення командного вікна.

• *computer* - дає довідку про тип комп'ютера (Для операційних систем після Windows 95 і Windows NT і їх спадкоємців команда практично даремна, оскільки повідомляє тільки тип операційної системи).

• *exit, quit*- ці команди припиняють роботу MATLAB.

• ; - дуже корисний символ. Виведення на екран результату виконання команди в робоче вікно відміняється, якщо команда закінчується цим символом.

2.1.2 Запис змінніх. Змінні: вектора і матриці

Імена змінних в MATLAB'і можуть позначатися довільним набором літер, цифр і знаків підкреслювання ('_'). Вони повинні починатися з букви і мати не більше 31 символу. При цьому не рекомендується використовувати імена операторів і функцій MATLAB'у та імена стандартних змінних, використовуваних пакетом:

- *i,j*- уявна одиниця ($\sqrt{-1}$);
- *inf* - невизначеність типу 1/0 (∞);
- *NAN* - невизначеність типу 0/0 (Not a number);
- *ans* - результат останньої виконуваної операції або функції (якщо в командному рядку відсутній оператор присвоювання, результат операції присвоюється змінній *ans* автоматично);
- *pi* - π = 4 * *atan(1)* = 3.1415926 ...;
- *rand* - псевдовипадкове число рівномірне розподілене на інтервалі [0,1];
- *eps* - відносна точність обчислень. За цю величину береться відстань від 1 до наступного дійсного числа доступного комп'ютеру;
- *realmin, realmax* - мінімальне і максимальне дійсні числа.

Основним об'єктом MATLAB'у є матриця. Таким чином, з точки зору середовища, вектором називається матриця, одна з розмірностей якої дорівнює одиниці, а скаляром – матриця 1x1.

Тому за замовчуванням усі основні функції та оператори MATLABу є матричними, окрім випадків, обумовлених особливо.

Для введення скаляру використовують оператор присвоювання: *a* = 2.34.

Для формування матриць і векторів використовуються символи «[…]». Наприклад, матриця 2х3 може бути записана у вигляді *a* = [1 3.3 2; 4.4 3 3]. Різні елементи розділяють пробілом або комою, рядки відокремлюють один від одного записом ';'. Різне число записів у рядках або стовпчих викликає помилку, про яку MATLAB повідомить користувача.

Згідно з вищесказаними, оператором *b* = [1 2 3] буде введено вектор-рядок, *c* = [1; 2; 3] – вектор-стовчик.

Такі ж правила використовуються й при формуванні блокових матриць. Наприклад, якщо виконані присвоювання попереднього абзацу, оператор *d = [a; b]* створить матрицю

$$d = \begin{bmatrix} 1 & 3.3 & 2 \\ 4.4 & 3 & 3 \\ 1 & 2 & 3 \end{bmatrix}$$

Звернення до одного з елементів матриці *a,* розташованого на перетині рядка *i* та стовпчика *j* має вигляд – *a(i,j)*. Вектор-стовпець, що співпадаэ з *j-м* стовпчиком матриці *a*, можна отримати, записавши *a(:,j)*, відповідно *i-й* вектор-рядок – *a(i,* :).

Підматрицю з матриці *a* можна виразити, якщо вказати замість індексів вектора *u* і *v*, які містять номери рядків і стовпчиків матриці *a,* на перетині яких розташовані елементи підматриці.

Приклад: для матриці d, що наведена раніше, і векторів *u* = [1 3] і *v* = [1 2]

$$a = d(u,v) = \begin{bmatrix} 1 & 3.3 \\ 1 & 2 \end{bmatrix} \qquad (2.1)$$

В загальному випадку значення компонентів векторів *u і v в* приведеному операторі при обчисленні індексів будуть округлені до найближчого цілого числа.

Блок матриці можна отримати також використавши символ перерахування ':'. Оператор

$$u = start : step : fin \qquad (2.2)$$

задає вектор арифметичної прогресії, перший член якої дорівнює *start,* крок – *step,* останнй – найближчий до величин *fin*, що належить проміжну, межами якого є *start* і *fin.* При *step* > 0 останнй компонент вектора, що сформований, буде максимальне число, що належить проміжку *[start, fin],* при *step* < 0 – мінімальний, що належить проміжку *[fin, start]* відповідно. Якщо значення кроку не вказано,*:* величина *step* – крок приймається 1. Таким чином, матрицю *a* з (1) можна також задати, наприклад, так:

$$a = d(1 : 1.6 : 3,1 : 2).$$

У MATLAB'і можна задати символьні змінні, що сприймаються середовищем, як вектора символів. Найпростіше задати таку змінну командою:

c ='KPI'

Вона еквівалентна команді *c=['K', 'P', 'I’].*

Необхідні операції над рядками виконуються за правилами векторних операцій, описаних в цьому розділі.

2.1.3 Прості арифметичні операції

Символьні позначення основних операцій, що проводяться над змінними:

- = – присвоювання;
- + – складання;
- * – множення;
- \ – ділення зліва (для матричних величин результат виконання операції *X*

$= A\backslash B$ приблизно дорівнює $A^{-1}*B$, еквівалентний розв'язку рівняння $A*X = B$ методом Гауса. Якщо матриця A буде погано обумовлена або вироджена буде виведено відповідне попередження;

- / – ділення справа (для матричних величин результат виконання операції $X = A/B$ – $A*B^{-1}$, і еквівалентний розв'язку матричного рівняння $X * B = A$ подібний до реалізації оператора \);
- ^ - піднесення до степеня;
- ' – транспонування;
- ./ - поелементне ділення справа;
- .^ - поелементне піднесення до степеня;

Вищеназвані дії виконуються з урахуванням правил матричної алгебри. Істотною перевагою пакету MATLAB є організована перевірка розмірності при обчисленнях. Невідповідність розмірностей операндів призводить до помилки трансляції.

Послідовність операцій визначається відповідно правилам алгебри, для виділення першочергових дії використовуються круглі дужки ( ).

2.1.4 Робоча область і операції над нею

Для змінних, що задані при роботі в пакеті, завжди виділяється область пам'яті комп'ютера, що називається робочою областю *(workspace).* Проглянути вміст робочої області можна командами *who* і *whos.*

•*who* - виводить список змінних, збережених в робочій області

•*whos* - виводить на екран список змінних з інформацією про їх тип та розмір. В якості прикладу можна навести результат роботи цієї команди, коли при роботі визначено такі змінні - скаляр *a1*, вектор-рядок *b3* з трьох елементів, матриця *c23* розмірністю *2×3* та комплексний вектор-стовпчик *dk* з двох елементів.

Name	Size	Bytes	Class
a1	lxl	8	double array
b3	1x3	24	double array
c23	2x3	48	double array

dk 2x1 32 double array (complex)

Grand total is 12 elements using 112 bytes

Перевірити наявність змінної *a1* в робочій області пам'яті можна, використовуючи функцію *exist ('a1'),* що повертає 1, якщо змінна *a1* існує в робочому просторі; 2 - якщо *a1* – це ім'я файлу на диску, 0 – якщо змінної *a1* не існує.

Для збереження і відновлення змінних, що розміщені в робочій області, використовуються команди *save* та *load.*

• *save* - зберігає усі змінні робочої області у файлі *matlab.mat.*

• *save <ім'я файлу>* – зберігає всі змінні робочої області е файлі із розширенням **.mat.* Цю ж дію можна зробити викликом пункту підменю *Save Workspace as* в пункті *File* головного меню.

• *save <ім'я файлу> <змінні>* зберігає змінні в mat-файлі з заданим ім'ям. В якості змінних задають список їх імен через пробіл.

• *save <ім'я файлу> <змінні> –ascii* зберігає змінні у файлі в кодах ASCII, при цьому ім'я файлу може мати довільне розширення.

• *load* – буде відновлено змінні з файлу *matlab.mat* .

• *load <ім'я файлу>* - відновлює змінні в робочу область з файлу *mat-файлу* з заданим ім'ям. Це можна також зробити викликом пункту підміню *Load Workspace у* пункті *File* головного меню.

• *load <ім'я файлу>-ascii* - читає у робочу область з файлу матрицю в кодах ASCII в вигляді таблиці, стовпчики якої розділені пробілом. За відсутності ключа *-ascii* результат виконання цієї операції буде переформовано в масив, ім'я якого співпадає з ім'ям файлу з відкинутим розширенням. Ім'я цього файлу повинне починатися з букви і не повинно містити знаків типу '-', '+' і тощо.

Видалити змінні з робочої області можна командою *clear*.

• *clear* - видаляє всі змінні робочої області.

• *clear < змінні>* видаляє перераховані списком через пробіл змінні.

2.1.5 Робоча папка

За відсутності вказаного повного шляху до файлів пакет MATLAB здійснює пошук файлу для читання і проводить запис в так звану робочу папку.

Всі версії MATLAB'у дозволяють встановити шлях до робочої папки, вибраний користувачем. Універсальним є спосіб з використанням команди *cd:*

- *cd <шлях>* - встановлює папку, вказану змінною *<шлях>* як робочу.

Останню вимогу можна порушити, але тоді ім'я файлу повинне бути скрізь присутнім тільки у вигляді символьної константи. Робота з цим ім'ям дуже ускладнена і опис її відсутній.

Проглянути вміст цієї папки дозволяють команди:

- *dir* - виводить список файлів, які містить робоча папка;
- *what*- виводить список файлів MATLAB'у, які містить робоча папка.

В сучасних версіях MATLAB'у є способи встановити робочу папку, з використанням інструментальної панелі. У випадаючому меню пункту *File* головного меню командного вікна є пункт встановлення шляхів *Set Path..*, з використанням якого можливо змінити параметр *Current directory*, який вказує шлях до робочої папки. Знак цієї дії виведений на панель інструментів. У всіх варіантах 4-ої версії спеціальний пункт меню є відсутнім, але для зміни робочої папки можна спробувати відкрити файл, використавши підпункт *Open* у випадаючому меню пункту *File* головного меню.

При такій спробі середовище запропонує вибрати шлях до папки, у якій знаходиться файл, що призначений користувачем для зчитування та редагування.

Після відкриття такого файлу редактором MatLab, папка, в якій він знаходився, встановлюється в якості робочої. Якщо користувач не бажає відкривати ніяких файлів, можна відмовитися від операції відкриття файлу, обравши у вікні відкриття файлу кнопку "*cancel"*.

Остання з папок, які переглядав користувач, буде встановлена в якості робочої.

2.1.6 Програмування в MATLAB'і. Сценарії і функції

На відміну від найпростішого способа роботи у системі MATLAB - послідовного введення команд в командний рядок, їх обробки пакетом і видачи результату, більш ефективним є попередня підготовка послідовності команд, запис її файл та подальше багаторазове виконання цих команд. У середовищі MATLAB є два типи файлів, що дають змогу реалізувати цю можливість, це *M-сценарій* та *M-функція.*

- *M-сценарієм* називається файл, який містить послідовність команд, операцій та функцій, що оперують зі змінними робочої області. M-сценарій не має вхідних і вихідних аргументів як таких.

•*M-функцією* ,або просто функцією називається послідовність операцій та команд у файлі зі спеціальним заголовком, що оперує тільки над вхідними змінними.

2.1.6.1 Сценарії

M-сценарій записуєтья у файл з розширенням **.m* та містить визначену послідовність операторів і не вимагає заголовків та/або коментарів. Приклад m-сценарію:

```
c1=5.5;
d1=a/c1*cos(b1);
e1=a*(c*sin(b1)+cos(pi-b1));
f2=e1/d1
```

Створіть в робочій папці файл з ім'ям *my1stprog.m* (у для цього можна використати *Editor/Debugger),* наберіть заданий текст та збережіть його на диску. Тепер введіть у командному вікні значення змінних *a = 11*; *b = pi/2;* наберіть *my1stprog,* буде запущено виконання послідовності записаних операцій над змінними робочої області. Всі використані змінні *c1, d1, e1* і *f2* також буде збережено в робочій області та доступні для подальшого використання.

У програмах, що працюють з великими масивами даних, збереження допоміжних змінних і масивів не є доцільним, оскільки це веде до засмічення оперативної пам'яті комп'ютера, тому при програмуванні в MATLAB'і застосовується поняття функцій.

2.1.6.2 Функції

Текст функції записують у файл з розширенням **.m,* проте до цього файла існують додаткові вимоги. Перший оператор повинний містити опис функції:

function [outparam]=programname (inparam)

Цей опис показує, що в файлі записано функцію з ім'ям *programname* з вхідними параметрами, які перераховано у списку *inparam,* а отримані параметри – у списку *outparam.* Переробимо описаний попередньо сценарій в функцію, вхідні параметри якої *a1* і *b1,* що обчислює величини *e2* й *f 2* .

Гарним тоном програмування вважається декілька рядків функції після початкового опису надати для коментарів. Ці рядки починають з символа '%', вони містять опис вигляду звернення до функції, також інформацію про вхідні та вихідні параметри й коментують методи обчислень. Вказані коментарі будуть видаватися у командне вікно при записі оператору *help «programe».* Коментарі у файлах можуть бути написані українською мовою, але при виведенні їх програмою командою *help* у різних версіях можуть виникати деякі труднощі.

Виконання функції буде припинено у таких випадках:

- усі оператори виконано до кінця файлу;
- виконання оператора *return* призведе до припинення виконання поточної функції та передасть керування функції, яка його викликала;
- завершення тіла функції може також бути у вигляді оголошення наступної функції.

```
function [e2,f2]=my1stprog(a1,b1)
% function [e2,f2]=my1prog(a1,b1)
%Thist is my1st function in MA TLAB
% Inputs: scalar constants a1 and b1
% Outputs: scalar constants e2 and f2
% Calculation formulas is in file.
c2=3.5;
d2=a1/c2*cos(b1);
e1=a1* (c2*sin(b1)+cos(pi-b1));
f2=e2/d2;
```

Так, звернення до функції із командного вікна може виглядати так:

```
[e1,f1] = my1stprog(a,b);
```

В якості формальних параметрів також можна було вказати їх значення 9 і *pi/3*.

Звернення *help mylstprog* в командному вікні викликає напис:

```
function [e2,f2]=my1prog(a1,b1)
Thist is my1st function in MA TLAB
Inputs: scalar constants a1 and b1
Outputs: scalar constants e2 and f2
Calculation formulas is in file.
```

Ряд зауважень, що пояснюють особливості роботи функцій у пакеті MATLAB.

1. При роботі початківцям бажано дотримуватись правила: один файл – одна функція.

2. Основним ім'ям функції є ім'я файлу, у якому записана функція. Якщо ім'я, що записано у описі функції, не співпаде з ім'ям файлу, при зверненні до функції середовище буде шукати файл із вказаним ім'ям. Тому необхідно, щобім'я функції та файлу співпадали.

3. Функція може викликати також іншу функцію, й число вкладень не обмежено, що дає можливості для автовиклика функції та зациклення програм. Основний недолік MATLAB'а у тому, що перервати виконання функції є можливість тільки в результаті аварійного виходу з середовища натисненням клавіш *CTRL — BREAK.*

4. Вхідні параметри передаються функції за їх значенням, довільні їх зміни під час виконання функції не відіб’ються на значеннях в командному вікні.

5. В MATLAB для кожної функції обов'язково існує дві внутрішні змінні – *nargin* і *nargout,* що містять інформацію про кількість параметрів функції.

2.1.7 Найбільш вживані стандартні функції

У MATLAB реалізовано велике різноманіття стандартних функцій, які об'єднано в пакети, так звані TOOLBOX'и. Усі TOOLBOX 'и зазвичай знаходяться у папці TOOLBOX. Основним TOOLBOX'ом є MATLAB, його файли розміщені у папках цієї папки, хочазначна їх частина –це тільки опис вбудованих функцій, виконання яких здійснено ядром програми. Набір таких функцій досить чималий, повний опис їх можна знайти у документації пакету

MATLAB [21] та у літературі [9–16]. Тому обмежимося коротким описом найбільш вживаних функцій середовища.

Першою опишемо функцію *disp(x)*: ця функція не має параметрів та виводить у робоче вікно значення змінної *x*.

2.1.7.1 Елементарні математичні функції

Аргументами всіх вказаних у цьому пункті функцій є матриці, так самом, як і обчислені результати. Функції будуть обчислені для матриць поелементно.

