Міністерство освіти і науки України
Сумський Державний університет

ЗНАЙОМСТВО З ПАКЕТОМ MATLAB

ЕЛЕМЕНТАРНІ ПРИЙОМИ РОБОТИ

Методичні вказівки
до виконання лабораторної роботи 8

2011

ЗМІСТ

ВСТУП 3

1 ОСНОВИ РОБОТИ З ПАКЕТОМ MATLAB 4

2 ДАНІ, ОПЕРАТОРИ ТА ФУНКЦІЇ 8

2.1 Дані та їх представлення 8
2.2 Оператори 9
2.3 Функції 12

3 КЕРУЮЧІ СТРУКТУРИ 15

4 ГРАФІКА ТА ІНТЕРФЕЙС КОРИСТУВАЧА 17

4.1 Побудова двовимірних графіків 17
4.2 Тривимірна графіка 17
4.3 Оформлення графіків 18
4.4 Об'єкти дескрипторної графіки 20
4.5 Команди для створення інтерфейсу користувача 22

5 ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ 25

6. ПОРЯДОК ВИКОНАННЯ І ПІДГОТОВКИ ЗВІТУ 29

ЛІТЕРАТУРА 24

ВСТУП

MATLAB — одна з найстарших і перевірених часом систем автоматизації математичних розрахунків, побудована на розширеному представленні і застосуванні матричних операцій. Це знайшло відображення в назві системи — MATrix LABoratory — матрична лабораторія. Однак синтаксис мови програмування системи продуманий настільки ретельно, що ця орієнтація майже не відчувається тими користувачами, яких не цікавлять безпосередньо матричні обчислення.

У цілому MATLAB — це унікальна колекція реалізацій сучасних чисельних методів комп'ютерної математики, створених за останні три десятки років. Вона увібрала в себе і досвід, і правила і методи математичних обчислень, накопичені за тисячі років розвитку математики. Систему з документацією до неї цілком можна розглядати як фундаментальний багатотомний електронний довідник по математичному забезпеченню ЕОМ — від масових персональних комп'ютерів до супер-ЕОМ.

Можливості MATLAB досить великі, а за швидкістю виконання задач система нерідко перевершує своїх конкурентів. Вона може застосовуватися для розрахунків практично в будь-якій області науки і техніки. У великому і постійно поповнюваному комплексі команд, функцій і прикладних програм (пакетів інструментів – toolbox), MATLAB 6 розглядається як один з toolbox усієї системи, що включає MATLAB 6, Simulink і інші пакети.

Важливими перевагами системи є її відкритість і розширюваність. Більшість команд і функцій системи реалізовані у вигляді текстових m-файлів (з розширенням .m) і файлів мовою Сі, причому усі файли доступні для модифікації. Користувачеві дана можливість створювати не тільки окремі файли, але і бібліотеки файлів для реалізації специфічних задач.

Мета даних методичних вказівок – дати огляд основних функцій і правил мови програмування пакета MATLAB, а також розглянути використання пакета MATLAB для рішення оптимізаційних задач, включаючи задачі навчання штучних нейроних мереж.

1 ОСНОВИ РОБОТИ З ПАКЕТОМ MATLAB

Після запуску MATLAB на екрані з'являється основне вікно системи MATLAB, показане на рис. 1.1. Звичайно це вікно розкрите не цілком і займає частину робочого столу. Ви можете розкрити вікно цілком, клацнувши на середній із трьох кнопок, розташованих наприкінці титульного (верхнього) рядка вікна.

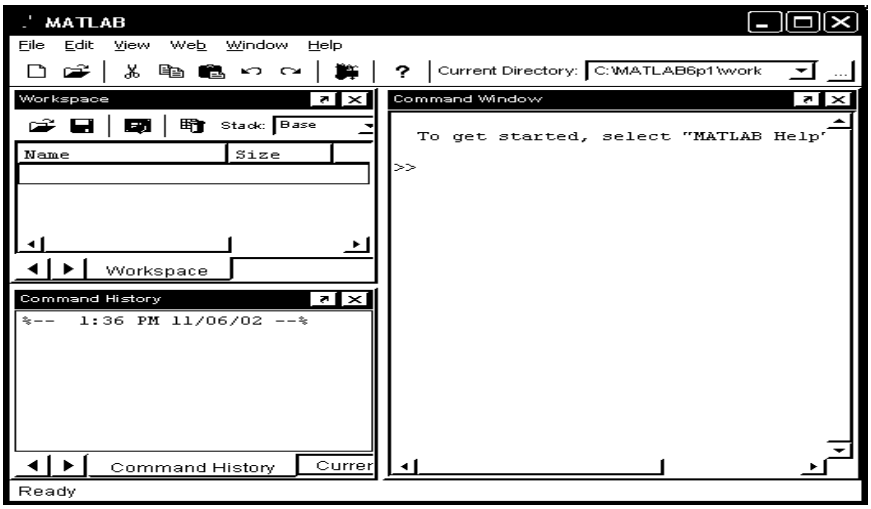

Рисунок 1.1 - Інтерфейс пакета MATLAB

Ліва кнопка звертає вікно в кнопку з ім'ям додатка, що поміщається в панель задач Windows, а права закриває вікно і припиняє роботу з MATLAB.
У MATLAB особливе значення мають файли двох типів — з розширеннями .mat і .m. Перші є бінарними файлами, у яких можуть зберігатися значення змінних. Другі являють собою текстові файли, що містять зовнішні програми, визначення команд і функцій системи. Саме до них відноситься велика частина команд і функцій, у тому числі ті, що задаються користувачем для рішення своїх специфічних задач. Нерідко зустрічаються і файли з розширенням .c (коди мовою Сі), файли з відкомпільованими кодами MATLAB з розширенням .tex і інші. Файли, що виконуються, мають розширення .exe.

Корисно знати, що на початку запуску автоматично виконується команда rnatlabrc, що виконує завантажувальний файл matlabrc.m і файл startup.m, якщо такий існує. Ці файли виконують початкове настроювання термінала системи і задають ряд її параметрів. Зокрема, можуть бути задані шляхи доступу до інших файлів, що необхідні для коректної роботи системи MATLAB. Таким чином, досвідчені користувачі можуть виконати настроювання системи під свій смак. Однак у більшості випадків особливої необхідності в цьому немає. Оскільки зазначені файли мають текстовий формат, їх легко переглянути за допомогою якого-небудь текстового редактора або за допомогою команди type у командному режимі роботи MATLAB.

Сеанс роботи з MATLAB прийнято іменувати **сесією** (session). Сесія, по суті, є поточним документом, що відбиває роботу користувача із системою MATLAB. У ній існують рядки введення, виведення і повідомлень про помилки.

Для очищення командного вікна використовують функцію clc.

Система MATLAB створена таким чином, що будь-які (часом досить складні) обчислення можна виконувати в режимі **прямих обчислень**, тобто без підготовки програми. Це перетворює

MATLAB у надзвичайно могутній калькулятор, що здатний робити не тільки звичайні для калькуляторів обчислення (наприклад, виконувати арифметичні операції й обчислювати елементарні функції), але й операції з векторами і матрицями, комплексними числами, рядами і поліномами. Можна майже миттєво задати і вивести графіки різних функцій — від простої синусоїди до складної тривимірної фігури.

Робота із системою в режимі прямих обчислень носить діалоговий характер і відбувається за правилом «поставив запитання, одержав відповідь». Користувач набирає на клавіатурі вираз, що обчислюється, редагує його (якщо потрібно) у командному рядку і завершує введення натисканням клавіші ENTER. Для указання введення вихідних даних використовується символ "»". Дані вводяться за допомогою найпростішого текстового редактора. Для блокування виведення результату обчислень деякого виразу після нього треба установити знак ";" (крапка з комою);

Якщо не зазначена змінна для значення результату обчислень, то MATLAB призначає таку змінну з ім'ям ans.

Знаком присвоювання є звичний математикам знак рівності "=".

MATLAB має інтерактивну **систему допомоги**, що реалізується в командному режимі за допомогою ряду команд. Однією з них є команда help, що виводить весь список папок, що містять m-файли з визначеннями операторів, функцій і інших об'єктів, властивих конкретної реалізації системи MATLAB.

Для одержання довідки про який-небудь конкретний об'єкт використовується команда: help ім'я.

Через достаток у системі MATLAB m-функцій, важливе значення має пошук m-функцій по ключових словах. Для цього служить команда: `lookfor Ключове слово`.

Одним з найефективніших методів знайомства зі складними математичними системами є ознайомлення з вбудованими прикладами їхнього застосування. Система MATLAB містить багато сотень таких прикладів — практично на кожен оператор або функцію. Найбільш повчальні приклади можна знайти в розділі demos, виконавши команду: help demos.

Оскільки MATLAB використовується для досить складних обчислень, важливе значення має наочність їхнього опису. Вона досягається, зокрема, за допомогою текстових коментарів. Текстові коментарі вводяться за допомогою символу %.

Змінні і визначення нових функцій у системі MATLAB зберігаються в особливій області пам'яті, що називається **робочою областю.**

MATLAB дозволяє зберігати значення змінних у вигляді бінарних файлів з розширенням .mat Для цього служить команда save, що може використовуватися в ряді форм:

save fname — записується робоча область усіх змінних у файлі бінарного формату з ім'ям fname.mat;

save fname X — записує тільки значення змінної X;

save fname X Y Z — записує значення змінних X, Y і Z.

Після цих параметрів можна вказати ключі, що уточнюють формат запису файлів:

-mat — двоїчний Мат-формат, що використовується за замовченням;

-ascii — ASCII-формат одиничної точності (8 цифр);

-ascii -double — ASCII-формат подвійної точності (16 цифр);

-ascii -double -tabs — формат з роздільником і мітками табуляції;

V4 — запис Мат-файлу у форматі версії MATLAB 4;

-append — додавання в існуючий Мат-файл.

Для завантаження робочої області раніше проведеної сесії (якщо вона була збережена) можна використовувати команду load:

load fname ... — завантаження раніше збережених у файлі fname.mat визначень зі специфікаціями на місці крапок, подібно описаним для команди save (включаючи ключ -mat для завантаження файлів з розширенням .mat звичайного бінарного формату, що використовується за замовченням);

1oad('fname'....) — завантаження файлу fname.mat у формі функції.

Якщо команда (або функція) load використовується в ході проведення сесії, то відбудеться заміна поточних значень змінних тими значеннями, що були збережені в мат-файлі, що зчитується.

