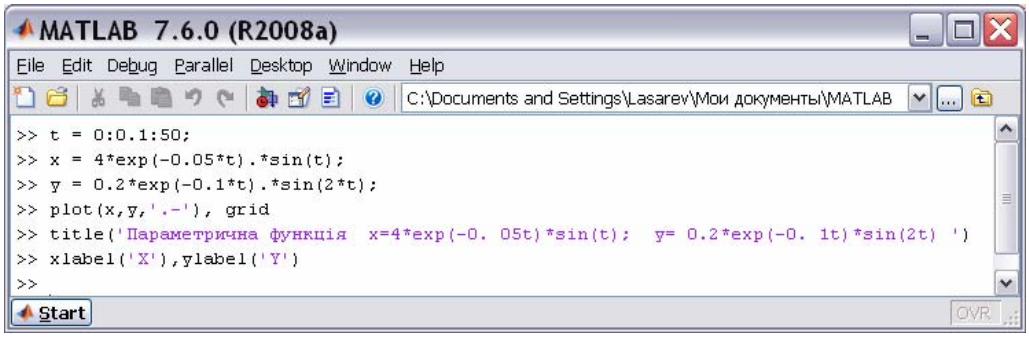
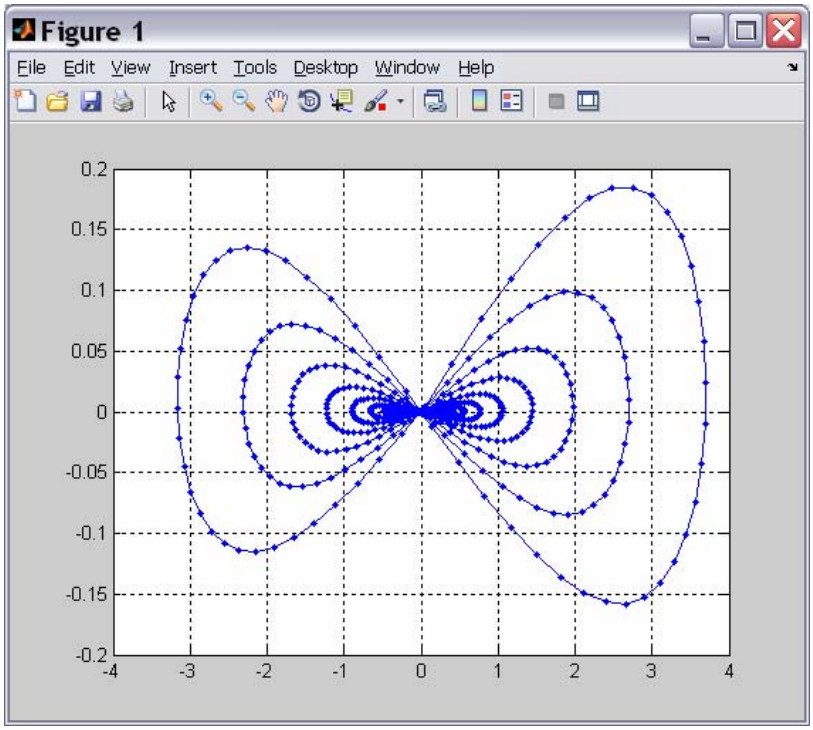
Ю. Ф. Лазарєв

MATLAB і моделювання динамічних систем

Навчальний посібник

Глава 1. Вступ у Matlab





Київ - 2009

УДК 681.3(0.75)
Л17

Лазарєв Ю. Ф.
Л17 MATLAB і моделювання динамічних систем. Навчальний посібник. Глава 1. Вступ у Matlab. – Київ: НТУУ "КПІ", 2009. – 81 с.

Зміст

1. ВСТУП У MATLAB

Комп'ютерна система Matlab наразі широко розповсюджена в інженерних і університетських колах завдяки видатним перевагам, до яких відносяться:

1) простота опановування і досяжність текстів практично усіх програмних засобів, окрім вбудованих;

2) велика бібліотека досяжних математичних програм, яка містить майже всі сучасні чисельні методи і функції;

3) можливість утворювати власні програмні засоби і навіть корегувати існуючі;

4) вельми зручний і пристосований для практичних потреб інженерів і науковців апарат графічного оформлення результатів обчислень;

5) наявність інтегрованого пакету програм ***Simulink*** візуального програмування, який дозволяє суттєво спростити процес утворення складних програм і автоматизувати процес організації чисельного інтегрування диференційних рівнянь досліджуваної системи.

Ознайомленню з головними особливостями і можливостями Matlab і присвячена ця глава.

1.1. Командне вікно

Після виклику Matlab із середовища Windows на екрані виникає зображення так званого "командного вікна" середовища Matlab (рис. 1.1).

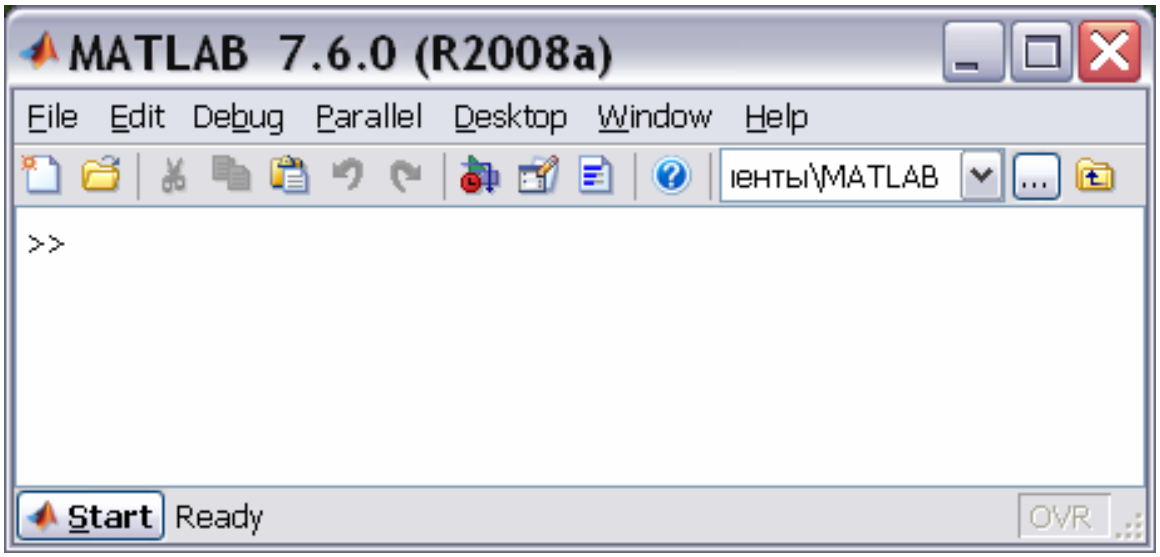


Рис. 1.1. Вид командного вікна Matlab

Це вікно є основним у Matlab. У ньому з'являються символи команд, що набираються користувачем на клавіатурі дисплея, відображуються результати виконання цих команд, текст програми, що виконується, і інформація про помилки виконання програми, розпізнані системою .

Ознакою того, що Matlab готова до сприйняття і виконання чергової команди, є поява в останньому рядку текстового поля вікна знака запрошення (»), після якого миготить вертикальна риса.

У верхній частині вікна (під заголовком) розміщений рядок меню, в якому містяться меню ***File***, ***Edit***, ***View, Windows***, ***Help***. Щоб відчинити якесь меню, потрібно встановити на ньому курсор миші і натиснути її ліву клавішу. Докладніше функції команд меню описані далі, у главі 3 "Команди загального призначення. Написання М-книг".

Тут відзначимо лише, що *для виходу із середовища Matlab* достатньо відчинити меню ***File*** і обрати у ньому команду ***Exit MATLAB*****,** або просто зачинити командне вікно, натиснувши ліву клавішу миші, коли курсор миші встановлений на зображенні верхньої крайньої правої кнопки цього вікна (з позначенням хрестика)**.**

1.2. Операції з числами

1.2.1. Введення дійсних чисел

Введення чисел із клавіатури здійснюється по загальних правилах, прийнятих для мов програмування високого рівня:

для відділення дробової частини мантиси числа застосовується десяткова крапка (замість коми при звичайному записі);

десятковий показник числа записується у вигляді цілого числа після попереднього запису символу "e";

між записом мантиси числа й символом "e" (який відокремлює мантису від показника) не повинно бути ніяких символів, включаючи і символ пропуску.

Якщо, наприклад, ввести в командному вікні Matlab рядок

1. 20357651e -17,

то після натискання клавіші <Enter> у цьому вікні з'явиться запис:

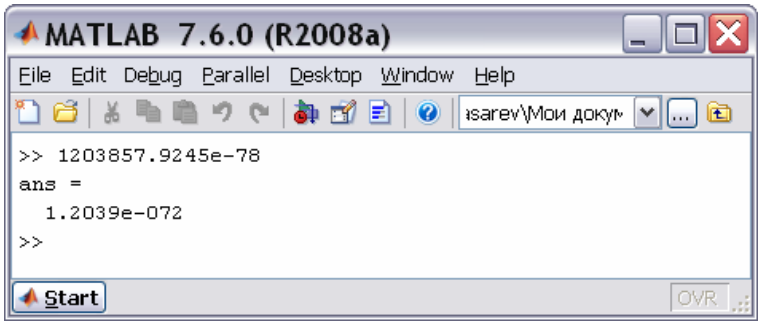


Рис. 1.2. Введення і відображування чисел

Слід зазначити, що результат виводиться у виді (форматі), що визначається попередньо встановленим форматом подання чисел. Цей формат може бути встановлений за допомогою команди ***Preferences*** меню ***File*** (рис. 1.3).

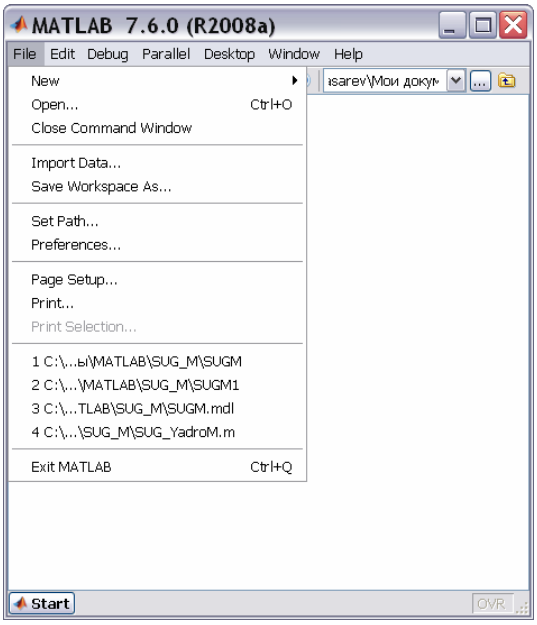


Рис. 1.3. Меню "File"

Після її виклику на екрані з'явиться однойменне вікно (рис. 1.4). У поділі ***Command Window*** одна з ділянок цього вікна має назву ***Numeric Format***. Її призначено для встановлення і змінювання формату подання чисел, які виводяться в командне вікно в процесі розрахунків. Передбачені такі формати:

Short (default) - стислий запис (застосовується за умовчанням);
Long - довгий запис;
Hex - запис у виді шістнадцяткового числа;
Bank - запис до сотих часток;
Plus - записується тільки знак числа;
Short E - стислий запис у форматі із плаваючою комою;
Long E - довгий запис у форматі із плаваючою комою;
Short G - друга форма стислого запису у форматі з плаваючою комою;
Long G - друга форма довгого запису у форматі з плаваючою комою;
Rational - запис у вигляді раціонального дробу.

Обираючи за допомогою мишки потрібний вид подання чисел, можна забезпечити надалі виведення чисел у командне вікно саме в цій формі.