- *y = exp(x)* - поелементна експонента ($y_{ij} = e^{x_{ij}}$)
- *y = sin(x)* - синус
- *y = cos(x)* - косинус
- *y = tan(x)* - тангенс
- *y = asin(x)* - арксинус
- *y = acos(x)* - арккосинус
- *y = atan(x)* - арктангенс
- *y = sqrt(x)* - квадратний корінь
- *y = abs(x)* - модуль
- *y = log(x)* – логарифм натуральний
- *y = log10(x)* - логарифм десятковий
- *y = sign(x)* - сігнум-функція

$$y = \begin{cases} 1, \text{ якщо } x_{ij} > 0 \\ 0, \text{ якщо } x_{ij} == 0 \\ -1, \text{ якщо } x_{ij} < 0 \end{cases}$$

2.1.7.2 Функції округлення і їм супутні.

- *y = round(x)* - округлення елементів *x* до цілого
- *y = ceil(x)* - округлення елементів *x* до цілого убік збільшення
- *y = fix(x)* - округлення елементів *x* до цілого убік нуля (для виділення цілої частини чисел)

- *y = floor(x)* - округлення елементів *x* до цілого убік зменшення
- *y = rem(x, y)* - залишок *rem*(*x, y)= x — y * M,* де *M = fix(x/y).*
- *y = gcd(x,y)* – знаходження найбільшого спільного дільника серед елементів матриць *x* та *y*.

2.1.7.3 Функції над комплексними числами

Комплексні числа задаються операціями типу b = *2*.9 + 5.697i, або c =*b + i*g,* якщо c і g –раніше присвоєні дійсні або цілі змінні. Комплексні числа можуть бути також і елементами матриці, тобто, якщо аргументами нижчевказаних функцій є комплексні матриці, функції будуть обраховані поелементно:

- *y = angle(x)* - аргумент комплексного числа.
- *y = conj(x)* - комплексне сполучення.
- *y = imag(x)* - уявна частина.
- *y = real(x)* - дійсна частина.
- *y = sign(x)* - сигнум-функція комплексного *x, sign(x)= x./abs(x).*

Перетворення типів змінних:

- *num2str* - числова змінна у рядок;

-*str2num* - рядок у числову змінну. Рядок буде містити цифри, десяткові крапки та знаки '+', '-', позначення уявної одиниці *i* або *j,* літеру 'e' при уявленні числа із плаваючою крапкою.

2.1.7.4 Формування векторів і матриць

- *y = linspace*(*xmin, xmax)* - формує вектор *y* зі ста елементів, які рівномірно розташовані між *xmin* та *xmax.* Якщо вектор містить інше число компонентів, синтаксис цієї функції повинний мати вигляд *y = linspace*(*xmin, xmax, N).*

- *y = logspace*(*xmin, xmax)* - формує вектор *y* зі ста елементів, що логарифмічно розташовані між *xmin* і *xmax.* Якщо вектор буде містити інше число

компонентів, то звернення до функції має вигляд *y = linspace*(*xmin, xmax, N).*

- *y = zeros(N, M)* - генерує нульову матрицю розмірністю *NXM.* За наявності одного єдиного операнда *N* буде сформовано квадратну матрицю розмірністю *N* x *N.*
- *y = ones*(*N, M)* - генерує матрицю розмірності *N* x *M,* усі елементи якої дорівнюють одиниці. За наявності одного операнда *N* буде згенерговано квадратну матрицю розмірністю *N* x *N.*
- *y = eye(N)* - генерує одиничну матрицю *I* розмірністю *N* x *N.*
- *y = rand*(*N, M)* - генерує матрицю розмірністю *N* x *M,* елементи якої – псевдовипадкові числа, які рівномірно розподілені у інтервалі (0..1).
- *y = diag*(*v,n)* - генгерує матрицю, на *n-й* наддіагоналі якої розташовано вектор *v.* Якщо *n* = 0, виконання цієї функції – діагональна матриця, на діагоналі якої лежать елементи вектора *v.* Якщо значення *n* < 0, елементи вектора *v* буде розташовано на піддіагоналі із номером |*n*|. Наприклад, *diag([2,3,6], 2)*сформує матрицю:

$$\begin{bmatrix} 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

а *diag([3, –2],–1*) – матрицю:

$$\begin{bmatrix} 0 & 0 & 0 \\ 3 & 0 & 0 \\ 0 & -2 & 0 \end{bmatrix}$$

- *y = diag(B, n)* - видаляє *n*-ту наддіагональ з матриці *B.* Матриця може бути не квадратною. При *n* < 0 виводиться піддіагональ з відповідним номером.

Для повернення векторів і матриць застосовують такі функції:

- *fliplr(B)* - переставляє стовпці матриці *B* симетрично відносно вертикальної осі

- *flipud(B)* - переставляє рядки матриці *B* симетрично відносно горизонтальної осі*;*

- *rot90(B, n)* - *n* разів обертає матрицю *B* на 90° проти годинникової стрілки. При відсутності параметра *n* поворот матриці відбувається на 90°. Якщо значеннч *n* – нецiле, функція не робить жодних дій, не видаючи ані попереджень, ані повідомлень про помилки.

Також існують функції, які генерують "іменні" матриці: Теплиця, Ганкеля, , Адамара, Уїлкинсона, Гильберта тощо.

Функції носять відповідні матрицям імена, про їх використання найпростішим чином можна дізнатися використанням оператора *help.* Слід взяти до уваги, що імена функцій пишуться у загальноприйнятійтлітерації: *toeplitz, hilb, hadamard, wilkinson, hankel* тощо.

2.1.7.5 Операції над векторами і матрицями

•*y = cross(u,v)* - векторний добуток векторів *u* і *v* довжиною 3;

Якщо для наступних функцій аргументи – вектора, результатом їх виконання буде скаляр. У випадку, коли аргументи – матриці, результатом обчислень буде вектор, j-й елемент якого - результат виконання функції до j-го стовчика матриці.

- *y = sum(x)* - сума елементів вектора ;
- *y = prod(x)* - добуток елементів вектора ;
- *y = min(x)* - мінімальний елемент вектора;
- *y = max* (*x)* - максимальний елемент вектора;
- *y = mean* (*x)* - середнє арифметичне елементів вектора;

- *y = std(x)* - середнє квадратичне відхилення елементів вектора;
- *y = median(x)* - медіанна елементу вектора. Під медіаною в даному випадку розуміють величину, отриману в результаті таких операцій: все *n* елементів вектора розташовуються за зростанням, в новому векторі обирається елемент з індексом (*n* + 1)*/2* для непарних *n* та середнє арифметичне елементів з індексами *n/2* і *n/2* + 1 – для парних *n;*
- *y = finite(x)* - повертає 1, якщо усі елементи вектора *x* кінцеві та 0 у зворотньому випадку;
- *y = isNaN(x)* - повертає 1, коли хоч би один з елементів вектора *x – NAN* та 0 у зворотньому випадку;
- *y = all(x)* - повертає 1, якщо всі елементи вектора *x* не дорівнюють один одному та 0 інакше;
- *y = any* (*x)* - повертає 1, коли хоч б один елемент вектора *x* не дорівнює 0 та 0 у зворотньому випадку;
- *y = find(x)* - повертає номери ненульових елементів вектора. У випадку, якщо *x* – матриця, застосовується наскрізна нумерація по рядках;

Якщо для нижчезазначених функцій аргументами є вектори, результатом виконання буде вектор. Якщо аргументами є матриці, результатом обчислення буде матриця, j-й стовпчик якої є результатом застосування відповідної функції до j-ома стовпчика початкової матриці.

• *y = cumsum(x)* - кумулятивна сума елементів (*i-й* елемент вектора *y* дорівнює сумі *i* перших елементів вектора *x*);

• *y = cumprod(x)* - кумулятивний добуток елементів вектора (*i-й* елемент вектора *y* дорівнює добутку *i* перших елементів вектора *x*);

• *y = sort(x)* - сортує елементи вектора за зростанням (стовпчики в матричному *x* сортуються незалежно один від одного);

• *y = diff(x)* – кінцево-різницеве диференціювання вектора. Для матриці *x* проводиться диференціювання стовпцчиків *y = x*(*2:N*,:) - *x*(1:*N*–1, :). При записі *y = diff(x, n)* – аналогічно обчислюється кінцево-різницева похідна;

• *[Px, Py] = gradient(x)* - обчислення кінцево-різницевих матричних градієнтів. При записі *[Px, Py] = gradient(x, Dx, Dy)* аргументи *Dx* та *Dy* задають кроки по вісям *x* і *y*.

• *z = trapz*(*x,y)* - обчислює методом трапецій інтеграл за значеннями ординат *y,* що інтерпретовані, як значення заданої функції у точках із абсцисами, що записані у *x*.

2.1.7.6 Норми векторів і матриць

y = norm(*v, N)* – обчислення норми вектора або матриці. Функція обчислює різні норми у залежності від того *v* – це вектор або матриця, та від значення, що приймає аргумент *N: v* - вектор (матриця один з розмірів якої рівний 1)

•якщо *N* = 2 або цей аргумент відсутній – евклидова норма:

$$\|v\|_2 = \left(\sum_{i=1}^{n} v_i^2 \right)^{1/2}$$

•якщо *N* = 1 - перша норма:

$$\|v\|_1 = \sum_{i=1}^{n} |v_i|$$

•якщо *N = inf* - нескінченна (чебишевська) норма:

$$\|v\|_\infty = \max_i |v_i|$$

•якщо *N = -inf* - норма вектора вигляду:

$$\|v\|_{-\infty} = \min_i |v_i|$$

•якщо •*N = p* - для будь-якого *p* це буде величина вигляду:

$$\|v\|_p = \left(\sum_{i=1}^{n} v_i^p \right)^{1/p}$$

y = norm(A, N), де *A* є матрицею

•якщо •*N* = 2 або цей аргумент відсутній – це спектральна норма матриці, дорівнює найбільшому сингулярному числу заданої матриці:

$$\|A\|_2 = \max_i \lambda_i^{1/2}(A^T A) = \max_i \sigma_i(A)$$

•якщо •*N* = 1 - перша (стовпчикова) норма матриці:

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |A_{ij}|$$

•якщо •*N = inf* - нескінченна (рядкова) норма матриці:

$$\|A\|_\infty = \max_i \sum_{j=1}^{m} |A_{ij}|$$

•якщо •*N =' fro'* - фробеніусова норма матриці:

$$\|A\|_F = trace^{1/2}(A^T A) = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} A_{ij}^2 \right)^{1/2}$$

2.1.7.7 Елементарні операції над матрицями

Транспонування матриці виконується оператором *' (*A^T *=A')***.**

- *[N, M] = size(A)* –повертає розміри матриці *(A* - матриця розміру *N* x *M;*
- *L = length(A)* – повертає максимальний розмір матриці *(L = max(size(A)));*
- *y = det(A)* – обраховує визначник матриці;
- *y = trace(A)* – обраховує слід матриці;
- *y = inv(A)* – проводить обернення матриці;
- *y = pinv(A)* – проводить псевдообернення матриці *(y =* $(A^T A)^{-1} A^T$*);*
- *y = sqrtm(A)* – повертає корінь квадратний з матриці;

- *y = expm (A)* – повертає експоненту від матриці;
- *y = logm (A)* – повертає логарифм матриці.
- *v = poly (A)* – повертає вектор коефіцієнтів характеристичного полінома (у порядку спадання степенів).
- *[V, D] = eig(A)* - обчислення власних чисел та власних векторів матриці Діагональні елементи матриці *D* (жорданової канонічної форми) – власні числа матриці *A,* стовпчики матриці *V* – власні вектори. За наявності одного єдиного вихідного параметра *L = eig(A)*–- вихід *L* - вектор-стовпчик власних чисел матриці;
- *[U, S, V] = svd(A)* – розрахунок сингулярного розкладання матриці. Матриця *A* представляється у вигляді добутку матриць $A = USV^T$, де *U* та *V* – ортогональні матриці, *S* – діагональна матриця сингулярних чисел:

$$\sigma_i(A) = \lambda_i^{1/2}(A^T A)$$

•*y = rank (A)* – повертає ранг матриці;

Слід пояснити деякі подробиці про те, що у MATLAB'і розуміється під рангом. MATLAB оперує з числовими оцінками та обчислювальними операціями. Відповідно, при достатньо малих ненульових значеннях детермінанта числове обернення матриці є неможливим. Таким чином, при обчисленнях вводиться узагальнене визначення рангу так: рангом вважається кількість сингулярних чисел матриці, які перевищують поріг *tol = max(size(A))* ||A||$_2$ * eps.*

З особливостями обчислювальних процедур при оберненні матриць, обчисленні власних чисел та інших операцій пов'язаний також ряд важливих функцій MATLAB'у. З теоретичними основами обчислювальних алгоритмів можна ознайомитися, наприклад, в [21] і [22]. Тут же ми наведемо тільки функції пакету, які виконують відповідні операції.

- *c = cond(A)* - повертає число обумовленості матриці $\mu = \left\|A^{-1}\right\|_2 * \left\|A\right\|_2 = \sigma_{\max}(A) / \sigma_{\min}(A)$

- *[T, B] = balance(A)* – операція балансування матриці, що використовується для покращення обумовленості. Процедура балансування полягає у тому, що обчислюється така діагональна матриця *T*, що матриця $B = T^{-1}*A*T$ має наближено рівні норми рядка й стовпчика $||B||_1 \approx ||B||_\infty$. Така процедура еквівалентна масштабуванню змінних.

Стандартна функція *eig* обрахування власних чисел автоматично робить балансування матриць. Для відмови від функції потрібно додати у звернення другий аргумент – текстову константу 'nobalance' ([*V,D*] = *eig(A,' nobalance').*

- *Z = null (A)* – розрахунок ортонормального базису нульового підпростору матриці. Базис нульового підпростору утворюють стовпчики матриці *Z.* Звернення *Z = null(A', r')* призводить до обчислення раціонального базису, тобто коли компонентами векторів будуть раціональні числа.

- *Q = orth(A)* – розрахунок ортонормального базису матриці. Стовпчики матриці утворюють той самий підпростір, що й стовпчики матриці *A*, $Q^TQ = I$.

Для ознайомлення наведемо деякі приклади обчислення матриць, що погано обумовлених. Погано обумовлена, з погляду обернення, є, наприклад, квадратна матриця Гільберта високого порядку, елементи якої обчислюються за формулою R = *1/(i+j - 1)*, *i*,*j* = 1... *n* - номери рядків й стовпчиків, а a_n- розмірність матриці. У середовищі MATLAB для формування такої матриці можна використати функцію *a = hilb(n),* де *n* - розмірність квадратної матриці. Розглянемо приклад матриці *a* Гільберта,

якщо $n = 6$ й матриці b, усі елементи якої дорівнюють елементам матриці a, окрім одного. Цей елемент – b_{16} буде відрізняється від елемент a_{16} на 1%. Розглянемо таку послідовність обчислень:

```
a=hilb(6)
a =
   1.0000   0.5000   0.3333   0.2500   0.2000   0.1667
   0.5000   0.3333   0.2500   0.2000   0.1667   0.1429
   0.3333   0.2500   0.2000   0.1667   0.1429   0.1250
   0.2500   0.2000   0.1667   0.1429   0.1250   0.1111
   0.2000   0.1667   0.1429   0.1250   0.1111   0.1000
   0.1667   0.1429   0.1250   0.1111   0.1000   0.0909
b=a
b(1,6)=b(1,6)*1.01
(b^(-1)–a^(-1))./a^(-1)

ans =

  -1.2762  -2.1878  -2.8715  -3.4033  -3.8287  -4.1768
  -1.2762  -1.6409  -1.9144  -2.1271  -2.2972  -2.4365
  -1.2762  -1.4586  -1.5953  -1.7017  -1.7867  -1.8564
  -1.2762  -1.3674  -1.4358  -1.4890  -1.5315  -1.5663
  -1.2762  -1.3127  -1.3401  -1.3613  -1.3783  -1.3923
  -1.2762  -1.2762  -1.2762  -1.2762  -1.2762  -1.2762

eig(a)

ans =

   0.0000
   0.0000
```

0.0006

0.0163

0.2424

1.6189

eig(b)

ans =

1.6191

0.2420

0.0165

0.0006

0.0000

-0.0000

d=cond(a)

d =

1.4951e+007

det(a)

ans =

5.3673e-018

У вищерозглянутому прикладі відносна помилка обчислення елементів зворотної матриці є більшою ніж 100%, при помилці завдання одного з елементів матриці в 1%. При цьому власні числа обох матриць обчислено цілком прийнятно. Основною ознакою можливої похибки у даному прикладі

можна вважати велике значення числа обумовленості й мале – визначника матриці.

