Державний біотехнологічний університет

Навчально-науковий інститут «Кіберпорт»

Кафедра автоматизації та комп'ютерно-інтегрованих технологій

Методичні вказівки

до практичних занять

з дисципліни

"Програмне забезпечення інженерної діяльності"

(для студентів, що навчаються за напрямком підготовки
141 - "Електроенергетика, електротехніка та електромеханіка")

2023

Укладач:

Абраменко Іван Григорович, к.т.н., доцент,

Рекомендовано до друку:

на засіданні кафедри автоматизації та комп'ютерно-інтегрованих технологій, протокол № від . .2023 р.

ЗМІСТ

ПЕРЕДМОВА

Робота на ПК звичайно проходить у формі діалогу людини з комп'ютером. Людина переглядає інформацію на екрані комп'ютера, указує на неї мишкою, натискає клавіші, набирає команди, вводить слова, числа, фрази і т.д. У відповідь комп'ютер виводить свою інформацію: повідомлення, меню, діаграми, малюнки, результати обчислень і обробки даних.

Робота ПК заснована на використанні програм. Програми для нього - це форма представлення даних і команд, призначених для одержання певних результатів або способу функціонування ПК.

Сукупність програм на конкретному комп'ютері визначається завданнями, які вирішує конкретний або колективний (з використанням мережних технологій) користувач.

До прикладного програмного забезпечення (application software) відносяться комп'ютерні програми, написані для користувачів або самими користувачами, для виконання необхідних перетворень інформації і розраховані на безпосередню взаємодію з користувачем. У більшості операційних систем прикладні програми не можуть звертатися до ресурсів комп'ютера прямо, а взаємодіють із устаткуванням за допомогою операційної системи.

Для установки будь-яких програм на ПК треба використовувати відповідні дистрибутиви від виробника.

При використанні ПК завжди треба пам'ятати про те, що перенос будь-якої інформації з одного комп'ютера на інший, або підключення до мережі пов’язані з можливістю зараження шкідливим програмним забезпеченням – комп'ютерними вірусами та програмами-шпигунами.

Практичне заняття № 1. Ознайомлення із системою наукових і інженерних розрахунків Matlab

Мета. Ознайомлення із системою наукових і інженерних розрахунків Matlab; одержання початкових відомостей про вікно керування, вікно вбудованого редактора; ознайомлення з найпростішими операціями із числами, векторами й матрицями, елементарними математичними функціями; створення *m*-файлів.

Теоретична частина. Система Matlab є інтерактивною системою для виконання інженерних і наукових розрахунків, орієнтованої на роботу з масивами даних. Система використовує математичний співпроцесор і допускає можливість звертання до програм, написаних на мовах FORTRAN, C и.C++.

Привабливою особливістю системи є те, що вона містить розвинену вбудовану матричну й комплексну арифметику. Система підтримує виконання операцій з векторами, матрицями й масивами даних, реалізує сингулярне й спектральне розкладання, розрахунки рангу й чисел обумовленості матриць, підтримує роботу з алгебраїчними поліномами, розв'язок нелінійних рівнянь і завдань оптимізації, інтегрування у квадратурах, розв'язок диференціальних і різницевих рівнянь, побудова різних видів графіків, тривимірних поверхонь і ліній рівня. У ній реалізоване зручне операційне середовище, що дозволяє формулювати проблеми й одержувати розв'язки у звичній математичній формі, не прибігаючи до рутинного програмування.

Основним об'єктом системи Matlab є прямокутний числовий масив, який допускає комплексні елементи й введення матриць без явної вказівки їх розмірів. Система дозволяє вирішувати багато обчислювальних завдань за значно менший час, ніж те, яке необхідно для написання відповідних програм на мовах FORTRAN, Basic і C.

Система Matlab виконує операції з векторами й матрицями навіть у режимі безпосередніх обчислень без якого-небудь програмування. Нею можна користуватися як наймогутнішим калькулятором, у якому поряд зі звичайними арифметичними й алгебраїчними діями можуть використовуватися такі складні операції, як звертання матриці, обчислення її власних значень і векторів, розв'язок систем лінійних алгебраїчних рівнянь і багато інших. Однак основна відмітна риса системи — це легкість її модифікації й адаптації до конкретних завдань користувача. Користувач може ввести в систему будь-яку нову команду, оператор або функцію й користуватися потім ними так само просто, як і вбудованими операторами й функціями. При цьому, на відміну від мов програмування, таких як Basic, Pascal або C, немає необхідності в їхньому попередньому описі. Нові програми, функції й процедури в системі Matlab зберігаються у вигляді файлів, що мають розширення .m. Це робить набір операторів і функцій практично необмеженим.

У базовий набір слів системи входять: спецзнаки; знаки арифметических і логічних операцій; арифметичні, тригонометричні й деякі спеціальні математичні функції; функції швидкого перетворення Фур'є й фільтрації; векторні й матричні функції; засобb для роботи з комплексними числами; оператори побудови графіків у декартовой і полярної системах координат, тривимірних поверхонь і т.д. Таким чином, Matlab надає користувачеві великий набір готових засобів (близько половини з них — зовнішні розширення у вигляді m-файлів).

Система Matlab має власна мова програмування, що нагадує Basic. Запис програм у системі є традиційною й тому звичною для більшості користувачів персональних комп'ютерів. До того ж система дає можливість редагувати програми за допомогою будь-якого звичного для користувача текстового редактора.

Matlab має більші можливості для роботи із сигналами, для розрахунків і проектування аналогових і цифрових фільтрів, для побудови їх частотних, імпульсних і перехідних характеристик. Є в наявності й засоби для спектрального аналізу й синтезу, зокрема, для реалізації прямого й зворотного перетворення Фур'є. Завдяки цьому система досить приваблива для проектування електронних пристроїв.

Із системою Matlab поставляється понад сотню *m*-файлів, які містять демонстраційні приклади й визначення нових операторів і функцій. Ця бібліотека, усі файли якої докладно прокоментовані, — справжня скарбниця прекрасних прикладів програмування мовою системи. Вивчення цих прикладів і можливість роботи в режимі безпосередніх обчислень значно полегшують знайомство із системою серйозних користувачів, зацікавлених у використанні математичних розрахунків.

Робота в середовищі Matlab може здійснюватися у двох режимах:

- у режимі калькулятора, коли обчислення проводяться безпосередньо після набору чергового оператора або команди Matlab; при цьому значення результатів обчислення можуть привласнюватися деяким змінним, або результати виходять безпосередньо, без присвоювання (як у звичайних калькуляторах);

- шляхом виклику програми, складеної й записаної на диску мовою Matlab, яка містить усі необхідні команди, що забезпечують введення даних, організацію обчислень і вивід результатів на екран (програмний режим).

В обох режимах користувачеві доступні практично всі обчислювальні можливості системи, у тому числі по виводу інформації в графічній формі. Програмний режим дозволяє зберігати розроблені обчислювальні алгоритми й, таким чином, повторювати обчислення при інших вихідних даних.

Після виклику MATLAB із середовища Windows на екрані з'являється командне вікно середовища MATLAB (рис. 1.1).

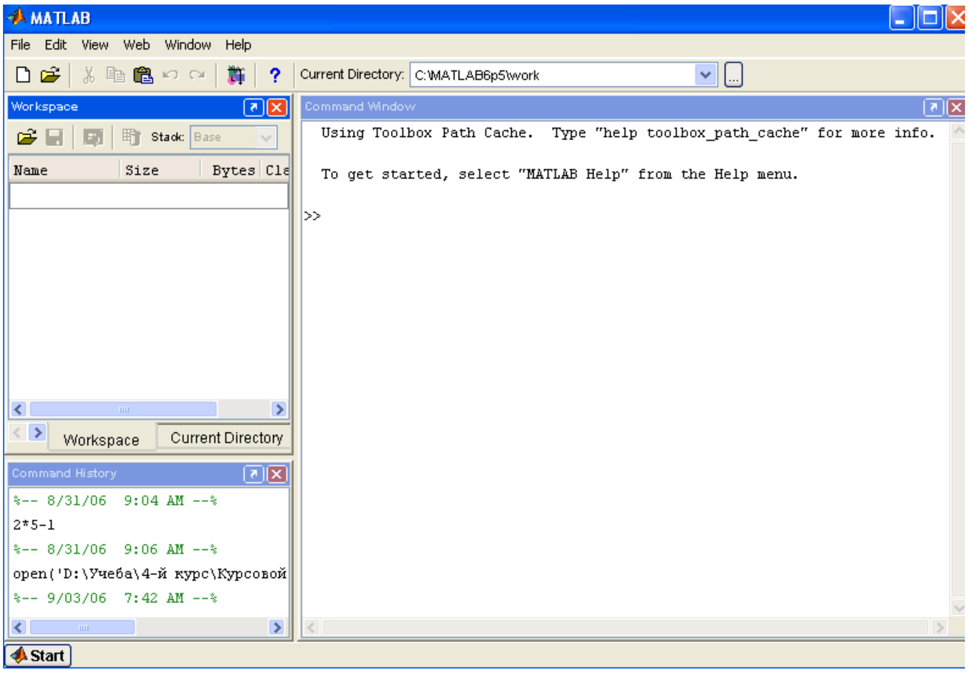


Рис. 1.1 – Командне вікно системи MATLAB

Це вікно є основним у системі MATLAB. У ньому відображаються символи команд, які набираються користувачем, результати виконання цих команд, текст програми, що виконується, а також інформація про помилки виконання програми, розпізнаних системою.

Ознакою того, що система готова до сприйняття й виконанню чергової команди, є наявність в останньому рядку текстового поля вікна знака запрошення (>>), після якого можливе введення символів команд і т.д.

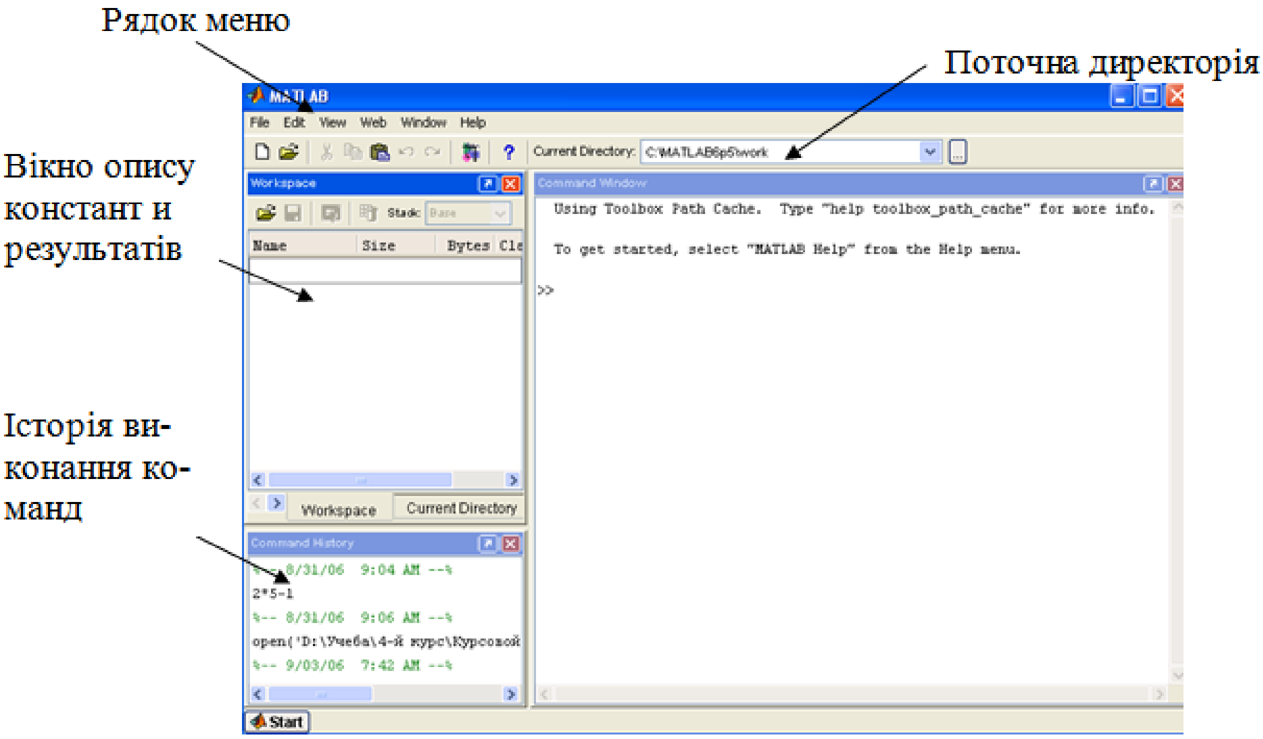


Рис. 1.2 – Інтерфейс системи MATLAB

Введення дійсних чисел.

Введення чисел із клавіатури проводиться за загальними правилами, прийнятим для мов програмування високого рівня:

- для відділення дробової частини числа застосовується десяткова крапка (замість коми при звичайному записі);
- десятковий показник числа записується у вигляді цілого числа після попереднього запису символу е (приклад: 1.2305e-5).

Найпростіші арифметичні операції:

\+ додавання;

\- вирахування;

* множення;

/ ділення зліва направо;

\ ділення справа наліво;

^ зведення в ступінь.

Використання MATLAB у режимі калькулятора може відбуватися шляхом простого запису в командний рядок послідовності арифметичних дій із числами, наприклад

*>> 4.52*7.23-3.14*10.4*

ans = 113.7515

Результат виконання дій виводиться як значення системної змінної *ans*.