У лівій частині вікна системи MATLAB 6.0 існує вікно спеціального **браузера робочої області** — Workspace Browser. Він служить для перегляду ресурсів робочої області пам'яті. Браузер дає наочну візуалізацію вмісту робочої області. Вікно браузера робочої області виконує й

інші важливі функції — дозволяє переглядати існуючі в пам'яті об'єкти, редагувати їхній вміст і знищувати об'єкти з пам'яті. Для виводу вмісту об'єкта досить виділити його ім'я за допомогою миші і клацнути на кнопці Open (Відкрити). Об'єкт можна відкрити і подвійним щигликом на його імені в списку. Відкриється вікно редагування масиву Array Editor.

Вікно редагування матриці дає зручний доступ для редагування будь-якого елемента матриці за правилами, прийнятим при роботі з електронними таблицями. Основне з них — швидкий доступ до будь-якого елемента матриці. Можна також змінювати тип значень елементів, вибираючи його зі списку, наданого меню Numeric format (Формат чисел). У вікні також виводяться дані про число рядків і стовпців матриці.

Слід зазначити, що перегляд робочої області можливий і в командному режимі, без звертання до браузера Workspace Browser.

Команда who виводить список визначених змінних, а команда whos — список змінних із вказівкою їхнього розміру й обсягу займаної пам'яті.

Для підготовки, редагування і налагодження m-файлів служить спеціальний **багатовіконний редактор**. Він виконаний як типовий додаток Windows. Редактор можна викликати командою edit з командного рядка або командою New => M-file з меню File. Після цього у вікні редактора можна створювати свій файл, користуватися засобами його налагодження і запуску. Перед запуском файлу його необхідно записати на диск, використовуючи команду File > Save as у меню редактора.

Після запису файлу на диск можна помітити, що команда Run у меню Tools (Інструменти) редактора стає активною (до запису файлу на диск вона пасивна) і дозволяє зробити запуск файлу. Запустивши команду Run, можна спостерігати виконання m-файлу.

Редактор також дозволяє встановлювати в тексті файлу спеціальні мітки, що називаються крапками переривання (breakpoints). При їхньому досягненні обчислення припиняються, і користувач може оцінити проміжні результати обчислень (наприклад, значення змінних), перевірити правильність виконання циклів і т.ін. Нарешті, редактор дозволяє записати файл у текстовому форматі й увічнити ваші праці у файловій системі MATLAB.

Для зручності роботи з редактором / відлагоджувачем тексту програми в ньому нумеруються в послідовному порядку. Редактор багатовіконний. Вікно кожної програми оформляється як вкладка.

Тут корисно відзначити, що m-файли, які створені редактором / відлагодчиком, поділяються на два класи:

- файли-сценарії, що не мають вхідних параметрів;
- файли-функції, що мають вхідні параметри.

Файл-сценарій або Script-файл не має списку вхідних параметрів і є прикладом простої процедури без параметрів. Він використовує глобальні змінні, тобто такі змінні, значення яких можуть бути змінені в будь-який момент сеансу роботи й у будь-якому місці програми.

Для запуску файлу-сценарію з командного рядка MATLAB досить вказати його ім'я в цьому рядку.

Файл-функція відрізняється від файлу-сценарію насамперед тим, що створена їм функція має вхідні параметри, список яких вказується в круглих дужках. Використовувані у файлі-функції змінні є локальними змінними, зміна значень яких у тілі функції ніяким образом не впливає на значення, що ті ж самі змінні можуть мати за межами функції.

Іншими словами, локальні змінні можуть мати ті ж імена (ідентифікатори), що і глобальні змінні (хоча правила культурного програмування не рекомендують змішувати імена локальних і глобальних змінних).

Повернення з функції відбувається після обробки всього тіла функції, тобто при досягненні кінця файлу функції. При використанні в тілі функції умовних операторів, циклів або перемикачів іноді виникає необхідність здійснити повернення функції раніш, ніж буде досягнутий кінець файлу. Для цього служить команда return. У будь-якому випадку, результатом, що повертається функцією, є значення вихідних параметрів, привласнені їм на момент повернення.

Структура М-файлу-функції:

```
function var=f_name(Список_параметров)
 %Основний коментар
 %Додатковий коментар
Тіло файлу з будь-якими виразами
var=вираз
```

Приведена форма файлу-функції характерна для функції з одним вихідним параметром. Якщо вихідних параметрів більше, то вони вказуються в квадратних дужках після слова function. При цьому структура модуля має такий вигляд:

```
function [varl,var2....]=f_name(Список_параметрів)
 %Основний коментар
 %Додатковий коментар
Тіло файлу з будь-якими виразами
var1=вираз
var2=вираз
```

Починаючи з версії 5.0 у функції системи MATLAB можна включати **підфункції**. Вони з'являються і записуються в тілі основних функцій і мають ідентичну їм конструкцію. Не слід плутати ці функції з внутрішніми функціями, вбудованими в ядро системи MATLAB. Підфункції визначені і діють локально, тобто тільки в межах m-файлу, що визначає основну функцію.

Отже, програмами в системі MATLAB є m-файлами текстового формату, що містять запис програм у вигляді програмних кодів. Для перегляду m-файлу варто використовувати будь-який текстовий редактор або команду: type Ім'я_M-файлу.

Змінні у файлах-сценаріях є глобальними, а у файлах-функціях — локальними. Часте застосування глобальних змінних у програмних модулях може приводити до побічних ефектів. Застосування локальних змінних усуває цю можливість і відповідає вимогам структурного програмування.

Однак передача даних з модуля в модуль у цьому випадку відбувається тільки через вхідні і вихідні параметри, що вимагає ретельного планування такої передачі. При створенні файлів-функцій часом бажане застосування глобальних змінних. Відповідальність за це повинен брати на себе програміст, що створює програмні модулі.

Команда global varl var2... дозволяє оголосити змінні модулі-функції **глобальними**. Таким чином, всередині функції можуть використовуватися і такі змінні, якщо це потрібно за умовою розв’язку вашої задачі.

Мова програмування системи MATLAB має наступні засоби: дані різного типу; константи і змінні; оператори, включаючи оператори математичних виразів; вбудовані команди і функції; функції користувача; керуючі структури; системні оператори і функції; засоби розширення мови.

Коди програм у системі MATLAB пишуться мовою високого рівня, яка досить зрозуміла для користувачів помірної кваліфікації в області програмування. Пакет MATLAB є типовим **інтерпретатором.** Це означає, що кожна інструкція програми розпізнається і відразу виконується, що полегшує забезпечення діалогового режиму спілкування із системою. Етап компіляції всіх інструкцій, тобто повної програми, відсутній. Висока швидкість виконання програм забезпечена наявністю свідомо відкомпільованого ядра, що зберігає в собі критичні до швидкості виконання інструкції, такі як базові математичні й інші функції, а також ретельним відпрацьовуванням системи контролю синтаксису програм у режимі інтерпретації.

Інтерпретація означає, що MATLAB не створює кінцевих програм, що виконуються. Вони існують лише у вигляді m-файлів. Для виконання програм необхідне середовище MATLAB. Однак для програм мовою MATLAB створені компілятори, що транслюють програми MATLAB у коди мов програмування C та C++. Це вирішує задачу створення програм, що виконуються, спочатку розроблених у середовищі MATLAB.

Слід особливо зазначити, що не всі інструкції MATLAB можуть компілюватися, так що перед компіляцією програми вимагають деякої доробки.

2 ДАНІ, ОПЕРАТОРИ ТА ФУНКЦІЇ

2.1 Дані та їх представлення

Число — найпростіший об'єкт мови MATLAB, що представляє кількісні дані. Числа можна вважати константами, імена яких збігаються з їх значеннями. Числа використовуються в загальноприйнятому представленні про них. Вони можуть бути цілими, дробовими, з фіксованою і плаваючою крапкою. Можливе представлення чисел у добре відомому науковому форматі з вказівкою мантиси і порядку числа.

Нижче приводяться приклади представлення чисел:

```
0 2 -3 2.301 0.00001 123.456e-24
```

Як неважко помітити, у мантисі чисел ціла частина відокремлюється від дробової не комою, а крапкою, як прийнято в більшості мов програмування. Для відділення порядку числа від мантиси використовується символ e. Знак «плюс» у чисел не проставляється, а знак «мінус» у числа називають *унарним мінусом.* Пробіли між символами в числах не допускаються.

Числа можуть бути *комплексними: z* =Re(x)+Im(x)*i. Такі числа містять дійсну Re(z) і мниму Im(z) частини. Мнима частина має множник *i* або *j, що* означає корінь квадратний з -1:

```
3i  2j  2+3i -3.141i  -123.456+2.7e-3i
```

Функція real (z) повертає дійсну частину комплексного числа, Re(z), а функція imag(z) — мниму, Im(z). Для одержання модуля комплексного числа використовується функція abs(z), а для обчислення фази — angle(Z).

У MATLAB не прийнято поділяти числа на цілі і дробові, короткій і довгі і т.ін., як це прийнято в більшості мов програмування, хоча задавати числа в таких формах можна. Взагалі ж операції над числами виконуються у форматі, що прийнято вважати форматом *з подвійною точністю.* Такий формат задовольняє переважній більшості вимог до чисельних розрахунків, але зовсім не підходить для символьних обчислень з довільною (абсолютною) точністю. Символьні обчислення MATLAB може виконувати за допомогою спеціального пакета розширення Symbolic Math Toolbox.

За замовченням MATLAB видає числові результати в *нормалізованій формі* з чотирма цифрами після десяткової крапки й однією до неї. Багатьох така форма представлення не завжди влаштовує. Тому при роботі з числовими даними можна задавати різні *формати* представлення чисел. Однак у будь-якому випадку всі обчислення проводяться з граничною, так званою *подвійною,* точністю.

Для встановлення формату представлення чисел використовується команда format name, де name — ім'я формату. Для числових даних name може бути наступним: short — коротке представлення у фіксованому форматі (5 знаків), shorte — коротке представлення в експонентному форматі (5 знаків мантиси і 3 знаки порядку), long — довге представлення у фіксованому форматі (15 знаків), longe — довге представлення в експонентному форматі (15 знаків мантиси і 3 знаки порядку), hex — представлення чисел у шістнадцятирічній формі; bank — представлення для грошових одиниць.