Приклад поганої обумовленості при розрахунку власних чисел також дає функція *gallery*. Ця функція дає змогу обчислити значення різних стандартних матриць. Список таких матриць й параметри можна подивитися командою *help gallery*.

Розглянемо приклад:

```
>       a=gallery(5)
a =

        -9         11        -21         63       -252
        70        -69        141       -421       1684
      -575        575      -1149       3451     -13801
      3891      -3891       7782     -23345      93365
      1024      -1024       2048      -6144      24572

        > b=a;
        > b(5,1)=b(5,1)*1.01;
        > [eig(a),eig(b)]
        > ans =
ans =

  -0.0408            -0.9201 +50.8135i
  -0.0119 + 0.0386i  -0.9201 -50.8135i
  -0.0119 - 0.0386i  -0.0000
   0.0323 + 0.0230i   0.9201 + 0.4053i
   0.0323 - 0.0230i   0.9201 - 0.4053i

> cond(a)
   ans =
   2.0253e+018
```

Власні числа матриць *a* і *b* в розглянутому прикладі відрізняються на два порядки. Ознакою такої поведінки власних чисел може служитм число обумовленості – 10^{18}.

В пакеті MATLAB реалізовано різноманіття функцій матриць, які здійснюють приведення до різних канонічних форм, таких як форми Фробеніуса, Шура, Жордана, Хессенберга, також часто використовувані функції розкладань *lu* і *qr*.

2.2 Програмування в пакеті MATLAB. Умовні переходи, цикли, перемикачі

Розділ присвячено організації основних елементів програмування у пакеті MATLAB: умовним переходам й організацію циклів.

2.2.1 Логічні змінні

В середовищі MATLAB існує 6 операцій порівняння: <, >, <=, >=, ==, ~=, які виконують поелементне порівняння двох матриць. Результатом операції буде матриця такого ж розміру, елементи якого дорівнюють 1, якщо відповідна умова виконується, та нулю інакше.

Аналогічно обумовлені й логічні операції:

– & - логічне І (AND). Результатом операції *A&B* буде матриця також ж розміру, кожний елемент якої дорівнює 1, якщо два відповідні елементм матриць *A* і *B* не дорівнюють 0, та нульовим інакше (тобто, коли хоч один з елементів дорівнює 0).

– | - логічне АБО (OR). Результатом операції *A*|*B* буде матриця тієж ж розмірності, кожний елемент якої дорівнює 1, коли хоча б один з відповідних елементів матриці *A* і *B* не дорівнюють 0, та дорівнює 0 інакше (тобто, коли обидва елементи дорівнюють 0).

– ~ - логічне НІ (NOT). Результатом операції ~ *A* буде матриця такого ж розміру, кожний елемент якої дорівнює 1, якщо відповідний елемент матриці *A* – 0, та дорівнює 0 інакше (тобто якщо елемент не дорівнює 0).

Наприклад

```
>> a=[1 2; 3 4]

a =

    1     2

    3     4

>> b=[4 2; 3 1]

b =

    4     2

    3     1

>> c= a>=b

c =

    0     1

    1     1


>> d = ~ (b <= c)&a

d =


    1     1

    1     0
```

2.2.2 Умовний оператор

В середовищі MATLAB для програмування розгалужень програми використовується умовний оператор, який синтаксично записується так:

if <Логічна змінна 1>,

<Інструкції 1>

elseif <Логічна змінна 2>,

<Інструкції 2>

else

<Інструкції 3>

end

Як логічна змінна може використовуватись й логічний вираз, тобто права частина присвоювання попереднього підрозділа.

Під інструкціями розуміють команди, що будуть виконнані, якщо всі елементи матриці – логічної змінної є ненульовими.

У наведеному умовному операторі інструкції, які починаються з *elseif* та *else*, можуть бути відсутні (коротка форма запису).

Приклади:

```
>> a=[1 0; 0 1]
a =

    1     0
    0     1
>> b=[1 1; 1 1]

b =

    1     1
    1     1

>> d=[1 1; 1 0]

d =
```

1 1

1 0

if d, c = a; else c = b; end; c

c =

1 1

1 1

>> if a <= b , c = a; else c = b; end; c

c =

1 0

0 1

Також умовні оператори використовуються й традиційним образом для скалярних величин, саме таке звернення до умовного оператора можна зустріти у програмах

```
if    a <b
      c=a;
      elseif
   a==b,
      c=1;
   else
```

```
    c=b;
end
```

2.2.3 Організація циклів

При програмуванні у пакеті MATLAB використовують два види циклів:

- цикл *for-end*
- цикл *while-end*

Цикл типу *for-end* використовується для обчислень із відомим числом повторень. Синтаксис циклу має такий вигляд:

```
for var = <Вираз>,
    <Інструкції>
end
```

В якості виразу у запропонованій конструкції можна використати матрицю розмірності $m \times n$. У цьому випадку <Інструкції> тіла цикл будуть повторені n разів, при цьому параметр *var* буде послідовно приймати векторні значення, що співпадають зі стовпчиками матриці. Це добре видно з прикладу:

```
>> a = [1 2 3; 4 5 6]
a =

    1    2    3
    4    5    6
>> for i = a; i, end

i =

    1
    4
```

i =

2

5

i =

3

6

Проте зазвичай <вираз> записують у традиційному вигляді:

for i = 1 : 3, for j = 1 : 3 ,
a(i,j)=i+j;
end
end

В результаті виконання програми буде отримано матрицю:

a =

2 3 4

3 4 5

(2.3)

4 5 6

Другий тип циклів призначений для виконання інструкцій у тілі цикла до тих пір, доки виконується задана умова. Синтаскис має такий вигляд:

while

<Логічна змінна>,

<Інструкції>

end

<Логічна змінна> має той самий сенс, що й для оператора *if* у підрозділі 2.2 та обчислення будуть продовжені до тих пір, доки у матриці, що використовується в якості логічної змінної немає нульових елементів. Так саме, як і в 2.2 у конструкції *while* прийнято використання умови з підпункту 2.1. Приведемо традиційне й нетрадиційне використання циклу *while.*

Приклад 1:

Традиційний приклад:

```
p = 1; k= 0;
 while p < 10,
 p = p * 2;
k =k + 1;
 end;
 k
```

Обчислюється мінімальний показник степеня *k* такий, що $2^k >= 10$. Після закінчення роботи циклу $k = 4$ і $p = 16$.

Прикдад 2:

```
a=[2.5,3;4,5];
 while    a,
 a=a-1;
 end
```

В даному випадку задана матриця перетвориться таким чином: усі елементи її зменшаться з кожним кроком на 1, доти, доки одне з них не буде дорівнювати нулю. Таким чином, процедуру буде завершено, коли хоч один з елементів матриці *a* буде дорівнювати 0. Якщо це не так, відбудеться зациклювання програми.

Для захисту від описаної вище ситуації використовується оператор *break,* що припиняє дію останнього вкладеного циклу, всередині якого єоператор *break.* Попередній приклад, перерваний таким чином, записаний у файл буде мати вигляд:

```
a =[2.5,3;4,5]
while a,
a = a - 1;
if a < 0, break
end
   end
```

2.2.4 Перемикач *switch-case-otherwise-end*

Конструкція *switch-case-otherwise-end* використовується для множинного вибору (або розгалуження).

В загальному випадку має синтаксис:

switch <switch-вираз>,

case <Case-вираз>

<Інструкції 1>

case { <case-вираз 1>, <Case-вираз 2>, <Case-вираз 3> ... }

<Інструкції 2 >

otherwise

<Інструкції 3 >

end

В якості switch-виразу використовують будь-який з названих раніше об'єктів пакету. Якщо поточне значення виразу співпаде з одним з <Case-виразів>, тоді будуть виконані оператори з відповідних <Інструкцій>, якщо ж жодний з Case-виразів не підходить, тоді виконуються <Інструкції>, записані за оператором *otherwise.*

Приклад – сценарій asw.m:

```
switch var
case {1,2,12,'January','February','December'}
disp('Winter')
```

```
case {3,4,5,'March','April','May'}
disp('Spring')
case {6,7,8,'June','July','August'}
disp('Summer')
case {9,10,11,'September','October','November'}
disp('Autumn')
otherwise
disp('It is not a month.')
   end
```

Звернення до сценарію прзиводить до таких результатів:

```
> var = 3;
> asw
Spring
> var = 'May';
> asw
Spring
> var = 10;
> asw
Autumn
> var = 'Autumn';
> asw
It is not a month.
```

2.3 Спеціальні можливості при зверненні до функцій

Докладний розгляд стандартних функцій MATLAB дає змогу зробити висновок щодо реалізації додаткових можливостей функцій пакету, наприклад, виконання різних обчислень в залежності від кількості й типу

аргументів та вихідних параметрів функцій, передати функції ім’я функції, що викликана, передати довільне число аргументів допоміжної функції тощо.

Виконання різних обчислень в залежності від кількості аргументів та вихідних параметрів нескладно організувати, використавши внутрішні змінні *nargin* та *nargout,* згадувані раніше, та оператори *if* або *switch,*. можна також перевірити й розмірність та тип аргументів.

Деякі особливості середовища MATLAB, що будуть використані для роботи з функціями, будуть описані нижче. Для того, щоб користуватися функціями, описаними в подальших підрозділах, читати цей розділ необов'язково, але його зміст допоможе зрозуміти, як реалізовано ті або інші функції й дасть читачеві змогу писати власні функції із аналогічними властивостями.

2.3.1 Звернення до функції за ім’ям

У ряді випадків, наприклад при розв’язку нелінійних рівнянь або числовому інтегруванні диференціальних рівнянь, є потреба передати функції, як аргумент, ім'я іншої функції, наприклад, ім'я функції, що обчислює праві або ліві частини рівняння, що буде викликана за цим ім’ям. Ім'я функції або змінної в такому випадку збережено у вигляді символьних одновимірних масивів (векторів), й може бути передано функції в якості формального параметра.

Виконання відповідної команди та/або виклик функції відтворено у функціях *eval* і *feval.*

Функція *eval* дає змогу виконати команду системі MATLAB. Синтаксис функції має вигляд:

eval(command)

Текстова змінна *command* містить текст команди. У прикладі наведено уривок програми, у якому створено 5 матриць Уілкінсона розмірністю від 5 до 10, імена яких встановлюються *M5, M6, M7, M8, M9, M10* відповідно.

for k=5:10

eval(['M',num2str(k), '=wilkinson(',num2str'(k));'])

end

Функція *feval* дозволяє за ім’ям викликати функцію MATLAB'у.Синтаксис має вигляд:

[outpars]=feval(FunName,inpar1,inpar2...)

Текстова змінна *FunName* містить ім'я функції, яка викликається, аргументи *inpar1,inpar2...* - аргументи функції, яка викликається. Через *[outpars]* позначено список результатів виконання викликаної функції.

Наприклад, при записі:

v=feval(fname, 0,10,50);

якщо *fname = 'linespace'* буде сформовано вектор, 50 елементів якого рівномірно розподілені в інтервалі [0,10], при *fname = 'logspace''* ці елементи будуть мати логарифмічний розподіл.

2.3.2 Багатовимірні масиви

Разом із традиційними двовимірними масивами – матрицями, у пакеті MATLAB є також і багатовимірні масиви, за допомогою яких у MATLAB'і можна відображати тензорні величини. Приділимо увагу таким масивам в зв'язку з тим, що вони застосовуються при записі результатів деякими функціями моделювання й дослідження систем керування.

Звернення до елементів багатовимірного масиву має синтаксис: *a(i1, i2, i3..., in).* Задати багатовимірний масив можливо, якщо збільшувати кількість розмірностей матриці. Наприклад:

>> a=[1 2; 3 4]

a =

1 2

3 4

>> a(:,:,2)=3

a(:,:,1) =

1 2

3 4

a(:,:,2) =

3 3

3 3

Також створюють багатовимірні масиви функції *ones, zeros, rand, randn.* При записі:

a = ones(n1, n2 .. ., nn)

ці функції повертають масив, розмірність якого буде дорівнювати числу аргументів функції, при цьому *n1, n2..., nn* – довжини відповідних розмірностей. Вищеопісані функції створюють багатовимірні масиви, всі елементи яких приймають значення, відповідно: для *ones* - одиниці; для *zeros* - нуля; для *rand* - випадкового числа, рівномірно розподіленого в інтервалі (0,1); для *randn* - випадкове число, нормально розподілене із нульовим середнім й одиничною дисперсією. Основні функції, які дають змогу працювати із багатовимірними масивами:

•функція

n = ndims(a)

визначає розмірність багатовимірного масиву.

•функція

n = size(a)

визначає розмір багатовимірного масиву за кожною розмірністю.

•Функція *B = cat(dim,A1,A2,A3,A4...)* дозволяє об'єднувати масиви *A1,A2,A3,A4...* в один вздовж розмірності *dim*.

Приклад:

>> A1=ones(2);

>> A2=eye(2);

>> B1=cat(1,A1,A2)

B1 =

1 1

1 1

1 0

0 1

>> B2=cat(2,A1,A2)

B2 =

1 1 1 0

1 1 0 1

>> B3=cat(3,A1,A2)

B3(:,:,1) =

1 1

1 1

B3(:,:,2) =

1 0

0 1

А першому випадку дія еквівалентна команді *B1 = [A1;A2],* у другому ж випадку – B2 = *[A1, A2], у* третьому функцією створено тривимірний масив *B3*.

- Функція *B = permute(A, order)* переставляє місцями розмірності масиву у порядку, який встановлено цілочисельним вектором *order*.

Наприклад, запис *C3 = permute*(*B3* [213]) для масиву *B3* з попереднього прикладу призведе до транспонування матриць *B3(*:,:, 1) і *B3(*:,:, 2).

- Функція *C = ipermute(B, order)* зворотня у відношенні до функції *permute*.

- Функція *B = shiftdim*(*B,n)* проводить зрушення розмірностей наліво на величину *n*. При *n* < 1 така дія еквівалентна циклічній перестановці розмірностей й збільшення їх числа не відбувається. При *n* < 0

відбувається зрушення розмірностей направо. При цьому число розмірностей збільшується на |*n*| й меншірозмірності будуть заповнюватися одиницями.

- Функція *C = squeese(B)* дозволяє видалити розмірності довжиною 1. Для MATLAB'у запис *B3(:,:,* 1) у використаному вище прикладі описує тривимірний масив, у той час, як це є матриця й розмірність її на 1 менша. Такий об'єкт не має мзоги використати при зверненні до стандартних операторів і функцій. Перетворення цього об'єкту, яке знижує розмірність проводить функція *squeese.*

2.3.3 Комірки і операції над ними

Масиви комірок використано у MATLAB'і для зберігання й перенесення даних різних типів. Створити масив комірок можливо двома методами: із застосуванням функції *cell* або введенням об'єктів у фігурні дужки. Наприклад, якщо у робочу область введено матрицю *a* розмірністю 2x2, така команда вводить масив комірок розмірністю 2 x 3 :

```
>>b={min(a) max(a) 3; 5 1 'text' }
b =
[1 . 2double] [1 . 2double] [ 3]

 [ 5] [ 1] 'text'
```

Запис *b = cell(N, M)* генерує порожній масив комірок розміру *N* x *M,* за відсутності аргументу *M* масив буде розміру *N* x *N.* Для звернення до елементів масиву застосовують фігурні дужки, й на такі звернення розповсюджено усі основні синтаксичні правила для роботи з традиційними матрицями:

```
>>b{2,3}
ans =
text
>>b{:,1}
```

ans =

1 2

ans =

5

Багатовимірний масив комірок зазвичай заповнюють згідно правилам, аналогічним до заповнення багатовимірних масивів, при цьому для масивів комірок існують функції *squeeze* і *shiftdim.*

В масиві комірок зберігаються копії об'єктів, й зміна самих об'єктів не змінює вмісту масиву комірок та навпаки, іншими словами масив комірок не треба плутати з масивом адрес.