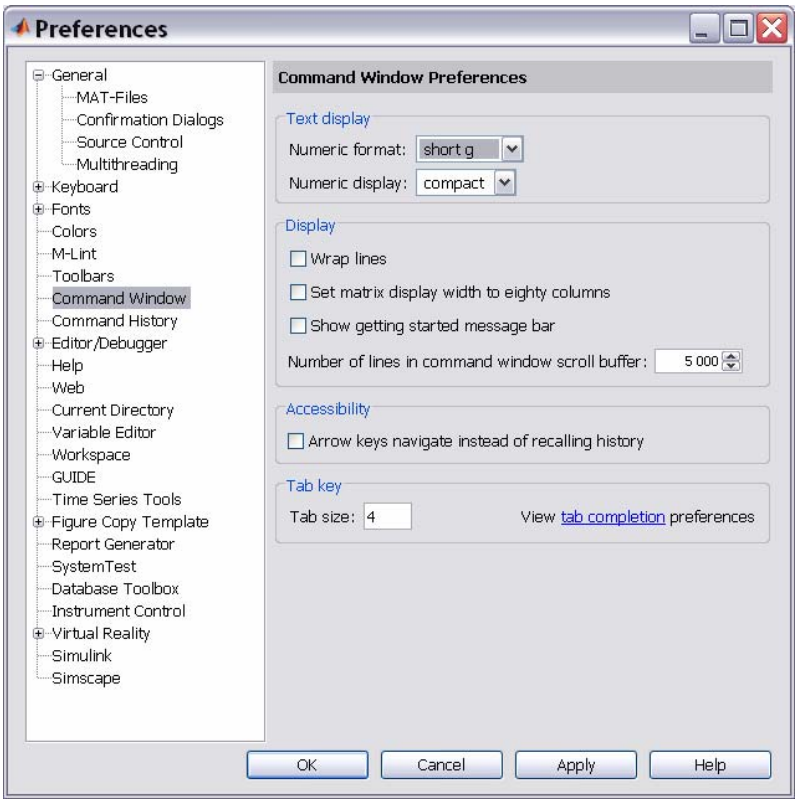


Рис. 1.4. Вікно Preferences меню "File"

Як видно з рис. 1.2, число, яке виведено на екран, не збігається з введеним. Це обумовлено тим, що встановлений за замовчуванням формат подання чисел (Short G) не дозволяє вивести більше 5 значущих цифр числа. Насправді введене число усередині Matlab зберігається з усіма введеними його цифрами. Наприклад, якщо обрати мишкою селекторну кнопку **Long E** (тобто установити цей формат подання чисел), то, повторюючи ті ж дії, одержимо (рис. 1.5):

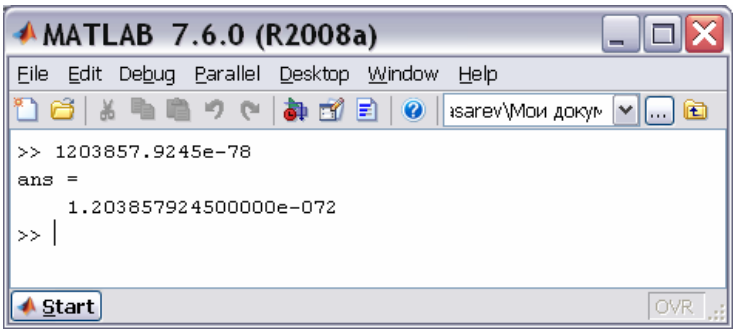


Рис. 1.5. Виведення числа у форматі Long E

де вже всі цифри відображені вірно.

Слід пам'ятати:

- *уведене число й результати всіх обчислень у системі Matlab зберігаються в пам'яті ПК із відносною похибкою біля 2.10^{-16} (тобто з точними значеннями в 15 десяткових розрядах);*

- *діапазон подання модуля дійсних чисел лежить у проміжку між* 10^{-308} *і* 10^{+308}.

1.2.2. Найпростіші арифметичні дії

В арифметичних виразах мови Matlab використовуються такі знаки арифметичних операцій:

\+ - додавання;
\- - віднімання;
* - множення;
/ - ділення зліва праворуч;
\ - ділення справа ліворуч;
^ - піднесення до степеня.

Використання Matlab у режимі калькулятора може відбуватися шляхом простого запису в командний рядок послідовності арифметичних дій з числами, тобто звичайного арифметичного виразу, наприклад:

(4.5)^2*7.23 - 3.14*10.4.

Якщо після введення із клавіатури цієї послідовності натиснути клавішу <Enter>, у командному вікні виникне результат виконання у виді, поданому на рис. 1.6, тобто на екран під ім'ям системної змінної ***ans*** виводиться результат дії останнього виконаного оператора.

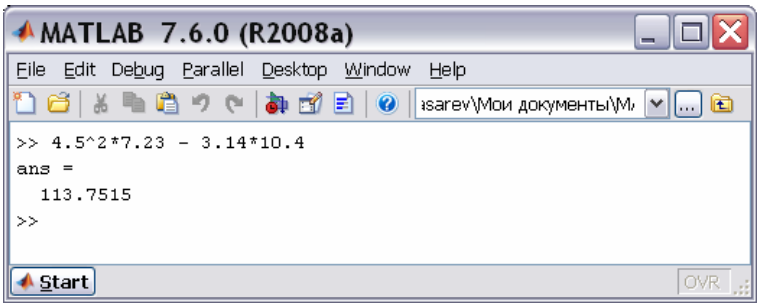


Рис. 1.6. Введення арифметичного виразу і відображення результату

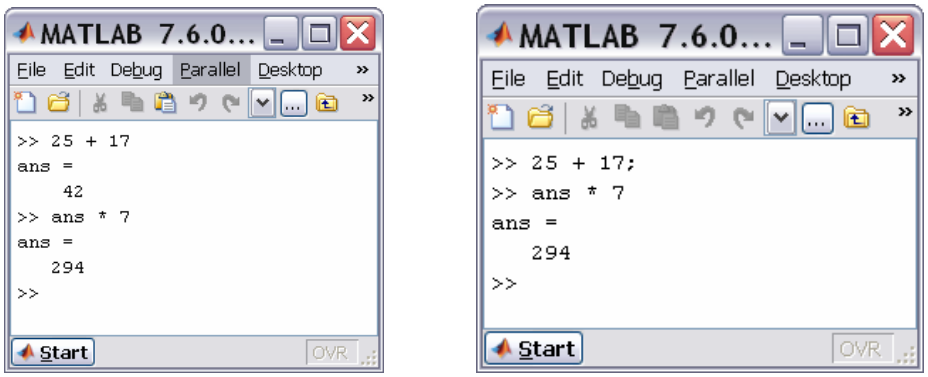
Взагалі виведення проміжної інформації у командне вікно підпорядковується таким правилам:

- *якщо запис оператора не закінчується символом ';', результат дії цього оператора одразу ж виводиться в командне вікно;*
- *якщо оператор закінчується символом ';', результат його дії не відображується в командному вікні;*
- *якщо оператор не містить знака присвоювання (=), тобто є просто записом деякої послідовності дій над числами і змінними, значення результату присвоюється спеціальній системній змінній за ім'ям* ***ans****;*
- *отримане значення змінної* ***ans*** *можна використовувати в наступних операторах обчислень, використовуючи це ім'я* ***ans****; при цьому варто пам'ятати, що значення системної змінної* ***ans*** *змінюється після дії чергового оператора без знака присвоювання;*
- *у загальному випадку форма подання результату в командне вікно має вид:*

<Ім'я змінної> = <результат>.

Приклад.

Нехай потрібно обчислити вираз (25+17)*7. Це можна зробити таким чином. Спочатку набираємо послідовність **25+17** і натискаємо <Enter>. Одержуємо на екрані результат у виді **ans = 42**. Тепер записуємо послідовність **ans*7** і натискаємо <Enter>. Одержуємо **ans = 294** (рис. 1.7а). Щоб запобігти виведення проміжного результату дії 25+17, достатньо після запису цієї послідовності додати символ ' ; '. Тоді будемо мати результати у виді, поданому на рис. 1.7б.



а) *б)*

Рис. 1.7. Використання системної змінної ***ans***

Особливістю Matlab як калькулятора є можливість використання імен змінних для запису проміжних результатів у пам'ять ПК. Для цього застосовується операція присвоювання, що вводиться знаком рівності ' = ' у відповідності зі схемою :

<Ім'я змінної> = <вираз>[;]

Ім'я змінної може містити до 30 символів і повинно не збігатися з іменами функцій, процедур системи і системних змінних. При цьому система роз-

різнює великі й малі букви в змінних. Так, імена 'amenu' , 'Amenu', 'aMenu' у Matlab позначають різні змінні.

Вираз справа від знака присвоювання може бути просто числом, арифметичним виразом, рядком символів (тоді ці символи потрібно укласти в апострофи) або символьним виразом. Якщо вираз не закінчується символом ' ; ', після натискання клавіші <Enter> у командному вікні виникне результат виконання у виді :

<Ім'я змінної> = <результат>.

Наприклад, якщо ввести в командне вікно рядок 'x = 25 + 17', на екрані то виникне запис (рис. 1.8) :

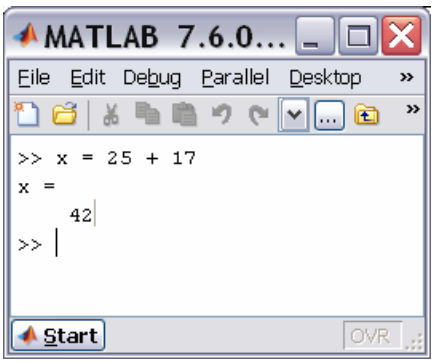


Рис. 1.8. Присвоювання значення змінній

Система Matlab має кілька імен змінних, що використовуються самою системою і входять до складу зарезервованих:

i, ***j*** - уявна одиниця (корінь квадратний з -1);

pi - число π (зберігається у виді 3.141592653589793);

inf - позначення машинної нескінченності;

NaN - позначення невизначеного результату (наприклад, типу 0/0 або inf/inf);

eps - похибка операцій над числами із плаваючою комою

ans - результат останньої операції без знака присвоювання;

realmax – максимальна величина числа, що може бути використана;

realmin – мінімальна величина числа, що може бути використана.

Ці змінні можна використовувати в математичних виразах.

1.2.3. Введення комплексних чисел

Мова системи Matlab, на відміну від багатьох мов програмування високого рівня, містить у собі дуже просту в користуванні вбудовану арифметику комплексних чисел. Більшість елементарних математичних функцій побудовано таким чином, що аргументи припускаються комплексними числами, а результати також формуються як комплексні числа. Ця особливість мови робить її дуже привабливою й корисною для інженерів і науковців.

Для позначення уявної одиниці в мові Matlab зарезервовано два ймення ***i*** і ***j***. Уведення із клавіатури значення комплексного числа здійснюється шляхом запису в командне вікно рядка виду:

*<ім'я комплексної змінної> = <значення ДЧ> + i [j] *<значення УЧ>,*

де ДЧ - дійсна частина комплексного числа, УЧ - уявна частина. Приклад наведений на рис. 1.9:

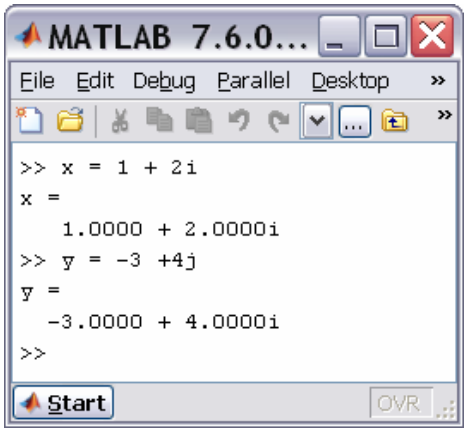


Рис. 1.9. Введення комплексних чисел і виведення результату

З наведеного прикладу очевидно, у якому виді система виводить комплексні числа на екран (і "до друку").