Вивід проміжної інформації в командне вікно підкоряється наступним правилам:

- якщо запис оператора не закінчується символом ";", результат дії цього оператора відразу ж виводиться в командне вікно;
- якщо оператор закінчується символом ";", результат його дії не відображається в командному вікні;
- якщо оператор не містить знака присвоювання (=), тобто є просто записом деякої послідовності дій над числами й змінними, то значення результату привласнюється спеціальної системної змінної ans;
- отримане значення можна використовувати в наступних операторах обчислень під іменем ans; при цьому слід пам'ятати, що значення системної змінної ans змінюється після дії чергового оператора без знака присвоювання;
- у загальному випадку форма виводу результату в командне вікно має вигляд

<ім'я змінної> = <результат>

Особливістю системи MATLAB як калькулятора є можливість використання імен змінних для запису проміжних результатів на згадку. Для цього застосовується операція присвоювання у відповідності зі схемою:

<ім'я змінної> = <вираз>; (наприклад x=25+17)

У системі MATLAB є кілька зарезервованих імен змінних:

i і j – уявна одиниця (корінь квадратний з -1);

pi – число π (3,1416);

inf – позначення машинної нескінченності;

NaN – позначення невизначеного результату (наприклад типу 0/0, inf/inf).

Введення комплексних чисел.

Уведення комплексного числа проводиться шляхом запису в командному вікні рядка наступного типу:

*<ім'я змінної> = <значення ДЧ> + i (j) * <значення МЧ>,*

де ДЧ - дійсна частина комплексної величини; МЧ – уявна частина.

Наприклад x=3+j*2

Найпростіші арифметичні операції з комплексними числами:

\+ додавання;

\- вирахування;

* множення;

/ ділення зліва направо;

\ ділення зправа наліво;

^ зведення в ступінь.

В MATLAB є кілька додаткових функцій, розрахованих тільки на комплексний аргумент:

- real (Z) виділяє дійсну частину комплексного аргументу Z;
- imag (Z) виділяє комплексну частину комплексного аргументу Z;
- angle (Z) обчислює значення аргументу комплексного числа Z (у радіанах від – π до +π);
- conj (Z) видає число, комплексно сполучене відносно Z.

Елементарні математичні функції.

Загальна форма виклику функцій MATLAB має такий вигляд:

<ім'я результату> = <ім'я функції> (<список імен аргументів або їх значень>)

У мові MATLAB передбачені такі елементарні математичні функції:

<u>Тригонометричні й гіперболічні функції</u>

sin(Z) синус числа Z

sinh(Z) гіперболічний синус

asin(Z) арксинус (у радіанах, у діапазоні від - – π/2 до +π/2)

asinh(Z) зворотний гіперболічний синус

cos(Z) косинус

cosh(Z) гіперболічний косинус

acos(Z) арккосинус (у діапазоні від 0 до π)

acosh(Z) зворотний гіперболічний косинус

tan(Z) тангенс

tanh(Z) гіперболічний тангенс

atan(Z) арктангенс (у діапазоні від - – π/2 до +π/2)

atan2(X,Y) чотириквадрантний арктангенс (кут у діапазоні від – π до + π між горизонтальним правим променем і променем, що проходять через крапку з координатами X і Y)

atanh(Z) зворотний гіперболічний тангенс

sec(Z) секанс

sech(Z) гіперболічний секанс

asec(Z) арксеканс

asech(Z) зворотний гіперболічний секанс
csc(Z) косеканс
csch(Z) гіперболічний косеканс
acsc(Z) арккосеканс
acsch(Z) зворотний гіперболічний косеканс
cot(Z) котангенс
coth(Z) гіперболічний котангенс
acot(Z) арккотангенс
acoth(Z) зворотний гіперболічний котангенс

Експоненті функції

exp(Z) експонента числа Z
log(Z) натуральний логарифм
loglo(Z) десятковий логарифм
sqrt(Z) квадратний корінь із числа Z
abs(Z) модуль числа Z
round(Z) звичайне округлення числа Z до найближчого цілого

Найпростіші операції з векторами й матрицями.

MATLAB є системою, яка спеціально призначена для здійснення складних обчислень із векторами, матрицями й поліномами. Під вектором в MATLAB розуміється одномірний масив чисел, а під матрицею - двовимірний масив.

Вихідні значення векторів-рядків можна задавати шляхом заелементного введення. Для цього спочатку вказують ім'я вектора, потім ставлять знак присвоювання (=) квадратну дужку ([), за нею значення вектора, відокремлюючи їх між собою пробілами або комами. Завершується запис закриваючої квадратною дужкою (]).

Наприклад *V=[1 2 3] або V=[1,2,3]*

V = 1 2 3

Довгий вектор можна вводити частинами, які потім поєднують за допомогою операції об'єднання векторів у рядок.

V1=[1 2 3]; V2=[4 5 6];

V=[V1 V2]

V = 1 2 3 4 5 6

Мова MATLAB дає користувачам можливість скороченого введення вектора, елементи якого є арифметичною прогресією.

V = nz : h : kz,

де nz – початкове значення прогресії (перший елемент вектора);

kz – кінцеве значення прогресії (останній елемент вектора);

h – різниця прогресії (крок).

Наприклад *V= -0.1:0.3:1.4*

V= -0.1 0.2 0.5 0.8 1.1 1.4

Вектор-стовпець задається аналогічно вектору рядку, але елементи відділяються друг від друга знаком «;».

Введення елементів матриці здійснюється по рядках. При цьому елементи рядка матриці відділяються друг від друга пробілами або комами, а рядки відділяються друг від друга знаком «;».

Наприклад *A=[1,2,3; 4,5,6; 7,8,9]*

A = 1 2 3
4 5 6
7 8 9

Формування векторів і матриць.

MATLAB має кілька функцій, які дозволяють формувати вектори й матриці певного вигляді:

zeros (M, N) – створює матрицю розміром M на N з нульовими елементами;

zeros (2,3)
ans = 0 0 0
0 0 0

ones (M, N) - створює матрицю розміром M на N з одиничними елементами

ones(2,3)
ans = 1 1 1
1 1 1

eye (M, N) - створює матрицю розміром M на N з одиницями по головній діагоналі й усіма іншими нулями

eye(2,3)
ans = 1 0 0
0 1 0

rand (M,N) - створює матрицю розміром M на N з випадкових чисел, рівномірно розподілених у діапазоні від 0 до 1

rand(2,3)
ans = 0.9501 0.6068 0.8913
0.2311 0.4860 0.7621

Добування й вставка окремих елементів матриць.

Звертання до будь-якого елемента матриці здійснюється вказівкою після імені матриці номера рядка й номера стовпця на перетинанні яких розташований елемент матриці.

Наприклад *A = 1 2 3*
4 5 6
7 8 9

X= A (2,3)
X=6

Якщо потрібно навпаки вставити на це місце яке-небудь число, то це можна зробити в такий спосіб:

A(2,3)=pi
A = 1 2 3
4 5 3.14
7 8 9

Нехай потрібно створити вектор V1, що полягає з елементів третього стовпця матриці А.

V1=A(:,3)

V1 = 3

3,14

9

Нехай потрібно створити матрицю В розміром 2:2, що полягає з елементів правого верхнього кута матриці А.

B=A(1:2, 2:3)

B= 2 3

5 3,14

Нехай потрібно внести елементи матриці В у лівий нижній кут матриці А.

A(2:3, 1:2)=B

A= 1 2 3

2 3 6

5 3,14 9

Розтягти матрицю в один вектор можна за допомогою наступного запису:

A= 1 2 3

4 5 6

V=A(:)

V=1 2 3 4 5 6

Дії над векторами.

Виділимо 2 групи дій над векторами: векторні дії й дії по поелементному перетворенню векторів.

Векторні дії:

Додавання векторів: x=[1 2 3]; y=[4 5 6]; v=x+y (v= 5 7 9)

Вирахування векторів: v=x-y (v= -3 -3 -3)

Транспонування вектора: x' ans = 1

2

3

Множення вектора на число: v = x*2 (v= 2 4 6)

Множення 2 векторів (для векторів однакової довжини, один рядок, а інший стовпець)

*x=[1 2 3]; y=[4 5 6]; v=x'*y v = 4 5 6*

8 10 12

12 15 18

*v=x*y’ v = 32*

Дії по заелементному перетворення векторів.

Усі ці операції перетворять елементи вектора як елементи звичайного одномірного масиву чисел. До таких операцій відносяться всі з перерахованих вище елементарних математичних функцій, що залежать від одного аргументу.

Наприклад *x=[-2,-1,0,1,2]*

V=sin(x); V =-0.9093 -0.8415 0 0.8415 0.9093

V=tan(x); V=exp(x) і ін.

Додавання (вирахування) числа до кожного з елементів (+, -);
Поелементне множення векторів (.*);
Пелементний розподіл векторів (./, .\)
Поелементне зведення в ступінь.(.^).

Програмування в середовищі MATLAB. Створення *m*-файлів.

Робота в режимі калькулятора в середовищі Matlab, незважаючи на досить значні можливості, має істотні недоліки. Неможливо повторити всі попередні обчислення й дії при нових значеннях вихідних даних без повторного набору всіх попередніх операторів. Не можна повернутися назад і повторити деякі дії або по деякій умові перейти до виконання іншої послідовності операторів. І взагалі, якщо число операторів велике, стає проблемою налагодити правильну їхню роботу через неминучі помилки при наборі команд.

Тому складні, з перериваннями, заплутаними переходами по певних умовах, із часто повторюваними однотипними діями обчислення, які, до того ж, необхідно проводити неодноразово при змінених вхідних даних, вимагають їхнього спеціального оформлення у вигляді записаних на диску файлів, тобто у вигляді програм. Перевага програм у тому, що стає можливим кількаразове звертання до тим самим операторів і до програми в цілому. Створення програм дозволяє значно спростити й скоротити процес підготовки повторюваних обчислень, зробити процес обчислень більш наочним і прозорим, а завдяки цьому — різко зменшити ймовірність появи принципових помилок при розробці програм. Крім того, у програмах з'являється можливість автоматизувати й процес зміни значень вхідних параметрів у діалоговому режимі.

У мові MATLAB є програми 2 типів: Script-файли (керуючі програми) і файл - функції (процедури).

За допомогою Script-файлів оформляються основні програми, що управляють від початку до кінця організацією обчислювального процесу. Як файл - функції оформляються окремі процедури й функції (тобто такі частини програми, які розраховані на кількаразове використання при змінюваних вхідних параметрах).

Файл-функція (процедура) повинна починатися з рядка заголовка:

function [<ПКВ>]=<ім'я процедури>(<ПВВ>), де ПКВ - перелік скінченних величин, ПВВ - перелік вхідних величин.

Дані програми називаються m-файлами. Усі m-файли мають розширення *.m. m-файл повинен зберігатися в директорії яка є поточною (рис. 2.2).

Для створення нового m-файлу необхідно викликати File->New->M-file.

Приклад m -файлу наведений далі

function y = F1 (x, a)

y = x+a/x

Потім необхідно зберегти даний файл із іменем F1.m

Так, якщо ввести в MATLAB команду y=F1(1,2), то y=3.

Завдання.

1. Здійснити введення дійсного числа 2,15*10-7.

2. Виконати просту арифметичну операцію 8,3/6*2,7-0,0012*3,14.

3. Здійснити введення комплексного числа, дійсна частина якого рівна 4, а уявна рівна -9.

4. Виконати просту арифметичну операцію із двома комплексними числами, використовуючи одну з додаткових функцій комплексного аргументу.

5. Обчислити значення однієї з елементарних математичних функцій.

6. Сформувати вектор з 5 будь-яких ненегативних елементів.

7. Сформувати матрицю розміром 3х4 з 1 по головній діагоналі й нульовими іншими елементами.

8. У створеній матриці добути елемент 2-го рядка й 3-стовпця.

9. Розтягти дану матрицю в один вектор.

10. Створити 2 вектора *x* і *y* по 3 елемента кожний і провести операції додавання, вирахування, транспонування векторів, і їх перемножування.

11. Створити *m*-файл, що реалізує обчислення наступної функції

$$y = d^3 * ctg(x) * \sqrt{\sin^4(x) - \cos^4(x)}.$$

Практичне заняття № 2. Матричні дії над матрицями. Операції з поліномами

Мета. Ознайомлення з матричними діями над матрицями в системі Matlab. Одержання практичних навичок для виконання операцій з поліномами в системі Matlab.

Теоретична частина. Базові дії з матрицями – додавання, вирахування, транспонування, множення матриці на число, множення матриць, зведення матриці в целую ступінь. Дані операції здійснюються в Matlab за допомогою звичайних знаків арифметичних операцій.

Важливо пам'ятати ряд умов, при яких ці операції можливі:

- при додаванні або вирахуванні матриць вони повинні мати однакові розміри;

- при множенні матриць число стовпців першого множника повинне збігатися із числом рядків другого множника.