Декілька функцій і команд для роботи з масивом комірок:

- *celldisp(C)* - виводить вміст масиву комірок *C.*
- *cellplot(C)* - виводить вміст масиву комірок *C* у вигляді графічного вікна.
- *C = cellstr(s)* - переробляє масив рядків *s* у масив символьних комірок *C.*
- *C = iscell(C)* - повертає *TRUE(1),* якщо *C* - масив комірок, і *FALSE(0)*– інакше.
- *C = num2cell(A)* - перетворює масив *A* в масив комірок, розміщуючи кожний елемент масиву в окремій комірці.

З поняттям масивів комірок зв'язана можливість опису списків вхідних і вихідних аргументів змінної довжини, для цього використовують змінні *varargin* і *varargout.*

Змінна *varargin* дає змогу об'єднати довільну кількість вхідних змінних, вона – це одновимірний масив комірок, що містить аргументи викликаної функції. Ця змінна повинна завершувати список вхідних аргументів функції.

Аналогічно, змінна *varargout* дає змогу об'єднати довільну кількість вихідних змінних. Це – одновимірний масив комірок, який містить аргументи

виходу викликаної функції. Так само, змінна повинна завершувати список вихідних аргументів функції.

Приклад опису такої функції:

function [z,varargout]=my1fun(a,b,varargin)

...

varargout=my2fun(b, varargin);

...

Прикладом використання цих змінних є звернення до функцій типу *ode*, що будуть розглянуті у підрозділі 2.5.

2.3.4 Глобальні змінні

У попередньому підрозділи був описаний спосіб передачі довільного числа змінних допоміжної функції, який у багатьох випадках не є досить ефективним [12]. Більш ефектим та швидшим є спосіб передачі змінних із використанням інструкції global.

Команда:

global < ім'я1>,<ім'я2> ...

визначає змінні із іменами *<ім'я1>, <ім'я2>* ... як глобальні. Якщо декілька функцій й сценарій оголошують змінну глобальною, всі вони використовують одну й ту саму копію змінної. Зміна її значення у одній з функцій призведе до того, що при подальшому зверненні до інших функцій змінна приймає вже змінене значення. Це ж саме значення приймає змінна із цим ім'ям у робочій області (яку можна подивитися у командному вікні).

Приклад використання глобальних змінних:

Запис цієї функції із командного вікна приводить до результату *» mylfun(3,4)*

5 7

При цьому у командному вікні і для функції *my2fun* змінна *c* не визначена або не змінює свого значення, якщо то було задано.

```
function z=my1fun(a,b)
global c
c=b;
d=my2fun(a);
disp ([c,d])
. . .
function d=my2fun(b);
d=my3fun(b);
. . .
function d=my3fun(b);
global c
d=b+c;
c=5;
```

Звернення до цієї функції із командного вікна призводить до результату:

» mylfun(3,4)

5 7

При цьому у командному вікні й у функції *my2fun* змінна *c* не визначена, або не змінює свого значення, якщо те було задане.

2.4 Графічні функції пакету MATLAB

2.4.1 Двовимірна графіка

2.4.1.1 Креслення графіків в декартових координатах

Основною функцією для креслення графіків у пакеті MATLAB є функція *plot.* Синтаксис цієї функції має вигляд

• *plot (x, y)* будує графік, при цьому відкладаючи по вісі абсцис значення

елементів вектора *x* і по вісі ординат значення елементів вектора *y,* яке має таку ж розмірність, що й вектор *x*. Надалі, у всіх подібних випадках будемо говорити про креслення графіка *y від x*.

Побудова графіка – це нанесення на координатну площину точок, з'єднаних прямими. Такий спосіб побудови не пов'язаний з обмеженнями при накресленні неоднозначних функцій.

- *plot (y)* креслить аналогічний графік з відкладанням значень елементів вектора *y* по вісі ординат, а по вісі абсцис відкладається номер відповідного елемента у векторі.

Для двох вищевказаних варіантів функції *plot* для заданої матриці *y* розмірністю *n* × *m* у одній координатній площині буде накреслено *m* графіків (по одному для кожного стовпчика матриці *y).* З перших семи графіків кожний наступний матиме свій колір, надалі кольори будуть циклічно повторюватися.

Наприклад:

```
t=0:0.05:3*pi;
plot(t,[sin(t);cos(t)]);
```

Викличе відкриття графічного вікна MATLAB'у з зображенням, наведеним на рисунку 2.1

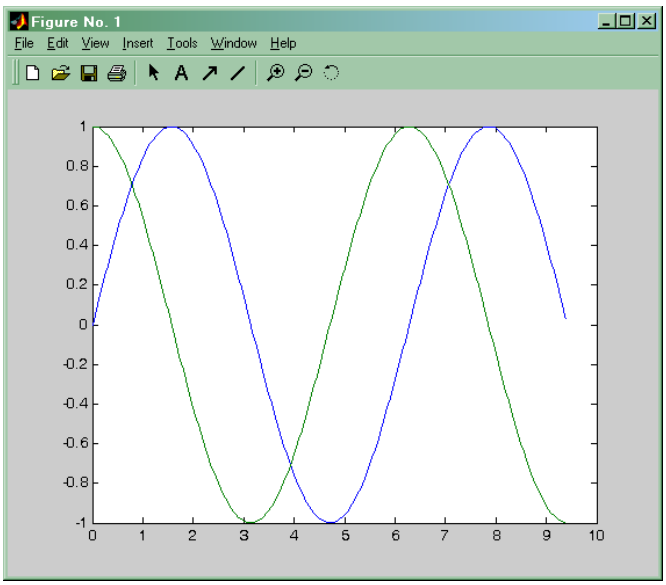


Рисунок 2.1 – Результат дії функції *plot*

Двовимірним може також бути й масив *x*. Якщо масив *y* є вектором, тоді буде побудовано графіки, на яких по вісі абсцис відкладено значення елементів векторів-стовпчиків матриці *x*, а по вісі ординат – значення елементів вектора *y*. Якщо ж *x* та *y* – це двовимірні масиви однакової розмірності, буде побудовано залежності стовпчиків матриці *y* від стовпчиків матриці *x*.

• *plot(x, y, s*).

Таке звернення дозволяє будувати графік функції, вказавши у текстовій константі *s* колір та спосіб відображення лінії, вигляд вузлових точок. Як і раніше вектора (або матриці) *x* та *y* задають значення абсцис і ординат точок на графіці, а текстова константа *s* може містити по одному символу з трьох наборів, які наведені у таблиці 2.1:

Таблиця 2.1 – значення змінної **s**

	Колір		**Вузлова точка**		**Вигляд лінії**
y	жовтий (yellow)	.	точка (●)	-	суцільна
m	фіолетовий (magenta)	o	коло (O)	:	пунктир
c	голубий (cyan)	x	хрест (x)	-.	штрих-пунктир
r	червоний (red)	+	плюс (+)	–	штрихова
g	зелений (green)	*	зірочка (*)		
b	синій (blue)	s	квадрат		
w	білий (white)	d	ромб		
k	чорний (black)	v	набла (∇)		
		^	трикутник (Δ)		
		<	трикутник (лівий)		
		>	трикутник (правий)		
		p	пятикінцева зірка		
		h	шестикінцева зірка		

Вищенаведені методи звернення функції *plot* мають недолік – вони не дозволяють побудувати у одних координатних вісях декілька графіків, якщо вектора абсцис та ординат мають різну розмірність. Можна подолати це такими способами.

Перший:

plot(x1, y1, s1, x2, y2, s2, x3, y3, s3. ..).

В цьому випадку на одних координатних вісях побудовані графіки *y1* від *x1* відповідно до текстової константи *s1*, та графіки *y2* від *x2* відповідно до текстової константи s*2* і т.д. При цьому повинна бути витримана відповідність розмірів масивів *x1* і *y1*, *x2 і y2* тощо, але число точок у різних парах масивів може не співпадати.

Приклад:

```
t=0:0.05:2*pi;
t1=0:0.05:3*pi;
plot(t,sin(t), 'b-',t1,cos(t1), 'r:');
```

В результаті у відкритому графічному вікні MATLAB'a буде побудовано зображення, приведене на рисунку 2.2.

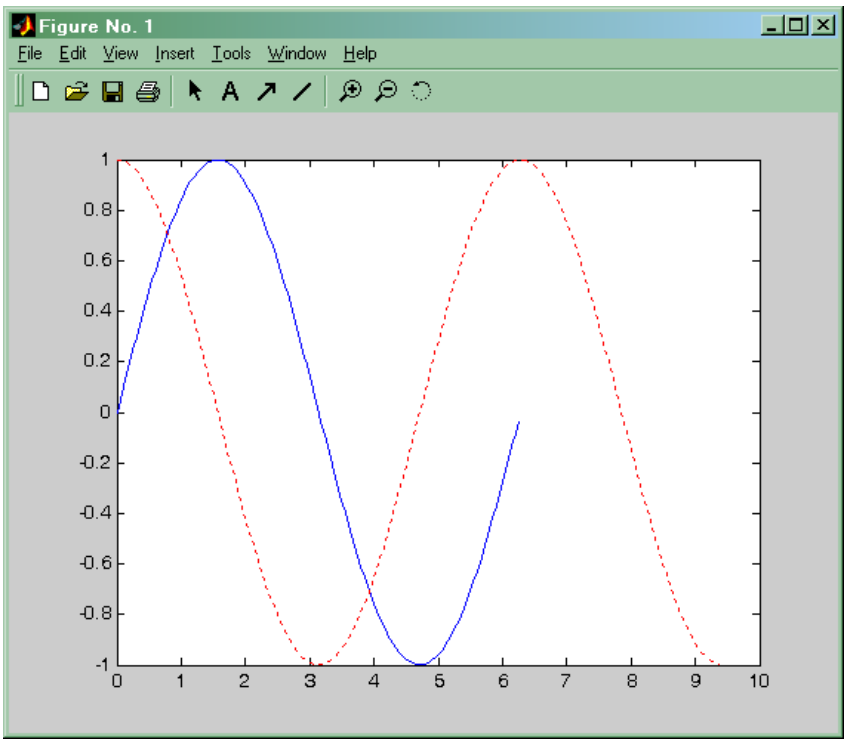


Рисунок 2.2 – Ще один приклад дії функції *plot*

Іншим методом побудови декількох графіків є поетапна побудова графіків у режимі збереження вмісту графічного вікна. При побудові чергового графіка пакет MATLAB за замовчуванням повністю переробляє зображення у графічному вікні, тобто, діє режим збереження координатних вісей. Для його включення існує оператор *hold.*

- Запис *hold {on/off}* включає або відключає режим збереження координатних вісей.
- Звернення *hold* змінює поточний стан режиму збереження координатних вісей на протилежний.
- Функція *k = ishold* дає інформацію про поточний стан режиму:

  k = 1 при включеному режимі, і *k = 0* у зворотньому випадку.

Зображення на рисунку 2.2 можна також отримати із використанням такої послідовності команд

```
t=0:0.05:2*pi;
hold on
plot(t,sin(t), 'b-')
t=0:0.05:3*pi;
plot(t,cos(t), 'V:');
hold off
```

Завершальний оператор *hold* відключає режим збереження вмісту графічного вікна, й використаний для відновлення первинного стану за замовчуванням.

Запис:

```
comet(x,y,p)
```

Будує графік залежності *y* від *x* поточково, демонструючи у часі послідовність накреслення точок на графіку.

Якщо записати приклад:

*t=0:0.0001:3*pi;*

*comet(sin(2*t),cos(t))*

можна побачити самому, як виглядає результат. Необов'язковий параметр *p* керує довжиною "хвоста".

2.4.1.2 Графіки в інших системах координат

В теорії керування й деякиї інших областях досить часто використовуються логарифмічні вісі, коли вздовж вісі відкладаються значення не величини, а її логарифму. Графіки у таких вісях будують наступні функції:

- *semilogx(x,y)* - будує графік із логарифмічним масштабом вісі абсцис;
- *semilogy(x,y)* - будує графік із логарифмічним масштабом вісі ординат;
- *loglog(x,y)* - будує графік із логарифмічним масштабом по обох вісях.

Усі ці функції мають такі ж форми синтаксису, як описана раніше *plot.*

Аналогічна побудова графіків й у полярних координатах, які задаються кутом *phi* та радіусом *rho,* функцією:

polar(phi,rho,s)

Як приклад наведемо команди, які будують криву, названу равликом Паскаля. Її рівняння у полярних координатах має вигляд

$\rho = a \cos \varphi + l$. При $a = 0.5$ і $l = 0.15$

*phi=0:0.05:2*pi; polar(phi,0.5*cos(phi) +0.15)*

Результат виконання цих команд приведений на рисунку 2.3.

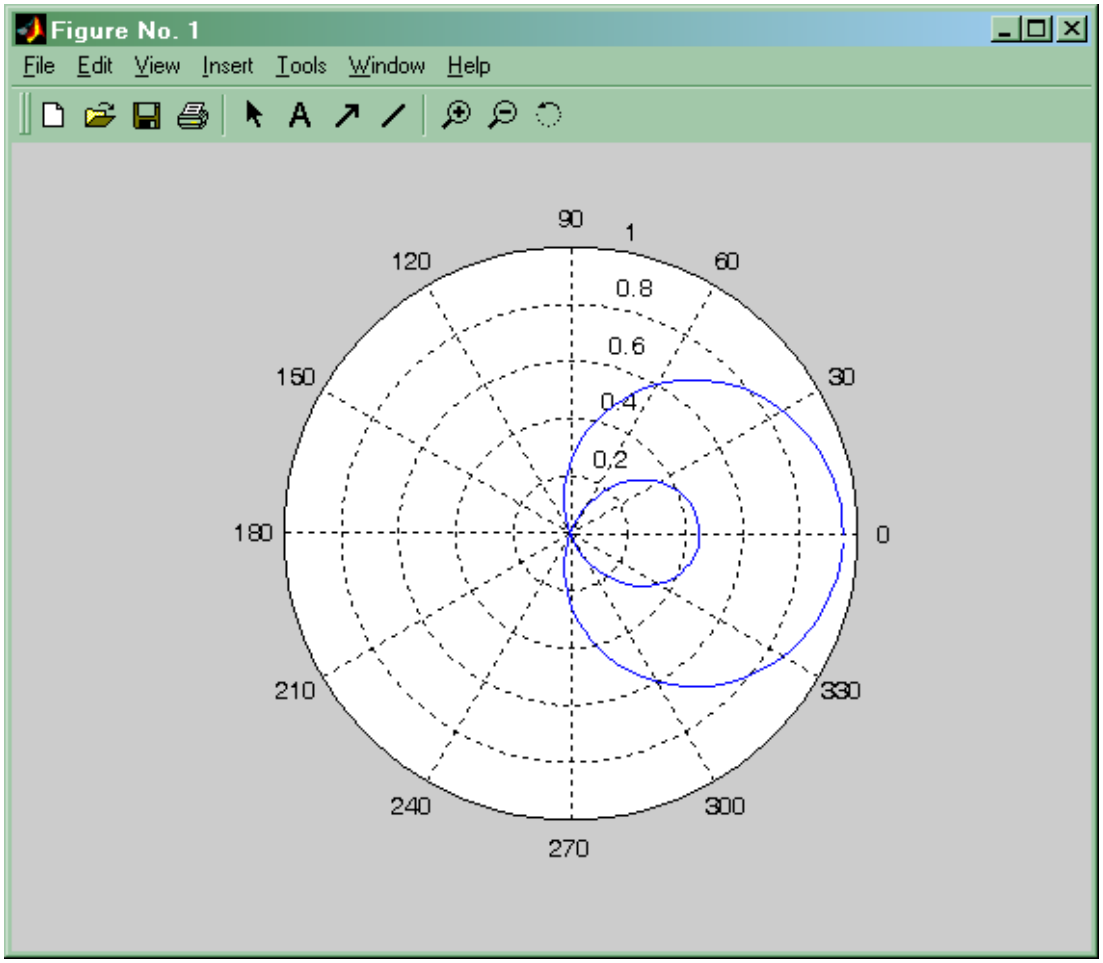


Рисунок 2.3 – Результат дії функції *polar*

2.4.1.3 Сітка, написи і пояснення на графіках

Усі попередні графіки, побудовані у попередній підпунктах, не мали сітки й не містили жодних написів, але вони необхідні при оформленні результатів.

Показати сітку на графіцідає змогу команда *grid*, щопослідовно проводить включення й відключення сітки. Команда *grid on* накреслює сітку на потому графіці, команда *grid off* відповідно відключає сітку.