Константа — це попередньо визначене числове або символьне значення, представлене унікальним ім'ям. Числа (наприклад 1, -2 і 1.23) є безіменними *числовими константами.*

Інші види констант у MATLAB прийнято назвати *системними змінними,* оскільки, з одного боку, вони задаються системою при її завантаженні, а з іншого боку - можуть перевикликатися. Основні системні змінні, що застосовуються в системі MATLAB, наведені нижче:

i або j — мнима одиниця (корінь квадратний з -1);

pi - число π=3.1415926...;

eps — похибка операцій над числами з плаваючою крапкою, (2^{-52});

realmin — найменше число з плаваючою крапкою, (2^{-1022});

realmax — найбільше число з плаваючою крапкою, (2^{1023});

inf — значення машинної нескінченності;

ans — змінна, що зберігає результат останньої операції і його відображення на екрані дисплея;

Nan — покажчик на нечисловий характер даних (Not-a-Number).

Якщо в апострофи поміщений математичний вираз, то він *не обчислюється* і розглядається просто як ланцюжок символів. Так що '2+3' не буде повертати число 5. Однак за допомогою спеціальних функцій перетворення символьні вирази можуть бути перетворені у вирази, що обчислюються. Відповідні функції перетворення будуть розглянуті надалі.

Змінні — це об'єкти, що мають імена та здатні зберігати деякі, звичайно різні за значенням, дані. У залежності від цих даних змінні можуть бути числовими символьними, векторними або матричними.

У системі MATLAB можна задавати змінним визначені значення. Для цього використовується операція *присвоювання:* Ім'я_змінної = Вираз.

Типи змінних заздалегідь не декларуються. Вони визначаються виразом, значення якого присвоюється змінній. Так, якщо цей вираз — вектор або матриця, то змінна буде векторною або матричною.

Ім'я змінної (її ідентифікатор) може містити скільки завгодно символів, але запам'ятовується й ідентифікується тільки 31 початковий символ. Ім'я будь-якої змінної не повинно збігатися з іменами інших змінних, функцій і процедур системи, тобто воно повинно бути унікальним. Ім'я повинно починатися з букви, може містити букви, цифри і символ підкреслення _.

Неприпустимо включати в імена змінної пробіли і спеціальні знаки, наприклад +,.-, *, / і т.ін., оскільки в цьому випадку правильна інтерпретація виразів стає неможливою.

Змінні можуть бути звичайними й *індексованими,* тобто елементами векторів або матриць. Можуть використовуватися і *символьні* змінні, причому символьні значення беруть в апострофи, наприклад s='Demo'.

У пам'яті комп'ютера змінні займають визначене місце, що називаєтья *робочою областю* (workspace). Для очищення робочої області використовується функція clear у різних формах, наприклад:

clear — знищення визначень усіх змінних;

clear x — знищення визначення змінної x;

clear a, b, c — знищення визначень декількох змінних.

Знищена (стерта в робочій області) змінна стає невизначеною. Використовувати невизначені змінні не можна, і такі спроби будуть супроводжуватися видачею повідомлень про помилку.

MATLAB — система, що спеціально призначена для проведення складних обчислень з векторами, матрицями і масивами. При цьому вона за замовченням припускає, що кожна задана змінна — це вектор, матриця або масив. Усе визначається конкретним значенням змінної. Наприклад, якщо задано X=1, те це значить, що X-це вектор з єдиним елементом, що має значення 1. Якщо треба задати вектор із трьох елементів, то їх значення треба перелічити в квадратних дужках, розділяючи пробілами. Так, наприклад, присвоєння V=[l 2 3] задає вектор V, що має три елементи зі значеннями 1, 2 і 3.

Завдання матриці вимагає вказівки декількох рядків. Для розмежування рядків використовується знак ; (крапка з комою). Цей же знак наприкінці введення запобігає виведенню матриці або вектора (і взагалі результату будь-якої операції) на екран дисплея. Так, введення M=[1 2 3; 4 5 6; 7 8 9]; задає квадратну матрицю:

1 2 3
4 5 6
7 8 9

Можливе введення елементів матриць і векторів у вигляді арифметичних виразів, що містять будь-які доступні системі функції, наприклад: V= [2+2/(3+4) exp(5) sqrt(l0)]:

2.2857 148.4132 3.1623

Для вказівки окремого елемента вектора або матриці використовуються вирази вигляду V(1) або M(i,j).

2.2 Оператори

Оператор — це спеціальне позначення для визначеної операції над даними — *операндами.* Наприклад, найпростішими арифметичними операторами є знаки суми +, віднімання -, множення * і ділення /. Оператори використовуються разом з операндами. Наприклад, у виразі 2+3 знак + є оператором додавання, а числа 2 і 3 — операндами. Слід зазначити, що більшість операторів відносяться до матричних операцій, що може служити причиною серйозних непорозумінь. Наприклад, оператори множення * і ділення / обчислюють добуток і частку від ділення двох багатомірних масивів, векторів або матриць.

Є ряд спеціальних операторів, наприклад, оператор \ означає ділення *справа наліво,* а оператори .* та ./ означають відповідно *поелементне* множення і *поелементне* ділення масивів.

Повний список операторів можна одержати, використовуючи команду help ops.

Арифметичні оператори є найпоширенішими і найвідомішими. На відміну від більшості мов програмування в системі MATLAB практично всі оператори призначені для виконання операцій над матрицями.

Таблиця 2.1 – Арифметичні оператори

Функція	**Назва оператору**	**Синтаксис**
Plus	Плюс +	M1+M2
Uplus	Унарний плюс +	+M
Minus	Мінус	M1-M2
Uminus	Унарний мінус	-M
Mtimes	Матричне множення *	M1*M2
Times	Поелементне множення масивів .*	A1.*A2
Mpower	Возведення матриці в ступінь	M1^x
Power	Поелементне возведення масиву в ступінь	A1.^x
Mldivide	Обернене (справа наліво) ділення матриць \	M1\M2
Mrdivide	Ділення матриць зліва направо /	M1/M2
Ldivide	Поелементне ділення масивів справа наліво . \	A1.\A2
Rdivide	Поелементне ділення масивів зліва направо . /	A1./A2
Kron	Тензорне множення Кронекєра kron	kron(X,Y)

Оператори *відношення* служать для порівняння двох величин, векторів або матриць. Всі оператори відношення мають два операнда, наприклад x та y, і записуються, як показано далі.

= = Дорівнює ~ = Не дорівнює

< Менше ніж > Більше ніж

<= Менше або дорівнює >= Більше або дорівнює

Дані оператори виконують поелементне порівняння векторів або матриць однакового розміру і повертають значення 1 (True), якщо елементи ідентичні, і значення 0 (False) у противному випадку.

Логічні оператори і відповідні їм функції служать для реалізації поелементних логічних операцій над елементами однакових по розміру масивів.

And -логічне І (AND) &

Or -логічне АБО (OR) |

Not -логічне НІ (NOT) ~

Хог – Виключаюче АБО (EXCLUSIVE OR)

Any -вірно, якщо всі елементи вектора дорівнюють нулю

All -вірно, якщо всі елементи вектора не дорівнюють нулю.

До класу операторів у системі MATLAB відносяться також *спеціальні символи.* Вони призначені для створення найрізноманітніших об'єктів вхідної мови і мови програмування системи і додання їм різних форм. Нижче представлений опис повного набору спеціальних символів.

: (двокрапка) — формування підвекторів і підматриць з векторів і матриць. Оператор : — один з найбільш часто використовуваних операторів у системі MATLAB.

Оператор : використовує наступні правила для створення векторів:

j:k — те ж, що і [j.j+l,...,k];

j:k — порожній вектор, якщо j>k;

j:i:k — те ж, що і [j, j+i ,j+2i .....k];

j:i:k — порожній вектор, якщо i>0 і j>k або якщо i<0 і j<k, де 1, j і k — скалярні величини.

Нижче показано, як вибирати за допомогою оператора : рядка, стовпці й елементи з векторів, матриць і багатомірних масивів:

A(:, j) — це j-й стовпець з A;

A(i,:) — це і-й рядок з A;

A(:,:) — еквівалент двовимірного масиву. Для матриць це аналогічно A;

A(j:k) - це A(j), A(j+l),...,A(k);

A(:J:k)-це A(:.j),A(:,j+l),...,A(:,k);

A(:,: ,k) — це *k-та* сторінка тривимірного масиву А;

A(i.j.k.:) — вектор, виділений з чотирьохмірного масиву А. Вектор включає елементи A(1, j.k.l), A(i. j,k,2), A(i. j,k.3) і т.ін.;

A(:) — записує всі елементи масиву А в вигляді стовпця.

Символи ( ) (круглі дужки) використовуються для завдання порядку виконання операцій в арифметичних виразах, вказівки послідовності аргументів функції і вказівки індексів елемента вектора або матриці. Якщо X і V — вектори, то X(V) можна представити як [X(V(D), X(V(2)).....X(V(n))]. Елементи вектора V повинні бути цілими числами, щоб їх можна було використовувати як індекси елементів масиву X. Помилка генерується в тому випадку, якщо індекс елемента менше одиниці або більше, ніж size(X). Такий же принцип індексування дійсний і для матриць. Якщо вектор V має *m* компонентів, а вектор W — *n* компонентів, то A(V,W) буде матрицею розміру mxn, сформованою з елементів матриці А, індекси якої — елементи векторів V і W.

Символи [ ] (квадратні дужки) використовуються для формування векторів і матриць:

Для формування матриць і виконання ряду матричних операцій виникає необхідність видалення окремих стовпців і рядків матриці. Для цього використовуються порожні квадратні дужки [ ].

[6.9 9.64 sqrt(-l)] — вектор, що містить три елементи, розділених пробілами;

[6.9. 9.64. i] —такий же вектор;

[1+j 2-j 3] і [1 +j 2 -j 3] — різні вектори: перший містить три елементи, а другий – п'ять;

[11 12 13:21 22 23] — матриця розміру 2x3. Крапка з комою розділяє перший і другий рядки.

Ще кілька прикладів:

A = [ ] — зберігає порожню матрицю в А;

A(m, :) = [] — видаляє рядок m з матриці А;

A(: ,n) = [ ] — видаляє стовпець n з матриці А.

Символи { } (фігурні дужки) використовуються для формування масивів чарунок (рос. – ячеек). Наприклад, {magic(3) 6.9 'hello'} — масив чарунок із трьома елементами.