1.2.4. Елементарні математичні функції

Загальна форма використання функції у Matlab така:

<ім'я результату> = <ім'я функції>(<перелік аргументів або їх значень>).

У мові Matlab передбачені наступні елементарні арифметичні функції.

Тригонометричні й гіперболічні функції

sin(Z) - синус числа Z;
sinh(Z) - гіперболічний синус;
asin(Z) - арксинус (у радіанах, у діапазоні від $-\pi/2$ до $+\pi/2$);
asinh(Z) - обернений гіперболічний синус;
cos(Z) - косинус;
cosh(Z) - гіперболічний косинус;
acos(Z) - арккосинус (у діапазоні від 0 до π);
acosh(Z) - обернений гіперболічний косинус;
tan(Z) - тангенс;
tanh(Z) - гіперболічний тангенс;
atan(Z) - арктангенс (у діапазоні від $-\pi/2$ до $+\pi/2$);
atan2(X,Y) - чотириквадрантний арктангенс (кут у діапазоні від $-\pi$ до $+\pi$ між горизонтальним правим променем і променем, що проходить через точку з координатами X і Y);
atanh(Z) - обернений гіперболічний тангенс;
sec(Z) - секанс;

sech(Z) - гіперболічний секанс;
asec(Z) - арксеканс;
asech(Z) - обернений гіперболічний секанс;
csc(Z) - косеканс;
csch(Z) - гіперболічний косеканс;
acsc(Z) - арккосеканс;
acsch(Z) - обернений гіперболічний косеканс;
cot(Z) - котангенс;
coth(Z) - гіперболічний котангенс;
acot(Z) - арккотангенс;
acoth(Z) - обернений гіперболічний котангенс.

Експоненціальні функції

exp(Z) - експонента числа Z;
log(Z) - натуральний логарифм;
log10(Z) - десятковий логарифм;
sqrt(Z) - квадратний корінь із числа Z;
abs(Z) - модуль числа Z.

Цілочислові функції

fix(Z) - округлення до найближчого цілого убік нуля;
floor(Z) - округлення до найближчого цілого убік від'ємної нескінченності;
ceil(Z) - округлення до найближчого цілого убік додатної нескінченності;
round(Z) - звичайне округлення числа Z до найближчого цілого;
mod(X,Y) - цілочислове ділення ділення X на Y;
rem(X,Y) - обчислення остачі від ділення X на Y;
sign(Z) - обчислення сигнум-функції числа Z
(0 при Z=0, -1 при Z<0, 1 при Z>0).

1.2.5. Спеціальні математичні функції

Крім елементарних у мові Matlab передбачений цілий ряд спеціальних математичних функцій. Нижче наведений перелік і стислий зміст цих функцій. Правила звернення до них і використання користувач може відшукати в описах цих функцій, що виводяться на екран, якщо набрати команду ***help*** і вказати в тому ж рядку ім'я функції.

Функції перетворення координат

cart2sph - перетворення декартових координат у сферичні;
cart2pol - перетворення декартових координат у полярні;
pol2cart - перетворення полярних координат у декартові;
sph2cart - перетворення сферичних координат у декартові.

Функції Бесселя

besselj - функція Бесселя першого роду;
bessely - функція Бесселя другого роду;
besseli - модифікована функція Бесселя першого роду;
besselk - модифікована функція Бесселя другого роду.

Бета-функції

beta - бета-функція;
betainc - неповна бета-функція;
betaln - логарифм бета-функції.

Гамма-функції

gamma - гамма-функція;
gammainc - неповна гамма-функція;
gammaln - логарифм гамма-функції.

Еліптичні функції й інтеграли

ellipj - еліптичні функції Якобі;
ellipke - повний еліптичний інтеграл;
expint - функція експоненціального інтегралу.

Функції похибок

erf - функція похибок;
erfc - додаткова функція похибок;
erfcx - масштабована додаткова функція похибок;
erfinv - обернена функція похибок.

Інші функції

gcd - найбільший загальний дільник;
lcm - найменше загальне кратне;
legendre - узагальнена функція Лежандра;
log2 - логарифм за основою 2;
pow2 - піднесення 2 до зазначеного степеня;
rat - подання числа у виді раціонального дробу;
rats - подання чисел у виді раціонального дробу.

1.2.6. Елементарні дії з комплексними числами

Найпростіші дії з комплексними числами - додавання, віднімання, множення, ділення й піднесення до степеня - здійснюються за допомогою звичайних арифметичних знаків +, -, *, /, \ і ^ відповідно.

Приклади використання наведені на рис. 1.11.

Примітка. У наведеному фрагменті використана функція ***disp*** (від слова 'дисплей'), яка також дозволяє виводити в командне вікно результати обчислень або деякий текст. При цьому чисельний результат, як очевидно, виводиться вже без указівки ймення змінної або ***ans***.

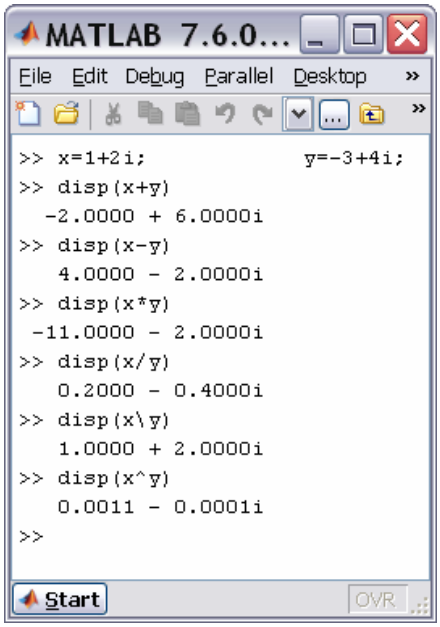


Рис. 1.10. Приклади використання функції ***disp***

1.2.7. Функції комплексного аргументу

Практично *всі* елементарні *математичні функції*, наведені в п. 1.2.4, *обчислюються при комплексних значеннях аргументу* й одержують у результаті цього комплексні значення результату.

Завдяки цьому, наприклад, функція ***sqrt*** обчислює, на відміну від інших мов програмування, квадратний корінь із від'ємного аргументу, а функція ***abs*** при комплексному значенні аргументу обчислює модуль комплексного числа. Приклади наведені на рис. 1.11.

У Matlab є кілька додаткових функцій, розрахованих тільки на комплексний аргумент:

real(Z) - виділяє дійсну частину комплексного аргументу Z;

imag(Z) - виділяє уявну частину комплексного аргументу;

angle(Z) - обчислює значення аргументу комплексного числа Z (у радіанах від $-\pi$ до $+\pi$);

conj(Z) - видає число, комплексно спряжене щодо Z.

Приклади наведені на рис. 1.12.

Крім того, у Matlab є спеціальна функція ***cplxpair***(V), що здійснює сортування заданого вектора V із комплексними елементами у такий спосіб, що комплексно-спряжені пари цих елементів розташовуються у вихідному векторі в порядку зростання їхніх дійсних частин, при цьому елемент із від'ємною уявною частиною завжди розташовується першим. Дійсні елементи завершують комплексно-спряжені пари.

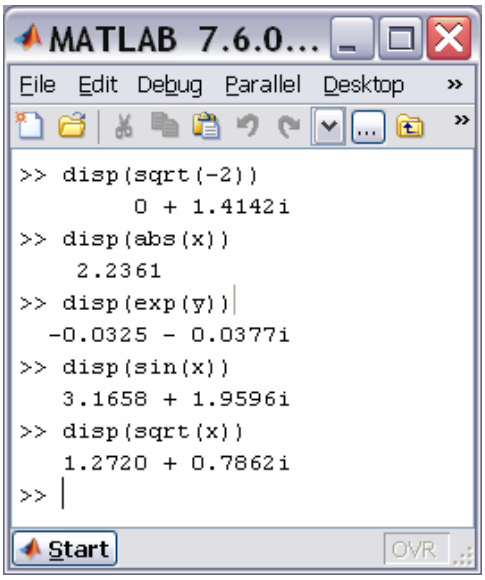


Рис. 1.11. Застосування функцій з комплексним аргументом

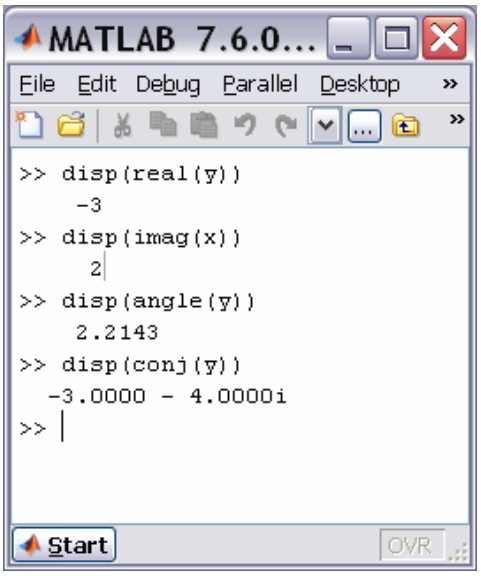


Рис. 1.12. Застосування функцій комплексного аргументу

Наприклад (*надалі в прикладах команди, що набираються із клавіатури, будуть написані масним шрифтом, а результат їхнього виконання - звичайним шрифтом*):

```
» v = [ -1, -1+2i,-5,4,5i,-1-2i,-5i]
v =
  Columns 1 through 4
  -1.0000          -1.0000 + 2.0000i                  -5.0000                   4.0000
  Columns 5 through 7
       0 + 5. 0000i          -1. 0000 - 2. 0000i       0 - 5. 0000i
» disp(cplxpair(v))
  Columns 1 through 4
  -1. 0000 - 2. 0000i  -1. 0000 + 2. 0000i       0 - 5. 0000i       0 + 5. 0000i
  Columns 5 through 7
  -5. 0000          -1. 0000          4. 0000
```

Пристосованість більшості функцій Matlab до оперування з комплексними числами дозволяє значно простіше будувати обчислення з дійсними числами, результат яких є комплексним, наприклад, знаходити комплексні корені квадратних рівнянь.

1.2.8. Знайомство з програмуванням в Matlab

Робота в режимі калькулятора в середовищі Matlab, незважаючи на досить значні можливості, має істотні незручності. Неможливо повторити всі попередні обчислення й дії при нових значеннях початкових даних без повторного набирання всіх попередніх операторів. Не можна повернутися назад і повторити деякі дії, або за деякою умовою перейти до виконання іншої послідовності операторів. І взагалі, якщо кількість операторів є значною, стає проблемою налагодити правильну їхню роботу через неминучі помилки при набиранні команд. Тому складні, із перериваннями, складними переходами по певних умовах, із часто повторюваними однотипними діями обчислення, які, до того ж, необхідно проводити неодноразово при змінених первинних даних, потребують

їхнього спеціального оформлення у виді записаних на диску файлів, тобто у виді програм. Перевага програм у тому, що, унаслідок того, що вони зафіксовані у виді записаних файлів, стає можливим багаторазове звернення до тих самих операторів і до програми в цілому. Це дозволяє спростити процес налагоджування програми, зробити процес обчислень більш наочним і прозорим, а завдячуючи цьому - різко зменшити можливість появи принципових помилок при розробці програм. Крім того, у програмах виникає можливість автоматизувати також і процес змінювання значень первісних параметрів у діалоговому режимі.

Створення програми в середовищі Matlab здійснюється за допомогою вбудованого редактора. Вікно цього вбудованого редактора виникає на екрані, якщо перед цим використано команду "M-file" із поділу *New* або обрано назву одного з існуючих М-файлів при виклику команди *Open M-file* із меню ***File*** командного вікна. У першому випадку вікно текстового редактора є порожнім (рис. 1.13), у другому - у ньому міститься текст викликаного М-файлу. В обох випадках вікно текстового редактора готове для введення нового тексту або коригування існуючого.