Зробити написи на рисунках можна за допомогою таких функцій:

- *title(stext)* - виводить у графічне вікно текстову константу *stext* в якості надрисункового напису;
- *xlabel(stext)* - виводить у графічне вікно текстову константу *stext* в якості підпису до вісі *x*;

- *ylabel(stext)* - виводить у графічне вікно текстову константу *stext* в якості підпису до вісі *y;*
- *text(x, y, stext)* - виводить у графічне вікно текстову константу *stext* в якості напису, лівий верхній кут якого знаходиться у точці із координатами *x,y.* Якщо *x* та *y* – вектори, текст поміщається на всі позиції, задані цими векторами. Координати текстів необхідно задавати у фізичних координатах екрану;
- *gtext(stext)* - виводить у графічне вікно текстову константу *stext* в якості напису, лівий верхній кут якого знаходиться у точці із координатами, що вказуються мишею;
- *legend(stext1, stext2...*) - генерує у графічному вікні пояснення до графіків. Відповідність констант *stext1, stext2 ...* лініям на графіку відповідає порядку виведення графіків. Також можливі такі форми виклику функції:

— *legend(M),* де *M* -це масив рядків однакової розмірності.

— *legend off* – видаляє пояснення.

— *legend(stext1, stext2..., n)* - аргумент *n* встановлює граничне значення позицій пояснення, *n* = - 1 - пояснення розміщується поза зоноюграфіка, *n* = 0 - пояснення розміщується у зоні графіка, якщо місця для цього досить.

Приклад:

```
t=0:0.05:2*pi;
plot(t,[sin(t);cos(t);sin(2*t)])
title('Тригонометричні функції')
xlabel('Абсциса')
ylabel('Ордината')
text(5.5,0.2,'2*pi')
```

legend('sin(t)','cos(t)','sin(2t)',0)

Результати виконання цих команд наведено на рисунку 2.4.

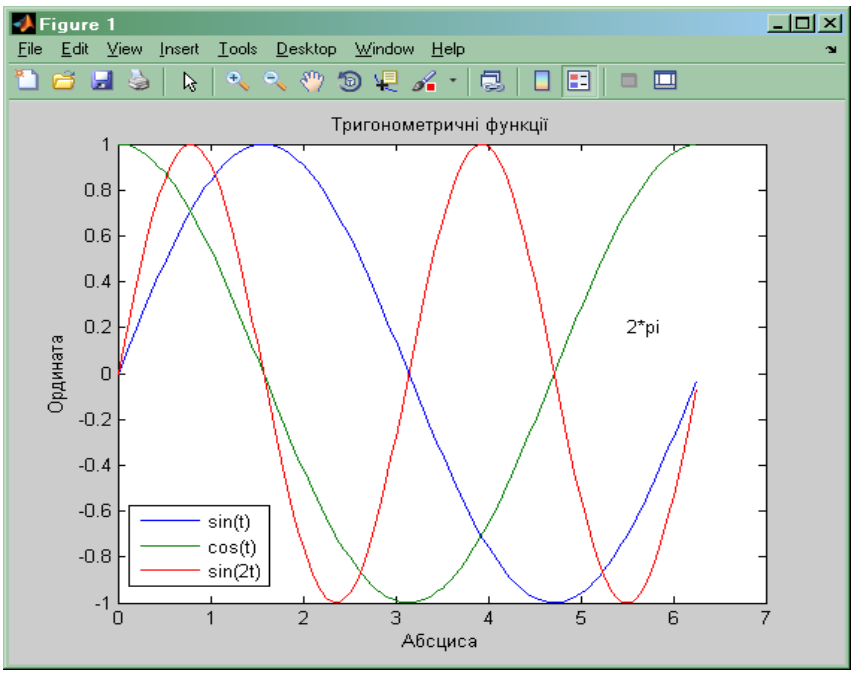


Рисунок 2.4 – Зразок написів до графіків

2.4.1.4 Декілька графіків у одному вікні

Для зручності аналізу результатів обчислень досить часто є необхідність побудови декількох графіків у одному графічному вікні. Робити це дозволяє функція *subplot.* Синтаксис цієї функції

- *hp = subplot(n,m,p)* і *hp = subplot(nmp)* - розбиття поточного графічного вікна на n×m підвікон. При цьому *n* – число підвікон по горизонталі, а m – число підвікон по вертикалі. У підвікні з номером *p* команда будує вісі й робить їх активними для подальшого використання функції *plot.* Номер *p* вказує на *j-е* підвікнов *i-м* рядку, де *i = floor(p/n),* а *j* – залишок від ділення *(j = rem (p,n)).* Тобто, для підвікон прийнята наскрізна нумерація за рядами зліва направо й зверху донизу. Повертає функція *subplot* покажчик *hp* - адреса структури для вісей, що розташовані у *p-ом* підвікні.

Виклик цієї функції для існуючої системи підвікон у разі, коли величини *n* і *m* не змінюються, встановлює активним підвікно із номером *p.* Якщо хоча б одне з чисел *m* або *n* змінилось, функція витирає попередній вміст вікна й проводить розбиття по новій.

- *subplot(hp)* - встановлює активним підвікно із покажчиком *hp.*
- *subplot('position'[left bottom width height])* - створює підвікно у області графічного вікна, що вказується програмно. В цьому випадку '*position'* – ключове слово, яке повідомляє функції про те, що надалі буде задано прямокутну область для розміщення вісей підвікна, а *left bottom width height* – аргументи, які задають положення лівого нижнього кута підвікна та його ширину й висоту у долях від розмірів графічного вікна.

Приклад:

```
t=0:0.05:4*pi;
h1=subplot(221);
plot(cos(t).*exp(-0.1*t),sin(t).*exp(-0.1*t))
h2=subplot(222);
polar(t,exp(-0.1*t))
h3=subplot('position',[0.1,0.1,0.8,0.35])
plot(t,sin(t).*exp(-0.1*t),'-',t,cos(t).*exp(-0.1*t),'_')
xlabel('Время,с, Угол, рад.')
ylabel(Координати, м")
subplot(h2)
title('Траекторія в полярних координатах ')
subplot(2,2,1)
xlabel('Дальність, м.')
ylabel('Висота, м')
```

title('Траєкторія в декартових координатах')

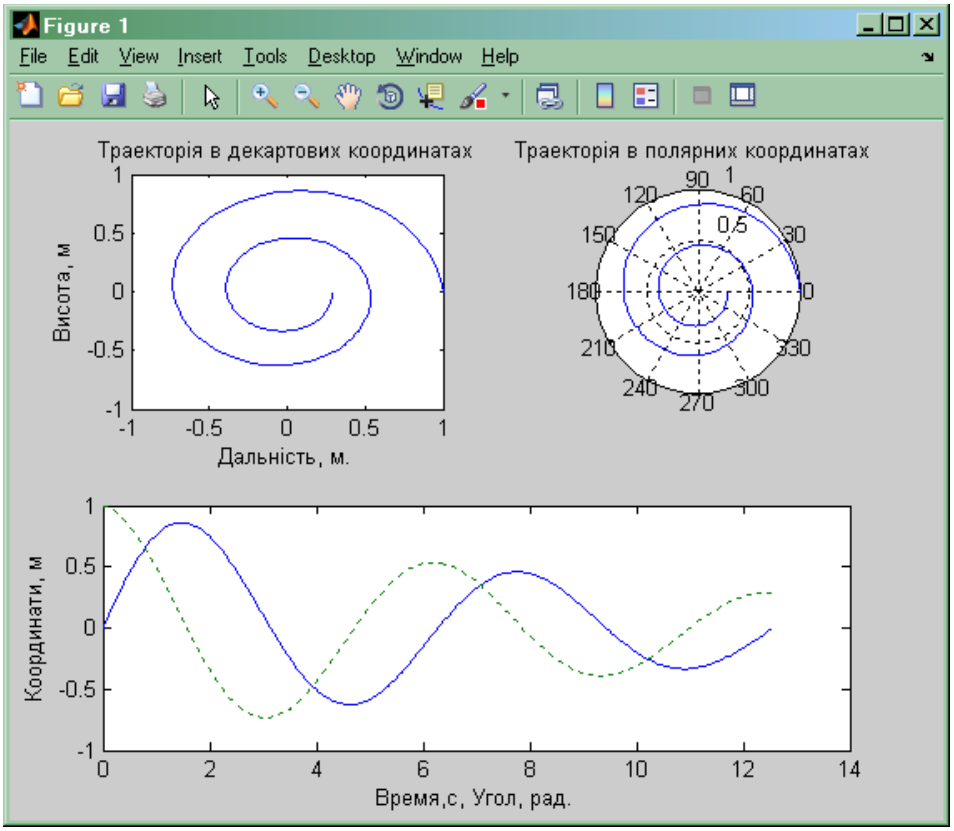


Рисунок 2.5 – Розбиття графічного вікна на "підвікна"

В даному прикладі надрисункові заголовки й підписи під координатними вісями можно було б зробити й відразу після побудови графіків за допомогою функції *plot*. Звернення до відповідних вісей проводиться для демонстрації звернення функцією *subplot*.

У підвікні можна будувати будь-яке зображення, в тому числі і тривимірне; робота із тривимірними зображеннями буде розглянута пізніше.

4.1.5 Графічні вікна і управління ними

MATLAB допускає одночасне створення декількох графічних вікон й креслення графіків в них.

• Для створення нового графічного вікна використовується функція *figure*. За відсутності формальних параметрів запис *h = figure* креслить нове графічне

вікно й робить його активним. Тепер усі викликані функції креслення графіків будуть будувати зображення у цьому графічному вікні. Запис *h = figure(N)* для цілого числа *N* створює нове графічне вікно із номером *N*, якщо вікна не існувало, й робить його активним. У обидвої випадках функція повертає покажчик на вказане ікно. Якщо *h* – покажчик на вікно звернення, *figure(h)* робить це графічне вікно активним.

• Дізнатися покажчик на активне графічне вікно дозволяє запис *h = gcf*.

• Очистити активне графічне вікно можна оператором *clf*.

• Закрити активне графічне вікно дає оператор *close*. Запис *close(h)* закриває графічне вікно із покажчиком *h*, а *close(N)* – графічне вікно із номером *N*. Запис *closeall* закриває усі відкриті графічні вікна.

Можливість розглянути деталі на графіках у графічному вікні дає оператор *zoom*. Звернення до нього має синтаксис:

• *zoom* - переключає стан режиму зміни масштабу графіку.

• *zoom on* - включає режим змінення масштабу графіку.

• *zoom off* - вимикає режим змінення масштабу графіку.

• *zoom xon* і *zoom yon*- включає режим змінення масштабу графіку по одній з вісей *x* або *y* відповідно.

• *zoom out* - відновлює початковий масштаб графіків у активному вікні.

• *zoom reset* - встановлює поточний масштаб графіків у активному вікні як резуюче (незменшуваного початку відліку).

• *zoom(F)* - встановлює масштаб графіків у активному вікні відповідно з параметром *F*, тобто збільшує зображення в *F* разів *(F* > 1).

Масштабування графіків при основних формах запису оператора *zoom* проводиться за допомогою миші. Для цього курсор миші необхідно підвести

до центру області графіка, що цікавить користувача. Якщо режим *zoom* включений, натиснення лівої клавіші збільшує масштаб вдвічі, а натиснення правої клавіші - зменшує вдвічі. При натиснутій лівій клавіші миші можливо виділити пунктирним прямокутником потрібна ділянку графіку – при відпуску клавіші ця ділянка з'явиться у збільшеному вигляді й у тому масштабі, який відповідає розмірам виділяючого прямокутника.

2.4.1.6 Діаграми, гістограми, вектора

Спеціальні можливості існують у пакеті й для побудови діаграм та гістограм і інших стандартних виглядів представлення інформації графічно.

• **Стовпчикова діаграма.** Будується з використанням функції *bar*.

— *bar(Y)* –будує *n* груп стовпчикових діаграм по *m* стовпчиків у групі ($n \times m$ – розмірність матриці *Y)*. По вісі абсцис відкладається номер стовпчика, який містить відповідне значення.

— *bar* (*x, Y)* – за матрицею *Y* також будує групи діаграм, але позиції стовпчиків визначаються вектором *x.*

— *bar* (*x, Y,' stack')* - будує групу діаграм, у якій для кожного рядка з *Y* будується один стовпчик діаграми.

— *bar(…,s)* - у текстовій константі *s* задається колір діаграми відповідно до таблиці, наведеної у описі функції *plot.*

- **Горизонтальна стовпчикова діаграма.** Діаграма будується із використанням функції *barh* й складається з горизонтальних стовпчиків, основні форми звернення такі ж, як у функції *bar*.
- **Кругові діаграми.** Запис *pie(x)* дає змогу побудувати кругову діаграму. Звернення *pie*(*x, explode)* дає змогу на круговій діаграмі відділити окремі сектори. Номери секторів задаються вектором *explode,* розмірність якого співпадає із довжиною вектора *x.* Ненульові

компоненти вектора *explode* визначають сектори, відокремлювані від діаграми.

•**Гістограми.** Функція побудови гістограм *hist* має декілька форм запису:

— *N = hist(Y, M)* - повертає вектор кількості попадань значень із вектора *Y* у *M* інтервалів, які рівномірно ділять інтервал *[min(Y), max(Y)]*. За відсутності аргументу *M* кількість інтервалів дорівнює 10. Якщо *Y* – матриця розмірності *n* x *m*, *N* –матриця розмірності *Mxm, і-й* стовпчик якої – гістограмма *і-го* рядка матіриці *Y*.

— *N = hist(Y, x)* - повертає вектор кількості попадань значень вектора *Y* у інтервали, середини яких задаються елементами вектора *x*.

— *[N, X] = hist(Y, M)* - повертає крім вектора числа попадань *N*, також вектор *X*, який містить середини інтервалів.

- **Кругові гістограми.** Функція кругової гістограми *rose* має форми, подібні до функції *hist*.
- **Ступінчастий графік,** *stairs*(*x, Y, s)* - показує залежність *Y* від *x* за допомогою ступінчастої функції. Форми звернення до неї подібні до функції *plot*.
- **Графік дискретних відліків.** *stem*(*x, Y, s)* -показує дискретні значення залежностей *Y* від *x*, сполучені вертикальною лінією із віссю абсцис. Форми звернення подібні до функції *plot*.
- **Графік з інформацією про погрішності.** *errorbar(x, Y, L, U)* – креслить дискретні значення залежності *Y* від *x* із вказанням інтервалів довіри значень функції. Верхні й нижні межі інтервалів задані аргументами *L* і *U*. При записі *errorbar(x, Y, V)* межі інтервалів довіри встановлені: *L = Y — V nU = Y+V*.
- **Графік проекцій векторів на площину.** Функція *feather*. При записі

feather(U, V) на координатній площині будуються вектори, початок яких знаходиться у точках із нульовими ординатамийі абсцисами *X0* =1 : *length(U)*, закінчення – у точках із координатами (*X0 + U,V).* Запис *feather(Z)* для комплексного вектора *Z* еквівалентний запису *feather(real(Z),imag(Z)).*

•**Графік векторів в полярних координатах.** Функція *compass*. *compass*(*U, V)* будує на площині вектори, початки яких знаходяться у точці із нульовими координатами, закінчення – у точках із координатами *(U,V).* Запис *compass(Z)* для комплексного вектора *Z* еквівалентний запису *compass(real(Z), imag(Z)).*

Опис, даний тут функціям, є неповним, й приведений список слугує тільки "маршрутним листом"для читача при ознайомленні із відповідними функціями.

2.4.2 Тривимірна графіка

2.4.2.1 Побудова кривих в просторі. Перше знайомство з функцією *plot3.*

Тривимірне зображення можна отримати в результаті креслення графіка параметрично заданої кривої у трьох вимірах. Для побудови такої кривої записують функцію *plot3,* синтаксис якої подібний до функції *plot,* розглянутої у розділі 2.4.1.1

plot3(x1, y1, z1, s).

В основному співпадає із записом функції графіка у двох вимірах та відрізняється появою вектора *z1*. Рядок *s* має значення, які вказано у таблиці розділу 2.4.1.1. Результат роботи функції можна продемонструвати на прикладі, рисунок 2.6.

```
x=(1:1000)/1000*3*pi;
plot3(x, sin(10*x),x.^2);
```

Синтаксично вірним також є звернення до решти усіх форм функції plot3, які копіюють форми plot.