Символ . (десяткова крапка) використовується для відділення дробової частини чисел від цілої. Наприклад, 314/100, 3.14 і .314e1 — те саме число.

Крім того, символ крапки . використовується для виділення полів структур. Наприклад, A.(field) і A(i).field, де А — структура, означає виділення полів структури з ім'ям «field».

Нижче перераховане призначення інших спеціальних символів MATLAB:

.. (батьківський каталог) — перехід по дереву каталогів на один рівень вище;

... (продовження) — три або більш крапки наприкінці рядка вказують на продовження рядка;

; (крапка з комою) — використовується всередині круглих дужок для поділу рядків матриць, а також наприкінці операторів для заборони виведення на екран результату обчислень;

, (кома) — використовується для поділу індексів елементів матриці й аргументів функції, а також для поділу операторів мови MATLAB. При поділі операторів у рядку кома може замінятися на крапку з комою з метою заборони виведення на екран результату обчислень;

% (знак відсотка) — використовується для вказівки логічного кінця рядка. Текст, що знаходиться після знака відсотка, сприймається як коментар і ігнорується (на жаль, за винятком російськомовних коментарів, що нерідко ведуть до помилкових команд);

! (знак оклику) — є покажчиком введення команди операційної системи. Рядок, що іде за ним, сприймається як команда операційної системи;

= (знак рівності) — використовується для присвоювання значень в арифметичних виразах;

' (одиночні лапки, апостроф) — текст у лапках представляються як вектор символів з компонентами, що є ASCII-кодами символів. Лапки усередині рядка задаються двома лапками;

' (транспонування з комплексним сполученням) — транспонування матриць, наприклад A' — транспонована матриця А. Для комплексних матриць транспонування доповнюється комплексним сполученням. Рядки транспонованої матриці відповідають стовпцям вихідної матриці;

.' (транспонування) — транспонування масиву, наприклад A.' — транспонований масив А. Для комплексних масивів операція сполучення не виконується;

[.] — горизонтальна конкатенація. Так, [A.B] — горизонтальна конкатенація (об'єднання) матриць А та В. А й В повинні мати однакову кількість *рядків.* [A B] діє аналогічно. Горизонтальна конкатенація може бути застосована для будь-якого числа матриць у межах одних

дужок: [A,B,C]. Горизонтальна і вертикальна конкатенації можуть використовуватися одночасно: [A,B:C];

[:] — вертикальна конкатенація. Так, [A:B] — вертикальна конкатенація (об'єднання) матриць A та B. A й B повинні мати однакове число *стовпців.* Вертикальна конкатенація може бути застосована для будь-якого числа матриць у межах одних дужок: [A:B:C]. Горизонтальна і вертикальна конкатенації можуть використовуватися одночасно: [A;B,C];

(),{} — присвоювання підмасиву. Приведемо кілька прикладів:

A(I)=B — привласнює значення елементів масиву B елементам масиву A, що визначаються вектором індексів I. Масив B повинний мати таку ж розмірність, як і масив I, або може бути скаляром;

A(I,J)=B — привласнює значення масиву B елементам прямокутної підматриці A, що визначаються векторами індексів I і J. Масив B повинний мати LENGTH(I) рядків і LENGTH(J) стовпців;

A{1}=B, де A — масив чарунок і I — скаляр, поміщає копію масиву B в задану чарунку масиву A. Якщо I має більше одного елемента, то з'являється повідомлення про помилку.

2.3 Функції

Функція cputime — повертає час роботи процесора (у секундах), що був використаний системою MATLAB з моменту її запуску. Це число може вийти за рамки внутрішнього представлення, і тоді відлік часу починається заново.

У системі MATLAB визначені наступні **алгебраїчні й арифметичні функції**:

abs(X) — повертає абсолютну величину для кожного числового елемента вектора X. Якщо X містить комплексні числа, abs(X) обчислює модуль кожного числа.

exp(X) — повертає експоненту для кожного елемента X. Для комплексного числа $z = x + i*y$ функція exp(z) обчислює комплексну експоненту: exp(z)=exp(x)*(cos(y)+i*sin(y)).

log(X) — повертає натуральний логарифм елементів масиву X. Для комплексного або негативного z, де z = x + y*i, обчислюється комплексний логарифм у вигляді log(z) = log(abs(z)) + i*atan2(y,x). Функція логарифма обчислюється для кожного елемента масиву. Область визначення функції включає комплексні і негативні числа, що здатні привести до непередбачених результатів при некоректному використанні.

log2(X) — повертає логарифм зо основою 2 елементів масиву X;

log10(X) — повертає логарифм за основою 10 для кожного елемента X;

mod(X, Y) — повертає залишок від ділення X на Y (тобто, X - Y.*floor(X./Y)) для ненульового Y, і X у противному випадку;

sqrt(A) — повертає квадратний корінь кожного елемента масиву X. Для негативних і комплексних елементів X функція sqrt(X) обчислює комплексний результат.

У системі MATLAB визначені також **тригонометричні і зворотні тригонометричні функції** (наприклад: sin, asin, cos, acos, tan, atan і ін.). Вхідний масив допускає комплексні значення, при цьому всі кути у функціях задаються в радіанах.

Ряд особливих функцій служать для виконання операцій **округлення числових даних і аналізу їхнього знака.**

fix(A) — повертає масив A з елементами, округленими до найближчого до нуля цілого числа. Для комплексного A дійсні і мнимі частини округляються окремо;

floor(A) — повертає A з елементами, що є найближчими меншими цілими числами попереднього масиву. Для комплексного A дійсні і мнимі частини перетворяться окремо;

ceil(A) — повертає найближче більше або рівне A ціле число. Для комплексного A дійсні і мнимі частини округляються окремо;

rem(X,Y) — повертає X - fix(X./Y).*Y, де fix(X./Y) — ціла частина від частки X/Y;

round(X) — повертає округлені до найближчого цілого елементи масиву X. Для комплексного X дійсні і мнимі частини округляються окремо;

sign(X) — повертає масив Y тієї ж розмірності, що і X, де кожний з елементів Y дорівнює: 1, якщо відповідний елемент X більше 0; 0, якщо відповідний елемент X дорівнює 0; -1, якщо відповідний елемент X менше 0.

Для **обчислення розміру масиву** використовується функція size:

M = size(A, DIM) повертає розмір розмірності, зазначеної скаляром DIM, у вигляді вектора-рядка розміром 2. Для двомірного або одномірного масиву A size(A,l) повертає число рядків, a size (A, 2) - число стовпців;

Для N-мірних масивів A при n>2 size(A) повертає N-мірний вектор-рядок, що відбиває сторінкову організацію масиву, остання складового цього вектора дорівнює N. У векторі відсутні дані про одиничні розмірності.

[M1, M2 ,M3,..., MN] = size(A) повертає розмір перших N розмірностей масиву A;

D = size(A), для m*n матриці A повертає двохелементний вектор-рядок, у якому перша складова — число рядків m, а друга складова — число стовпців n;

[m.n] = size(A) повертає число рядків і стовпців у різних вихідних параметрах (вихідних аргументах у термінології MATLAB).

Функції обробки й аналізу даних:

max(A) — повертає найбільший елемент, якщо A — вектор; або повертає вектор-рядок, що містить максимальні елементи кожного стовпця, якщо A — матриця, у багатомірних масивах працює з першої не одиничної розмірності;

max(A,B) — повертає масив того ж розміру, що A та B, кожен елемент якого є максимальний з відповідних елементів цих масивів;

max(A,[ ],dim) — повертає найбільші елементи по стовпцях або по рядках матриці в залежності від значення скаляра dim.

min(A) — повертає мінімальний елемент, якщо A — вектор; або повертає вектор-рядок, що містить мінімальні елементи кожного стовпця, якщо A — матриця;

min(A,B) — повертає масив того ж розміру, що A та B, кожен елемент якого є мінімальний з відповідних елементів цих масивів;

min(A,[ ],dim) — повертає найменший елемент по стовпцях або по рядках матриці в залежності від значення скаляра dim.

Елементарна **статистична обробка даних** у масиві звичайно зводиться до підрахування їхнього середнього значення, медіани (серединного значення) і стандартного відхилення. Для цього в системі MATLAB визначені наступні функції:

mean(A) — повертає арифметичне середнє значення елементів масиву, якщо A — вектор; або повертає вектор-рядок, що містить середні значення елементів кожного стовпця, якщо A — матриця. Арифметичне середнє значення є сума елементів масиву, ділена на їхнє число;

median(A) — повертає медіану, якщо A — вектор; або вектор-рядок медіан для кожного стовпця, якщо A — матриця;

std(X) — повертає стандартне відхилення елементів масиву, що обчислюється по формулі якщо X — вектор. Якщо X — матриця, то std(X) повертає вектор-рядок, що містить стандартне відхилення елементів кожного стовпця;

Основні функції символьних даних.

В основі представлення символів у рядках лежить їхнє кодування за допомогою змінних таблиць кодів. Такі таблиці ставлять в однозначну відповідність кожному символові деякий код зі значенням від 0 до 255. Вектор, що містить рядок символів, у системі MATLAB задається в такий спосіб: S= 'Any Characters' - вектор, компонентами якого є числові коди, що відповідають символам.

До основних **строкових функцій** відносяться наступні:

char(X) — перетворить масив X позитивних цілих чисел (числових кодів від 0 до 65 535) у масив символів системи MATLAB;

char(C) — перетворить кожен елемент строкового масиву чарунок у ряди масиву символів, якщо рядки масиву чарунок різного розміру, до них наприкінці додаються пробіли (здійснюється набивання (padding) у термінах MATLAB) так щоб у кожному рядку масиву символів було однакове число символів;

char(Tl, T2, T3), де T — рядки, повертає масив символів, при цьому копії рядків T1, T2, T3 перетворяться в рядки масиву символів додаванням при необхідності пробілів наприкінці рядків масивів символів;

double(S) — перетворить символи рядка S у числові коди 0—65535 і повертає вектор з цими числовими кодами;

ischar(S) — повертає логічну одиницю, якщо S є символьною змінною, і логічний нуль у противному випадку;

deblank(str) — повертає рядок, отриманий з аргументу — рядка str з вилученими з його кінця пробілами;

deblank(c) — застосовує функцію deblank до кожного елемента строкового масиву чарунок c.