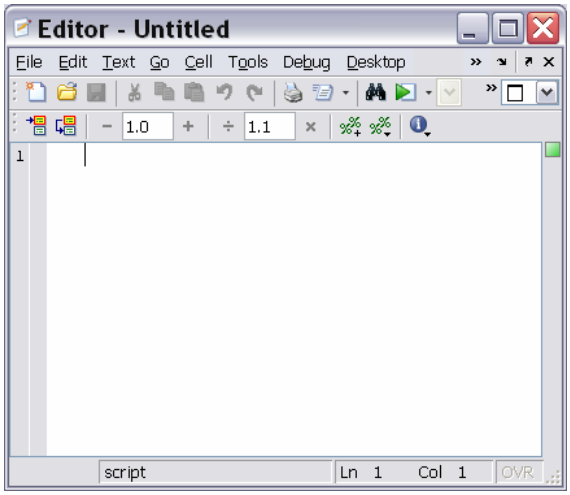


Рис. 1.13. Вікно вбудованого текстового редактора Matlab

Запис тексту програми (М-файлу) мовою Matlab має підпорядковуватися наступним правилам.

1. Зазвичай кожний оператор записується в окремому рядку тексту програми. Ознакою кінця оператора є символ (він не виникає у вікні) повернення каретки й переходу на наступний рядок, який вводиться в програму при натисканні клавіші <Enter>, тобто при переході при записі тексту програми на наступний рядок.

2. Можна розміщувати кілька операторів в одному рядку. Тоді попередній оператор у тому ж рядку має закінчуватися символом ' ; ' або ' , '.

3. Можна довгий оператор записувати в декілька рядків. При цьому попередній рядок оператора має завершуватися трьома крапками (' ... ').

4. Якщо черговий оператор не закінчується символом ' ; ', результат його дії при виконанні програми буде виведений у командне вікно. Щоб запобігти

виведенню на екран результатів дії оператора програми, запис цього оператора в тексті програми має закінчуватися символом ' ; '.

5. Рядок програми, що починається із символу ' % ', не виконується. Цей рядок сприймається системою Matlab як *коментар*. Тому для введення коментарю в будь-яке місце тексту програми достатньо почати відповідний рядок із символу ' % '.

6. Рядки коментарю, які передують першому виконуваному (тобто такому, що не є коментарем) оператору програми, сприймаються системою Matlab як опис програми. Саме ці рядки виводяться в командне вікно, якщо в ньому набрано команду

help <ім'я файла>

7. У програмах мовою Matlab *відсутній символ закінчення тексту програми.*

8. У мові Matlab *змінні не описуються і не оголошуються*. Будь-яке нове ім'я, що зустрічається в тексті програми при її виконанні, сприймається системою Matlab як ім'я матриці. Розмір цієї матриці встановлюється при введенні значень її елементів або визначається діями по встановленню значень її елементів, описаними у попередньому операторі або процедурі. Ця особливість робить мову Matlab дуже простою у вжитку і привабливою. У мові MatLAB неможливо використання вхідної матриці або змінної, у якій попередньо не введені або обчислені значення її елементів (а значить - і визначені розміри цієї матриці). У противному випадку при виконанні програми Matlab з'явиться повідомлення про помилку - "Змінна не визначена".

9. Імена змінних можуть містити лише букви латинського алфавіту або цифри і мають починатися з букви. Загальна кількість символів в імені може сягати 30. В іменах змінних можуть використовуватися як великі, так і малі букви. Особливістю мови Matlab є те, що *великі й малі букви в іменах розрізнюються системою*. Наприклад, символи "a" і "A" можуть використовуватися в одній програмі для позначення різних величин.

1.2.9. Завдання

Завдання 1.1. Обчисліть зазначений арифметичний вираз. Укажіть послідовність натискання клавіш. Порівняйте отриманий результат із наведеною відповіддю.

	Відповідь
1. $\dfrac{\left(12\frac{1}{6}-6\frac{1}{27}-5{,}25\right)13{,}5+0{,}111}{0{,}02}$.	599,3
2. $\dfrac{\left(1\frac{1}{12}+2\frac{5}{32}+\frac{1}{24}\right):9{,}6+2{,}13}{0{,}0004}$.	179,5

3. $\dfrac{\left(6{,}6-3\dfrac{3}{14}\right)5\dfrac{5}{6}}{(21-1{,}25):2{,}5}$. 2,5

4. $\dfrac{2{,}625-\dfrac{2}{3}\cdot 2\dfrac{5}{14}}{\left(3\dfrac{1}{12}+4{,}375\right):19\dfrac{8}{9}}$. 2,8095

5. $\dfrac{0{,}134+0{,}05}{18\dfrac{1}{6}-1\dfrac{11}{14}-\dfrac{2}{15}\cdot 2\dfrac{6}{7}}$. 0,00115

6. $\dfrac{\left(58\dfrac{4}{15}-56\dfrac{7}{24}\right):0{,}8+2\dfrac{1}{9}\cdot 0{,}225}{8{,}75\cdot 0{,}6}$. 0,56071

7. $\dfrac{\left(\dfrac{0{,}216}{0{,}15}+0{,}56\right):0{,}5}{\left(7{,}7:24{,}75+\dfrac{2}{15}\right)4{,}5}$. 2

8. $\dfrac{1\dfrac{4}{11}\cdot 0{,}22:0{,}3-0{,}96}{\left(0{,}2-\dfrac{3}{40}\right)1{,}6}$. -4,35

9. $\dfrac{\left(\dfrac{3}{5}+0{,}425-0{,}005\right):0{,}12}{30{,}5+\dfrac{1}{6}+3\dfrac{1}{3}}$. 0,25

10. $\dfrac{3\dfrac{1}{3}+2{,}5}{2{,}5-1\dfrac{1}{3}}\cdot\dfrac{4{,}6-2\dfrac{1}{3}}{4{,}6+2\dfrac{1}{3}}:\left(\dfrac{0{,}05}{\dfrac{1}{7}-0{,}125}+5{,}7\right)$. 0,19231

11. $\dfrac{0{,}725+0{,}42(6)}{0{,}128-6{,}25-(0{,}0345:0{,}12)}\cdot 0{,}25$. -0,04492

12. $\dfrac{\left(4{,}5\cdot 1\dfrac{2}{3}-6{,}75\right)\cdot 0{,}(6)}{\left(3{,}333\cdot 0{,}3+0{,}222\cdot\dfrac{4}{9}\right)2\dfrac{2}{3}}$. 0,17068

13. $\dfrac{\left(5\frac{4}{45}-4\frac{1}{6}\right):5\frac{8}{15}}{\left(4\frac{2}{3}+0{,}75\right)3\frac{9}{13}}\cdot 34\frac{2}{7}$. 0,28571

14. $\dfrac{1\frac{4}{11}\cdot 0{,}22:0{,}3-0{,}96}{\left(0{,}2-\frac{3}{40}\right)1{,}68}$. 0,19048

15. $\dfrac{\left(40\frac{7}{30}-38\frac{5}{12}\right):10{,}9+\left(0{,}875-\frac{7}{30}\right)\cdot\frac{20}{11}}{0{,}008}$. 166,67

16. $\dfrac{(68{,}023-66{,}028):6\frac{1}{9}+\frac{7}{40}\cdot 4{,}5}{0{,}042+0{,}086}$. 8,7028

17. $\dfrac{(2{,}1-1{,}965):(1{,}2\cdot 0{,}045)}{0{,}00325:0{,}013}-\dfrac{4}{0{,}2\cdot 073}$. -17,397

18. $\dfrac{(1{,}88+2{,}127)\cdot 0{,}01875}{0{,}625-\frac{13}{18}:3{,}13}+8{,}29$. 8,2441

19. $\dfrac{3:0{,}4-0{,}009:(0{,}15:2{,}5)}{0{,}32\cdot 6+0{,}033-(5{,}3-3{,}88)}$. 13,79

20. $\dfrac{(34{,}06-33{,}81)\cdot 4}{6{,}84:(28{,}57-25{,}15)}+1{,}33:\dfrac{4}{21}$. 7,4825

21. $\dfrac{8{,}8077}{20-(28{,}2:(13{,}333\cdot 0{,}3+0{,}0125))2{,}004}$. 1,4889

22. $\dfrac{\left(1{,}75:\frac{2}{3}-1{,}75\cdot 1{,}125\right):\frac{7}{12}}{(0{,}2012-0{,}0325):400}$. 2667,5

23. $\dfrac{\left(26\frac{1}{3}-18{,}02\cdot 0{,}75\right)\cdot 2{,}4:0{,}88}{1{,}37-23\frac{2}{3}:1{,}82}$. -3,005

24. $26:\dfrac{3:(0{,}48-0{,}27)}{2{,}52(1{,}38+2{,}45)}+1{,}27$. 18,836

25. $\left(16{,}5-13\frac{7}{9}\right)\frac{6}{11}+2{,}2:(0{,}241-0{,}91)$. -1,8036

Завдання 1.2. Проведіть обчислення по заданій формулі при заданих значеннях параметрів. Зазначте необхідну послідовність дій. Порівняйте отриманий результат із наведеною відповіддю.

Вказівка. *У системі Matlab декілька останніх команд запам'ятовуються. Виклик цих команд у командне вікно здійснюється натисканням клавіш* <↓> і <↑>. Використовуйте цю можливість для повторного звернення до набраної функції.

1. $3m^2 + \sqrt[3]{2n^2} : m$; а) $m = -\frac{14}{5},\ n = tg\frac{\pi}{8}$; б) $m = 2{,}2 \cdot 10^{-2},\ n = \frac{1}{3{,}1}$.

ВІДПОВІДЬ: а) 23,27; б) 26,938.

2. $\frac{4}{3}l^3 \sin^2\frac{\alpha}{2}\sqrt{\cos\alpha}$; а) $l = 1{,}7 \cdot 10^3,\ \alpha = 18^\circ$; б) $l = \frac{16}{21},\ \alpha = \frac{\pi}{5}$.

ВІДПОВІДЬ: а) 1. 5633e+008; б) 5. 0651e-002.

3. $\sqrt{\frac{a\sqrt{b}}{\sqrt[3]{tg\alpha}}}$; а) $a = 1{,}5,\ b = 0{,}8,\ \alpha = 61^\circ$; б) $a = 3 \cdot 10^{-2},\ b = 0{,}71,\ \alpha = \frac{3}{7}\pi$.

ВІДПОВІДЬ: а) 1. 0498e+000; б) 1. 2429e-001.

4. $\frac{3a^2\sqrt{6{,}8 \cdot (a-b)}}{4(a+b)^3}$; а) $a = 4{,}13 \cdot 10^{-1},\ b = \frac{1}{261}$;

б) $a = \sin\frac{5\pi}{8},\ b = -tg12^\circ$.

ВІДПОВІДЬ: а) 2. 9464e+000; б) 4. 9445e+000.

5. $\frac{c^3}{6}\cos\frac{\alpha}{2}\sqrt{\sin\alpha}$; а) $c = \lg 2{,}38,\ \alpha = \frac{\pi}{5}$; б) $c = e^{-0{,}3},\ \alpha = 65^\circ$.

ВІДПОВІДЬ: а) 3. 4657e-004; б) 2. 2120e-002.

6. $\sqrt{\frac{n^3}{16{,}3\sin\alpha\sin 2\alpha}}$; а) $n = 3{,}1516 \cdot 10^{-2},\ \alpha = 5^\circ$; б) $n = e^{3{,}5},\ \alpha = \frac{2\pi}{13}$.

ВІДПОВІДЬ: а) 1. 1265e-002; б) 7. 6324e+001.