Менш вдалим прикладом побудови кривої у просторі є рисунок 2.7. Цю криву записують кодом:

```
x=(1:1000)/1000*3*pi;
plot3(sin(25* x), cos(25* x), cos(2* x));
```

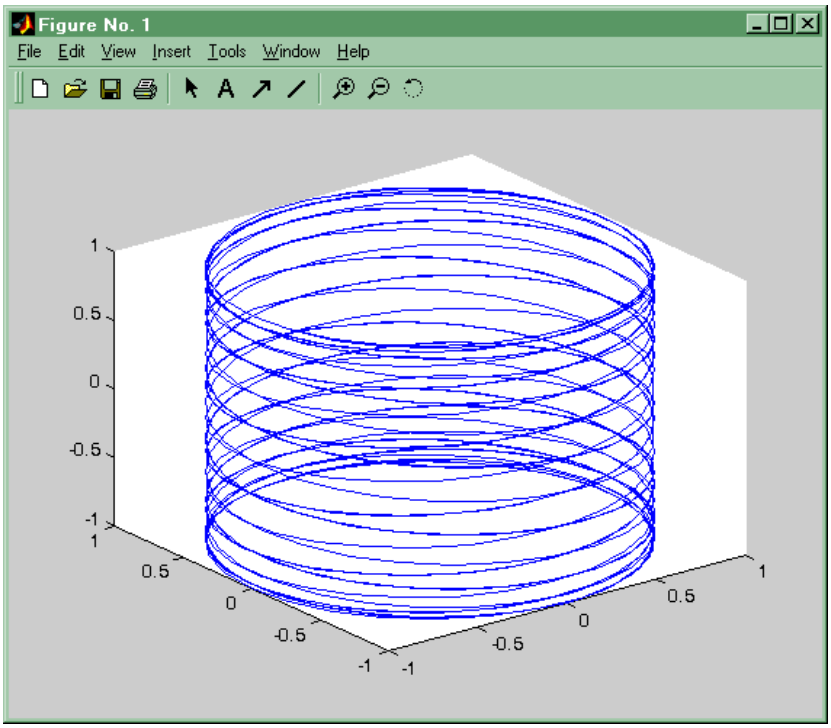


Рисунок 2.6 – Результати роботи функції *plot3*

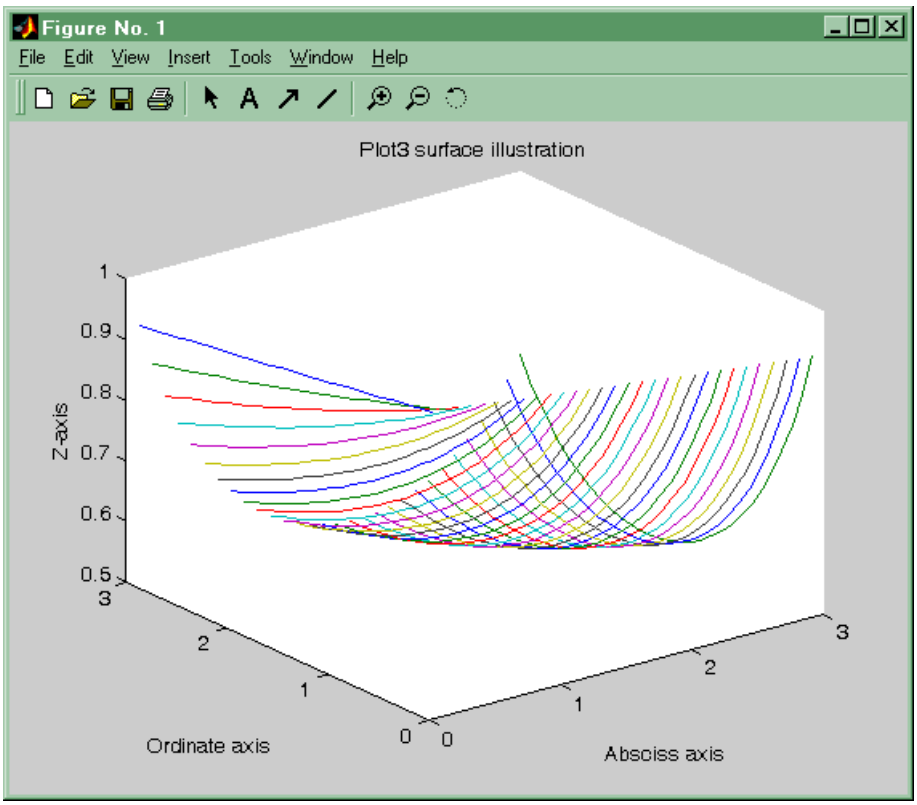


Рисунок 2.7 – Незрозумілий результат дії оператора plot3

Для приведеного прикладу неможливо зрозуміти, де початок й кінець побудованої кривої, також незрозуміло, яким чином будуються криві, оскільки функція *plot "*будує" зображення у пам'яті комп'ютера та демонструє на екрані отриманий результат. Як побудова графіка розгортається у часі, можливо можливо побачити за допомогою функції comet3:

*comet3(sin(25 * x), cos*(25 * *x*), *cos*(2 * *x*));

При цьому потрібно слідкувати за тим, щоб на екрані одночасно було виведено командне й поточне графічні вікна MATLAB'у. Результат побудови буде таким самим, але відстеживши те що відбувається на екрані монітору, є можливітсь зрозуміти, що побудова вказаної кривої аналогічна намотуванню нитки на котушку й далі за виглядом аргументів приблизно обрахувати кількість витків, число "рухів руки"із ниткою вниз та вгору тощо.

Написи до зображення й конкретних вісей для 3D графіків записуються відомим методом, описаним у розділі 2.4.1.3. Для 3D графіки додані функції *zgrid* і *zlabel* аналогічні *xgrid, ygrid* і *xlabel, ylabel,* описаним раніше.

2.4.2.2 Поверхні у просторі

Найпростіший метод побудови поверхні у просторі представляє функція *plot3.* Якщо в якості аргументів цієї функції використати матриці однакової розмірності, функція *plot3* побудує набір кривих, сукупність яких будуть утворювати поверхню.

Приклад:

```
X=0.1:0.1:3;
for i=2:30, X(i,:)=X(1,:); end
Y=X';
Z=sin(0.3*X.*Y); Z=1./(sin(0.3*X.*Y)+1);
plot3(X,Y,Z)
title('Plot3 surface illustration')
```

xlabel('Absciss axis')

ylabel('Ordinate axis')

zlabel('Z-axis')

Для приведеного прикладу є можливість побудувати також і сітчасту поверхню. Для цього використовують симетрією масивів *X* і *Y* та запис *plot3*(*X, Y, Z,'k', Y, X, Z,' k').*

В результаті у графічному вікні з'явиться зображення представлене на рисунку2.8.

Для зручного графічного уявлення тривимірних поверхонь у MATLAB'і є набір таких спеціальних функцій:

- *mesh*(*X,Y, Z)* - будує кольорову сітчасту поверхню *Z(X,Y),* колір вузлів задається висотою поверхні.
- *mesh*(*X, Y, Z)* - будує кольорову сітчасту поверхню *Z(X, Y)* того ж типу, що і *mesh,* а на площині *X, Y* проводяться лінії рівня поверхні.
- *meshz*(*X,Y, Z)* - будує кольорову сітчасту поверхню *Z*(*X,Y),* того ж типа, що і *mesh,* краї якої відмічені стовпчасто. Для ілюстрації роботи цієї функції непогано використовувати поверхню вигляду:

Z1 = *sin*(0.3 * *X*. * *Y*)+ 1;.

- *surf*(*X,Y,Z)* - будує кольорову поверхню у вигляді комірок *Z(X,Y), дн* колір комірок задається як висота поверхні.
- *surf*(*X, Y, Z)* - будує кольорову поверхню у вигляді комірок *Z(X, Y*) такого ж типу, що і *surf,* де на площині *X, Y* буде проведено лінії рівня поверхні.
- *surfl*(*X,Y, Z)* - будує кольорову поверхню у вигляді комірок *Z*(*X,Y)* того ж типу що і *surf,* але колір буде імітувати підсвічування додаткового джерела світла. При записі *surfl*(*X, Y, Z, S)* вектор *S* =

[Sx, Sy, Sz] буде задавати положення джерела світла у декартовій системі координат, а *S = [Az, El]* – відповідно у сферичній системі координат.

- *water fall*(*X,Y, Z)* будує кольорову листкову поверхню *Z*(*X,Y)* подібнк до *meshz,* але стиль креслення більше нагадує порізаний вертикальними шматками пиріг.

- *contour3(X, Y, Z)* будує кольорову поверхню у вигляді листа *Z(X, Y*) у якій шари відокремлені лініями рівного рівня, накресленими у оксанометричній проекції. У зверненні *contour3*(*X, Y, Z, n)* аргумент *n* буде задавати кількість рівнів.

Для усіх, перелічених вище функцій допустимі звернення типу:

•*surf*(*x, y, Z),* де *x* і *y* вектора однакової розмірності *n*, а *Z* - квадратна матриця розмірності *nxn*. У цьому випадку функція сама створює матриці *X* та *Y*.

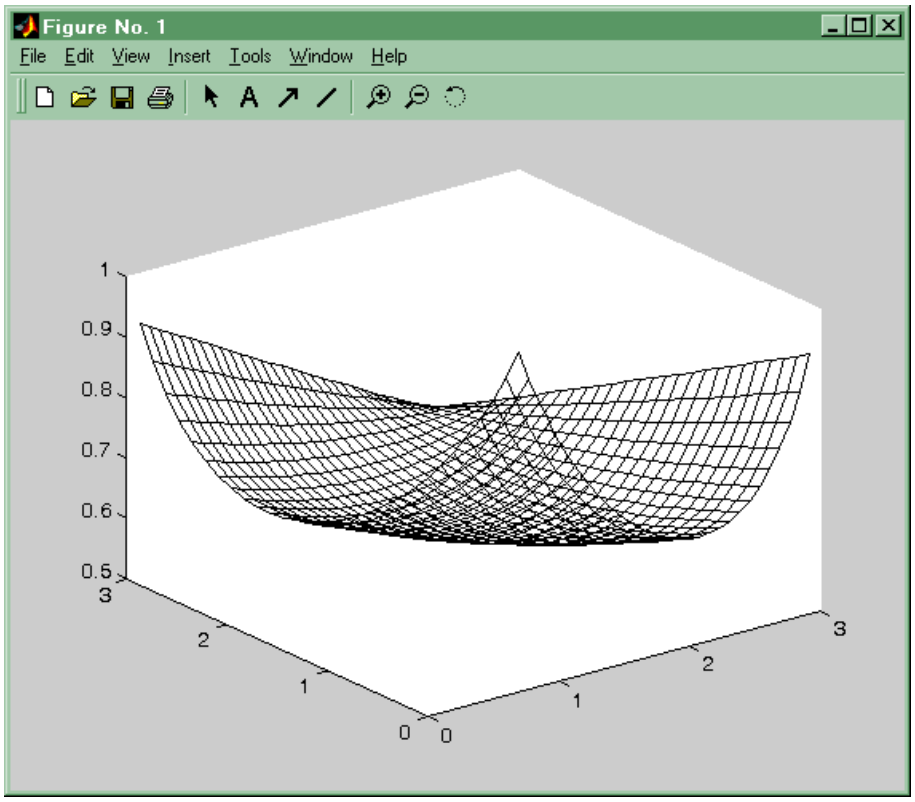


Рисунок 2.8 – Результат будування поверхні функцією plot3

surf(Z) будує поверхню з відкладанням по вісях абсцис та ординат номери стовпчиків і рядків матриці *Z.*

Побудувати лінії рівня на площині *xy* для даної поверхні можна за допомогою функції *contour,* синтаксис якої цілком співпадає зі зверненням до функції *contour3.* Ці функції повертають матрицю *Cs* з описом контурних ліній. Маркірувати ці лінії для результуючих графіків функцій можна за допомогою функції *clabel* – *clabel(Cs),* де матрицю *Cs* розраховують та повертають функції *contour* і *contour3.* При записі *clabel(Cs, v)* маркіруються висоти, які вказані у векторі *v.* Функції побудови ліній рівня можуть також при зверненні *[Cs, H] = contour*(*X, Y, Z)* повертати вектор, який містить інформацію про висоти. При записі *clabel(Cs, H)* і *clabel*(*Cs,H,v)* написи ліній рівня з відсутністю додаткових маркерів будуть встановлені у розриви контурних ліній.

Представити характер поверхні також допомагає функція *quiver.* При записі *quiver*(*X, Y, U, V)* у графічному вікні накреслиться зображення градієнтного поля. У точках, які задані координатами матриць *X* та *Y* відображаються вектори, величини й напрями яких задаються матрицями *U* і *V. З*ручно використовувати таку функцію для відображення градієнтного поля поверхні, яка задана матрицею *Z,* так:

[px,py] = gradient(Z, hx, hy);

quiver(X, Y,px,py)

У приведеному раніше прикладі значення кроків *hx* і *hy* по вісях абсцис та ординат слід вказувати рівними 0.1.

Для побудови матриць *X* і *Y* за векторами *x* та *y* можли використати спеціальну функцію, яка має синтаксис:

[X, Y] = meshgrid(*x,y).*

2.4.2.3 Вигляд зображення з різних боків

Після побудови тривимірної поверхні користувачу зручно буде розглянути її з іншого боку. Обрати точку огляду, відмінну від стандартної оксанометричної проекції, можна за допомогою функції *view*. Синтаксис функції:

•*view(Cv),* де *Cv* - вектор, що задає напрям перегляду.

Для тривимірного вектора *Cv = [Cx, Cy, Cz]* - *Cv* - декартові координати точки перегляду. Лінія візування – це промінь, що проведено з початку координат *Cv,* який проходить через точку.

Для двовимірного вектора *Cv = [Az, El]* - величини *Az* і *El* задають лінію візування, використовуючи кути азимута й піднесення. Так само буде сприйнято і запис *view*(*Az, El).*

Зображення, що буде виведено на екран є проекцією на площину, яка ортогональна лінії візування, проведеної поза межами об'єкту.

- *view(2)* встановлює штатне положення точки перегляду для 2D графіки (*Az* = 0°, *El* = 90°).
- *view(3)* встановлює штатне положення точки перегляду для 3D графіки (оксанометрична проекція – *Az = -37.5°, El = 30°)*.
- *[Az, El] = view* - повертає поточне значення кута азимуту й піднесення.
- *T = view* повертає поточне значення узагальненої матриці *T* перетворень, використовуваної для отримання зображення.

T = viewmtx(*Az, El).* Звернення *view(T)* встановлює положення точки перегляду, яке задається матрицею *T.*

Приклад:

*x=(1:100) /100*3*pi-pi;*

y=(1:100) /100-0.5;

[X, Y]=meshgrid(x,y);

*Z=sin(X. *Y);*

mesh(X, Y, Z)

Звернення *view(2)* дає уявлення про вигляд зверху на зображену поверхню, *view*(30, 30) дозволяє поглянути на неї "збоку", а *view*(*45* -60) - знизу. Забавно, що в останньому випадку MATLAB будує координатні осі для *Az* = 60.

Розібратися в ситуації з картинкою, окрім вашої просторової уяви дозволяє функція *colorbar,* що додає до поточного графіка шкалу палітри. При записі до цієї функції без аргументів або – *colorbar* (*'vert')* ця шкала вертикальна. При записі *colorbar*(*'horiz')* – горизонтальна.

Шкала палітри, зображена в графічному вікні, є окремим графіком, накресленим у подвікні, яке описане в підпункті 2.5. Це підвікно залишається активним й після закінчення відпрацювання функції *colorbar,* це необхідно враховувати при подальшому кресленні зображень у графічному вікні.

2.4.2.4 Друк, зберігання і експорт зображень

Надрукувати зображення з графічного вікна на принтері, приєднаному до комп'ютера, на якому працює користувач, нескладно. Для цього достатньо у меню графічного вікна вибрати позицію "File", а у тому, що відкриється, підменю обрати позицію "Print". Подальші дії також відповідають стандартним діям системи Windows.