Приклад додавання й вирахування матриць:

A=[1 2 3 4 5; 6 7 8 9 11]

A= 1 2 3 4 5
6 7 8 9 11

B=[0 -1 -2 -3 -4; 5 6 7 8 9]

B = 0 -1 -2 -3 -4
5 6 7 8 9

A+B

ans = 1 1 1 1 1
13 15 17 20

A-B

ans = 1 3 5 7 9
1 1 1 2

Приклад множення матриці на число

*A*5*

ans = 5 10 15 20 25
35 40 45 55

Приклад транспонування матриці

A'

ans = 1 6
7
8
9
11

Приклад множення матриці на матрицю

*A'*B*

ans = 30 35 40 45 50
40 45 50 55

45 50 55 60
50 55 60 65
61 67 73 79

Функція обернення матриці – inv (A) – обчислює матрицю зворотну заданій матриці А. Вихідна матриця А повинна бути квадратною і її визначник повинен бути відмінний від нуля.

Приклад зведення матриці в ступінь

A^2

Досить цікавими в мові Matlab є операції розподілу матриць ліворуч праворуч і праворуч ліворуч .(/ і \)

Операція А\У рівносильна сукупності операцій inv(A)*B, яка є розв'язком матричного рівняння: A*X=B.

Для прикладу розглянемо розв'язок системи лінійних алгебраїчних рівнянь:

$x_1 + 2x_2 + 3x_3 = 14$
$2x_1 - x_2 - 5x_3 = -15$
$x_1 - x_2 - x_3 = -4$

A=[1 2 3; 2 -1 -5; 1 -1 -1]
A= 1 2 3
2 -1 -5
1 -1 -1

B= [14; -15; -4]
B = 14
-15
- 4
x = A \ B
x = 1
2
3

Тобто $x_1 = 1$ $x_2 = 2$ $x_3 = 3$ – коріння системи рівнянь.

У системі Matlab передбачені можливості математичного оперування з поліномами.

Поліном (багаточлен) як функція визначається наступним виразом

$$P(x) = a_n * x^n + ... + a_2 * x^2 + a_1 * x + a_0$$

В Matlab поліном задається й зберігається у вигляді вектора, елементами якого є коефіцієнти полінома від an до a_0

$$P = [an_{\ ...}\ a_2\ a_1\ a_0]$$

Введення поліномів здійснюється також як і введення вектора довжиною n+1, де n – порядок полінома.

Система Matlab має функцію *roots(P)* яка обчислює вектор, елементи якого є коренями заданого полінома, по вектору коефіцієнтів.

Нехай потрібно знайти корені полінома $P(x) = x^5+8x^4+31x^3+80x^2+94x+20$

P=[1 8 31 80 94 20]

roots(P)

-1.0000+3.0000i

-1.0000+3.0000i

-3.7321

-2.0000

-0.2679

Зворотна операція – побудова вектора P коефіцієнтів полінома по заданому вектору його корінь – здійснюється функцією *poly*.

P = poly(R),

де R – заданий вектор корінь полінома, P – обчислений вектор коефіцієнтів полінома.

Приклад:

P = [1 8 31 80 94 20]

P = 1 8 31 80 94 20

R= roots (P)

R = -1.0000+3.0000i

-1.0000+3.0000i

-3.7321

-2.0000

-0.2679

P1 = poly(R)

P1 = 1.0000 8.0000 31.0000 80.0000 94.0000 20.0000

Для обчислення значення полінома за заданим значенням його аргументу в Matlab передбачена функція *polyval*. Звертання до неї відбувається за схемою:

y = polyval (P, x),

де P – вектор коефіцієнтів полінома;

x – значення аргументу полінома.

Приклад

P = [1 8 31 80 94 20]

x = 2

y = polyval (P, x)

y = 936

Обчислення похідної від полінома проводиться функцією *polyder*. Ця функція створює вектор коефіцієнтів полінома, що представляє собою похідну від заданого полінома.

dp = polyder (P)

dp = 5 32 93 160 94

Завдання.

1. Провести операції додавання, вирахування, перемножування довільних матриць розміром (4x4), множення матриці на ціле позитивне число, транспонування матриці.

2. Знайти зворотну матрицю для довільної матриці.

3. Використовуючи функцію обертання матриці й функцію розподілу матриць розв'язати систему лінійних рівнянь

$$\begin{cases} 5x_1 + 2x_2 + 15x_3 + 10x_4 = 22 \\ x_1 - x_2 + 3x_3 - 5x_4 = 4 \\ -x_1 + 8x_2 + x_3 = 7 \\ 6x_1 + 5x_2 + 4x_3 + 3x_4 = 2 \end{cases}.$$

4. Знайти корены полінома *P(x) = 2x6+3x5+4x4+12x3+5x2+6x+5*.

5. По заданому вектору коренів полінома (R = [2; 4; 8; 16; 32]) знайти його коефіцієнти.

6. Обчислити значення полінома із пункта 4 при аргументі рівному 5.

7. Обчислити похідну від полінома P(x).

Практичне заняття № 3. Візуалізація обчислень у системі Matlab

Мета. Ознайомлення з можливостями візуалізаціїї результатів в системі Matlab.

Теоретична частина. Система MATLAB має багаті можливості графічного представлення інформації. Вона дозволяє будувати двомірні й тривимірні графіки функцій, заданих в аналітичному вигляді, у вигляді векторів і матриць; дає можливість побудови безлічі функцій на одному графіку; дозволяє представляти графіки різними кольорами, типами крапок і ліній і в різних системах координат.

Система здатна будувати діаграми, гістограми й графіки спеціальних функцій.

Двомірна графіка.

Основною функцією, що забезпечує побудова графіків, є функція plot.

Загальна форма звертання до цієї функції наступна:

plot (x, y)

plot (x, в, s)

plot (x l, yl, s1, x2, y2, s2, . . . , xn, yn, sn),

де: x - аргумент функції, що задається у вигляді вектора;

y - функція, представлена в аналітичному виді або у вигляді вектора або матриці;

s - вектор стилів графіка; константа, що визначає колір ліній графіка, тип крапок і тип лінії;

x1, x2,..., xn - аргументи п функцій, зображуваних на одному графіці;

y1, в2, ..., yn - функції, зображувані на одному графіку.

Розглянемо більш докладно функції двомірної графіки й приведемо приклади.

Функція *plot(x,y)*

Функція дозволяє будувати графік при завданні функції y = f(x) в аналітичному виді, у вигляді вектора або матриці.

У математичних розрахунках знаходить широке застосування. Найбільш часто використовується в наступних випадках:

- вибір області ізоляції кореня рівняння f(x) = 0;

- визначення координат особливих крапок функції (максимумів, мінімумів, крапок перегину, розривів непреривностей);

- перевірка вірогідності вибору функції інтерполяції;

- якісна оцінка точності представлення функції степенним рядом.

Функція *plot (x, y, s)*

Функція аналогічна функції plot (x, y) і відрізняється лише наявністю вектора констант s, що визначає колір ліній графіка, тип крапок і ліній функції, тобто стиль графіка. Стиль графіка s можна не задавати.

У таблиці 3.1 наведені стилі графіків системи MATLAB.

Таблиця 3.1 - Стилі графіків

Тип точки		Колір лінії		Тип лінії	
.	Крапка	Y	Жовтий	-	Суцільна
O	Окружність	M	Фіолетовий	:	Подвійний пунктир
×	Хрест	C	Блакитний	-.	Штрих-унктир
+	Плюс	R	Червоний	--	Штрихова
*	Восьмиконечна сніжинка	G	Зелений		
S	Квадрат	B	Синій		
D	Ромб	W	Білий		
V, ^, <, >	Трикутник нагору, униз, уліво, вправо	K	Чорний		
P	П'ятикутна зірка				
H	Шестикінечна зірка				

При завданні стилю символ s представляється у вигляді вектора, елементами якого є: тип крапки, колір лінії й тип лінії, розділені комами й виділені одиночними лапками.

Наприклад

*plot (x, в, [‘R',   ‘ * ‘,   ‘ - .’])*

Це графік червоного кольору (R), крапки графіка у вигляді зірочок (*), лінії штрихпунктирні (-.).

Функція *plot(x1, y1, s1, x2, y2, s2, ...xn, yn, sn)*

Ця функція дозволяє будувати велике число математичних функцій на одному графіку. Позначення мають наступний сенс:

x_i - і-й масив аргументів, заданий у вигляді вектора;

yi - і-й масив значень функції для заданого масиву аргументів;

Si - стиль графіка для і-й функції.

Графіки в системі Matlab завжди виводяться в окремому графічному вікні, яке називається фігурою.

Приклад. Нехай потрібно вивести графік функції y=3sin(x+π/3) на проміжку від -3π до +3π із кроком π/100.

Спочатку треба сформувати масив значень аргументу x, потім обчислити масив відповідних значень функції й, нарешті, побудувати графік залежності y(x).

У системі Matlab дана послідовність операцій буде виглядати в такий спосіб

*x = -3*pi : pi/100 : 3*pi;*

*y = 3*sin(x+pi/3);*

plot (x,y)

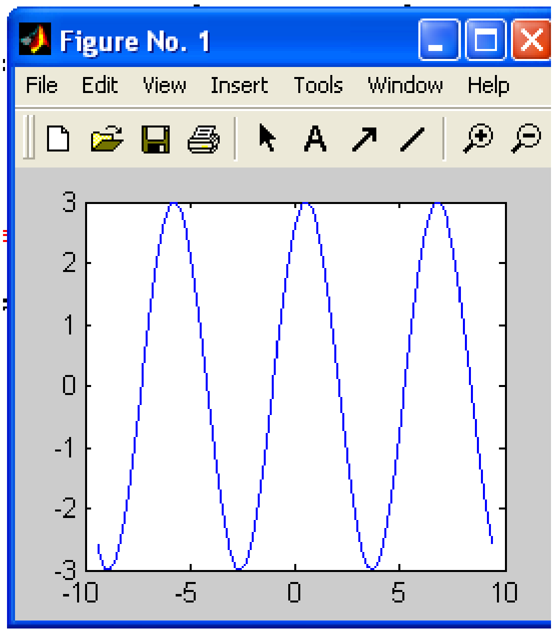


Рис. 3.1

Графік має кілька недоліків:

- не нанесена сітка з координатних ліній, що утрудняє «читання» графіка;
- немає загальної інформації про графік (його заголовка);
- невідомо які величини відкладені по осях.

Перший недолік усувається за допомогою функції grid, яка записується відразу після функції plot.

Заголовок графіка виводиться за допомогою процедури title, яка також записується після функції plot:

title (‘текст’)

Назва осей наносяться двома функціями *xlabel (‘x') і ylabel (‘y').*

Приклад:

*x= -3*pi : pi/100 : 3*pi;*
*y = 3*sin(x+pi/3);*
plot (x,y), grid
title ('function y=f(x)');
xlabel ('x');
ylabel ('y')

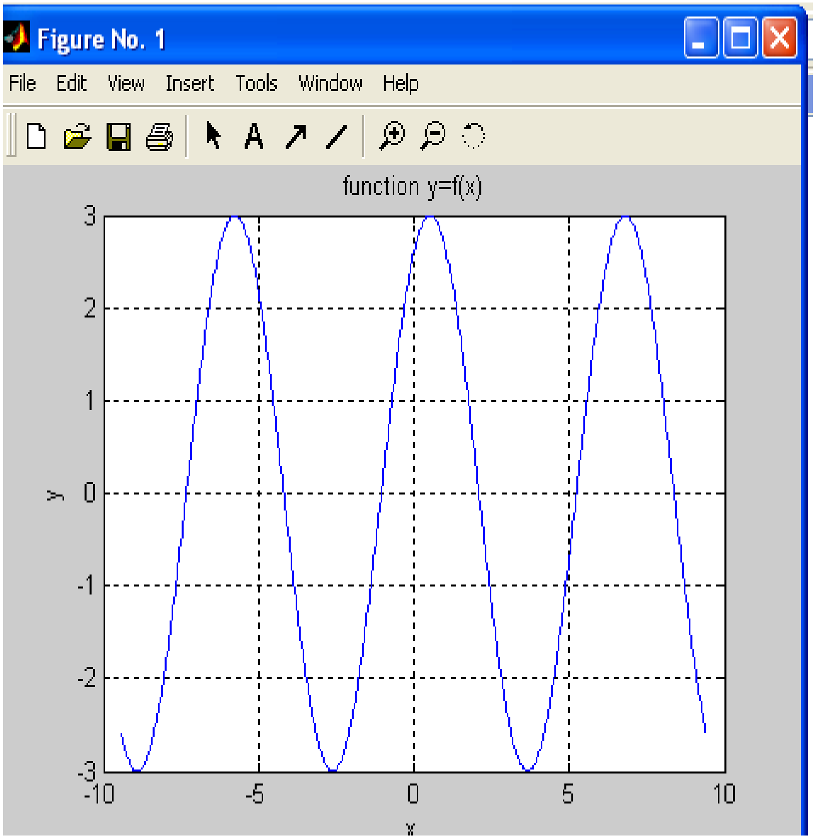


Рис. 3.2

Графічне представлення у вигляді ступінчастого графіка здійснюється за допомогою функції *stairs (x, y).*

Спочатку треба сформувати масив значень аргументу x, а потім здійснити вивід функції у вигляді ступінчастого графіка

Приклад:

x = 0:.25:10;

stairs (x,sin(x))