7. $5\sin 35^\circ \sqrt{\frac{S^3\cos 36^\circ}{\pi^3 tg\alpha}}$; а) $S = \ln 3,\ \alpha = 44^\circ$; б) $S = \frac{18}{25},\ \alpha = \frac{7}{12}\pi$.

ВІДПОВІДЬ: а) 5. 4283e-001; б) 8. 9703e-018+ 1. 4650e-001i.

8. $\left|\lg(1+\sin\alpha) + \ln(1-\sin\beta)\right|$; а) $\alpha = \frac{3\pi}{7},\ \beta = 83^\circ$; б) $\alpha = \frac{2}{3}\pi,\ \beta = 16^\circ$.

ВІДПОВІДЬ: а) 4. 6035e+000; б) 5. 1546e-002.

9. $\sqrt[3]{\sin^2(\alpha+\beta) - \sin^2(\alpha-\beta)}$; а) $\alpha = \frac{5}{7}\pi,\ \beta = 0{,}3\pi$; б) $\alpha = 12^\circ,\ \beta = 220^\circ$

ВІДПОВІДЬ: а) 4. 8756e-001+ 8. 4448e-001i; б) 7. 3715e-001.

10. $\left(\log_a(b+1{,}4)\right)^{-3/4}$; а) $a = 3{,}56,\ b = e^{0{,}316}$; б) $a = 2,\ \mathrm{b} = 2{,}1649 \cdot 10^{-2}$.

ВІДПОВІДЬ: а) 1. 1790e+000; б) 1. 6630e+000.

11. $3\left(p^{-2/3}+q^{-1/2}\right)\sqrt[3]{pq}$; а) $p=\ln 3,\ q=\lg 3$; б) $p=0{,}013,\ q=1{,}4\cdot 10^2$.

ВІДПОВІДЬ: а) 5. 7737e+000; б) 6. 6559e+001.

12. $\frac{2}{3}m\sqrt{m\sqrt[3]{m\sqrt[4]{m}}}$; а) $m=3{,}6485\cdot 10^2$; б) $m=\frac{24}{37}$.

ВІДПОВІДЬ: а) 1. 5880e+004; б) 5. 4516e-001.

13. $\frac{8}{3}S\sqrt{\frac{S}{\pi}}\sin^6\frac{\alpha}{2}$; а) $S=e^{1{,}11},\ \alpha=\frac{7}{11}\pi$; б) $S=5{,}403,\ \alpha=28^\circ$.

ВІДПОВІДЬ: а) 2. 8187e+000; б) 3. 7879e-003.

14. $2\sqrt{\frac{F}{\pi}tg\alpha}\sin^2\frac{\alpha}{2}$; а) $F=\frac{1}{0{,}03},\ \alpha=\frac{5}{7}\pi$; б) $F=\ln 7,\ \alpha=1{,}34^\circ$.

ВІДПОВІДЬ: а) -6. 6313e+000; б) 5. 0346e-006.

15. $\frac{1}{12}\cdot\frac{m^3\cos\alpha}{(\sin\alpha+\cos\alpha)^3}$; а) $m=-20{,}1,\ \alpha=20^\circ$; б) $m=\lg 13{,}6,\ \alpha=1{,}48$.

ВІДПОВІДЬ: а) -3. 0201e+002; б) 8. 5792e-003.

16. $\frac{\sqrt{3h^3}}{\cos^2\alpha}\sin(\alpha+30^\circ)\sin(\alpha-30^\circ)$;

а) $h=0{,}28,\ \alpha=41^\circ$; б) $h=e^{0{,}415},\ \alpha=237.^\circ$

ВІДПОВІДЬ: а) 8. 1284e-002; б) 4. 9334e+000.

17. $\frac{\alpha}{3}(\lg(d+2)-tg\alpha)^2$;

а) $d=6{,}178,\ \alpha=20^\circ$; б) $d=-2{,}2461\cdot 10^{-2},\ \alpha=1{,}146.$

ВІДПОВІДЬ: а) 3. 5028e-002; б) 1. 4003e+000.

18. $d^3ctg\alpha\sqrt{\sin^4\alpha-\cos^4\alpha}$; а) $d=10{,}6,\ \alpha=50^\circ$; б) $d=e^{2{,}3},\ \alpha=1$.

ВІДПОВІДЬ: а) 4. 1645e+002; б) 4. 1101e+002.

19. $\frac{a^2\sqrt{3}}{4}(\sec\alpha+\cos ec\alpha)^4$;

а) $a=5{,}08,\ \alpha=25^\circ$; б) $a=\ln 1{,}37,\ \alpha=\frac{12}{25}\pi$

ВІДПОВІДЬ: а) 1. 6193e+003; б) 3. 5238e+003.

20. $\frac{\sqrt{\pi}}{3}\cdot\frac{1}{(ctgA+ctgB)^2}$; а) $A=51^\circ,\ \text{B}=39^\circ$; б) $A=0{,}643,\ \text{B}=\frac{\pi}{7}$.

ВІДПОВІДЬ: а) 1. 4132e-001; б) 5. 0772e-002.

21. $\lg\left(3^{x^2-x-9}+\frac{8}{27}\right)$; а) $x=e^{1{,}648}$; б) $x=tg1{,}21$.

ВІДПОВІДЬ: а) 6. 1109e+000; б) -5. 1927e-001.

22. $\dfrac{\sqrt[5]{5e^{4a}\left(a+12{,}36\right)^2}}{\ln\left(a+7\right)}$; а) $a = 2{,}1754\cdot 10^2$; б) $a = \cos 17^\circ$.

ВІДПОВІДЬ: а) 8. 5511e+075 ; б) 4. 0272e+000;

23. $\lg^2 x - \left(\dfrac{27}{8}\right)^{x-1} \sin\sqrt{x}$; а) $x = e^{2{,}145}$; б) $x = 2{,}468\cdot 10^{-1}$.

ВІДПОВІДЬ: а) -2. 0936e+003; б) 1. 7858e-001.

24. $\sqrt[5]{(x-y)^2}\sqrt[3]{1/(y-x)}$; а) $x = e^{-0{,}37}$, $y = \ln 2{,}1517$; б) $x = 37^\circ$, $y = \cos\dfrac{7}{24}\pi$.

ВІДПОВІДЬ: а) 3. 4445e-001; б) 2. 6745e-001.

25. $\dfrac{\sin A + tgB}{\sqrt[5]{\left(A-3B\right)^2}}$; а) $A = 5{,}6,\ B = \lg 25$; б) $A = \dfrac{8}{9}\pi,\ B = \dfrac{\pi}{10}$.

ВІДПОВІДЬ: а) 4. 4466e+000; б) 5. 2145e-001.

Завдання 1.3. Виконайте такі дії (див. таблицю 1.1):

а) число z_1, задане в алгебричній (експоненціальній) формі, перекладіть в експоненціальну (алгебричну) форму і запишіть результат;

б) число z_2, задане в експоненціальній (алгебричній) формі, перекладіть в алгебричну (експоненціальну) форму і запишіть результат;

в) обчисліть заданий вираз; запишіть результат в алгебричній і експоненціальній формах, причому аргумент результату забезпечте в межах між $(-\pi)$ і $+\pi$.

Примітка. Внаслідок складності і значної кількості дій для виконання цього завдання складіть програму у вигляді М-файлу.

Таблиця 1.1

Варіант	Комплексне число				Вираз
	z_1	z_2	z_3	z_4	
1	$4+3i$	$2{,}71e^{i\pi/12}$	$1{,}82e^{-1{,}2i}$	$\sqrt{3}-2i$	$z_1^2\cdot z_2 : z_3 + z_4$
2	$0{,}8-2i$	$3{,}08e^{i7\pi/12}$	$8{,}01e^{2i}$	$-5+\sqrt{2}i$	$z_1^2 : z_2 + z_3 - z_4$
3	$-0{,}7+4i$	$1{,}74e^{i0{,}3\pi}$	$3+4i$	$2{,}1e^{-2{,}3i}$	$\sqrt{z_1 : z_2}\cdot z_3 + z_4$
4	$-3-2i$	$3{,}21e^{15^\circ i}$	$1{,}2+3i$	$2{,}71e^{-78^\circ i}$	$\sqrt{z_1\cdot z_2} : z_3 + z_4$
5	$2{,}71e^{i\pi/12}$	$-0{,}7+4i$	$1{,}31e^{-i5\pi/12}$	$-8-3i$	$\sqrt{z_1} : z_2\cdot z_3 - z_4$
6	$3{,}08e^{i7\pi/12}$	$-3-2i$	$2{,}03e^{i14/13}$	$\sqrt{3}+\sqrt{2}i$	$\left(z_1+z_2\right)^4\cdot z_3 : z_4$
7	$1{,}74e^{0{,}3\pi i}$	$0{,}8-2i$	$3{,}28e^{-1{,}2i}$	$\sqrt{3}-\sqrt{2}i$	$\left(\sqrt{z_1}+z_2\right)\cdot z_3 : z_4$

8	$3{,}21e^{15i}$	$4+3i$	$\sqrt{3}-4i$	$1{,}23e^{111^\circ i}$	$(z_1-z_2)^3\cdot z_3+z_4$
9	$1+i\pi/2$	$1{,}2e^{107^\circ i}$	$0{,}8-2i$	$2{,}5e^{14^\circ i}$	$(z_1:z_2+z_3)^3\cdot z_4$
10	$\sqrt{5}-i$	$0{,}7e^{1{,}7i}$	$1{,}2e^{0{,}9i}$	$-3-2i$	$(z_1:z_2+z_3)^2-z_4$
11	$0{,}187-3{,}94i$	$0{,}3e^{-107^\circ i}$	$-0{,}7+4i$	$1{,}5e^{23^\circ i}$	$\sqrt[3]{z_1+z_2-z_3}\cdot z_4$
12	$-1+\sqrt{5}i$	$2{,}1e^{211^\circ i}$	$0{,}4e^{32^\circ i}$	$4+3i$	$\sqrt[3]{z_1}\cdot z_2:z_3+z_4$
13	$-\sqrt{3}-4i$	$1{,}25e^{-0{,}8i}$	$-3-2i$	$0{,}75e^{0{,}7i}$	$(\sqrt[3]{z_1\cdot z_2}+z_3):z_4$
14	$1{,}2e^{1{,}7i}$	$0{,}18-3{,}9i$	$0{,}71e^{4i}$	$0{,}8-2i$	$(\sqrt[3]{z_1:z_2}+z_3)\cdot z_4$
15	$0{,}3e^{-97^\circ i}$	$-1+\sqrt{5}i$	$-0{,}7+4i$	$5{,}2\,e^{71^\circ i}$	$(\sqrt{z_1\cdot z_2}-z_3):z_4$
16	$1{,}25e^{0{,}6i}$	$-\sqrt{3}-4i$	$4+3i$	$2{,}5e^{3{,}8i}$	$(\sqrt{z_1:z_2}-z_3)\cdot z_4$
17	$1{,}05e^{-0{,}4i}$	$\sqrt{5}-i$	$2{,}7e^{0{,}8i}$	$-0{,}7+4i$	$\sqrt{(z_1:z_2+z_3)\cdot z_4}$
18	$2{,}1e^{73^\circ i}$	$1+i\pi/2$	$\sqrt{2}+\sqrt{3}i$	$1{,}93e^{192^\circ i}$	$\sqrt{(z_1\cdot z_2-z_3):z_4}$
19	$2{,}7+0{,}8i$	$2e^{-\sqrt{3}i}$	$0{,}81e^{i\pi/7}$	$-\sqrt{2}-\sqrt{3}i$	$\sqrt{(z_1+z_2):z_3\cdot z_4}$
20	$-0{,}8+2{,}7i$	$-2e^{\sqrt{3}i}$	$0{,}9e^{i5\pi/7}$	$3{,}1-2{,}1i$	$\sqrt{(z_1+z_2)\cdot z_3:z_4}$
21	$-1{,}1-3{,}2i$	$0{,}33e^{-1{,}9i}$	$2e^{\sqrt{2}i}$	$2{,}08+i$	$\sqrt{z_1-z_2}\cdot z_3:z_4$
22	$2{,}1-3{,}2i$	$0{,}68e^{148^\circ i}$	$\sqrt{5}+\sqrt{2}i$	$2{,}73e^{23^\circ i}$	$\sqrt{z_1-z_2}:z_3\cdot z_4$
23	$1{,}1e^{-0{,}8i}$	$\sqrt{5}-2i$	$-1{,}7+i$	$0{,}97e^{\sqrt{2}i}$	$((z_1+z_2)^2-z_3):z_4$
24	$2{,}1e^{0{,}8i}$	$-\sqrt{5}+2i$	$1{,}7e^{\sqrt{3}i}$	$0{,}8e^{2{,}5i}$	$((z_1-z_2)^2+z_3):z_4$
25	$1{,}1e^{-2{,}1i}$	$\pi/8-2{,}1i$	$2{,}71+0{,}4i$	$1{,}71e^{-\sqrt{3}i}$	$(z_1-z_2)^3:z_3+z_4$

Завдання 1.4. Знайдіть корені квадратного рівняння

$$a\cdot x^2+b\cdot x+c=0$$

при заданих значеннях коефіцієнтів *a*, *b* і *c* (див. таблицю 1.2).