Перенести зображення на комп'ютер із тією ж версією MATLAB'у вдається, зберігши його. Для цього необхідно у меню графічного вікна обрати позицію File, а у тому, що відкриється, підменю позиції Save або Save as. Різні версії MATLAB'у пропонують різноманітні способи збереження зображень: версії, раніші ніж MATLAB 5.2, використовують для збереження файли **.m* і **.mat*, версія MATLAB 5.3 - **.m* і **.fig* файли. При відкритті цих **.fig* файлів із використанням позиції підменю Open зображення з'являється у поточному графічному вікні. Для відновлення зображення з **.m* і **.mat*

файлів слід виконати відповідний сценарій з *.*m* файлу. Проте не всі версії MATLAB'у нормально читають чужі файли.

Проблеми виникають у випадку, якщо необхідно включити зображення у звіт. При наборі звіту у редакторові MS Word можна скористатися стандартним шаблоном M-book, який поставляється спільно із системою MATLAB. Для використання інших редакторів необхідно експортувати зображення, використовуючи clipboard Windows. Для цього у меню графічного вікна обрати позицію Edit, а в тому, що відкриється, підменю позицію Copy Figure. Залежно від установок проводиться копіювання зображення в одному з форматів Windows Metafile або Windows Bitmap. Ці зображення можна зобразити у будь-якому графічному редакторові із використанням пункту підменю Paste або комбінації клавіш Ctrl-v. Для формату Windows Metafile при відновленні зображення у редакторах Paint, Paintbrush, Adobe Photoshop й подібних спостерігається спотворення шрифтів у написах. При використанні формату Windows Bitmap шрифти передаються без спотворень. Для установки відповідного формату необхідно у меню графічного вікна обрати позицію File, а в тому, що відкриєтья, підменю, позицію Preferences.... У вікні, що знов відкрилося, слід обрати сторінку з закладкою Copying Options й у розділі Clipboard Format встановити обраний вами формат.

2.5 Числовий розв'язок задачі Коші для звичайних диференціальних рівнянь

В пакеті MATLAB є багато можливостей розв'язку задачі Коші для звичайних диференціальних рівнянь: стандартні ODE-вирішувачі, спеціалізовані програми дослідження моделей для керованих систем, засоби імітації аналогового моделювання у середовищі SIMULINK. Докладений опис засобів середовища SIMULINK може бути предметом окремого обговорення (див. [11], [18-19]). Дві перші можливості розглянемо тут докладніше.

2.5.1 Стандартні ODE-вирішувачі

Стандартні ODE-вирішувачі – це пакет програм, який містить М-функції: *ode45, ode23, ode113, ode15s, ode23s, ode23t, ode23tb,* надалі позначені через *solver*

Вказані вище функції реалізують такі методи розв’язку систем звичайних диференціальних рівнянь:

- *ode45* - явні методи Рунге-Кутти четвертого й п’ятого порядків. Це класичний метод, що рекомендується для початкового наближення розв'язку, й зазвичай дає задовільні результати.
- *ode23* - явні методи Рунге-Кутта другого й третього порядків. При помірних вимогах до точності розв’язук метод може дати виграш у швидкості розв'язку.
- *ode113*- багатокроковий метод Адамса-Башворта-Мултона змінного порядку – адаптивний метод, викликаний забезпечити високу точність розв'язку.
- *ode15s* - багатокроковий метод змінного порядку – від першого до п’ятого. Адаптивний метод, що дає високу точність розв'язку для жорстких систем.
- *ode23s* - однокроковий модифікований метод Розенброка 2-го порядку, що викликаний забезпечити високу швидкість обчислень для жорстких систем при низькій точності.
- *ode23t*- метод трапецій з інтерполяцією. Метод дає хорошу точність при рішенні жорстких задач.
- *ode23tb* - неявний метод Рунге-Кутта на початковій стадії розв'язку й метод, що використовує формули зворотного диференціювання другого порядку в подальшому. При низькій точності для жорстких систем цей метод може опинитися ефективніше, ніж *ode23s.*

Вказані вище функції призначені для числового розв'язку задачі Коші систем звичайних диференціальних рівнянь вигляду.

$$\begin{cases} \dfrac{dy}{dt} = f(y,t) \\ y(0) = y_0 \end{cases} \qquad (2.4)$$

де y – це вектор розмірністю n.

Вирішувачі *odel5s, ode23s, ode23t, ode23tb* також призначені для розв'язку завдання Коші для систем звичайних диференціальних рівнянь неявного вигляду:

$$\begin{cases} M(t)\dfrac{dy}{dt} = f(y,t) \\ y(0) = y_0 \end{cases} \qquad (2.5)$$

де *M(t)* - матриця масових коефіцієнтів. Для функції *ode23s* матриця *M* не повинна залежати від аргументу *t*.

Звернення до функцій числового інтегрування може приймати такі форми:

[T, Y] = solver('Fun',tspan,y0)

[T,Y] = solver('Fun',tspan,y0, options)

[T,Y] = solver(Fun,tspan,y0,options,p1,p2 …)

[T,Y,TE,YE,IE] = solver('Fun',tspan,y0,options,p1,p2…)

•*Fun* – ім'я файлу, в якому поміщена М-функція обчислення правих частин системи диференціальних рівнянь. Опис цієї функції приймає одну з наступних форм

function f = Fun(t, y) (2.6)

function f = Fun(t, y, flag) (2.7)

function f= Fun(t,y, flag,p1,p2…) (2.8)

- *tspan* – вектор значень моментів часу. Якщо вектор *tspan* складається з двох компонент *[t0,tfin]*, то ці значення сприймаються як початкове і фінальне значення аргументу. Для вектора більшої довжини *[t0,t1...,tfin]*, компоненти якого розташовані в порядку зростання або спадання, результати числового інтегрування видаються для вказаних значень аргументу.
- *y0* – вектор початкових значень.
- *options* – аргумент задаючий параметри для числового інтегрування. Цей аргумент задається функцією *odeset*, використання якої обговорюватиметься нижче.
- *p1,p2...* – довільні параметри, що передаються у функції *Fun.*
- *T* – вектор значень аргументу, для якого видаються значення функції *y.*
- *Y* – матриця розв'язків, кожен рядок якої відповідає значенню аргумента, поверненому в *Y.*
- *TE, YE, IE* - вихідні матриці, в яких виводяться значення аргументу *t,* функції *y* і номери події (event) для роботи опції фіксації подій.

Приклад розв'язку задачі Коши

Перш ніж розбирати далі різні можливості рішення задач числового інтегрування систем звичайних диференціальних рівнянь, приведемо простій приклад, що може послужити зразком для розв'язку 90% завдань такого типу. Розв'язок рівняння Ван дер Поля:

$$x'' = -x + 0.3 * x' * (1 - x^2)$$

з початковими умовами:

$$x(0) = 1;\ x'(0) = 0$$

Зробимо заміну змінних $y_1 = x$ і $y_2 = x'$. Перетворимо завдання до вигляду

$$\begin{cases} y_1' = y_2 \\ y_2' = -y_1 + 0.3 * y_2 * (1 - y_1^2) \\ y_1(0) = 1 \\ y_2(0) = 0 \end{cases} \quad (2.9)$$

Для цієї системи створимо функцію обчислення правих частин з описом типу 9, помістивши цю функцію у файл *vanderp.m* у робочій папці

```
function f=vanderp(t,x)
% function f=vanderp(t,x)
% right part for VanDerPol equation calculation
% dx1/dt=x2
% dx2/dt=-x1+e*x2*(1-x1^ 2)
f(1,1)=x(2);
f(2,1)=-x(1)+0.3*x(2)*(1-x(1)^2);
```

Результат отримаємо набравши в командному вікні MATLAB'а команду:

ode45('vanderp ',[0,30], [1,0])

В результаті такого звернення MATLAB відкриє графічне вікно і виводитиме в нього залежність *x(t)* у міру числового інтегрування. Результат приведений на рисунку 2.9. Графіки різних компонентів вектора *y* виводяться різними кольорами. Маркерами відмічено точки, у яких обчислені значення.

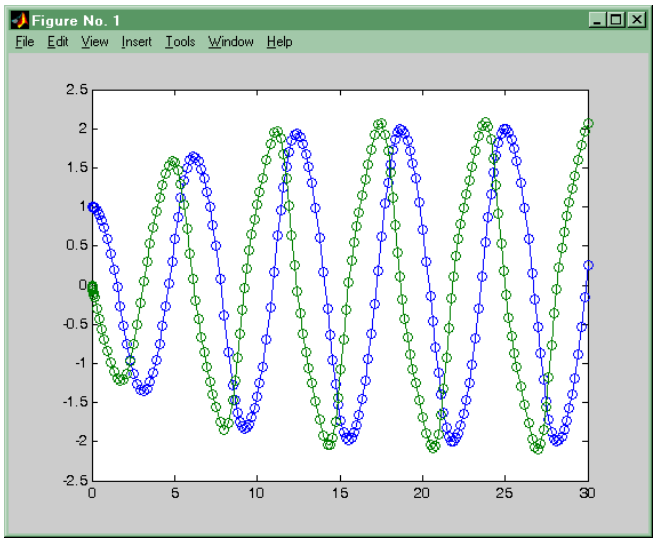


Рисунок 2.9 – Результат дії функції ode45

Щоб отримати числові значення розв'язку, використовуємо звернення:

[T, Y]=ode45('vanderp',[0,30],[1,0]);

В цьому випадку зображення на екрані не відображається, але вихідні масиви *T, Y* заповнені. З їх використанням можна отримати, наприклад, фазовий портрет знайденого вирішення. Для цього побудуємо залежність y_2 від y_1 за допомогою команди

plot(Y(:, 1),Y(:,2))

Результат побудови приведений на рисунку 2.10.

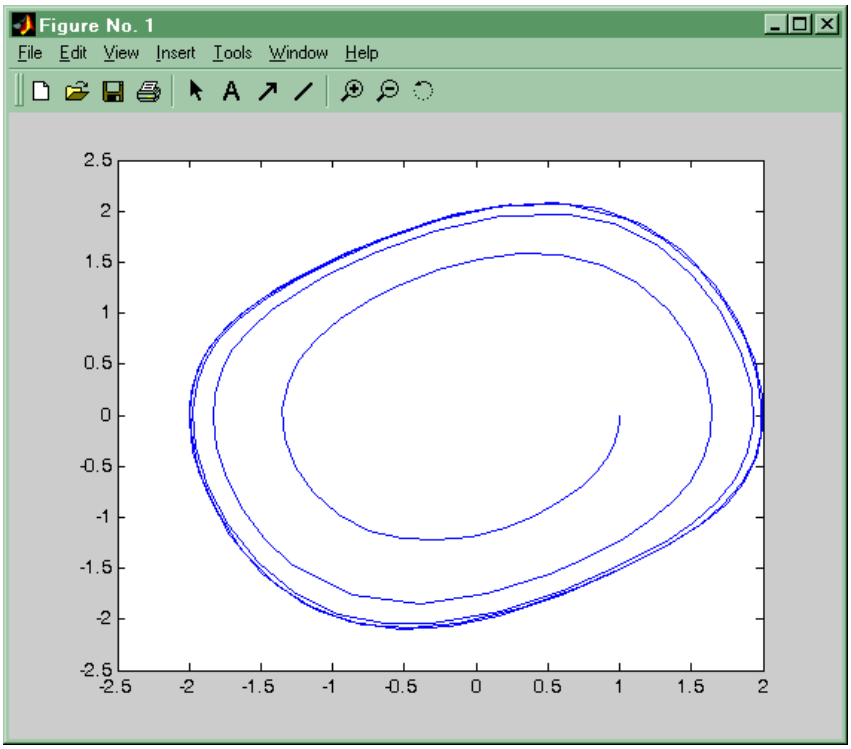


Рисунок 2.10 – Фазовий портрет розв'язків рівняння Ван дер Поля.

Можна також подивитися залежність довжини кроку інтеграції від номера кроку, виконавши команду *plot(diff(T)).* Отриманий графік наочно продемонструє змінність кроку в ході числового інтегрування. Подібний автоматизм у виборі кроку може надати погану послугу при інтеграції систем з розривною правою частиною. Автоматичний вибір кроку при частому "переключенні" приводить точні методи до непомірного збільшення часу обчислень.

2.5.2 Опції вирішувача

У цьому підпункті обговоримо деякі можливості використання аргументу *options*. За допомогою цього масиву встановлюються різні спеціальні параметри *ode-вирішувачів.* Перерахуємо ті з них, які представляються найбільш важливими:

1. *RelTol*- відносна погрішність. Скалярна величина за умовчанням приймається рівною 10^{-3}. Помилка на кожному кроці інтеграції оцінюється величиною $ei < max(RelTol * abs(y_i), AbsTol_i)$.

2. *AbsTol* - вектор абсолютних погрішностей (за умовчанням всі його компоненти рівні 10^{-6}).

3. *Refine* - позитивне ціле число - чинник якості, що підвищує кількість значень аргументу, що виводяться. Використовується для згладжування висновку. За замовчуванням вирішувачі окрім *ode45* використовують значення 1, а *ode45* - 4. Опція *Refine* не використовується, якщо довжина вектора *tspan* більше 2.

4. *OutputFcn* - текстова константа *'AFun',* що містить ім'я додаткової функції виведення. Вирішувач викликатиме вказану функцію після закінчення кожного кроку інтегрування. Зазвичай за умовчанням додаткова функція виведення відсутня, але за відсутності вихідних параметрів вона приймає значення '*odeplot'*. Її роботу ми і бачили в першому з простих прикладів, що наводилися. Для того, щоб на екран комп'ютера виводився графік за наявності вихідних параметрів необхідно встановити значення опції *OutputFcn* рівне '*odeplot'*. При необхідності можна використовувати також функції:

   - *odephas2* - зображає двовимірний фазовий портрет;
   - *odephas3* - зображає тривимірний фазовий портрет;
   - *odeprint* - виводить в командне вікно проміжні результати числового інтегрування.

Користувач може самостійно написати функцію такого типу. Єдиною вимогою до такої функції є те, що її опис повинен мати вигляд:

function AFun(t, z)

де *z* - вектор *y* змінних стану вирішуваної системи рівнянь або вектор меншої довжини, який складається з компонент вектора *y* порядок яких задається опцією *OutputFcn.*.

5. *OutputSel* - вектор номерів компонент вектора стану *y,* використовуваних вирішувачем при зверненні до функції додаткового виведення *OutputFcn.* За замовчуванням використовуються всі компоненти вектора стану.
6. *Stats* - відображає обчислювальну статистику після закінчення роботи вирішувача (приймає значення *on/off).*
7. *Jacobian*- ознака можливості обчислення Якобіана (*on/off).* Для використання цієї опції необхідно, щоб звернення до функції *Fun* обчислення правих частин системи рівнянь мало вигляд (2.7). При цьому при значенні змінної *flag* рівному '*Jacobian'* вказана програма повинна обчислювати матрицю Якобі, компоненти якої мають вигляд $\partial f_i / \partial x_j$. Матриця Якобі є такою, що повертається параметром функції *Fun.* Використання цієї опції може значно понизити число кроків і час числового інтегрування.
8. *JConstant* - ця опція повинна приймати значення '*on'* у випадку, якщо матриця Якобі для системи диференціальних рівнянь є постійною.
9. *Events* - ознака використання обробника подій *on/off.* Для використання цієї опції необхідно, щоб звернення до функції *Fun* обчислення правих частин системи рівнянь мало вигляд (2.7). При значенні змінної *flag* рівному '*Events'* вказана програма повинна повертати наступні величини

- *value* - вектор величин, для яких повинно бути зафіксований момент

зміни знаку

- *isterminal* - логічний вектор, компоненти якого приймають значення 0 або 1 залежно від того, чи слід зупиняти числове інтегрування при зміні знаку відповідною компоненти вектора *value.*
- *direction* - вектор тієї ж довжини, вказуючий напрям зміни знаку, при якому відповідна подія буде зафіксована. Компоненти цього вектора приймають значення 1 і -1 залежно від напряму зміни знаку. Значення відповідною компоненти рівне 0 означає, що напрям зміни знаку неістотний.

10. *Mass* - ознака використання обчислюваної матриці M(t) (інерційних коефіцієнтів при похідних в лівій частині рівнянь) [*on|{off}*]. Для використання цієї опції необхідно, щоб звернення до функції *Fun* обчислення правих частин системи рівнянь мало вигляд (2.7). При значенні змінної *flag* рівному '*Mass'* вказана програма повинна повертати матрицю інерційних коефіцієнтів - "мас". Цей параметр обробляють функції *ode15s, ode23s, ode23t, ode23tb.* (Відмітимо, що змінну матрицю мас обробляє тільки функція *ode15s).*

11. *MassConstant* -признак використання постійної матриці "мас"(інерційних коефіцієнтів при похідних в лівій частині рівнянь) [*on|{off}*]. Цей параметр повинен приймати значення '*on'*, якщо при зверненні до функції *Fun* із змінній *flag* рівною '*Mass'*. Це значення інформує про те, що функція *Fun* повертає постійну матрицю інерційних коефіцієнтів ("мас").