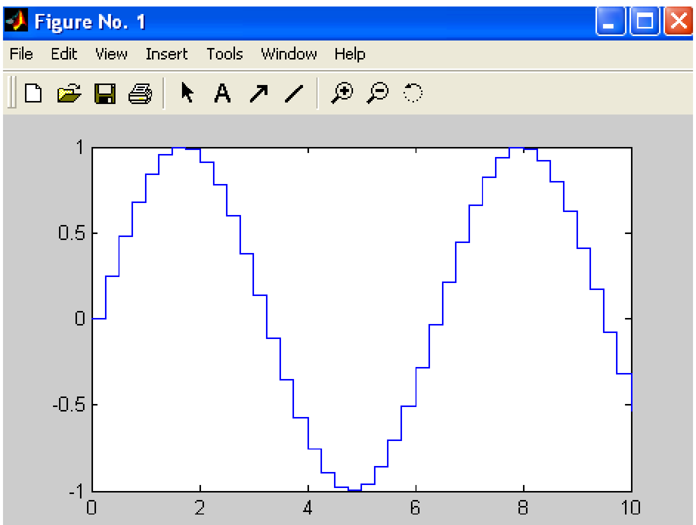


Рис. 3.3

Наступним прикладом є побудова графіка у вигляді стовпчастої діаграми. Це здійснюється за допомогою функції *bar(x, y)*

x = -2.9:0.2:2.9;
*bar(x,exp(-x.*x))*
colormap hsv

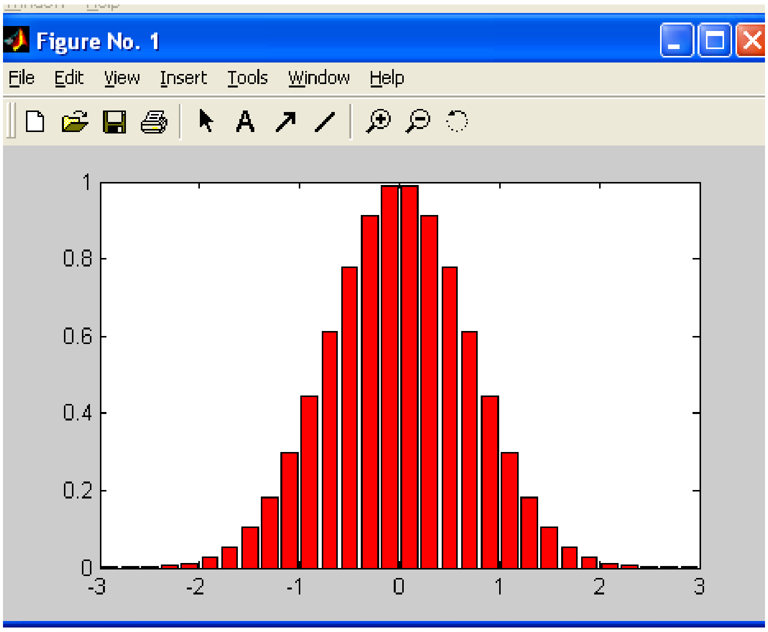


Рис. 3.4

Іншим прикладом є побудова графіка у вигляді гістограми. Це здійснюється за допомогою функції *hist(y, x)*

Побудуємо гістограму випадкових величин, які формуються функцією randn.

x = -2.9:0.1:2.9;
y = randn(10000,1);
hist(y,x)

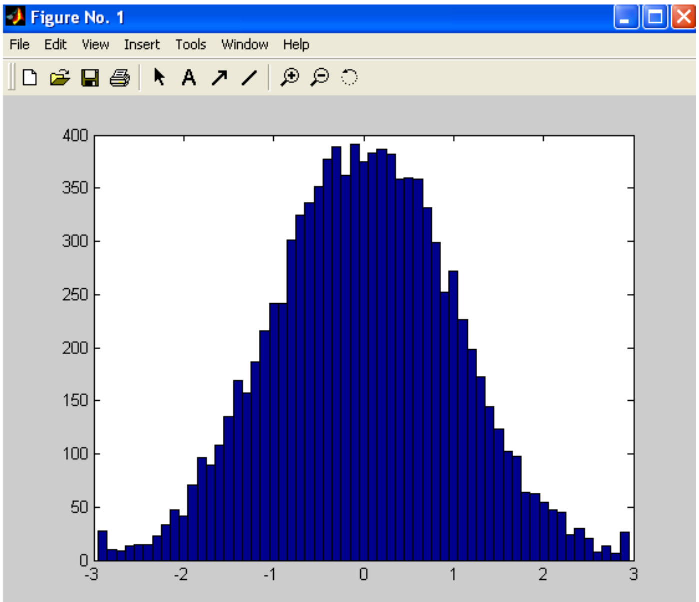


Рис. 3.5

Для побудови декількох графіків у тому самому вікні використовується функція *subplot*. Синтаксис якої наступний

subplot (m,n,p),

де m – число графіків по горизонталі, n – по вертикалі, p – поточна позиція графіка.

Приклад:

*x= -3*pi : pi/100 : 3*pi;*
*y1 = 3*sin(x+pi/3);*
*y2= 3*cos (2*x-pi/4);*
subplot(1,2,1), plot (x,y1), grid;
subplot (1,2,2), plot (x,y2), grid

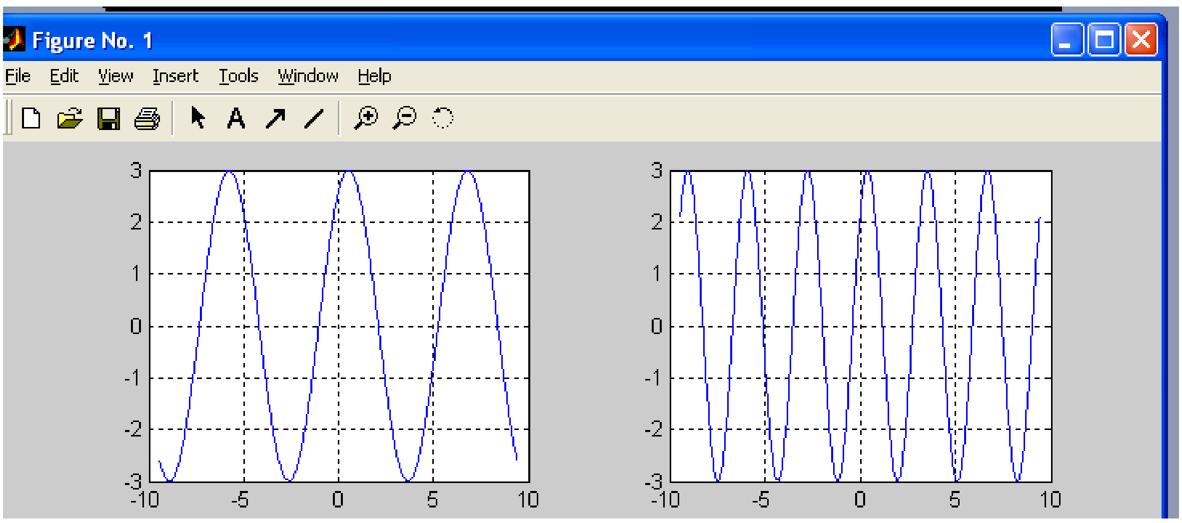


Рис. 3.6

Тривимірна графіка

Система MATLAB має багаті можливості побудови тривимірних графіків. Ми розглянемо тільки кілька функцій, що дозволяють створювати тривимірні графіки.

Для створення тривимірного графіка z = f(x, y) необхідно мати матриці значень змінних x, y.

Для цього призначені наступні функції:

[X,Y] = meshgrid (x, y)
[X,Y] = meshgrid (x)
[X,Y,Z] = meshgrid (x, y, z)

Функція *[X,Y] = meshgrid(x,y)* — перетворить область векторів x, в у масиви X, Y, які використовуються для обчислення функції z = f(x, y) і побудови графіків.

Рядкb масиву X є копіями вектора x, а стовпці масиву Y - копіями вектора y.

Функція *[X,Y,Z] = meshgrid(x, y, z)* повертає тривимірний масив для побудови тривимірного графіка.

Графіки тривимірних поверхонь будуються за допомогою наступних функцій:

plot3 (x, y, z)
plot3 (X, Y, Z)
plot3 (X, Y, Z, s)
plot3 (x l, yl, zl, s1, x2, y2, z2, s2, . . . , xn, yn, zn, sn)

де:

- x, в, z — вектори аргументів функції;
- X, Y, Z — матриці однакового розміру;
- s- стилі ліній і крапок графіка, аналогічно функції plot ().

Наведені функції будують крапки тривимірного графіка й з'єднують їх відрізками прямих відповідно до заданого стилю.

Функція plot3(xl,yl,zl,sl,x2,y2,z2,s2,...,xn, yn, zn,sn) будує на одному малюнку n функцій.

Приклад: побудувати графік функції $z = ln\ x + ln\ в$ у діапазоні аргументів [-4; 4] із кроком h = 0.1.

[X,Y] = meshgrid([-4:0.1:4]);
Z = log (X) + log (Y);
plot3(X,Y,Z)

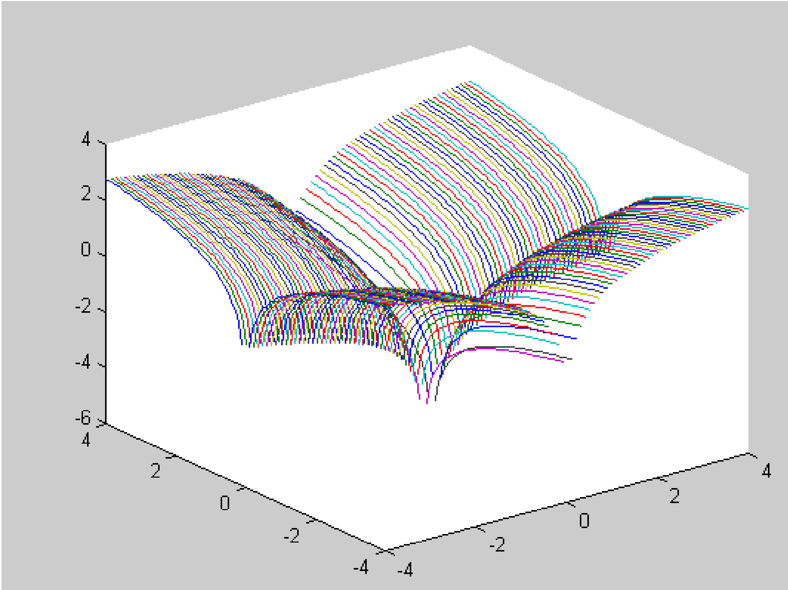


Рис. 3.7

В Matlab існують функції, спеціально призначені для формування каркаса поверхні й самої поверхні по заданому масиву даних.

mesh (Z),
surf (Z), *де Z – масив даних заданий матрицею.*

Приклад1: *z=peaks(25);*
mesh(z);

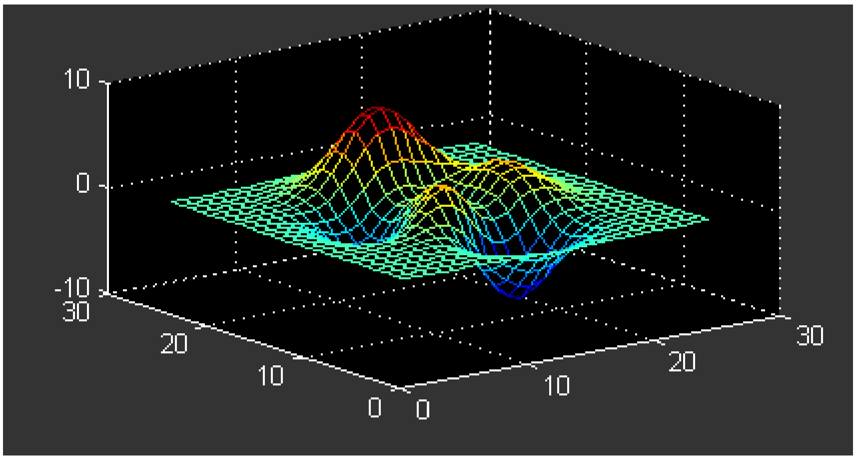


Рис. 3.8

Приклад2: *z=peaks(25);*
surf(z);

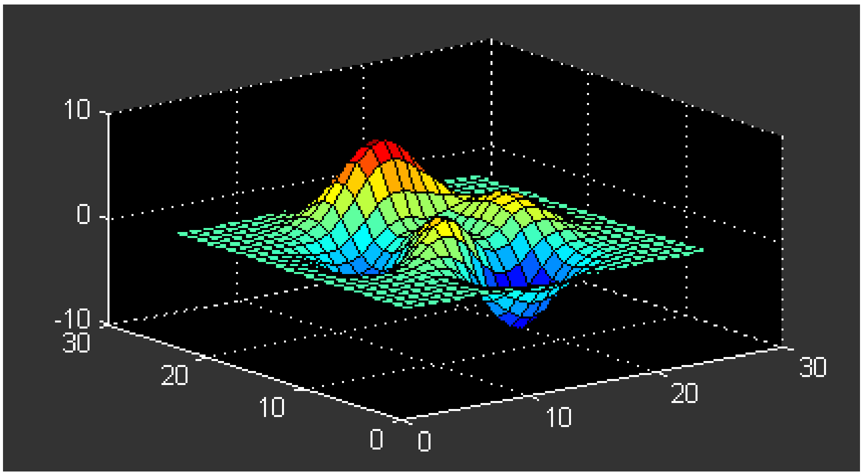


Рис. 3.9

Завдання.

1. Побудувати графік функції $y = (cos\ (x/\pi + \pi) + sin(x))/2$; на проміжку від -3π до +3π із кроком π/50.