Таблиця 1.2

Варіант	*a*	*b*	*c*
1	0.56	1. 2e-4	4.08
2	1	0.1	100
3	4. 2e-3	8. 03e-4	1.06
4	7. 1e3	9. 4e4	8. 3e10
5	5.09	4.32	256
6	8.3	5.34	693

7	27	27	1276
8	3.08	0.2	30
9	5.3	10.6	876
10	0.45	0. 034	121
11	4.3	10.7	3. 4e3
12	13	0.8	287
13	6. 035	5.2	875
14	2.3	7.9	324
15	1	0.02	16.57
16	1.3	0.56	18.8
17	0.13	0. 056	18.8
18	17	12	956
19	0. 085	1	1. 3e3
20	1.2	0.32	15
21	7.1	6.4	256
22	0.2	0. 002	2.9
23	1. 4e-3	3.9	2. 6e2
24	0.86	3.2	5. 4e2
25	7. 3e3	8. 2e2	3. 5e8

1.2.9. Запитання

1. Як подаються дійсні числа при обчисленнях у системі Matlab?
2. Як змінити формат подання дійсних чисел у командному вікні?
3. Яким чином оголошуються змінні в мові Matlab?
4. Як зробити так, щоб результат дій, записаних у черговому рядку
   а) виводився в командне вікно; б) не виводився на екран?
5. Яку роль грає системна змінна ***ans*** ?
6. Як повернути в командний рядок раніше введену команду ?
7. Як увести значення комплексного числа й у якому виді воно виведеться на екран ?
8. Як мовою Matlab забезпечити додавання, віднімання, множення, ділення й піднесення до степеня комплексних чисел ?
9. Які функції роботи з комплексними числами є в мові Matlab ?

1.3. Операції з векторами й матрицями

Matlab є системою, спеціально призначеною для здійснювання складних обчислень із векторами, матрицями й поліномами.

Під вектором у Matlab розуміється одновимірний масив чисел, а під матрицею - двовимірний масив. При цьому за замовчуванням передбачається, що будь-яка задана змінна є вектором або матрицею. Наприклад, окреме задане число система сприймає як матрицю розміром (1*1), а вектор-рядок із N елементами - як матрицю розміром (1*N).

1.3.1. Введення векторів і матриць

Початкові значення векторів можна задавати із клавіатури шляхом поелементного введення. Для цього в рядку треба спочатку вказати ім'я вектора, потім поставити знак присвоювання ' =', а далі, - відкриваючу *квадратну* дужку, а за нею ввести задані значення елементів вектора, *відділяючи їх пропусками або комами*. Закінчується рядок записом квадратної дужки, що закриває.

Наприклад, запис рядка V = [1.2 -0. 3 1. 2e-5] задає вектор V, що містить три елементи зі значеннями 1.2, -0. 3 і 1. 2e-5 (рис. 1.14):

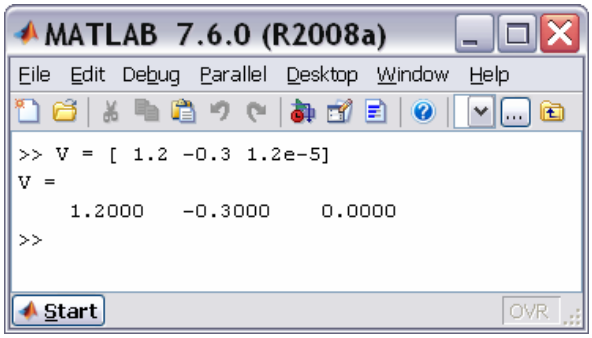


Рис. 1.14. Введення вектора

Після введення вектора система виводить його на екран. Те, що в наведеному прикладі останній елемент виведений як 0, обумовлено встановленим форматом short, відповідно до якого виводяться дані на екран.

Довгий вектор можна вводити частинами, які потім об'єднувати за допомогою операції *об'єднання векторів у рядок* : v = [v1 v2]. Приклад наведений на рис. 1.15.

Мова Matlab дає користувачеві можливість *скороченого введення вектора, значення елементів якого є арифметичною прогресією*. Якщо позначити:

nz - початкове значення цієї прогресії (значення першого елемента вектора);

kz - кінцеве значення прогресії (значення останнього елемента вектора);

h - різницю прогресії (крок),

то вектор можна ввести за допомогою короткого запису

V = nz : h : kz .

Наприклад, введення рядка V = - 0.1 : 0.3 : 1.4 приведе до результату, показаному на рис. 1.16.

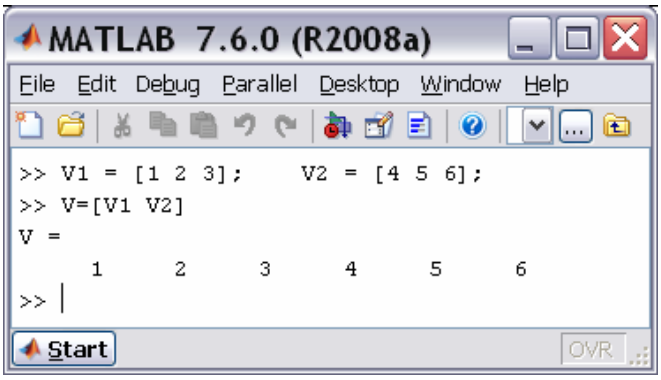


Рис. 1.15. Конкатенація (об'єднання) векторів

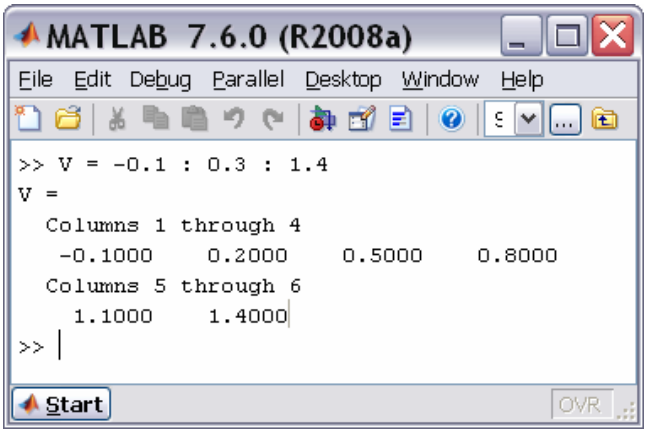


Рис. 1.16. Введення вектора арифметичної прогресії

Якщо середній параметр (різниця прогресії) не зазначений, то він за замовчуванням приймається рівним одиниці. Наприклад, команда

>> -2. 1 : 5

приводить до формування такого вектора

```
ans =
  Columns 1 through 7
   -2.1000 -1.1000 -0.1000 0.9000 1.9000 2.9000 3.9000
  Column 8
    4. 9000
```

У такий спосіб уводяться вектори-рядки. *Вектор-стовпець* вводиться аналогічно, але *значення елементів відокремлюються знаком ";".*

Введення значень елементів *матриці* здійснюється в Matlab у квадратних дужках, *по рядках*. При цьому елементи рядка матриці один від одного відокремлюються пропуском або комою, а *рядки один від одного відокремлюються знаком ";"* (рис. 1.17).

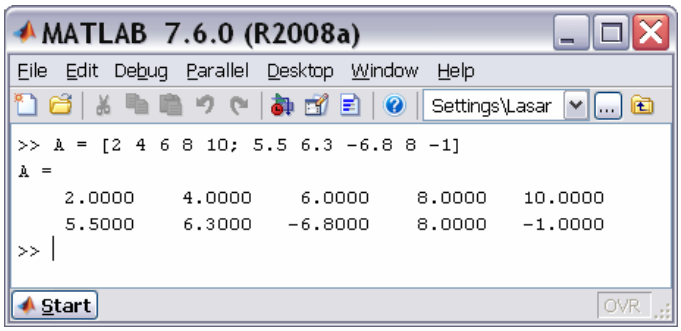


Рис. 1.17. Приклад введення матриці

1.3.2. Формування векторів і матриць

Matlab має декілька функцій, що дозволяють формувати вектори й матриці деякого певного виду. До таких функцій належать:

zeros(M,N) - створює матрицю розміром (M*N) із нульовими елементами, наприклад:

```
» zeros(3,5)
ans =
   0   0   0   0   0
   0   0   0   0   0
   0   0   0   0   0
```

ones(M,N) - створює матрицю розміром (M*N) з одиничними елементами, наприклад:

```
» ones(3,5)
ans =
   1   1   1   1   1
   1   1   1   1   1
   1   1   1   1   1
```

eye(M,N) - створює матрицю розміром (M*N) з одиницями по головній діагоналі й інших нульових елементах, наприклад:

```
» eye(3,5)
ans =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
```

rand(M,N) - створює матрицю розміром (M*N) із випадкових чисел, рівномірно розподілених у діапазоні від 0 до 1, наприклад:

```
» rand(3,5)
ans =
  2.1896e-001   6.7930e-001   5.1942e-001   5.3462e-002   7.6982e-003
  4. 7045e-002  9. 3469e-001  8. 3097e-001  5. 2970e-001  3. 8342e-001
  6. 7886e-001  3. 8350e-001  3. 4572e-002  6. 7115e-001  6. 6842e-002
```

randn(M,N)- створює матрицю розміром (M*N) із випадкових чисел, розподілених за нормальним (гауссовим) законом із нульовим математичним сподіванням і стандартним (середньоквадратичним) відхиленням, рівним одиниці, наприклад:

```
» randn(3,5)
ans =
```

```
 1.1650e+000   3.5161e-001   5.9060e-002    8.7167e-001    1.2460e+000
 6. 2684e-001  -6. 9651e-001  1. 7971e+000  -1. 4462e+000  -6. 3898e-001
 7. 5080e-002   1. 6961e+000  2. 6407e-001  -7. 0117e-001   5. 7735e-001
```

hadamard(N) - створює матрицю Адамара розміром (N*N), наприклад:

```
» hadamard(4)
ans =
    1    1    1    1
    1   -1    1   -1
    1    1   -1   -1
    1   -1   -1    1
```

hilb(N) - створює матрицю Гілберту розміром (N*N), наприклад:

```
» hilb(4)
ans =
  1.0000e+000   5.0000e-001   3.3333e-001   2.5000e-001
  5.0000e-001   3.3333e-001   2.5000e-001   2.0000e-001
  3. 3333e-001  2. 5000e-001  2. 0000e-001  1. 6667e-001
  2. 5000e-001  2. 0000e-001  1. 6667e-001  1. 4286e-001
```

invhilb(N) - створює обернену матрицю Гільберту розміром (N*N), наприклад:

```
» invhilb(4)
ans =
      16        -120         240        -140
    -120        1200       -2700        1680
     240       -2700        6480       -4200
    -140        1680       -4200        2800
```

pascal(N) - створює матрицю Паскаля розміром (N*N), наприклад:

```
» pascal(5)
ans =
    1    1    1    1    1
    1    2    3    4    5
    1    3    6   10   15
    1    4   10   20   35
    1    5   15   35   70.
```