До **операцій над рядками** звичайно відносять пошук входжень одних рядків в інші, заміну регістрів символів, об'єднання рядків і т.ін. Наступні функції здійснюють операції над рядками:

findstr(str1,str2) — забезпечує пошук початкових індексів більш короткого рядка всередині більш довгого і повертає вектор цих індексів. Індекси вказують положення першого символу більш короткого рядка в більш довгому рядку;

lower('str') — повертає рядок символів str, у якій символи верхнього регістра переводяться в нижній регістр, а всі інші символи залишаються без змін;

upper('str') — повертає рядок символів str, у якому усі символи нижнього регістра переводяться у верхній регістр, а всі інші символи залишаються без змін;

strcat(sl,s2,s3,...) — виконує горизонтальне об'єднання відповідних рядків масивів символів s1, s2, s3 і т.ін. (причому пробіли наприкінці кожного ряду відкидаються) і повертає об'єднаний рядок результуючого масиву символів, пробіли додаються заново після аналізу рядків в отриманому масиві. Усі вхідні масиви повинні мати однакове число рядків (в окремому випадку повинні бути представлені у вигляді одного рядка символів), але якщо один із вхідних аргументів — не масив символів, а строковий масив чарунок, то кожний з інших вхідних аргументів може бути скаляром або будь-яким масивом тієї ж розмірності і того ж розміру. Якщо вхідний масив складається тільки із символів, то вихідний масив також буде масивом символів. Якщо кожний з вхідних масивів є строковим масивом чарунок, то функція strcat повертає строковий масив чарунок, сформований з об'єднаних відповідних елементів масивів si, s2, s3. при цьому кожний з елементів може бути скаляром тощо;

strvcat(t1,t2,t3....) — виконує вертикальне об'єднання рядків tl, t2, t3,.. у масив символів S аналогічно char(tl,t2.t3....);

strcmp('str1', 'str2') — повертає логічну одиницю, якщо два порівнювані рядки str1 і str2 ідентичні, і логічний нуль у противному випадку;

TF = strcmp(S,T) — повертає строковий масив чарунок TF, що містить одиниці для ідентичних елементів масивів S і T и нулі для всіх інших, причому якщо один з масивів — не масив символів, а строковий масив чарунок, то перед порівнянням з порівнюваних копій рядів масиву символів видаляються пробіли наприкінці рядків. Масиви S і T повинні мати однаковий розмір, або один з них може бути скалярною чарункою.

Перетворення символів і рядків:

int2str(X) — округляє елементи масиву X до цілих чисел і повертає масив символів, що містять символьні представлення округлених цілих чисел. Аргумент X може бути скаляром, вектором або матрицею;

mat2str(A) — перетворить матрицю A в єдиний рядок; якщо елемент матриці не скаляр, то він заміняється на [ ], при цьому враховуються 15 знаків після десяткової крапки;

mat2str(A, n) — перетворить матрицю A в рядок, використовуючи точність до n цифр після десяткової крапки. Функція eval(str) здійснює зворотне перетворення;

num2str(A) — виконує перетворення масиву A в рядок символів str з точністю до чотирьох десяткових розрядів і експонентним представленням, якщо потрібно;

num2str(A, precision) — виконує перетворення масиву A в рядок символів str з максимальною точністю, визначеної аргументом precision. Аргумент precision визначає число розрядів у вихідному рядку;

num2str(A, format) — виконує перетворення масиву чисел A, використовуючи заданий формат format. За замовченням приймається формат, що використовує чотири розряди після десяткової крапки для чисел з фіксованою або плаваючою крапкою;

str2double('str') — виконує перетворення чисельного рядка s, що представлений в ASCII-символах, у число з подвійною точністю. При цьому + і - можуть бути тільки на початку рядка;

str2num(s) — виконує перетворення чисельного масиву символів — матриці або рядка s, що представлений у ASCII-символах, у матрицю (масив розмірності 2).

За допомогою функції `eval 'строковий вираз'` рядок, що представляє математичний вираз, може бути обчислено.

3 КЕРУЮЧІ СТРУКТУРИ

Крім програм з лінійною структурою (інструкції, які виконуються строго одна за одною) існує безліч програм, структура яких нелінійна. При цьому частини програм можуть виконуватися в залежності від визначених умов, іноді з кінцевим числом повторень — циклів, іноді у вигляді циклів, що завершуються при виконанні заданої умови.

Практично будь-яка серйозна програма має нелінійну структуру. Для створення таких програм необхідні спеціальні керуючі структури. Вони існують в будь-якій мові програмування, і, зокрема, у MATLAB.

Умовний оператор if у загальному вигляді записується в такий спосіб:

```
if Умова
   Інструкції_1
elself Умова
   Інструкції_2
else
   Інструкції_3
end
```

Ця конструкція допускає кілька особливих варіантів.

У найпростішому випадку: `if Умова Інструкції end`

Якщо Умова повертає логічне значення 1 (тобто «істина»), виконуються Інструкції, що складають тіло структури if...end. При цьому оператор end указує на кінець переліку інструкцій. Інструкції в списку розділяються оператором "," (кома) або ";" (крапка з комою). Якщо Умова не виконується (дає логічне значення 0, «неправда»), то Інструкції також не виконуються.

Ще одна конструкція:

```
if Умова
    Інструкції_1
else
    Інструкції_2
end
```

виконує Інструкції_1, якщо виконується Умова, або Інструкції_2 - у противному випадку.

Умови записуються у вигляді:

`Вираз_1 Оператор_відношення Вираз_2,`

причому як Оператор_відношення використовуються наступні оператори: ==, <, >, <=, >= або ~=. Усі ці оператори являють собою пари символів без пробілів між ними.

Цикли типу for...end звичайно використовуються для організації обчислень із заданим числом повторюваних циклів. Конструкція такого циклу має такий вигляд:

```
for var=Вираз. Інструкція. .... Інструкція end
```

Вираз найчастіше записується у вигляді s:d:e, де s — початкове значення змінної циклу var, d — збільшення цієї змінної та e — кінцеве значення керуючої змінної, при досягненні якого цикл завершується. Можливий і запис у вигляді s:e (у цьому випадку d=l). Список виконуваних у циклі інструкцій завершується оператором end.

Оператор **continue** передає керування в наступну ітерацію циклу, пропускаючи оператори, що записані за ним, причому у вкладеному циклі він передає керування на наступну ітерацію основного циклу. Оператор **break** може використовуватися для дострокового переривання виконання циклу. Як тільки він зустрічається в програмі, цикл переривається.

Цикл типу while виконується доти, поки виконується Умова:

```
while Умова Інструкції end
```

Конструкція перемикача (або розгалуження) використовується для здійснення множинного вибору:

```
switch switch_Вираз
case case_Вираз
     Список_інструкцій
...
otherwise
     Список_інструкцій
end
```

Якщо вираз після заголовка switch має значення одного з виразів case_вираз, то виконується блок операторів case, у противному випадку — список інструкцій після оператора otherwise. При виконанні блоку case виконуються ті списки інструкцій, для яких case_Вираз збігається з switch_виразом. Зверніть увагу на те, що case_Вираз може бути числом, константою, змінною, вектором чарунок або навіть рядковою змінною. В останньому випадку оператор case істинний, якщо функція strcmp (значення, вираз) повертає логічне значення «істина».

Приклад: Розробити програму, яка генерує матрицю псевдовипадкових чисел розміром 5*5 та знаходить кількість елементів головної діагоналі матриці, менших за число 3. Виконати завдання з використанням матричних операцій та функцій, а також без них.

Програма:

% підрахунок кількості елементів головної діагоналі матриці, менших за число 3 без використання матричних функцій

```
x=5*randn(5,5)      % генерація матриці
    c=0;            % в змінній c зберігається потрібна кількість
for i=1:1:5
    if x(i,i)<3
        c=c+1;
    end;
end;
c                   % вивід значення c на екран
```

% підрахунок кількості елементів головної діагоналі матриці, менших за число 3 з використанням матричних функцій

```
    y=diag(x);          % y – масив чисел, що складається з елементів
                        головної діагоналі матриці x
    length(find(y<3)) % в результаті виконання find(y<3) отримується
                        масив чисел, менших за 3, а length() повертає
                        кількість елементів цього масиву, яку і
                        необхідно було підрахувати
```

4 ГРАФІКА ТА ІНТЕРФЕЙС КОРИСТУВАЧА

4.1 Побудова двовимірних графіків

Функції однієї змінної y(x) знаходять широке застосування в практиці математичних і інших розрахунків, а також у техніці комп'ютерного математичного моделювання. Для відображення таких функцій використовуються графіки в декартовій (прямокутній) системі координат. При цьому звичайно будуються дві осі і задаються координати x та y, що визначають вузлові крапки функції y(x) . Ці крапки з'єднуються одна з одною відрізками прямих, тобто при побудові графіка здійснюється лінійна інтерполяція для проміжних крапок. Оскільки MATLAB - матрична система, сукупність крапок y(x) задається векторами x і y однакового розміру.

Команда plot служить для побудови графіків функцій у декартовой системі координат. Ця команда має ряд параметрів, розглянутих нижче.

plot (x, y) — будує графік функції y(x) , координати крапок (x, y) які беруться з векторів однакового розміру y і x. Якщо x або y - матриця, то будується сімейство графіків за даними, що утримуються в колонках матриці.

plot(x,y,s) — аналогічна команді plot(x.y), але тип лінії графіка можна задавати за допомогою строкової константи s.

Значеннями константи s можуть бути наступні символи.

колір лінії: y -жовтий, m -фіолетовий, з - голубий, r - червоний, g - зелений, b - синій, w - білий, k - чорний;

тип крапки: . - крапка, o - окружність, x - хрест, + -плюс, * -зірочка, s -квадрат, d - ромб, v - трикутник (униз), ^ - трикутник (нагору), < - трикутник (уліво), > - трикутник (вправо), p - п'ятикутник, h - шестикутник;

тип лінії: - суцільна, : подвійний пунктир, -. штрих-пунктир, -і штрихова.

Таким чином, за допомогою строкової константи s можна змінювати колір лінії, представляти вузлові крапки різними оцінками (крапка, окружність, хрест, трикутник з різною орієнтацією вершини і т.ін.) і змінювати тип лінії графіка.