Цей графік виконати зеленим кольором, крапки графіка у вигляді зірочок, лінія суцільна.

2. Додати до отриманого графіка координатну сітку, заголовок і назви осей.

3. Здійснити вивід функції $y = cos\ (x/\pi + \pi)$ у вигляді ступінчастого графіка в діапазоні від 0 до 100 із кроком 0.5

4. Побудувати графік функції $y = e^x$ у вигляді стовпчастої діаграми на відрізку від -3 до 3.

5. Побудувати 4 графіка довільних функцій в одному графічному вікні.

6. Побудувати графік функції $z = sin\ x + 2cos$ у діапазоні аргументів [-3; 3] із кроком h = 0.05.

7. Виконати побудова каркаса поверхні й самої поверхні. Вихідними даними є матриця (5x5) з випадкових чисел, рівномірно розподілених у діапазоні від 0 до 1.

Практичне заняття № 4. Алгоритми й технології обчислення інтегралів

Мета. Одержати практичні навики обчислення інтегралів.

Теоретична частина. Система MATLAB дозволяє обчислювати невизначені й визначені інтеграли, первообразні яких задані у вигляді аналітичних виразів.

Вона також має велику кількість способів чисельного інтегрування. Чисельне інтегрування необхідне в наступних випадках:

- первообразна не виражається через елементарні функції;
- аналітичне вираження інтеграла занадто складне;
- підінтегральна функція задана в табличній формі або у вигляді матриці.

При обчисленнях інтегралів чисельними методами підінтегральгу функцію доцільно представляти в найбільш простому виді. Це може прискорити обчислення. Спрощення цієї функції можна виконати, скориставшись функцією *simplify(y)*.

Мають місце випадки, коли система до спрощення не може обчислити невизначений інтеграл і легко його визначає після спрощення.

Метод обчислення інтеграла вибирає користувач. У цьому особливість системи MATLAB. За допомогою MATLAB студент має можливість порівнювати різні методи чисельного інтегрування.

Існує ряд способів чисельного інтегрування. У всіх таких способах обчислення здійснюється по наближених формулах, називаних квадратурними. Приведемо деякі з них.

Формули прямокутників.

Формули прямокутників представляються в наступному вигляді

$$\int_a^b f(x)dx = \begin{Bmatrix} h\sum_{k=0}^{n-1} y_k \\ h\sum_{k=1}^{n} y_k \end{Bmatrix},$$

де:

- h - крок інтегрування;
- y_k - значення підінтегральної функції при аргументі x_k, *k=0,1,2,..., n*;
- *n=(b-a)/h* - число частин, на які розбивається область інтегрування *a, b*.

Одна з формул дає значення інтеграла з надлишком, інша з недоліком.

Яка з них видає розв'язок з надлишком або з недоліком, залежить від вигляду підінтегральної функції.

Формула трапецій.

Ця формула має вигляд

$$\int_a^b f(x)dx = h\left(\frac{y_0}{2} + \sum_{k=1}^{n-1} y_k + \frac{y_n}{2}\right)$$

де:

- y_0 - значення підінтегральної функції при $x = a$;
- y_n - значення підінтегральної функції при $x = b$;
- h - крок інтегрування.

Формула парабол (Сімпсона).

Ця формула має вигляд

$$\int_a^b f(x)dx = \frac{h}{3}\left(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + 4y_5 + \ldots\right)$$

У цій формулі ординати з непарними індексами множаться на 4, а з парними - на 2. Передбачається, що n - число парне.

При непарному n формула має вигляд

$$\int_a^b f(x)dx = \frac{h}{3}\left(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + 4y_5 + \ldots + 2y_{n-1} + y_n\right)$$

Крайні ординати мають коефіцієнт, рівний 1.

Існує багато інших квадратурних формул обчислення інтегралів - Котеса, Чебишева, Гаусса й ін.

У системі MATLAB обчислення інтегралів реалізоване чисельними методами трапецій, парабол (Сімпсона) і Ньютона - Котеса.

Метод трапецій

Метод трапеції реалізований в MATLAB декількома функціями, наведеними нижче.

Функція *cumtrapz(y)*.

Здійснює обчислення інтеграла у випадку, коли значення функції y задані у вигляді вектора або матриці необмежених розмірів. Відгуком цієї функції є n інтегралів, де n - число елементів вектора або число елементів у кожному стовпці матриці.

Таке обчислення інтеграла називається інтегруванням з нагромадженням.

Приклад 1: Нехай функція *y(x)* має значення, представлені у вигляді наступного вектора: *y= [1,2,3,4,5,б,7,8,9,10]*. Необхідно обчислити $\int_a^b y(x)dx$.

При цьому *a=1; b=1,2, 3,..., 10*.

Функція обчислення інтеграла методом трапецій буде мати вигляд

y=[1,2, 3,4,5,6,7,8,9,10];
cumtrapz (y)

ans =
0 1.5000 4.0000 7.5000 12.0000 17.5000 24.0000 31.5000 40.0000 49.5000

Приклад 2: Нехай необхідно обчислити інтеграл виду $\int_0^{10}(3e^x + \ln x + 1)dx$.

Щоб обчислити цей інтеграл за допомогою функції *cumtrapz()*, слід спочатку обчислити 10 ординат підінтегральної функції, представивши їх у вигляді вектора.

Програма обчислення інтеграла з нагромадженням буде мати вигляд

x=1:1:10;
*y=3*exp(x)+log(x)+1;*
cumtrapz(y)

ans =
*1.0e+04 **
0 0.0017 0.0060 0.0174 0.0481 0.1311 0.3564 0.9684 2.6313 7.1510

Існує модифікація даної функції *cumtrapz(x, y)*.

Основним недоліком методу трапецій є суттєва погрішність результату обчислення інтеграла.

Функція *trapz(y)*.

Відмінність даної функції від функції *cumtrapz(y)* полягає в тому, що здійснюється просте інтегрування без нагромадження, тобто *trapz(y)* повертає не стільки інтегралів, наскільки кроків розбивається область інтегрування, а загальне значення інтеграла.

Приклад 1: обчислити інтеграл виду $\int_0^{10}(xe^x + \ln x + 1)dx$ із кроком 0,5.

x=1:0.5: 10;
*y=x.*exp(x) + log(x) + 1;*
trapz(y)
ans =
4.0657e+05

Існує модифікація даної функції *trapz (x, y)*. Слід мати у виді, що при обчисленні інтеграла за допомогою функції *trapz (x, y)* його значення залежить від кроку інтегрування.

Метод парабол (Сімпсона).

Для його реалізації в системі MATLAB використовуються наступні функції:

```
quad('fun', a, b)
quad('fun', a, b, tol)
quad('fun', a, b, tol, trace)
dblquad('fun', a, b, c, d)
dblquad('fun', a, b, c, d, tol)
```

У цих функціях прийняті позначення:

- *'fun'* - підінтегральна функція, узята в одинарні лапки;
- *a, b* - межі інтегрування;
- *tol* - відносна погрішність, що задається користувачем (за замовчуванням tol = 10e $^{-3}$);
- *c,d* - межі інтегрування по іншій змінній (зовнішній) при обчисленню подвійного інтеграла;
- *trace* - число, відмінне від нуля, по якому система показує хід обчислювального процесу.

Розглянемо перераховані функції й приведемо приклади.

Функція *quad('fun', a, b).*

Функція обчислює певний інтеграл $\int_a^b f(x)dx$ з погрішністю, що не перевищує 10^{-3}.

Приклад 1.

Підінтегральна функція має вигляд $f(x) = e^x + x^2 + 2\sin x - 5$.

Необхідно обчислити інтеграл

$$\int_1^5 f(x)dx$$

Розв'язок:

```
y='exp(x)+ x.^2+2.*sin(x)-5';
quad(y,1,5)
ans =
  167.5415
```

Функція може бути представлено одному рядком

```
quad('exp(x)+ x.^2+2.*sin(x)-5',1,5)
```

Функція *dblquad('fun',a,b,c,d).*

У функції dblquad ('fun', a, b, c, d) прийняті наступні позначення:

- 'fun' - це функція із двома змінними;
- a, b - межі по внутрішній змінній;
- c, d - межі по зовнішній змінній.

<u>Приклад 1</u>: Нехай функція двох змінних має вигляд $z = x^2 + y^2 - 2$.

Необхідно обчислити інтеграл $\int_1^2\int_0^3 z(x,y)dxdy$.

Розв'язок:

```
z='x.^2+y.^2-2';
dblquad (z,1,2,0,3)

ans =10.
```

Аналітичні методи обчислення інтеграла.

Функція *int()* обчислення невизначеного й визначеного інтегралів.

Обчислення інтеграла аналітичними методами здійснюється в системі MATLAB за допомогою функцій *int(),* які мають вигляд:

- *int(y(x))*
- *int(y(x), a, b),*

де:

- *y(x)* - підінтегральна функція;
- *a, b* - межі інтегрування.

Ці функції обчислюють:

- невизначений інтеграл;
- невизначений інтеграл із символьними змінними;
- визначений інтеграл із символьними значеннями меж інтегрування;
- визначений інтеграл від алгебраїчних функцій;
- кратні інтеграли;
- невласні інтеграли.

Технологія обчислення інтегралів досить проста й полягає в наступному:

- визначення символьних змінних за допомогою функції *syms*.
- введення підінтегрального виразу із присвоєнням йому імені *y=f(x);*
- введення функції *int(y),* якщо обчислюється невизначений інтеграл, або функції *int(c, a, b),* якщо обчислюється певний інтеграл у межах *[a; b];*
- одержання розв'язку шляхом натискання клавіші <Enter>.

<u>Приклад 1</u>: Необхідно обчислити інтеграл

$$\int_a^b \frac{x}{1+x^2}dx\,.$$

```
syms x;
y=x/(1+x^2);
int(y)

ans =
log(x^2 + 1)/2
```

Приклад 2 : Обчислити значення визначеного інтеграла

$$\int_1^5 \frac{x}{1+x^2}dx$$

```
syms x a b;
y=x/(1+x^2);
int(y,a,b)

ans =
log(b^2+1)/2 -log(a^2+1)/2
```

Приклад 3. Обчислити визначений інтеграл

$$\int_1^5 \frac{x}{1+x^2}dx$$

```
syms x;
y=x/(1+x^2);
int(y,1,5)

ans =
log(13)/2
```

Обчислення кратних інтегралів.

Найбільше просто кратний інтеграл обчислити шляхом інтегрування відповіді, отриманого від попереднього значення інтеграла.

Приклад 1. Нехай необхідно обчислити подвійний невизначений інтеграл

$$\iint \frac{x}{1-x^2}dx$$

Розв'язок:

```
syms x;
y=x/(1-x^2);
int(int(y))

ans=
x-atanh(x)-(x*log(x^2-1))/2
```

Завдання.

1. Провести інтегрування з нагромадженням (крок інтегрування рівний 0,5) для інтеграла $\int_3^9 x*e^x-\ln(x)+6$

2. Обчислити значення інтеграла (інтегрування з нагромадженням) від функції представленої у вигляді вектора коренів полінома

$$P(x) = x^5+8x^4+31x^3+80x^2+94x+20.$$

3. Обчислити за допомогою методу трапеції (крок інтегрування рівний 1) значення інтеграла

$$\int_1^5 \frac{15e^x+\ln x+1}{5x}dx.$$

4. Підінтегральна функція має вигляд $f(x)=-e^{x}+8x^{4}+3ctg\,x+1.$

Обчислити методом Сімпсона значення інтеграла від *f(x)* з точністю 10^{-5}. Межі інтегрування *[1;10]*.

5. Обчислити методом парабол значення подвійного інтеграла від функції *z=ln(x)+ln(y).*

Межі інтегрування по 1 змінній *[1, 5]*, а по зовнішній змінній *[2; 4]*.

6. За допомогою аналітичного методу знайти значення невизначеного інтеграла $\int \frac{x^2}{a+bx^3-cd}\,dx$.

7. За допомогою аналітичного методу обчислити значення визначеного інтеграла $\int_1^3 \frac{x}{c+dx^2}\,dx$.

8. Обчислити інтеграл $\iiiint \frac{\ln x+e^x}{3x^2}$.

Практичне заняття № 5. Розв'язок диференціальних рівнянь. Методи і комп'ютерні технології інтерполяції

Мета. Одержати практичні навики розв'язку диференціальних рівнянь. Ознайомитись з методами і комп'ютерними технологіями інтерполяції.

Теоретична частина. Диференціальним рівнянням називається рівняння, що зв'язує шукану функцію однієї або декількох змінних, ці змінні й похідні різних порядків даної функції.

Приклади диференціальних рівнянь:

$$\frac{dx}{dt} + t * x^2 + \sin x = 5$$

$$y'''+2y'-8y*t = 0$$

У загальному випадку розв'язок диференціального рівняння n-го порядку

$$y^n = f(x, y, y', ..., y^{n-1}) \tag{5.1}$$

полягає у відшуканні функції $y=f(x)$, при підстановці якої в рівняння (5.1) останнє звертається в тотожність. Порядок старшої похідної, що входить у диференціальне рівняння, називається порядком диференціального рівняння.