У мові Matlab передбачено декілька функцій, що дозволяють формувати матрицю на основі іншої (заданої) або, використовуючи деякий заданий вектор. До таких функцій належать:

fliplr(A) - формує матрицю, переставляючи стовпчики відомої матриці A щодо вертикальної осі, наприклад, якщо

```
A =
    1    2    3    4    5    6
    7    8    9   10   11   12
   13   14   15   16   17   18
```

то застосування цієї функції приводить до результату

```
» fliplr(A)
ans =
    6    5    4    3    2    1
   12   11   10    9    8    7
   18   17   16   15   14   13
```

flipud(A) - формує матрицю, переставляючи рядки заданої матриці A щодо горизонтальної осі, наприклад:

```
» flipud(A)
ans =
   13   14   15   16   17   18
    7    8    9   10   11   12
    1    2    3    4    5    6
```

rot90(A) - формує матрицю шляхом "повороту" заданої матриці A на 90 градусів проти годинникової стрілки:

```
» rot90(A)
ans =
    6   12   18
    5   11   17
    4   10   16
    3    9   15
    2    8   14
    1    7   13
```

reshape(A,m,n) - утворює матрицю розміром (m*n) шляхом послідовної вибірки елементів заданої матриці A по стовпчиках і наступному розподілі цих елементів по 'n' стовпчиках, кожний з яких містить 'm' елементів; при цьому число елементів матриці A повинно дорівнювати m*n, наприклад:

```
» reshape(A,2,9)
ans =
    1   13    8    3   15   10    5   17   12
    7    2   14    9    4   16   11    6   18
```

tril(A) - утворює нижню трикутну матрицю на основі матриці A шляхом онулювання її елементів вище головної діагоналі:

```
» tril(A)
ans =
    1    0    0    0    0    0
    7    8    0    0    0    0
   13   14   15    0    0    0
```

triu(A) - утворює верхню трикутну матрицю на основі матриці A шляхом онулювання її елементів нижче головної діагоналі:

```
» triu(A)
ans =
    1    2    3    4    5    6
    0    8    9   10   11   12
    0    0   15   16   17   18
```

hankel(V) - утворює квадратну матрицю Ганкеля, перший стовпчик якої збігається із заданим вектором V, наприклад:

```
>> V = [-5 6 7 4]
V =
   -5    6    7    4

» hankel(V)
ans =
   -5    6    7    4
    6    7    4    0
    7    4    0    0
    4    0    0    0
```

Процедура ***diag(x)*** - формує або витягає діагональ матриці.

Якщо ***x*** - вектор, то функція ***diag(x)*** створює квадратну матрицю з вектором ***x*** на головній діагоналі:

```
» diag(V)
ans =
   -5    0    0    0
    0    6    0    0
    0    0    7    0
    0    0    0    4
```

Щоб установити заданий вектор на іншу діагональ, при звернені до функції необхідно зазначити ще один параметр (ціле число) - номер діагоналі (при цьому *діагоналі відлічуються від головної нагору*), наприклад:

```
» diag(V,  -1)
ans =
    0    0    0    0    0
   -5    0    0    0    0
    0    6    0    0    0
    0    0    7    0    0
    0    0    0    4    0
```

Якщо ***x*** - матриця, то функція ***diag*** створює вектор-стовпчик, що складається з елементів головної діагоналі заданої матриці ***x,*** наприклад, для матриці A, зазначеної перед прикладом застосування процедури ***fliplr***:

```
» diag(A)
ans =
    1
    8
   15
```

Якщо при цьому зазначити додатково номер діагоналі, то можна одержати вектор-стовпчик з елементів будь-якої діагоналі матриці ***x***, наприклад:

```
» diag(A,3)
ans =
    4
   11
   18
```

Функція ***zeros***(1,N) формує (створює) вектор-рядок із N нульових елементів. Аналогічно ***zeros***(N,1) створює вектор-стовпчик із N нулів.

Вектори, значення елементів яких є випадковими, рівномірно розподіленими, формуються в такий спосіб: ***rand***(1,n) - для вектора-рядка і ***rand***(m,1) - для вектора-стовпчика.

1.3.3. Витягання й вставляння частин матриць

Насамперед зазначимо, що *звернення до будь-якого елемента певної матриці в Matlab здійснюється шляхом указівки (у дужках, через кому) після йменння матриці двох цілих додатних чисел, що визначають відповідно номери рядка й стовпця матриці, на перетинанні яких розташований цей елемент.*

Нехай маємо деяку матрицю A:

```
>> A = [ 1 2 3 4; 5 6 7 8; 9 10 11 12]
A =
    1    2    3    4
```

```
    5     6     7     8
    9    10    11    12
```

Тоді одержати значення елемента цієї матриці, розташованого на перетинанні другого рядка із третім стовпчиком, можна в такий спосіб:

```
>> A(2,3)
ans =
    7
```

Якщо потрібно, навпаки, встановити на це місце деяке число, наприклад, π, то це можна зробити так:

```
>> A(2, 3) = pi; A
A =
   1.0000    2.0000    3.0000    4.0000
   5. 0000   6. 0000   3. 1416   8. 0000
   9. 0000  10. 0000  11. 0000  12. 0000
```

Іноді потрібно створити меншу матрицю з більшої, формуючи її шляхом витягання з останньої матриці елементів її кількох рядків і стовпчиків, або, навпаки, вставити меншу матрицю таким чином, щоб вона стала певною частиною матриці більшого розміру. Це в Matlab робиться за допомогою знака двокрапки (" : ").

Розглянемо ці операції на прикладах.

Нехай потрібно створити вектор V1, що складається з елементів третього стовпчика попередньої матриці A. Для цього зробимо такі дії:

```
>> V1 = A(:, 3)
V1 =
    3.0000
   3. 1416
  11. 0000
```

Щоб створити вектор V2, який складається з елементів другого рядка матриці A, роблять так:

```
>> V2 = A(2, : )
V2 =
   5. 0000   6. 0000   3. 1416   8. 0000
```

Припустимо, що необхідно з матриці A утворити матрицю B розміром (2*2), яка складається з елементів лівого нижнього рогу матриці A. Тоді роблять таке:

```
>> B = A(2:3, 1:2)
B =
    5     6
    9    10
```

Аналогічно можна вставити матрицю B в верхню середину матриці A:

```
>> A(1:2,2:3)=B
A =
    1    5     6     4
    5    9    10     8
    9   10    11    12
```

Як очевидно, для цього *замість указівки номерів елементів матриці можна вказувати діапазон змінювання цих номерів шляхом указівки нижньої й верхньої меж, розділяючи їх двокрапкою*.

Примітка. *Якщо верхньою межею змінювання номерів елементів матриці є її розмір у цьому вимірі, замість нього можна використовувати службове слово* ***end.*** Наприклад:

```
>> A(2:end,2:end)
```

```
ans =
    9   10    8
   10   11   12
```

Ці операції дуже зручні для формування матриць, більшість елементів яких однакові, зокрема, так званих розріджених матриць, що складаються, в основному, із нулів, за винятком окремих елементів. Для прикладу розглянемо формування розрідженої матриці розміром (5*7) з одиничними елементами в її центрі:

```
>> A = zeros(5,7);
>> B = ones(3,3);
>> A(2:4,3:5)=B
A =
    0    0    0    0    0    0    0
    0    0    1    1    1    0    0
    0    0    1    1    1    0    0
    0    0    1    1    1    0    0
    0    0    0    0    0    0    0
```

"Розтягнути" матрицю (A) *у єдиний* вектор (V) можна за допомогою звичайного запису *"V = A(:)".* При цьому створюється вектор-стовпчик із кількістю елементів (m*n), у якому стовпчики заданої матриці розміщені зверху униз у порядку самих стовпчиків:

```
» A = [1 2 3; 4 5 6]
A =
    1    2    3
    4    5    6
» v = A(:)
v =
    1
    4
    2
    5
    3
    6
```

Нарешті, "розширювати" матрицю, укладаючи її з окремих заданих матриць ("блоків") можна теж досить просто. Якщо задані кілька матриць-блоків A1, A2,... AN з однаковою кількістю рядків, то з них можна "зліпити" єдину матрицю A, об'єднуючи блоки в один "рядок" у такий спосіб:

A = [A1, A2,... , AN].

Цю операцію називають *горизонтальною конкатенацією* (зчепленням) матриць. Аналогічно, *вертикальна конкатенація* матриць реалізується (за умови, що всі складові блоки-матриці мають однакову кількість стовпчиків) аналогічним чином, шляхом застосовування для відділяння блоків замість ко-ми крапки з комою:

A = [A1; A2;... ; AN].

Наведемо приклади. Приклад горизонтальної конкатенації :

```
>> A1 =  [1 2 3; 4 5 6; 7 8 9];
>> A2 = [10;11;12];
>> A3 = [14 15; 16 17; 18 19];
>> A = [A1, A2, A3]
A =
    1    2    3   10   14   15
    4    5    6   11   16   17
    7    8    9   12   18   19
```

Приклад вертикальної конкатенації:

```
>> B1 = [1 2 3 4 5];
>> B2 = [ 6 7 8 9 10; 11 12 13 14 15];
>> B3 = [17 18 19 20 21];
>> B = [ B1; B2; B3]
B =
    1    2    3    4    5
    6    7    8    9   10
   11   12   13   14   15
   17   18   19   20   21
```

1.3.4. Дії над векторами

Розрізнюватимемо дві групи дій над векторами:

а) векторні дії - тобто такі, що передбачені векторним зчисленням у математиці;

б) *дії по перетворенню елементів* - це дії, що перетворюють елементи вектора, але не є операціями, дозволеними математикою з векторами.

Векторні дії над векторами

Додавання векторів. Як відомо, складатися (підсумовуватися) можуть тільки вектори однакового типу (тобто такі, які обидва є або векторами-рядками, або векторами-стовпцями), що мають однакову довжину (тобто однакову кількість елементів). Якщо X і Y є саме такими векторами, то їхню суму Z можна одержати, увівши команду Z = X + Y, наприклад:

```
» x = [1 2 3] ;    y = [ 4 5 6];
» v = x + y
v =    5    7    9
```

Аналогічно за допомогою арифметичного знака " - " здійснюється ***відні-мання векторів***, що мають однакову структуру (Z = X - Y).

Наприклад:

```
» v = x - y
v =   -3   -3   -3
```

Транспонування вектора здійснюється застосуванням знака апострофу, що записується відразу за записом імені вектора, який транспонується . Наприклад:

```
» x'
ans =
          1
          2
          3
```

Множення вектора на число здійснюється в Matlab за допомогою знака арифметичного множення (*) у такий спосіб: Z = X*r або Z = r*X, де r - деяке дійсне число.

Приклад:

```
» v = 2*x
  v =    2    4    6
```

Множення двох векторів визначено у математиці тільки для векторів однакового розміру (довжини) і лише тоді, коли один із векторів-множників є

рядком, а другий - стовпчиком. Тобто, якщо вектори X і Y є рядками, то математичний зміст мають лише дві форми множення цих векторів: U = X' * Y і V = X * Y'. Причому в першому випадку результатом буде квадратна матриця, а в другому - число.