Наприклад, наступні команди побудують графік функції y=sin x на інтервалі $x \in [0;2\pi]$ лініями червоного кольору:

```
>> x=0:0.01:2.*pi; y=sin(x); plot(x,y,'r-');
```

4.2 Тривимірна графіка

Тривимірні поверхні звичайно описуються функцією двох змінних z(x, y). Специфіка побудови тривимірних графіків вимагає не просто завдання ряду значень x та y, тобто векторів x та y. Вона вимагає визначення для x і y двовимірних масивів. Для створення таких масивів служить функція meshgrid. В основному вона використовується разом з функціями побудови графіків тривимірних поверхонь.

Функція meshgrid записується в наступних формах:

[x,y] = meshgrid(x) — аналогічна [x,y] = meshgrid(x,x);

[x,y,z] = meshgrid(x,y,z) — повертає тривимірні масиви, що використовуються для обчислення функцій трьох змінних і побудови тривимірних графіків;

[x.y] = meshgrid(x,y) — перетворить область, задану векторами x та y, у масиви x і y, що можуть бути використані для обчислення функції двох змінних і побудови тривимірних графіків. Рядки вихідного масиву x є копіями вектора x; а стовпці *y* — копіями вектора *y*.

Команда plot3(...) є аналогом команди plot (...), але відноситься до функції двох змінних z(x, y). Вона будує аксонометричне зображення тривимірних поверхонь і представлена наступними формами:

plot3(x,y,z) — будує масив крапок, представлених векторами x, y та z, з'єднуючи їх відрізками прямих. Ця команда має обмежене застосування;

plot3(x,y,z), де x, y і z — три матриці однакового розміру, будує крапки з координатами x(i,:), y(i,:) і z(i,:) і з'єднує їх відрізками прямих.

Наприклад:

[x y z]=sphere(k); surf(x,y,z); генерує тривимірну сферу де к- кількість сегментів.

4.3 Оформлення графіків

Після того як графік уже побудований, MATLAB дозволяє виконати його форматування або оформлення в потрібному вигляді. Відповідні цьому засоби описані нижче.

title('string') — установка на двовимірних і тривимірних графіках титульного напису, заданого строковою константою 'string'.

Для установки написів біля осей x, y та z використовуються наступні команди:

xlabel('string') – напис біля осі x;

ylabel('string') – напис біля осі y;

zlabel('string') – напис біля осі z.

Часто виникає необхідність додавання тексту у визначене місце графіка, наприклад для позначення тієї або іншої кривої графіка. Для цього використовується команда text:

text(x, y, 'string') — додає в двовимірний графік текст, заданий строковою константою 'string', так що початок тексту розташований у крапці з координатами (x, y). Якщо x і y задані як одномірні масиви, то напис розташовується в усі позиції [x(i) ,y(i)];

text(x, y, z, 'string') — додає в тривимірний графік текст, заданий строковою константою 'string', так що початок тексту розташований у позиції, заданої координатами x, y і z.

Пояснення у вигляді відрізків ліній з довідковими написами, що розташоване всередині графіка або біля нього, називається легендою. Для створення легенди використовуються різні варіанти команди legend:

legend(stringl, string2, strings,...) — додає до поточного графіка легенду у вигляді рядків, зазначених у списку параметрів;

legend (h, stringl, string2, strings,...) — поміщає легенду на графік, що містить об'єкти з дескрипторами h, використовуючи задані рядки як мітки для відповідних дескрипторів;

legend(ax . ...) — поміщує легенду в осях (об'єкт класу axes) з дескриптором ax;

legend(m) — розміщає легенду, використовуючи дані зі строкової матриці m;

legend off — усуває раніше виведену легенду;

legend — перемальовує поточну легенду, якщо така існує;

legend(legendhandle) — перемальовує легенду, зазначену дескриптором legendhandle;

legend (....pos) — поміщає легенду в точно визначене місце, специфіковане параметром pos:

pos=0 — краще місце, що обирається автоматично;

pos=l — верхній правий кут;

pos=2 — верхній лівий кут;

pos=3 — нижній лівий кут;

pos=4 — нижній правий кут;

pos=-l — праворуч від графіка.

[legh, objh]=legend(...) — ця функція повертає дескриптор об'єкта для легенди (legh) і матрицю objh, що містить дескриптори об'єктів, з яких складається легенда.

Звичайно графіки виводяться в режимі автоматичного масштабування. Наступні команди класу axis змінюють цю ситуацію:

axis([xmin xmax ymin ymax]) — установка діапазонів координат по осях x та y для поточного двовимірного графіка;

axis([xmin xmax ymin ymax zmin zmax]) - установка діапазонів координат по осях x, y та z поточного тривимірного графіка;

axis auto — установка параметрів осей за замовченням;

axis manual — «заморожує» масштаб в поточному стані, щоб при використанні команди hold on наступні графіки використовували ті ж параметри осей;

axis tight — установлює діапазони координат по осях відповідно до діапазонів зміни даних;

axis ij — задає «матричну» прямокутну систему координат з початком координат у лівому верхньому куті, вісь i — вертикальна, що розмічається зверху вниз, вісь j — горизонтальна і розмічається зліва направо;

axis xy — установлює декартову систему координат з горизонтальною віссю x, що розмічається зліва направо, і вертикальною віссю y, що розмічається знизу вверх. Початок координат розміщається в нижньому лівому куті;

axis equal — включає масштаб з однаковою відстанню між мітками по осях x, y та z;

axis image — встановлює масштаб, при якому пікселі зображення стають квадратами;

axis square — установлює поточні осі у вигляді квадрата (або куба в тривимірному випадку) з однаковою відстанню між мітками й однаковою довжиною осей;

axis normal — відновлює масштаб, скасовуючи установки axis equal і axis square;

axis vis3d — «заморожує» пропорції осей для можливості повороту тривимірних об'єктів;

axis off — забирає з осей їхні позначення і маркери;

axis on — відновлює раніше введені позначення осей і маркери;

v=axis — повертає вектор-рядок, що містить коефіцієнти масштабування для поточного графіка. Якщо поточний графік двовимірний, то вектор має 4 компоненти, якщо тривимірний — 6 компонентів.

У математичній, фізичній і іншій літературі при побудові графіків на додаток до розмітки осей часто використовують масштабну сітку. Команди grid дозволяють задавати побудову сітки або скасовувати цю побудову:

grid on - додає сітку до поточного графіка;

grid off - відключає сітку;

grid - послідовно робить включення і відключення сітки.

У багатьох випадках бажана побудова багатьох накладених один на одного графіків у тому самому вікні. Для цього служить команда продовження графічних побудов hold. Вона використовується в наступних формах:

hold on — забезпечує продовження виведення графіків у поточне вікно, що дозволяє додавати наступні графіки до вже існуючих;

hold off — скасовує режим продовження графічних побудов;

hold — працює як перемикач, послідовно включаючи режим продовження графічних побудов і скасовуючи його.

Буває, що в одному вікні треба розташувати декілька координатних осей з різними графіками без накладення їх один на одного. Для цього використовуються команди subplot, яку треба застосувати перед побудовою графіків.

Для зміни масштабу двовимірних графіків використовуються команди класу zoom:

zoom — переключає стан режиму інтерактивної зміни масштабу для поточного графіка;

zoom (FACTOR) встановлює масштаб відповідно до коефіцієнта FACTOR;

zoom on — включає режим інтерактивної зміни масштабу для поточного графіка;

zoom off — виключає режим інтерактивної зміни масштабу для поточного графіка;

zoom out — забезпечує повний перегляд, тобто встановлює стандартний масштаб графіка;

zoom xon або zoom yon — включає режим зміни масштабу тільки по осі x або по осі y;

zoom reset — запам'ятовує поточний масштаб як масштаб за замовченням для даного графіка;

zoom(FIG,OPTION) — застосовується до графіка, заданого дескриптором FIG, при цьому OPTION може бути кожним з перерахованих вище аргументів.

Для освоєння функцій побудови та оформлення графіків розглянемо два приклади.

Приклад1: Побудувати декілька графіків на одній формі.

```
% створюємо масиви даних:
    x=1:10; y=x.*x;y1=20.*x;
% будуємо графік червоними суцільними лініями:
    plot(x,y,'r-');
hold on;   % включаємо режим накладення графіків
plot(x,y1,'b:');      % будуємо графік синім пунктиром
grid on;         % включаємо відображення координатної сітки
legend('y=x*x','y=20*x');        % додаємо легенду
title('graphs');            % додаємо підпис до графіка
% додаємо підписи до осей:
xlabel('x'); ylabel('y');
```

Приклад2: Побудувати декілька графіків в одному вікні (за допомогою функції subplot- побудова підграфіка):

```
x=0:0.01:2.*pi;
y=sin(x);
y1= 3*sin(x)-x;
y2= sin(5*x)-10;
y3= sin(5*x)+ cos(x);
subplot(2,2,1); plot(x,y,'r-'); legend(' y=sin(x)'); grid on;
subplot(2,2,2); plot(x,y1,'b-'); legend('3*sin(x)-x '); grid on;

subplot(2,2,3); plot(x,y2,'g-'); legend(' sin(5*x)-10'); grid on;

subplot(2,2,4); plot(x,y3,'y-'); legend(' sin(5*x)+ cos(x) '); grid on;
```

Приклад3: Згенерувати набір значеннь, який занести до масиву x, обчислити засобами пакету MATLAB значення функції y(x), та побудувати її графік

$$y = \begin{cases} x^5, \ 1 \le x < 4, \text{або } 8 < x < 16, \\ a\cos\left|5 - x^7\right|, \ \text{інакше,} \end{cases} \quad \text{при } x \in [-10;20], \ \Delta x = 0{,}33;$$

Програма (варіант1):

```
x=-10:0.33:20;              % генерація масиву x
    y=x;
    k=0;
for i=-10:0.33:20           % обчислення значень функції y(x)
    k=k+1;
        if (((i>=1)&(i<4))|((i>8)&(i<16)))
       y(k)=(x(k)).^5;
      else
       y(k)=acos(5-(x(k)).^7);
    end;
end;
plot(x,y)                   % побудова графіка
    xlabel('x')                     % підпис по осі x
    ylabel('y')                     % підпис по осі y
title('график')             % підписуємо заголовок графіка
```

Програма (варіант2):

```
x=-10:0.33:20; % генерація масиву x
y=-10:0.33:20; %  початкові значення масиву y (можна y=x)
[m,n] = size(x); % визнач. кількості рядків(m, в данному випадку 1)  та  елементів в рядку
(n)
for k=1:n                   %  цикл обчислення значень функції y(x)

         if (((x(k)>=1)&( x(k)<4))|(( x(k)>8)&( x(k)<16)))
       y(k)=(x(k)).^5;
      else
       y(k)= acos(5-(x(k)).^7);
    end;
end;
```

```
plot(x,y)              % побудова графіка
    xlabel('x')                % підпис по осі x
    ylabel('y')                % підпис по осі y
title('график')          % підписуємо заголовок графіка
```

Приклад 4.1: Побудувати тривимірний графік функції z=sin(x)/y , якщо x та a змінюються від -8 до 8 з кроком 0,5 .