Кожне диференціальне рівняння має незліченна безліч розв'язків, тому для знаходження приватного розв'язку необхідно вказати початкові умови, а саме, задати значення $y, y', \ldots, y^{n-1}$ при $x=x_0$

$$y_0 = y(x_0),\ \ y_0' = y'(x_0),\ \ldots,\ y_0^{n-1} = y^{n-1}(x_0). \tag{5.2}$$

Завдання відшукання розв'язку рівняння (5.1) при заданих значеннях початкових умов (5.2) називається завданням Коші для звичайного диференціального рівняння.

Рівняння (5.1) зводиться до системи n звичайних диференціальних рівнянь першого порядку заміною y' на невідому функцію p. Наприклад, рівняння другого порядку $y'' - 4y' + 5y = 3x$ можна записати у вигляді системи двох рівнянь

$$\begin{cases} p = y', \\ p' = 4p - 5y + 3x. \end{cases}$$

Диференціальні рівняння вирішують наближеними методами. Існують дві групи наближених методів - аналітичні й чисельні.

Аналітичні методи:

- метод послідовного диференціювання (розкладання в ряд Тейлора);

- метод послідовних наближень.

Чисельні методи:

- метод Ейлера;
- удосконалений метод Уйлера (метод Уйлера – Коші);
- метод Рунге – Кутта.

Метод Рунге – Кутта.

Даний метод має більш високу точність, ніж методи Ейлера за рахунок зменшення методичних помилок. Ідея методу полягає в наступному.

По методу Ейлера розв'язок диференціального рівняння першого порядку визначається зі співвідношення

$$y_{i+1} = y_i + \Delta y_i$$

Тоді збільшення Δу може бути знайдене шляхом інтегрування

$$\Delta y_i = \int_{x_i}^{x_{i+1}} y'(x, y)dx = \int_{x_i}^{x_{i+1}} f(x, y)dx .$$

Або остаточно

$$y_{1+i} = y_i + \int_{x_i}^{x_{i+1}} f(x, y)dx$$

Потім обчислюється інтеграл по методу прямокутників.

У методі Рунге - Кутта шуканий інтеграл представляється у вигляді наступної кінцевої суми

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y)dx = y_i + \sum_{i=1}^{q} p_i K_i(h)$$

де p_i - деякі числа, що залежать від q;

K(h) - функції, що залежать від вигляду підінтегральної функції f(x,y) і кроку інтегрування h.

Система MATLAB дозволяє вирішувати диференціальні рівняння й системи високого порядку з табличним і графічним представленням результатів.

Найбільше часто диференціальними рівняннями описуються процеси, що протікають у часі. Тоді змінною інтегрування є час. Функції ode () реалізують чисельні методи Рунге - Кутта третього, четвертого, п'ятого й шостого порядків з автоматичним вибором кроку.

Функціями розв'язку диференціальних рівнянь є ode23() і ode45(), що мають вигляд:

[t, y]=ode23(‘fun’ t_0, tf, y_0)
[t, y]=ode45(’fun’ t_0, tf, y_0)
[t, y]=ode23(‘fun’ t_0, tf, y_0, tol, trace)
[t, y]=ode45(‘fun’, t_0, tf, y_0, tol, trace)

де:
- fun' - ім'я m-файлу, у якому втримуються праві частини системи диференціальних рівнянь;
- t_0 - початкове значення аргументу;
- tf - кінцеве значення аргументу;
- y_0 - вектор початкових умов;
- tol - точність, що задається, за замовчуванням, для ode23() - 10^{-3} для ode45() – $l0^{-6}$;
- trace - видача проміжних результатів.

Технологія розв'язку диференціальних рівнянь у системі MATLAB така:
- створення нової функції, що представляє собою m-файл обчислення правих частин системи диференціальних рівнянь;
- введення функції ode ();
- одержання розв'язку натисканням клавіші <Enter>.

<u>Приклад1</u>: розв'язати диференціальне рівняння $\frac{dx}{dt} + x = \sin x * t$

x(0) = 1,5-початкове значення; інтервал інтегрування [0, 35].
Створюємо m-файл

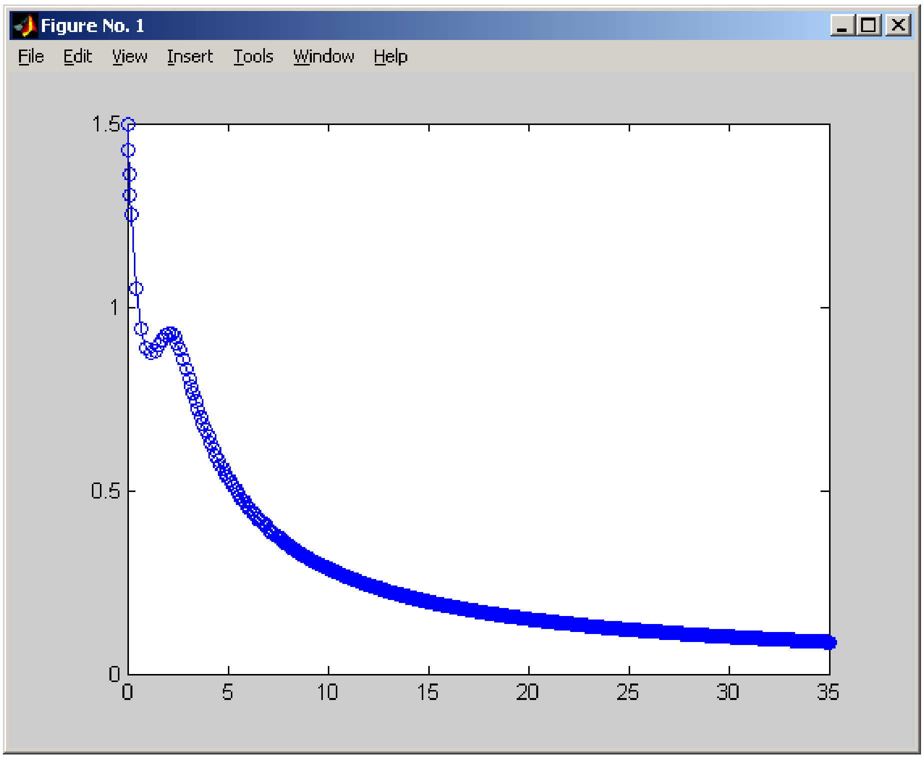
```
function y=D(t,x)
y=-x+sin(x*t)
end
```

Викликаємо функцію

а) *ode23('D', [0, 35], 1.5)*
б) *ode45('D', [0, 35], 1.5)*
в) *ode45(@D, [0, 35], 1.5)*
в) *[T,Y] = ode45(@D, [0, 35], 1.5)*
   plot (T,Y)

@ D-це посилання на створений нами m-файл утримуючий функцію f(x,t), тобто праву частину нашого диференціального рівняння.

Видається масив розв'язків у і будується наступний графік.



Інтерполяція функцій.

Однієї з найважливіших завдань у процесі математичного моделювання є обчислення значень функцій, що входять в опис моделі. Для складних моделей подібні обчислення можуть бути досить трудомісткими навіть при використанні ЕОМ.

Використовувані в математичних моделях функції задаються як аналітичним способом, так і табличним, при якому значення функції відомі тільки при дискретних значеннях аргументів.

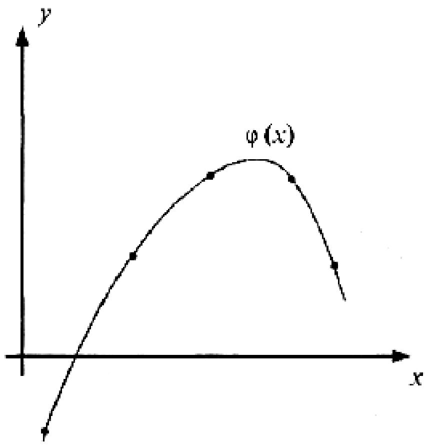
Відзначені проблеми вирішуються шляхом наближеної заміни функції f(x) більш простою функцією φ(x), яку неважко обчислювати при будь-якому значенні аргументу x у заданому інтервалі його зміни.

Функцію φ(x), яку називають апроксимуючою функцією, можна використовувати не тільки для наближеного визначення чисельних значень f(x), але й для проведення аналітичних викладок при теоретичному дослідженні моделі.

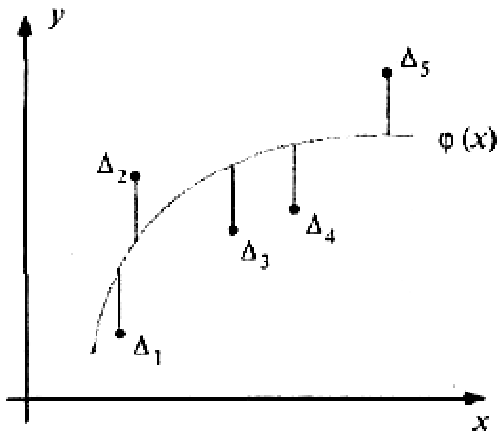
Допустимо в нас є таблиця значень функції при конкретних значеннях аргументу. У найпростішому випадку під інтерполяцією розуміється відшукання значень функції f(x), відповідних до проміжних значень аргументу, відсутнім у таблиці, тобто «читання між рядками».

Основними видами інтерполяції є: точна у вузлах і наближена у вузлах:

При інтерполяції, точної у вузлах, значення апроксимуючої функції збігаються зі значеннями вихідної функції у вузлах інтерполяції.



При інтерполяції, наближеної у вузлах, значення апроксимуючої функції не збігаються зі значеннями вихідної функції у вузлах інтерполяції.



Інтерполяція точна у вузлах. Сплайн-Інтерполяція.

Інтерполяція кубічними сплайнами в середовищі MATLAB здійснюється за допомогою функції spline (). Функція має вигляд

Yi = spline (x,y, xi)

де:

- x — вектор вузлів інтерполяції;
- y — вектор значень функції у вузлах інтерполяції;
- xi- вектор аргументів функції y=f(x) з області її визначення, що задається користувачем.

Замість вектора функція y=f(x) може бути задана у вигляді формули. Функція spline () не дозволяє одержати функцію інтерполяції у вигляді формули. У цьому її істотний недолік.

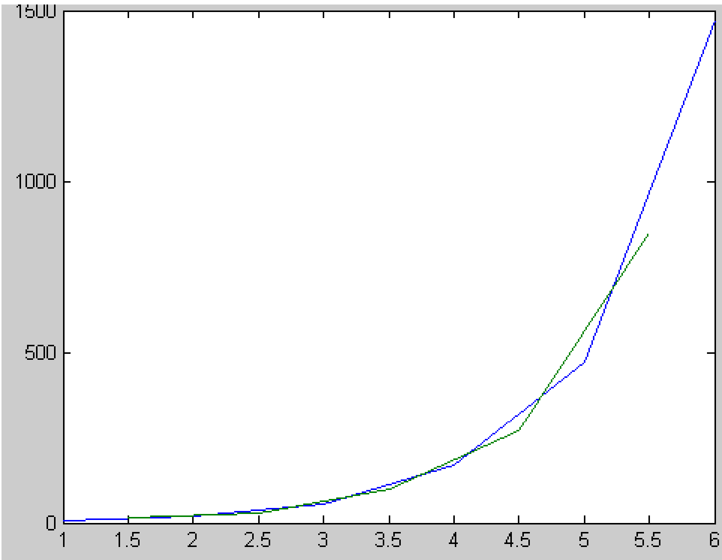
Приклад1.

Нехай функція задана таблицею

x	1	2	3	4	5	6
У	6,5	20	53,5	167	473	1470

Знайти значення функції при x = 1.5, 2.5, 3.5, 4.5, 5.5.

x=[1, 2, 3, 4, 5, 6];
y = [6.5, 20, 53.5, 167, 473, 1470];
xi = [1.5, 2.5, 3.5, 4.5, 5.5];
yi = spline (x, y, xi)
plot(x,y,xi,yi)
yi =   15.2333   29.7667   98.7000 270.9958 847.7542



<u>Приклад 2.</u>

Нехай функцією є *y=sin x*, задана при x = 1, 2, 3, 4, 5. Необхідно знайти її значення при x=1.5, 1.8,2.3,3,3.5.

x = [1, 2, 3, 4, 5];
xi = [1.5, 1.8, 2.3, 3, 3.5];
yi = spline (x, sin(x), xi)

yi =   1.0222   0.9843   0.7358   0.1411   -0.3414.

З відповіді видні погрішності інтерполяції: значення sinx не може бути більше одиниці.

<u>Інтерполяція точна у вузлах.</u>

Функція interp() дозволяє вирішувати завдання інтерполяції декількома методами. Вона має вигляд

yi = interp1 (x, y, xi, метод)

де:

- x, y — вектори значень вузлів і функції;
- xi — вектор значень аргументів, що задається користувачем;
- *метод* — аргумент, що дозволяє користувачеві вибрати метод інтерполяції.

Методами інтерполяції є:

- nearest -східчаста;
- linear — лінійна;
- cubic — кубічна;
- spline — кубічними сплайнами.

Якщо метод не зазначений, то реалізується лінійна інтерполяція. Приведемо приклад розв'язку завдання цими методами.