12. *MaxStep* - позитивний скаляр - максимальне значення кроку інтеграції (за умовчанням приймає значення десятої частини інтервалу часу tspan).

13. *Initial Step* - позитивний скаляр - початкове значення кроку інтеграції. За умовчанням програми визначають початковий розмір кроку автоматично.

14. *MaxOrder* - максимальний порядок для інтегруючої функції *ode15s* і

приймає одне із значень [1 | 2 | 3 | 4 | 5].

2.5.3 Програми числового інтегрування лінійних моделей керованих систем

Одним з основних достоїнств пакету MATLAB є наявність широкого набору бібліотек - TOOLBOX'ів - призначених для розв'язку спеціальних типів математичних і інженерних завдань. У поточному параграфі розглянемо використання декількох функцій бібліотеки "Control system toolbox" для розв'язку систем лінійних диференціальних рівнянь. Детальніше з функціями Control system toolbox можна ознайомитися у [19].

Модель лінійної динамічної керованої системи запишемо у вигляді системи диференціальних рівнянь першого порядку:

$$\begin{aligned} \frac{dx}{dt} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \qquad (2.10)$$

де x - вектор стану системи, u - вектор управлінь, y - вектор виходів, A, B, C, D - матриці з постійними коефіцієнтами.

У бібліотеці "Control" представлені наступні функції, що дозволяють вирішувати задачі інтегрування:

- *initial* - функція рішення задачі Коші для за заданих початкових умов. Звернення до цієї функції може мати вигляд:

[Y, T, X] = initial(A, B, C, D, x0, Tf)

Тут *A, B,C,D* - матриці, що описують систему вигляду (2.10); *x*0 - вектор початкових умов, *Tf* - значення моменту часу, до якого здійснюється інтегрування. Вихідними параметрами є матриці *Y* і *X,* в яких поміщаються порядково значення векторів *y і x* для моментів часу з відповідного вектора *T.*

•*impulse*- обчислення вагової функції системи (2.10). Функція розв'язку задачі про поведінку системи при імпульсному збуренні входу з номером *i* сигналом типу дельта-функції.

$$u_i(t) = \delta(t) = \begin{cases} 1 \textit{ при } \text{t=0} \\ 0 \textit{ при } \text{t} \neq 0 \end{cases}$$

при нульових початкових умовах.

Запис цієї функції може мати вигляд:

[Y, T, X] = impulse(A, B, C, D, i, Tf)

Тут *A,B,C,D* - матриці, що описують систему вигляду (2.10); *i* - номер збурюваного входу, *Tf* - значення моменту часу, до якого здійснюється інтегрування. Вихідними параметрами є матриці *Y* і *X*, в яких поміщаються відрядкові значення векторів *y* і *x* для моментів часу з відповідного вектора *T*.

•*step*- обчислення перехідної функції системи (2.10). Функція розв'язку задачі про поведінку системи при збуренні входу з номером *i* сигналом у вигляді одиничної сходинки:

$$u_i(t) = \delta(t) = \begin{cases} 1 \textit{ при } \text{t} < 0 \\ 0 \textit{ при } \text{t} \geq 0 \end{cases}$$

Звернення до цієї функції може мати вигляд:

[Y,T,X] = step(A,B,C,D,i,Tf)

Аргументи і вихідні параметри функції *step* мають той же сенс, що і для функції *impulse.*

• *lsim* - обчислює рішення системи для випадку довільного вхідного сигналу, заданого масивом своїх відліків - матрицею *U*. Звернення до цієї функції може мати вигляд:

[Y, T, X] = lsim(A, B, C, D, U, T, x0)

Тут *A,B,C,D* - матриці, що описують систему вигляду (10); *U* і *T* - матриця значень векторної функції *u* і відповідних ним моментів часу. Вихідними параметрами є матриці *Y* і *X*, в яких поміщаються відрядкові значення векторів *y* і *x* для моментів часу з відповідного вектора *T*. Матриця *X* і вектор *T* з числа вихідних параметрів може бути опущений. Якщо аргументом *x*0 відсутній, програма проводить обчислення з нульовими початковими умовами.

У всіх випадках матриця X *з числа* вихідних параметрів може бути опущена. Якщо аргументом *Tf* є вектор, програма сприймає його як вектор моментів часу, в які виводиться рішення.

Ефективність використання вказаних процедур пов'язана з тим, що вони здійснюються не за допомогою чисельної інтеграції, а в результаті рішення задачі про власні числа і власні вектори матриці *A* і подальшого використання аналітичних залежностей для розв'язку вказаної системи.

2.6 Індивідуальні завдання для розв’язку в середовищі MatLab

Завдання 1.Розрахувати заданий інтеграл з точністю

Варіант	Інтеграл	Точність
1.	$\int_1^3 \frac{x^2}{2x+3}dx$	0,001
2.	$\int_1^4 x^2\sqrt{x+2}dx$	0,1
3.	$\int_{0.2}^1 \frac{x}{\sin^2 3x}dx$	0,01
4.	$\int_2^3 x\cdot e^{0.8x}dx$	0,1
5.	$\int_1^2 \frac{1}{x\sqrt{x^2+0.25}}dx$	0,01

6.	$\int_1^2 \frac{x^2}{(2x+0.3)^2}dx$	0,001
7.	$\int_3^5 \frac{x}{0,5x+0.1}dx$	0,01
8.	$\int_0^1 x^2 \cdot \sin(x)dx$	0,001
9.	$\int_1^4 x \cdot 2^{3x}dx$	0,1
10.	$\int_0^2 \frac{x}{(x+3)^2}dx$	0,001
11.	$\int_{0.2}^1 \frac{\sqrt{4-x^2}}{x}dx$	0,01
12.	$\int_0^{\pi/4} x \cdot \sin(x)dx$	0,001
13.	$\int_0^1 2^{3x}dx$	0,01
14.	$\int_1^5 \frac{\ln^2 x}{x}dx$	0,01
15.	$\int_1^3 \frac{x^2}{2x+3}dx$	0,001
16.	$\int_1^4 x^2\sqrt{x+2}dx$	0,1
17.	$\int_{0.2}^1 \frac{x}{\sin^2 3x}dx$	0,01
18.	$\int_2^3 x \cdot e^{0.8x}dx$	0,1
19.	$\int_1^2 \frac{1}{x\sqrt{x^2+0.25}}dx$	0,01

Завдання 2 Розв'язок систем алгебраїчних рівнянь

Варіант	Система рівнянь	Початкові значення
1	2	3
1	$\left.\begin{array}{l} 3x^2 + xy - 1 = 0 \\ x - 3y + 5 = 0 \end{array}\right\}$	x_0=1 y_0=–2
2	$\left.\begin{array}{l} xy - 2y + y^2 = 0 \\ x + xy = 0 \end{array}\right\}$	x_0=0 y_0=0.2
3	$\left.\begin{array}{l} xy^2 + y - 7 = 0 \\ x^2 + y^2 - 12 = 0 \end{array}\right\}$	x_0=4.2 y_0=–1
4	$\left.\begin{array}{l} 7x + 9y - 15{,}6 = 0 \\ 3{,}5x^2 + 20{,}8y - 30 = 0 \end{array}\right\}$	x_0=2.1 y_0=–3
5	$\left.\begin{array}{l} 12x - xy - 8 = 0 \\ xy - 3x - 7 = 0 \end{array}\right\}$	x_0=6 y_0=2.5
6	$\left.\begin{array}{l} 3x^2 + xy - 1 = 0 \\ x - 3y + 5 = 0 \end{array}\right\}$	x_0=1 y_0=–2
7	$\left.\begin{array}{l} xy - 2y + y^2 = 0 \\ x + xy = 0 \end{array}\right\}$	x_0=0 y_0=0.2
8	$\left.\begin{array}{l} xy^2 + y - 7 = 0 \\ x^2 + y^2 - 12 = 0 \end{array}\right\}$	x_0=4.2 y_0=–1
9	$\left.\begin{array}{l} 7x + 9y - 15{,}6 = 0 \\ 3{,}5x^2 + 20{,}8y - 30 = 0 \end{array}\right\}$	x_0=2.1 y_0=–3
10	$\left.\begin{array}{l} 12x - xy - 8 = 0 \\ xy - 3x - 7 = 0 \end{array}\right\}$	x_0=6 y_0=2.5
11	$\left.\begin{array}{l} 3x^2 + xy - 1 = 0 \\ x - 3y + 5 = 0 \end{array}\right\}$	x_0=1 y_0=–2

12	$\left.\begin{array}{l} xy-2y+y^2=0 \\ x+xy=0 \end{array}\right\}$	$x_0=0$ $y_0=0.2$
13	$\left.\begin{array}{l} xy^2+y-7=0 \\ x^2+y^2-12=0 \end{array}\right\}$	$x_0=4.2$ $y_0=-1$
14	$\left.\begin{array}{l} 7x+9y-15,6=0 \\ 3,5x^2+20,8y-30=0 \end{array}\right\}$	$x_0=2.1$ $y_0=-3$
15	$\left.\begin{array}{l} 12x-xy-8=0 \\ xy-3x-7=0 \end{array}\right\}$	$x_0=6$ $y_0=2.5$
16	$\left.\begin{array}{l} 12x-xy-8=0 \\ xy-3x-7=0 \end{array}\right\}$	$x_0=6$ $y_0=2.5$
17	$\left.\begin{array}{l} 3x^2+xy-1=0 \\ x-3y+5=0 \end{array}\right\}$	$x_0=1$ $y_0=-2$
18	$\left.\begin{array}{l} xy-2y+y^2=0 \\ x+xy=0 \end{array}\right\}$	$x_0=0$ $y_0=0.2$
19	$\left.\begin{array}{l} xy^2+y-7=0 \\ x^2+y^2-12=0 \end{array}\right\}$	$x_0=4.2$ $y_0=-1$

Завдання 3 – точкова апроксимація функції методом найменших квадратів

Вар	Значення x						Значення y					
№	1	2	3	4	5	6	1	2	3	4	5	6
1	-7	-5	3	5	5,5	7	3281	1173	-300	-1327	-1765	-3580
2	0,3	0,7	1	1,2	1,8	2,1	2	-0,75	-2	-2,4	-1,85	-0,45
3	0,2	0,5	1	1,5	2	2,5	-0,76	0,125	3	7,62	14	22,12
4	0,4	0,7	0,9	1,3	1,8	2	6,6	9,3	11,1	14,7	19,2	21
5	0	0,5	1	1,6	2	2,3	3,75	5	6,15	7,8	8,8	9,6
6	0	0,3	0,8	1	1,5	2,2	2	3,125	6	7,5	12,2	20,7
7	-7	-5	4	4,5	6	7	-78	-14	76	98	195	286

8	-6,5	-5	-2	4	6	7	96	14	-22	-148	-382	-562
9	0	0,5	0,9	1,2	1,6	2	-2,5	-3,12	-3,08	-2,75	-1,86	-0,5
10	0,3	0,7	1,2	1,5	2,2	2,5	-6,4	-6,61	-4,2	-1,25	9,74	16,25
11	0	0,5	0,9	1,2	1,6	2	-2,5	-3,12	-3,08	-2,75	-1,86	-0,5
12	0,2	0,6	0,9	1,5	2	2,1	-1,3	0,1	1,2	3,33	5	5,33
13	0,3	0,5	0,8	1	1,2	2	-4	-4,5	-5,33	-5,9	-6,41	-8,49
14	0,5	1	1,3	1,7	2	2,2	1,5	0,65	0,2	-0,4	-0,9	-1,2
15	0	0,3	0,8	1	1,5	2,2	2	3,125	6	7,5	12,2	20,7
16	0,3	0,7	1	1,2	1,8	2,1	2	-0,75	-2	-2,4	-1,85	-0,45
17	0,2	0,5	1	1,5	2	2,5	-0,76	0,125	3	7,62	14	22,12
18	0	0,5	1	1,6	2	2,3	3,75	5	6,15	7,8	8,8	9,6
19	0,3	0,7	1,2	1,5	2,2	2,5	-6,4	-6,61	-4,2	-1,25	9,74	16,25

Перелік посилань

1. http://ablbook.com/index.php?newsid=5507 Макаров Е.Г., Самоучитель MathCad 14 від 18.12.2015 р.
2. Е. Любимов. / Mathcad. Теория и практика проведения электротехнических расчетов в среде Mathcad и Multisim. / Москва, 2014. - 400 с.: ил.
3. Кирьянов Д.В. / Mathcad 15/Mathcad Prime 1.0. - СПб.: БХВ-Петербург, 2012. - 432 с.: ил.
4. Методичні вказівки до проведення комп'ютерного практикуму з дисципліни "Методи комп'ютерного розрахунку обладнання целюлозно-паперового виробництва" для студентів напряму 6.050503 "Машинобудування" спеціальність 7.05050303 "Обладнання лісового комплексу": [Електронний ресурс]: / НТУУ „КПІ; уклад Я.В. Гробовенко, А.Р. Степанюк. – Київ: НТУУ „КПІ, 2015. – 49 с.
5. Очков В. Ф. О-94 Mathcad 14 для студентов, инженеров и конструкторов. — СПб.: БХВ-Петербург, 2007. — 368 с.: ил.
6. Дьяконов В. Mathcad 2001: учеб. курс. - СПБ.: Питер, 2001 - 624с.
7. Кудрявцев Е.М. MathCAD 2000 Pro. – М.: ДМК „Пресс", 2001. – 576с.
8. Кирьянов Д.В. Самоучитель MathCAD 2001 – СПб: БХВ-Петербург, 2001.
9. Потемкин В.Г. Система MATLAB.// М.: Изд. "Диалог-МИФИ", 1997.
10. Потемкин В.Г. MATLAB 5 для студентов.// М.: Изд. "Диалог-МИФИ", 1998.
11. Потемкин В.Г. Система инженерных и научных расчетов MATLAB 5.x.// в 2-х томах. М.: Изд. "Диалог-МИФИ", 1999.
12. Дьяконов В.П. Справочник по применению системы PC MATLAB.// М.: Физматлит, 1993.
13. Дьяконов В.П., Абраменкова И.В. MATLAB 5.0/5.3. Система символьной математики. // М.: Нолидж, 1999.

14. Мартынов Н.Н., Иванов А.П. MATLAB 5.X. Вычисления, визуализация, программирование.-М.: КУДИЦ-ОБРАЗ, 2000.

15. Конспект лекцій з кредитного модуля «Основи програмування для проектування нафтопереробного обладнання» для студентів напрямку підготовки 050503 Машинобудування: [Електронний ресурс]: / Укладач: Сачок Р.В., К.: НТУУ "КПІ", 2016 – 61 с., назва з екрану, доступ: http://ci.kpi.ua/METODA/osnovy_programuvannya_pno_konpekt.pdf

16. Коробова Н.Л., Загашвили Ю.В. Комплекс автоматизированного проектирования MATLAB-CTRL.// СПб, Изд. СПГААП, 1993.

17. Андрушевский Б.Р., Фрадков А.Л. Элементы математического моделирования в программных средах MATLAB и Scilab.// СПб, изд. Наука, 2001.

18. Гультяев А. MATLAB 5.2. Имитационное моделирование в среде Windows. // СПб.: КОРОНАпринт, 1999.

19. Медведев В.С. Потемкин В.Г. Control System Toolbox. MATLAB 5 для студентов.// М.: Изд. "Диалог-МИФИ", 1999.

20. Селезнев А.В. Эффективное программирование в среде MATLAB.// в кн. Тезисы докладов всероссийской научной конференции "Проектирование научных и инженерных приложений в среде MATLAB"// М.: ИПУ РАН. 2002. с.199-200.

21. PC MATLAB: User's Guide. MathWorks Inc., 1998.

22. Голуб Дж., Ван Лоун Ч. Матричные вычисления.// М.: Мир. 1999.

23. Воеводин В.В., Кузнецов Ю.А. Матрицы и вычисления.// М.: Наука. 1984.