Програма:

```
[X, Y] = meshgrid(-8:.5:8 );
Z = sin(X)./Y
mesh(X,Y,Z)
      xlabel('x')                % підпис по осі x
      ylabel('y')                % підпис по осі y
title('поверхня Z = sin(X)./Y ')      % підписуємо заголовок графіка
```

Приклад 4.2: Побудувати тривимірний графік функції z=sin(x)/y , якщо x змінюються від -8 до 8 з кроком 0,5 і y змінюються від -20 до 20 з кроком 1

Зміни до програми 4.1:

```
[X, Y] = meshgrid(-8:.5:8, -20:2:20 );
```

4.4 Об'єкти дескрипторної графіки

Як уже відзначалося, графічні засоби MATLAB базуються на **низькорівневій графіці**, що називається **дескрипторною** (описовою), або handle графікою. Власне кажучи, ця графіка забезпечує об’єктно-орієнтоване програмування як усіх розглянутих вище графічних команд, так і користувальницького інтерфейсу.

Графічний редактор дескрипторної графіки Property Editor у MATLAB 6 є основним редактором графіки. Хоча звичайний користувач може навіть не знати про існування дескрипторної графіки через те, що слово «дескрипторна» у скорочену назву графічного редактора не входить, але все-таки треба враховувати, що саме дескрипторна графіка дає нові, часом унікальні можливості створення користувальницьких графічних програм MATLAB 6, не говорячи вже про те, що вона допомагає зрозуміти, яким образом реалізовані графічні засоби системи.

Центральним поняттям дескрипторної графіки є **графічний об'єкт.** Існують наступні **типи** таких об'єктів:

root (корінь) — первинний об'єкт, що відповідає екранові комп'ютера;

figure (малюнок) — об'єкт створення графічного вікна;

uicontrol (елемент керування, визначений користувачем) — об'єкт створення елемента користувальницького інтерфейсу;

axes (осі) — об'єкт, що задає область розташування графіка у вікні об'єкта figure;

uimenu (визначене користувачем меню) — об'єкт створення меню;

uicontextmenu (визначене користувачем контекстне меню) - об'єкт створення контекстного меню;

image (образ) — об'єкт створення растрової графіки;

line (лінія) — об'єкт створення лінії;

patch (латка) — об'єкт створення зафарбованих фігур;

rectangle (прямокутник) - об'єкт створення зафарбованих прямокутників;

surface (поверхня) — об'єкт створення поверхні;

text (текст) — об'єкт створення текстових написів;

light (світло) — об'єкт створення ефектів освітленості.

Об'єкти часом взаємозалежні і можуть звертатися один до іншого для одержання того або іншого графічного ефекту.

Для **створення графічних вікон і керування ними** використовуються наступні функції:

figure — відкрити чисте графічне вікно;

gcf — одержати дескриптор графічного вікна figure;

elf — очистити графічне вікно;

shg — показати раніше згорнуте графічне вікно;

close (закрити) — закрити графічне вікно;

refresh (обновити) — оновити графічне вікно.

До графічних об'єктів застосовується **ряд операцій:**

set — установка властивостей (параметрів) графічного об'єкта;

get — виведення властивостей графічного об'єкта;

reset — відновити властивості графічного об'єкта за замовченням;

delete — видалити створений графічний об'єкт;

gco — повертає дескриптор поточного графічного об'єкта;

gcbo — повертає дескриптор об'єкта, функція якого в даний момент виконується;

gcbf — повертає дескриптор вікна, що містить об'єкт, функція якого в даний момент виконується;

drawnow — виконати чергу затриманих графічних команд;

findobj — знайти об'єкти з заданими властивостями;

copyobj — скопіювати об'єкт і породжені ним об'єкти.

Крім того, існують три **утиліти**, зв'язані з операціями над об'єктами:

closereq — закрити вікно по запиту;

ishandle — перевірити дескриптор на істинність;

newplot — відновити властивості об'єкта, змінені nextPlot.

4.5 Команди для створення інтерфейсу користувача

У пакеті MATLAB дескрипторна графіка дозволяє конструювати деталі користувальницького інтерфейсу. Повний список команд і функцій для проектування користувальницького інтерфейсу (**GUI**) можна одержати, виконавши команду help uitools.

Нижче перераховані основні функції GUI:

ulcontrol — створення керуючого елемента;

ulmenu — створення користувальницького меню;

ginput — графічне введення за допомогою миші.

dragrect — створення прямокутника, що виділяється, за допомогою миші;

rbbox — розтягування прямокутника мишею;

selectmoveresize — інтерактивне виділення, переміщення і копіювання об'єктів за допомогою миші;

waitforbuttonpress — чекання натискання клавіші клавіатури або кнопки миші у вікні;

waltfor — припинення виконання програми в очікуванні знищення заданого графічного об'єкта або зміни його властивостей;

uiwait — припинення виконання програми в очікуванні виклику функції uiresume або закриття заданого графічного вікна;

inresume — відновити виконання після блокування;

uisuspend — припинення інтерактивного стану фігури;

uirestore — поновлення інтерактивного стану фігури;

guide — створення GUI;

align — вирівняти положення об'єктів інтерфейсу;

cbedit — зміна повторного виклику об'єктів;

menuedit — зміна меню;

propedit — зміна властивостей об'єктів;

dialog — створення діалогового вікна;

axlimdlg — обмеження розмірів діалогового вікна;

errordlg — створення вікна з повідомленням про помилку;

helpdlg — створення довідкового вікна;

inputdlg — створення вікна діалогу введення;

listdlg — створення вікна діалогу для вибору варіантів параметра зі списку;

menu — створення меню діалогового введення;

msgbox — створення вікна повідомлень;
questdlg — створення вікна запиту;
warndlg — створення вікна попередження;
uigetfile — створення стандартного вікна відкриття файлів;
uiputfile — створення стандартного вікна запису файлів;
uisetcolor — створення вікна вибору кольору;
uisetfont — створення вікна вибору шрифту;
pagedlg — створення діалогового вікна параметрів сторінки;
printdlg — створення діалогового вікна друку;
waitbar — створення вікна з індикатором прогресу.
makemenu — створити структури меню;
menubar — установлювати типові властивості для об'єкта MenuBar;
umtoggle — змінювати статус параметра "checked" для об'єкта uimenu;
winmenu — створити підменю для меню Window;
btngroup — створити кнопку панелі інструментів;
btnstate — запросити статус кнопки;
btnpress — керування кнопкою;
btndown — натиснути кнопку;
btnup — відпустити кнопку;
clruprop — видалити властивість об'єкта;
getuprop — запросити властивість об'єкта;
setuprop — установити властивість об'єкта;
all child — запросити всі породжені об'єкти;
findall — знайти всі об'єкти;
hidegui — сховати/відкрити об'єкти GUI;
edtext — інтерактивне редагування об'єктів text;
getstatus — запросити властивості рядка об'єкта figure;
setstatus — установити властивості рядка об'єкта figure;
popupstr — запросити властивості рядка випадаючого меню;
remapflg — змінити положення об'єкта figure;
setptr — установити покажчик на об'єкт figure;
getptr — одержати покажчик на об'єкт figure;
overobj — запросити дескриптор об'єкта, над яким знаходиться курсор миші.

5. ЗАВДАННЯ ДЛЯ ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ

5.1.Завдання 1.

Задача 1. 1. Побудувати графік функції згідно варіанту.

Задача 1.2.Побудувати три графіка функцій (N, N-1,N+1 де N- номер варіанта) в одному вікні.

Задача 1.3.Побудувати три графіка функцій (N, N-1, N+1 де N- номер варіанта) на одному рисунку, але кожен в окремому вікні(встовпчик і врядочок) .

Варіанти завдань 1

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***h***
01	$z = 0.9x + 5\sin x + a + 3b$	2.5	*2*	*-10*	*10*	*0,01*
02	$p = -x + \sin 7x + a$	10	-	0	10	0,01
03	$z = 3x + \sin 5x$			0	10	0,01
04	$z = 0.33x + \sin x$			-10	10	0,01
05	$z = 5x + 0{,}5\sin x + a$	10	-	-20	20	0,01
06	$z = 0.33x^2 + 0.77a\cos x^2$	3,4		0	10	0,01
07	$f = 0.75\lvert x - a\rvert + 10a$	5	-	0	10	0,01
08	$f = 0.77x + \sin x - b$		5,5	0	10	0,01
09	$q = ab + \lvert x\rvert - \sqrt{4a}$	2	3,6	-5	5	0,01
10	$p = 0.33a + be^x$	5	2,8	0	10	0,01
11	$z = 0.33x + 0.77a\cos x^2$	10	-	0	10	0,01
12	$p = -x + a$ -b	1	4	0	10	0,01
13	$p = \cos x + 5e^x$	-	-	0	10	0,01

Продовження таблиці.

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***h***
14	$z = ax + \cos x^2 - b$	0,33	2,54	0	10	0,01
15	$p = x - be^x + a$	7,75	5	0	10	0,01
16	$f = \lvert x - 5\rvert - ab$	0,33	15	0	10	0,01
17	$z = ax + \sin 10x$	5	-	0	10	0,01
18	$u = x - \cos x^2$			0	10	0,01
19	$s = 0.25\sin x^2 + ax + b$	1,77	5,2	0	10	0,01
20	$p = \sin 25x - 0{,}33$	-	-		10	0,01

21	$f = 0.75\|x-3\| + 10ab$	0,75	7,1	0	10	0,01

5.2 Завдання 2.

Задача 2. 1. Побудувати графік функції згідно варіанту.

Задача 2.2.Побудувати (різними кольорами) графіки функцій (N, N-1,N+1 де N- номер варіанта) в одному вікні. Передбачити необхідні оформлення графікив.

Задача 2.3.Побудувати три графіка функцій (N, N-1, N+1 де N- номер варіанта) на одному рисунку, але кожен в окремому вікні (встовпчик і врядочок).