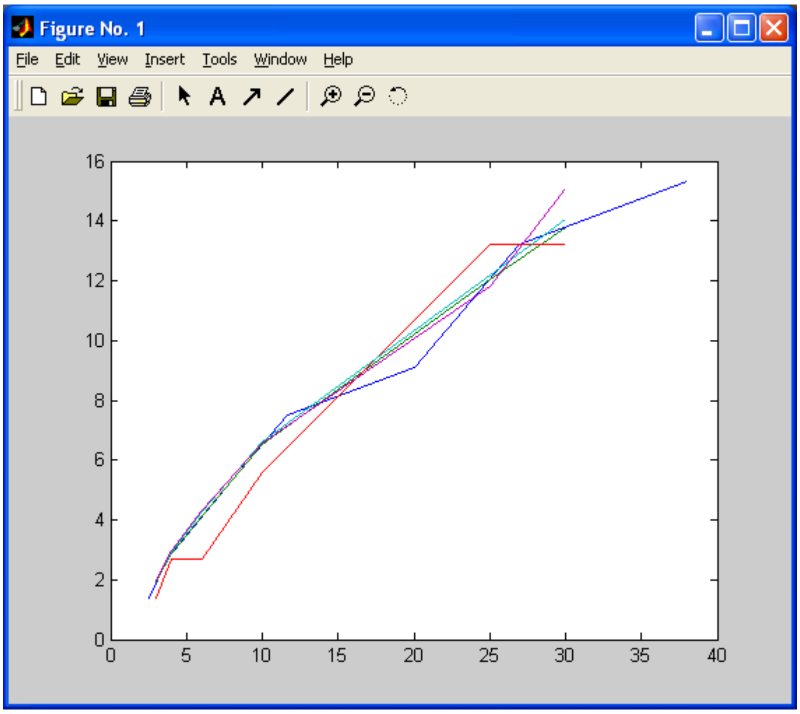
Приклад1.

Функція y=f(x) задана таблично

X	2.5	3.7	8.4	11.7	20	27	38
y	1.4	2,7	5,6	7,5	• 9,1	13,2	15.3

Необхідно визначити значення функції при значеннях аргументів x = 3, 4, 6, 10, 25, 30, використовуючи різні методи.

x=[2.5, 3.7, 8.4, 11.7, 20, 27, 38];
y=[1.4, 2.7, 5.6, 7.5, 9.1, 13.2, 15.3];
xi=[3,4, 6, 10, 25, 30];
y1=interp1(x, y, xi, 'linear');
y2=interp1(x, y, xi, 'nearest');
y3=interp1(x, y, xi, 'cubic');
y4=interp1(x, y, xi, 'spline1');
plot (x,y,xi,y1,xi,y2, xi,y3,xi,y4)



Інтерполяція, наближена у вузлах. Поліноміальна апроксимація

Апроксимація поліномами в середовищі MATLAB здійснюється за допомогою функції polyfit(), яка має вигляд

polyfit (x, y, n)
де:
- x — вектор вузлів інтерполяції;
- y — вектор значень функції у вузлах інтерполяції;
- n — ступінь полінома.

Відгуком при реалізації функції polyfit () є вектор коефіцієнтів полінома. Функція *y = f(x)* може бути також представлена в аналітичному вигляді. Приведемо приклад інтерполяції за допомогою функції polyfit ().

Приклад1.

Функція задана таблично.

S	30	40	50	60	70	80	90	100	ПО	120
G	6.36	6.85	7.34	7.84	8.08	8.32	8.57	8.70	882	8.04

Тоді процедура інтерполяції будуть мати вигляд

x=[30,40,50,60,70,80,90,100,110,120];
в =[6.36, 6.85, 7.34, 7.84, 8.08, 8.32, 8.57, 8.7, 8.82, 8.94] ;
P=polyfit(x, в, 2)

Після натискання клавіші <Enter> відповідь одержимо в наступному вигляді

P = -0.003   0.0719   4.4747

Тоді функцією інтерполяції буде наступний поліном другого порядку

$$\varphi(G) = 4.4747 + 0.0719S - 0.003S^2 \, ..$$

Перевіримо правильність отриманих розв'язків за допомогою функції *polyval (P,x).*

F= polyval (P,x)

F = 6.3695   6.884   7.34   7.7373   8.0761   8.3564   8.5780 8.7412 8.8457   8.8917.

Інтерполяція кубічними поліномами.

Інтерполяція кубічними поліномами реалізується за допомогою функції icubic (), що має вигляд

yi = icubic (x, y, xi)
де:

- x, y — аргумент і функція, задані у вигляді векторів, або у вигляді вектора, а y- у вигляді формули;

- xi — вектор значень аргументу, для яких обчислюється значення функції; xi повинне бути в діапазоні значень x.

Приклад1.

Функція задана у вигляді таблиці.

x	1	3	6	9	12	15	18
y	I	2	5	5	!0	14	19

Необхідно знайти значення функції при значеннях аргументу x = 2, 5, 7, 13, 17.

x=[1 3 6 9 12 15 18];
в=[1 2 5 8 10 14 19];
xi=[2 5 7 13 17];
yi=icubic(x, в, xi)

Після натискання клавіші <Enter> одержимо розв'язок у наступному вигляді

yi = 1.1246 3.0928 5.3590 10.7824 17.1211

Завдання.

1. Знайти розв'язки диференціального рівняння *2y'- 4cosy+3y=5* використовуючи методи Рунге-Кутта 2 і 4 порядків. Побудувати графіки для знайдених розв'язків. Зрівняти їх і з'ясувати чому вони відрізняються.

2. Нехай функція задана таблично

x	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2
y	-1.1	0.2	0.5	0.8	0.7	0.6	0.4	0.1

Провести інтерполяцію кубічними сплайнами для значень x = -0.5 : 0.01 : 0.2.

3. Нехай функцією є *y=(cosx+x)/2*, задана при x = 1, 5, 10, 15, 20. Використовуючи кубічні сплайны, необхідно знайти її значення при x=2.5, 7.5, 12.5, 17.5, 22.5.

4. Функція y=f(x) задана таблично

x	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2
y	-1.1	0.2	0.5	0.8	0.7	0.6	0.4	0.1

5. Необхідно визначити значення функції при значеннях аргументів x = -0.5 : 0.01 : 0.2, використовуючи різні методи (інтерполяція точна у вузлах).

6. Апроксимувати функцію, задану у вигляді таблиці поліномами від 1 до 5 ступеня. Перевірити правильність отриманих розв'язків за допомогою функції polyval (P,x).

x	1	2	3	4	5	6	7	8
y	-1.1	0.2	0.5	0.8	0.7	0.6	0.4	0.1

Практичне заняття № 6. Застосування прямого й зворотного перетворення Фур'є для спектрального аналізу

Мета. Одержати практичні навики використання функцій Matlab для прямого й зворотного перетворення Фур'є для спектрального аналізу.

Теоретична частина. Деякі важливі універсальні процедури в Matlab використовують у якості змінного параметра ім'я функції, з якою вони оперують. Такі процедури називаються функціями функцій.

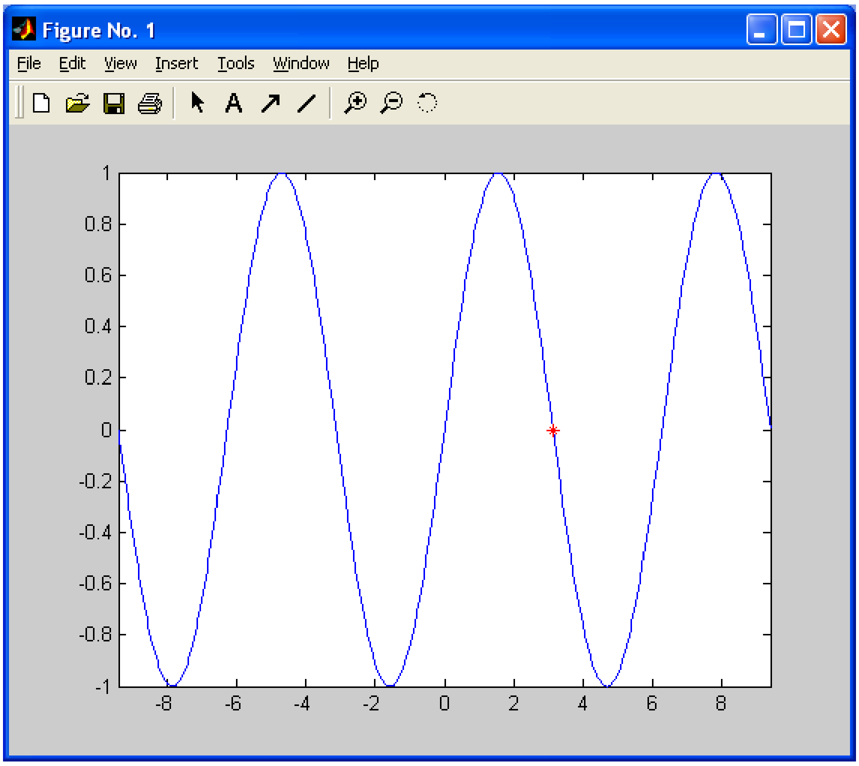
Прикладами можуть бути:

- знаходження розв'язків диференціальних рівнянь із використанням функцій ode23, ode113, ode45;
- обчислення мінімумів, і коренів (нулів) функцій одного аргументу.

Приклад 1.

Знайти корінь функції y=sin(x) в околиці точки x = π. На проміжку від -3π до +3π. Побудувати графік функцій з виділенням кореня червоним кольором.

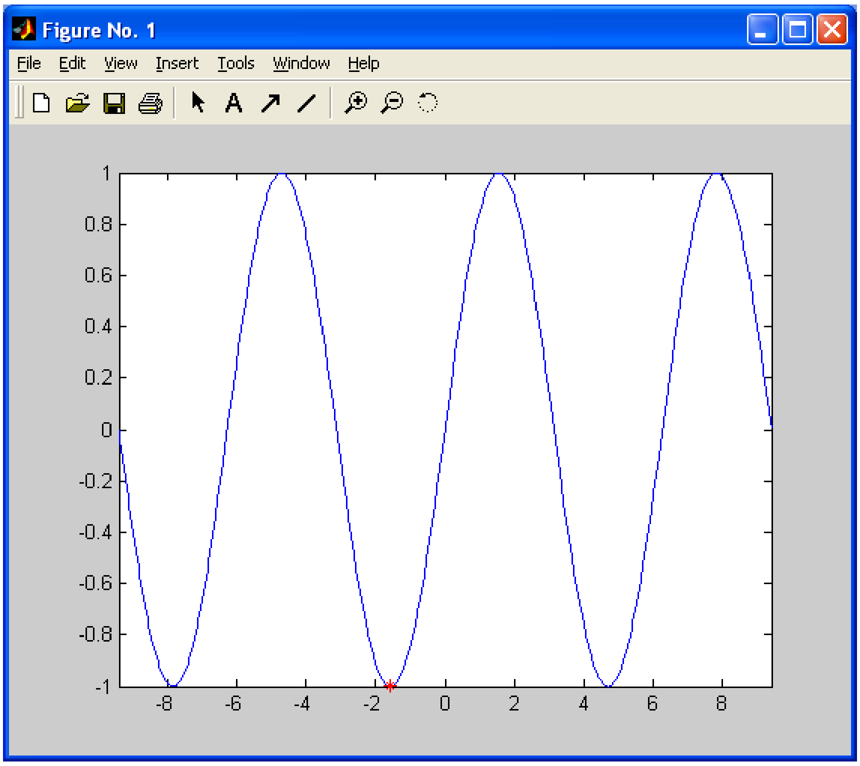
```
z=fzero('sin(x)', pi);
fplot('sin(x)',[-3*pi, 3*pi]);
hold on;
plot(z,0,'r*');
hold off
```



Приклад2.

Знайти локальний мінімум функції y=sin(x) на проміжку від -3π до +3π із кроком π/20. Побудувати графік функцій з виділенням мінімуму червоним кольором.

*z=fmin('sin(x)', -3*pi,3*pi);*
*fplot('sin(x)',[-3*pi, 3*pi]);*
hold on;
plot(z, sin(z),'r');*
hold off



У системі Matlab є функції, спеціально призначені для проведення спектрального аналізу різних сигналів.

Для одержання частотного спектра сигналу використовується функція fft, що реалізує пряме швидке перетворення Фур'є.

А для відновлення вихідного сигналу із частотного спектра використовується зворотне перетворення Фур'є, реалізоване функцією ifft.

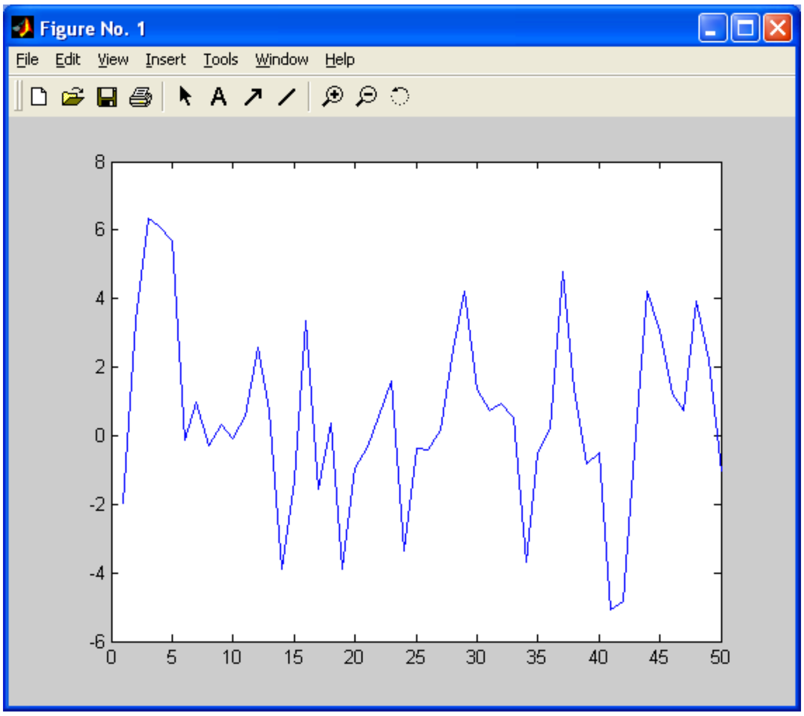
<u>Приклад 3.</u>

Даний приклад покаже використання функції fft для спектрального аналізу.