Варіанти до завдання 2

Таблиця 5.1.

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***h***
01	$y = \begin{cases} a^2+b^2 & \text{при } x < ab \\ \dfrac{ax+b}{a+56} & \text{при } x \ge ab \end{cases}$	2.5	2	-10	10	0.01
02	$z = \begin{cases} \dfrac{a+100}{b+1} & \text{при } x < ab \\ x + \dfrac{a+5}{b+8} & \text{при } x \ge ab \end{cases}$	1	2.5	1.2	16.7	0.01
03	$\mathrm{f} = \begin{cases} \mathrm{abx}+100 & \text{при } \mathrm{x} \le 0 \\ \dfrac{\mathrm{x}+5\mathrm{a}^2}{\mathrm{b}+100} & \text{при } \mathrm{x} > 0 \end{cases}$	0	1.5	-8	17	0.01
4	$\mathrm{p} = \begin{cases} \dfrac{\mathrm{x}^2+1}{5\mathrm{x}+10} & \text{при } \mathrm{x} > 0 \\ \mathrm{a} & \text{при } \mathrm{x} \le 0 \end{cases}$	-1	-	-10	10	0.01
05	$s = \begin{cases} \|x+100\| & \text{при } \mathrm{x} < 0 \\ \dfrac{x^2+\sin x}{abx+5} & \text{при } x \ge 0 \end{cases}$	1	2.4	-20	20	0.01
06	$r = \begin{cases} \dfrac{x+x^2+1}{100x+1.5} & \text{при } \mathrm{x} > 0 \\ 1993a & \text{при } x \le 0 \end{cases}$	10	-	-14	14	0.01
07	$p = \begin{cases} 0.33\left(x^2+x+1\right) & \text{при } \mathrm{x} < 0 \\ \|10-x\| + a & \text{при } x \ge 0 \end{cases}$	100	-	-6.8	16.2	0.01

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***h***
08	$s=\begin{cases}\dfrac{x+1}{1000+b} & \text{при } x\le 0\\ 100x+5 & \text{при } x>0\end{cases}$	-	1.74	-15	15	0.01
09	$z=\begin{cases}b^3 & \text{при } x\le -10\\ \lvert x\rvert+0{,}33x+\sin^2 x & \text{при } x>-10\end{cases}$	-	-1.2	-25	15	0.01
10	$f=\begin{cases}\lvert 1-x^2\rvert & \text{при } x<0\\ \dfrac{a+b}{a-b}x & \text{при } x\ge 0\end{cases}$	100	24.8	-19	49	0.01
11	$z=\begin{cases}0.33x^2+0.77x^2 & \text{при } x\le a\\ \dfrac{x+1}{a+b} & \text{при } x>a\end{cases}$	10	4.6	0	25	0.01
12	$p=\begin{cases}\lvert 1+x\rvert & \text{при } x<0\\ \dfrac{100x+1}{a+b} & \text{при } x\ge 0\end{cases}$	1	16.8	-12	12	0.01
13	$f=\begin{cases}a+\lvert x\rvert & \text{при } x\le 0\\ 0.77x^3+\sin^2 x & \text{при } x>0\end{cases}$	5	-	-17	7	0.01

Продовження таблиці.

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***H***
14	$q=\begin{cases}ab+\lvert x\rvert & \text{при } x\le 0\\ x+5 & \text{при } x>0\end{cases}$	0.33	2.54	-16	16	0.01
15	$p=\begin{cases}0.25\lvert x\rvert & \text{при } x<0\\ 0.33a+ax & \text{при } x\ge 0\end{cases}$	0.75	-	-8	28	0.01
16	$q=\begin{cases}0.77\sqrt[3]{\lvert x\rvert}+\dfrac{x+1}{a+b} & \text{при } x\le 0\\ abx & \text{при } x>0\end{cases}$	0.33	15	-6	18	0.01
17	$y=\begin{cases}\lvert x+1\rvert & \text{при } x<0\\ \dfrac{x+1}{a+b}+5 & \text{при } x\ge 0\end{cases}$	95	-9.3	-15	15	0.01
18	$z=\begin{cases}\sqrt[3]{\lvert x+1\rvert}+a & \text{при } x<0\\ 0.73ax+b & \text{при } x\ge 0\end{cases}$	0.5	8.4	-10	10	0.01
19	$p=\begin{cases}\sin x^2+1 & \text{при } x\le a\\ \dfrac{x+1}{ab+100} & \text{при } x>a\end{cases}$	1.77	5.2	-14	14	0.01
20	$y=\begin{cases}\dfrac{x-100}{a+b} & \text{при } x<a\\ \dfrac{x+1}{ab+x} & \text{при } x\ge a\end{cases}$	1	6.9	-8	32	0.01

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***H***
21	$f=\begin{cases}\lvert x-100\rvert-ab & \text{при } x<ab\\ a^3+a^2+x & \text{при } x\ge ab\end{cases}$	15	1.7	10	40	0.01
22	$p=\begin{cases}\dfrac{5+\lvert a+x\rvert}{a+b} & \text{якщо } x<a\\ 0.7x^2+0.3x & \text{якщо } x\ge a\end{cases}$	-1	5.2	-9	9	0.01
23	$q=\begin{cases}\lvert x-a\rvert & \text{при } x<a\\ \dfrac{x+a}{a+b}+0.75 & \text{при } x\ge a\end{cases}$	0.75	7.1	-2	3	0.01
24	$s=\begin{cases}0.33x^2+0.25x & \text{при } x<0\\ 0.25\sin x^2 & \text{при } x\ge 0\end{cases}$	-	-	-3	4	0.01
25	$p=\begin{cases}0.75\lvert x\rvert+0.5a & \text{при } x<ab\\ \dfrac{x+100}{0.33ab+7.5} & \text{при } x\ge ab\end{cases}$	17.5	0.5	2	22	0.01
26	$s=\begin{cases}\dfrac{x^2+ab}{a+b} & \text{при } x\le 0\\ \ln a+\ln b & \text{при } x>0\end{cases}$	2	7	-6	6	0.01
27	$z=\begin{cases}0.01x^3+\lvert x\rvert & \text{при } x<a\\ \dfrac{5a+8bx}{0.33ab+1}+a^2 & \text{при } x\ge a\end{cases}$	12	0.5	5	20	0.01

Продовження таблиці.

Варіант	В и д ф у н к ц і ї	Вхідні дані				
		a	***b***	$x_{поч}$	$x_{кін}$	***H***
28	$f=\begin{cases}0.75\lvert x\rvert+10a & \text{при } x\le a\\ 0.33x & \text{при } x>a\end{cases}$	1	-	-7	7	0.01
29	$s=\begin{cases}0.1x+0.25ab & \text{при } x>a+b\\ \dfrac{0.7a+0.3b}{5a+b}+b & \text{при } x\le a+b\end{cases}$	-9	3.6	-10	10	0.01
30	$y=\begin{cases}\lvert x-a-b\rvert+\sqrt[3]{a} & \text{при } x\le 0\\ \dfrac{a+1}{b+100}+x & \text{при } x>0\end{cases}$	16	6.7	-8	8	0.01

5.3 Завдання 3.

Задача 3.1. Побудувати тривимірний графік функції 2-х змінних згідно варіанту.

Задача 3.2.Побудувати 4 графіка функцій (N, N-1, N+1, N+2 де N- номер варіанта) на одному рисунку, але кожен в окремому вікні (матрицею 2*2)

Варіанти до завдання 3

Варіант	В и д ф у н к ц і ї
01	$z = 5\sin x + a$
02	$p = -x + \sin 7x + a$
03	$z = ax + \sin 5x$
04	$z = 0.33ax + \sin x$
05	$z = a(5x + 100\sin x)$
06	$z = 0.33x^2 + 25a\cos x^2$
07	$p = \cos x + ae^x$
08	$u = ax - 100\cos x^2$
09	$r = x - a\cos x^2$
10	$p = 0.33ae^x$
11	$z = 0.33x + 0.77a\cos x^2$
12	$z = 3\sin x + 7a$
13	$p = \cos x + 33ae^x$
14	$z = ax + \cos x^2 - b$
15	$p = 100\cos x - 0{,}01ae^x$
16	$z = x^2 - 10a$
17	$z = ax + \sin 10x$
18	$u = ax - 15\cos x^2$

6. ПОРЯДОК ВИКОНАННЯ, ПІДГОТОВКИ ЗВІТУ І ЗАХИСТУ РОБОТИ

- Ознайомитись з теоретичними положеннями.
- Виконати і розібрати контрольні приклади.
- Вирішити всі задачі за вказаними варіантами.

- Програми зберегти в М-файлах, давши їм зрозумілі імена, наприклад прізвище(4зн.)_lr1_z3_1
- Підготувати презентацію
- Скласти і роздрукувати звіт за схемою:

- Титульний лист;
- Звіт про вирішення задачі (всього 8 задач) за схемою:
- Умова задачі (повна)
- Текст програми (з коментарем)
- Результат

ДЛЯ ЗАХИСТУ:
1.пред'явити:

- презентацію
- роздрукований звіт
- по кожній задачі – М-файл і результат його роботи

2.відповісти на запитання викладача.

ЛІТЕРАТУРА

1. Дьяконов В. Matlab 6: Учебный курс. - СПб.; Питер, 2002. - 592с.
2. Дьяконов В., Круглов В. Математические пакет расширения Matlab. Специальный справочник.- СПб.: Питер, 2001. - 480с.
3. Лавров К.Н., Циплякова Т.П. Финансовая аналитика. Matlab 6 /Под общ. ред. В.Г. Потёмкина.- М.: 2001. - 368с.
4. Лазарев Ю.Ф. Matlab 5.x.- К.: Изд. Группа BHV, 2000. - 384с.
5. Потемкин В.Г. Введение в Matlab.- М.: Диалог - МИФИ, 2000.- 247с.
6. Потемкин В.Г., Рудаков П.И. Matlab 5 для студентов.- 2-е изд., испр. и доп.- М.: Диалог-МИФИ, 1999. - 448с.
7. Потемкин В.Г. Система инженерных и научных расчетов Matlab 5.x.: В 2-х т.- М.: Диалог-МИФИ, 1999.

Інформацію про пакет Matlab можна знайти також на сайтах:

http://www.matlab.ru
http://www.exponenta.ru
http://www.mathworks.com
http://www.csit.narod.ru