Використання fft виділяє частотні складові заданого сигналу, що зберігаються в часовій області сигналу. Для цього:

- сформуємо часову вісь для нашого сигналу в межах від 0 до 2 секунд із кроком 1 милісекунда;
- сформуємо вхідний сигнал, що представляє собою суму двох синусоїд із частотами 50 і 120 Гц. І зробимо сигнал зашумленным (додаємо випадкову складову функцією randn);
- побудуємо графік сигналу;
- знайдемо частотний спектр сигналу;
- по отриманому частотному спектру відновимо вихідний сигнал.

t = 0:.001:2;
*x = sin(2*pi*50*t) + sin(2*pi*120*t);*
*y = x+2*randn(size(t));*
plot(y(1:50))



За графіком дуже складно визначити частотні складові, тому застосуємо спектральний аналіз

Y = fft(y, 256);

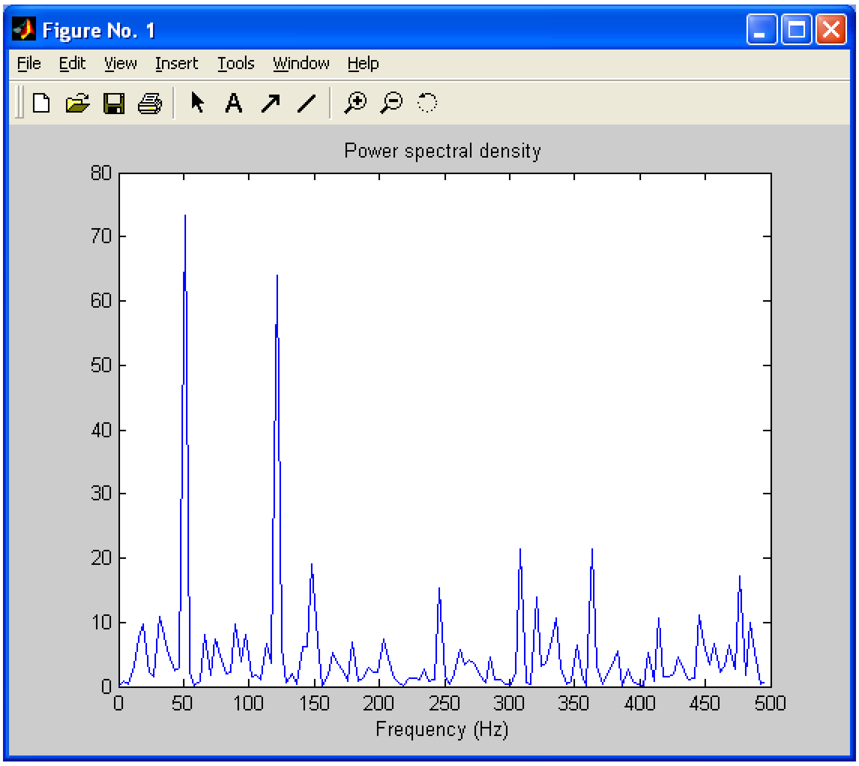
Обчислюємо спектральну щільність енергії сигналу по різних частотах (Pyy), використовуючи комплексно сполучені значення Y

*Pyy = Y.*conj(Y)/256;*

Сформуємо частотну вісь для дискретних значень аргументу швидкого перетворення Фур'є (для 128 значень, тому що для інших значень аргументу спектральна щільність енергії симетрична).

Будуємо графік залежності спектральної щільності енергії від частоти

f = 1000/256(0:127);*
plot(f,Pyy(1:128))
title('Power spectral density')
xlabel('Frequency (Hz)')



На графіку бачимо, що помітні піки в області частот 50 і 120 Гц. Це первісні частоти синусоїд нашого сигналу.

Завдання.

1. Знайти корінь функції $y=(\cos(x/\pi+\pi)+\sin(x))/2$ в околиці точки x=-π. Побудувати графік цієї функції на проміжку від -3π до +3π.

2. Знайти локальний мінімум функції $y=\cos(x/\pi)+2$ на проміжку:

а) від -3π до +3π

б) від 0 до 3π

в) від -3π до 0.

Побудувати графіки функцій з виділенням мінімумів зеленим кольором.

3. Сформувати сигнал, що представляє собою суму синусоїди із частотою ω = 80 Гц і косинусоиды із частотою 150 Гц (вивести графік сигналу)

Зробити сигнал зашумленным (вивести графік сигналу).

За допомогою прямого швидкого перетворення Фур'є визначити частотні складові сигналу (вивести частотний спектр сигналу).

Відновити сигнал використовуючи зворотне перетворення Фур'є (вивести графік відновленого сигналу).

Практичне заняття № 7. Режими програмування в Matlab

Мета. ознайомлення із застосуванням команд керування потоками в системі Matlab.

Теоретична частина. Робота в режимі калькулятора в середовищі Matlab, незважаючи на досить значні можливості, має істотні недоліки. Неможливо повторити всі попередні обчислення й дії при нових значеннях вихідних даних без повторного набору всіх попередніх операторів. Не можна повернутися назад і повторити деякі дії або по деякій умові перейти до виконання іншої послідовності операторів. І взагалі, якщо число операторів велике, стає проблемою налагодити правильну їхню роботу через неминучі помилки при наборі команд.

Тому складні, з перериваннями, заплутаними переходами по певних умовах, із часто повторюваними однотипними діями обчислення, які, до того ж, необхідно проводити неоднаразово при змінених вхідних даних, вимагають їхнього спеціального оформлення у вигляді записаних на диску файлів, тобто у вигляді програм. Перевага програм у тому, що стає можливим кількаразове звертання до тим самим операторів і до програми в цілому. Створення програм дозволяє значно спростити й скоротити процес підготовки повторюваних обчислень, зробити процес обчислень більш наочним і прозорим, а завдяки цьому — різко зменшити ймовірність появи принципових помилок при розробці програм. Крім того, у програмах з'являється можливість автоматизувати й процес зміни значень вхідних параметрів у діалоговому режимі.

У мові MATLAB є програми 2 типів: Script-файли (керуючі програми) і файли - функції (процедури).

За допомогою Script-файлів оформляються основні програми, що управляють від початку до кінця організацією обчислювального процесу.

Файлами - функціями оформляються окремі процедури й функції (тобто такі частини програми, які розраховані на кількаразове використання при змінюваних вхідних параметрах).

Файл-функція (процедура) повинна починатися з рядка заголовка

function [<ПКВ>]=<ім'я процедури>(<ПВВ>),

де ПКВ - перелік кінцевих величин, ПВВ - перелік вхідних величин.

Для переходу в режим програмування у вікні керування вибираємо пункт File і входимо в редактор: *New* (створити новий m-файл) або *Open* (відкрити існуючий файл із розширенням .m).

Надалі може проводитися звичайний набір тексту програми або його коректура й дії відповідно до меню (збереження під поточним або іншим іменем, запуск на виконання у звичайному й отладочному режимах і ін.).

MATLAB має п’ять видів структур керування потоками:

- onepaгop if;
- onepaгop *switch;*
- цикли *for;*
- цикли *while;*

- оператор *break.*

Оператор *if* розраховує логічний вираз і виконує групу операторів, якщо вираз істинний. Необов'язкові ключові слова *elseif* і *else* служать для виконання альтернативних груп операторів Ключове слово *end,* яке узгодиться з *if,* завершує останню групу операторів. Таким чином, усі групи операторів укладені між чотирьох ключових слів, без використання фігурних або звичайних дужок.

Умовний оператор виступає в одній з наступних форм

```
if <умова>
        <команди>
end
```

```
if <умова>
        <команди>
else
        <команди>
end
```

```
if <умова>
        <команди>
elseif <умова>
        <команди>
else
        <команди>
end
```

У ролі умови може використовуватися будь-який логічний вираз, побудований на основі операцій відносин й логічних операцій. Якщо значення цього виразу є масивом, то умова вважається дійсним, якщо всі його елементи дійсні (істина – 1, неправда – 0).

Приклад 1.

```
if rem(n,2) = 0
M = odd_magic(n)
else
M = double_even_magic(n)
end
```

Приклад 2.

```
if A > B
' greater '
elseif A < B
' less'
elseif A == B
' equal '
else
error ( ' Непередбачена ситуація' )
end
```

Оператор перемикання узагальнює умовний оператор на випадок більше двох умов і має конструкцію

```
switch <вираз>
case <значення 1>
<команди>
```

case <значення 2>
<команди>
otherwise % може бути відсутнім
<команди>
end

Приклад 3.

```
switch k
case 0
t=1
case (1, 2,5)
t=2
otherwise
t=0
end
```

Оператор циклу із заданим числом повторень, в основному використовуваний у формах

```
for V=A:H:B
    <команди>
end
```

```
for V=A:B
<команди>
end
```

Приклад 4.

```
for i=1:n
for j=1:m
a(i,j)=x(i)^j;
end
end
```

Приклад 5.

```
for i=1:n-1
for k=i+1:n
if a(i)<a(k)
m=a(i)
a(i)=a(k)
a(k)=m
end
end
```

end

Приклад 6.

```
        k=1;
for i=[0 5 7]
        x(k)=2^i;
        k=k+1;
end;
        X=1 32 128
```

Оператор циклу із передумовою while.

Цикл *while* повторює оператори певне число раз, поки виконується логічна умова. Ключове слово *end* окреслює використовувані one раторы.

Оператор циклу із предусловием має традиційну конструкцію

while <умова>
<команди>
end

і забезпечує виконання команд тіла циклу, поки істинно перевіряєма умова.

Приклад 7.

```
while a<1
n=n+1
end
```

Робота будь-якого циклу може бути перервана (вихід із внутрішнього або зовнішнього циклу) оператором break

Приклад 8.

```
while a<1
n=n+1
if n>250
break
end
```

Вихід з функції в програму забезпечується виконанням останнього її оператора або командою return.

Крім згаданих основних операторів, традиційних для будь-якої системи програмування, зупинимося на ряді операторів забезпечення користувацького інтерфейсу.

Введення із клавіатури реалізується командою виду

<змінна>= input (‘підказка’)

Припинення виконання програми може бути передбачена включенням у текст команди:

pause (припинення до натискання будь-якої клавіші),

pause (n) (припинення на n сек),

keyboard (припинення з можливістю виконувати практично будь-які команди з наступним поверненням у програму командою return).

Можна побудувати вибір варіанта із клавіатури створенням меню

<змінна>=menu('заголовок','вибір1','вибір2',...)

Наприклад, команда

k=menu('Використовувати метод','Гаусса','Краута','Простиї ітерації')

створить на екрані спливаюче меню із зазначеними пунктами клавішами й клацання по клавіші задасть значення змінної k, рівне 1, 2 або 3.

Завдання.

1. Є 3 змінні a, b і i.

b=15.

Змінна a=i/2.

Змінна i=1. На кожному кроці i збільшується на 1.

Визначити число кроків, за яке, a досягнеться більшого, ніж b значення.

2) Є 2 змінні n і m.

Змінна n може приймати одне із двох значень 0 (m=n) або 1 (m=n+n/2)

Використовуючи оператор перемикання для змінної n, визначити значення змінної m у кожному із цих випадків.

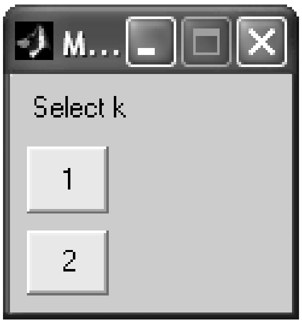
3. Дана матриця розміром n x m. Зробити підсумовування всіх елементів матриці крім елементів останнього стовпця й останнього рядка (використовуючи вкладені цикли).

4. Створюємо функцію залежності y від k і t.

Якщо k=1, то y=(k*t)/2

Якщо k=2, то y=0

Створюємо меню вибору значення змінної k (може приймати значення 1 або 2).



Вибираємо одне зі значень.

Викликаємо m-файл, що зберігає нашу функцію. Значення параметра t=1.

5. Створити файл-функцію для розрахунків факторіала числа 8.

СПИСОК ДЖЕРЕЛ

1. Бороденко В.А. Применение ЭВМ в энергетике / В.А. Бороденко. – Павлоград: ПГУ, 2002. – 36 с.

2. Гарнаев А. Ю. Ms Excel' 2002: разработка приложений / А.Ю. Гарнаев. – Спб.: БХВ – Петербург, 2003. – 768 с.

3. Советов Б. Я., Цехановский В. В. Информационные технологии / Б.Я. Советов, В.В. Цехановский. – М.: Высшая школа, 2003. – 263 с.

4. Компьютерные информационные технологии в электроэнергетике: учеб. пособие / И.Г. Абраменко, О.Г. Гриб, О.Н. Довгалюк, Н.П. Пан. - Харьков: ХГАГХ, 2003. – 176с.