У MatLAB множення векторів здійснюється застосуванням звичайного знака множення (*), який записується між множниками-векторами.

Приклад:

```
» 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
```

Для *трикомпонентних векторів* у Matlab існує функція ***cross***, яка дозволяє знайти ***векторний добуток двох векторів***. Для цього, якщо задані два трикомпонентних вектори v1 і v2, достатньо ввести оператор

cross(v1, v2).

Приклад:

```
» v1 = [1 2 3];   v2 = [4 5 6];
» cross(v1,v2)
ans =   -3    6   -3
```

На цьому перелік припустимих математичних операцій з векторами вичерпується.

Поелементне перетворення векторів

У мові Matlab є ряд операцій, які перетворюють заданий вектор в інший того ж розміру й типу, але в той же час не є математичними операціями з вектором як математичним об'єктом. Усі ці операції перетворюють елементи вектора як елементи звичайного одновимірного масиву чисел. До таких операцій належать, наприклад, *усі елементарні математичні функції*, наведені в розділі 1.2.4 і які залежать від одного аргументу. У мові Matlab запис, наприклад, виду Y = sin(X), де X - деякий відомий вектор, приводить до формування нового вектора Y, що має той самий тип і розмір, але елементи якого дорівнюють синусам відповідних елементів вектора-аргументу X. Наприклад:

```
» x = [ -2,-1,0,1,2];
» y = sin(x)
y =   -0. 9093   -0. 8415        0    0. 8415    0. 9093
» z = tan(x)
z =    2. 1850   -1. 5574        0    1. 5574   -2. 1850
» v = exp(x)
v =         0. 3679    1. 0000    2. 7183    7. 389
```

Крім цих операцій у MatLAB передбачено декілька операцій поелементного перетворення, що здійснюються за допомогою знаків звичайних арифметичних дій. *Ці операції застосовуються до векторів однакового типу й розміру. Результатом їх є вектор того ж типу й розміру*.

Додавання (віднімання) числа до (із) кожного елемента вектора. Здійснюється за допомогою знаку " + " (" - ").

Поелементне множення векторів. Проводиться за допомогою сукупності знаків " .* ", що записується між іменами векторів, які перемножуються. У результаті утворюється вектор, кожний елемент якого є добутком відповідних елементів векторів -"співмножників".

Поелементне ділення векторів. Здійснюється за допомогою сукупності знаків ". /". Результат - вектор, кожний елемент якого є часткою від ділення відповідного елемента першого вектора на відповідний елемент другого вектора.

Поелементне ділення векторів в оберненому напрямку. Здійснюється за допомогою сукупності знаків ".\". В результаті одержують вектор, кожний елемент якого є часткою від ділення відповідного елемента другого вектора на відповідний елемент першого вектора.

Поелементне піднесення до степеня. Здійснюється за допомогою сукупності знаків ". ^". Результат - вектор, кожний елемент якого є відповідним елементом першого вектора, піднесеним до степеня, розмір якого дорівнює значенню відповідного елемента другого вектора. Приклад :

```
» x = [1,2,3,4,5];          y = [-2,1,4,0,5];
» disp(x + 2)
    3    4    5    6    7
» disp(y - 3)
   -5   -2    1   -3    2
» disp(x. *y)
   -2    2   12    0   25
» disp(x. /y)
Warning: Divide by zero
  -0. 5000   2. 0000   0. 7500      Inf   1. 0000
» disp(x. \y)
  -2. 0000   0. 5000   1. 3333        0   1. 0000
» disp(x. ^y)
        1         2        81         1      3125
```

Вищевказані операції дозволяють дуже просто обчисляти (а потім - будувати графіки) складних математичних функцій, не використовуючи при цьому оператори циклу, тобто робити побудову графіків у режимі калькулятора.

Для цього достатньо задати значення аргументу як арифметичну прогресію так, як це було показано в поділі 1.3.1, а потім записати потрібну функцію, використовуючи знаки поелементного перетворення векторів.

Наприклад, нехай потрібно обчислити значення функції:

$$y = a \cdot e^{-hx} \cdot \sin x$$

при значеннях аргументу ***x*** від 0 до 10 із кроком 1. Обчислення масиву значень цієї функції у зазначених умовах можна здійснити за допомогою лише двох простих операторів :

```
» a = 3; h = 0.5;
» x = 0:1:10;
» y = a * exp(-h*x) . * sin(x)
y =
  Columns 1 through 7
```

```
    0 1.5311 1.0035 0.0945 -0.3073 -0.2361 -0.0417
Columns 8 through 11
  0. 0595    0. 0544    0. 0137   -0. 0110
```

1.3.5. Поелементне перетворення матриць

Для поелементного перетворення матриці придатні всі зазначені раніше в п. 1.2.4 алгебричні функції. Кожна така функція формує матрицю того самого розміру, що й задана, кожний елемент якої обчислюється як зазначена функція від відповідного елемента заданої матриці. Крім цього, у Matlab визначені операції ***поелементного множення*** матриць однакового розміру (сполученням ".*", що записується між іменами матриць, що перемножуються), ***поелементного ділення*** (сполучення "./" і ".\"), ***поелементного піднесення до степеня*** (сполучення ".^"), коли кожний елемент першої матриці підноситься до степеня, який дорівнює значенню відповідного елемента другої матриці.

Наведемо кілька прикладів:

```
» A = [1,2,3,4,5; -2, 3, 1, 4, 0]
A =
    1    2    3    4    5
   -2    3    1    4    0
» B = [-1,3,5,-2,1; 1,8,-3,-1,2]
B =
   -1    3    5   -2    1
    1    8   -3   -1    2
» sin(A)
ans =
   0. 8415    0. 9093    0. 1411   -0. 7568   -0. 9589
  -0. 9093    0. 1411    0. 8415   -0. 7568         0
» A . * B
ans =
   -1    6   15   -8    5
   -2   24   -3   -4    0
» A . / B
ans =
  -1. 0000    0. 6667    0. 6000   -2. 0000    5. 0000
  -2. 0000    0. 3750   -0. 3333   -4. 0000         0
» A . \ B
Warning: Divide by zero
ans =
  -1. 0000    1. 5000    1. 6667   -0. 5000    0. 2000
  -0. 5000    2. 6667   -3. 0000   -0. 2500       Inf
» A . ^ B
ans =
  1. 0e+003 *
   0. 0010    0. 0080    0. 2430    0. 0001    0. 0050
  -0. 0020    6. 5610    0. 0010    0. 0002         0
```

Оригінальною в мові Matlab є операція додавання до матриці числа. Вона записується в такий спосіб: A + x, або x + A (A - матриця, а x - число). Такої операції немає в математиці. У Matlab вона є еквівалентною до сукупності операцій

$$A + x * E,$$

де E - позначення матриці, що складається саме з одиниць, тих самих розмірів, що і матриця A. Наприклад:

```
» A = [ 1 2 3 4 5; 6 7 8 9 11 ]
A =
    1    2    3    4    5
    6    7    8    9   11
» A + 2
ans =
    3    4    5    6    7
    8    9   10   11   13
» 2 + A
ans =
    3     4         5   6    7
    8    9  10  11  13
```

1.3.6. Матричні дії над матрицями

До матричних дій над матрицями відносять такі операції, які використовуються в матричному численні в математиці і не суперечать йому.

Базові дії з матрицями - ***додавання, віднімання, транспонування, множення матриці на число, множення матриці на матрицю, піднесення матриці до цілого степеня*** - здійснюються в мові Matlab за допомогою звичайних знаків арифметичних операцій. При використовуванні цих операцій *важливо пам'ятати* умови, за яких ці операції є можливими:

при додаванні або відніманні матриць вони повинні мати однакові розміри;

при множенні матриць кількість стовпчиків першої матриці повинна збігатися з кількістю рядків другої матриці.

Невиконання цих умов призведе до появи в командному вікні повідомлення про помилку. Наведемо кілька прикладів.

Приклад додавання й віднімання:

```
» 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
   11    13    15    17    20
» A - B
ans =
    1    3    5    7    9
    1    1    1    1    2.
```

Приклад множення на число:

```
» 5*A
ans =
    5   10   15   20   25
   30   35   40   45   55
» A*5
ans =
    5   10   15   20   25
   30   35   40   45   55.
```

Приклад транспонування матриці:

```
» A'
ans =
    1   6
    2   7
    3   8
    4   9
    5   11.
```

Приклад множення матриці на матрицю:

```
» A' * B
ans =
   30   35   40   45   50
   35   40   45   50   55
   40   45   50   55   60
   45   50   55   60   65
   55   61   67   73   79
» C = A * B'
C =
  -40   115
  -94   299.
```

Функція ***обернення матриці - inv***(A) - обчисляє матрицю, обернену до заданої матриці А. Початкова матриця А повинна бути квадратною, а її визначник не повинен дорівнювати нулеві.

Наведемо приклад:

```
» inv(C)
ans =
 -2. 6000e-001  1. 0000e-001
 -8. 1739e-002  3. 4783e-002
```

Перевіримо слушність виконання операції обернення, застосовуючи її ще раз до отриманого результату:

```
» inv(ans)
ans =
 -4. 0000e+001  1. 1500e+002
 -9. 4000e+001  2. 9900e+002
```

Як бачимо, ми одержали початкову матрицю С, що є ознакою правильності виконання обернення матриці.

Піднесення матриці до цілого степеня здійснюється в Matlab за допомогою знака " ^ ": A^***n***. При цьому матриця має бути квадратною, а ***n*** має бути цілим (додатним або від'ємним) числом. Ця матрична дія є еквівалентною до множення матриці А на себе ***n*** разів (якщо ***n*** - додатне) або множенню оберненої матриці на себе (при ***n*** від'ємному).

Наведемо приклад:

```
» A^2
ans =
    8    -3   -10
   -5    10    16
   -2     4     9
» A^(-2)
ans =
  1.5385e-001  -7.6923e-002   3.0769e-001
  7. 6923e-002  3. 0769e-001 -4. 6154e-001
  2. 1328e-018 -1. 5385e-001  3. 8462e-001
```

Дуже оригінальними в мові Matlab є дві нові, невідомі в математиці функції ***ділення матриць***. При цьому вводяться поняття ***ділення матриць зліва направо*** і ***ділення матриць справа наліво***. Перша операція записується за

допомогою знаку " / " , а друга - " \ ", які розташовуються між іменами матриць, які діляться.

Операція B / A еквівалентна послідовності дій B * ***inv***(A), де функція ***inv*** здійснює *обернення матриці.* Її зручно використовувати для розв'язування матричного рівняння:

$$X * A = B.$$

Аналогічно операція A\B рівносильна сукупності операцій ***inv***(A)*B, що є розв'язком матричного рівняння:

$$A * X = B.$$

Для прикладу розглянемо задачу відшукування коренів системи лінійних алгебричних рівнянь:

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

У середовищі Matlab це можна зробити таким чином:

```
» 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
```

1.3.7. Матричні функції

Обчислення ***матричної експоненти*** (e^A) здійснюється за допомогою функцій ***expm, expm1, expm2, expm3.*** Ці функції варто відрізняти від раніше розглянутої функції ***exp***(A), яка формує матрицю, кожний елемент якої дорівнює ***e*** в степені, що дорівнює відповідному елементу матриці A.

Функція ***expm*** є вмонтованою функцією Matlab. Функція ***expm1***(A) є M-файлом, який обчислює матричну експоненту шляхом використання розкладення Паде матриці A. Функція ***expm2***(A) обчислює матричну експоненту, використовуючи розкладення Тейлора матриці A. Функція ***expm3***(A) обчислює матричну експоненту на основі використання спектрального розкладу A.

Наведемо приклади використання цих функцій:

```
» A = [1,2,3; 0, -1,5;7, -4,1]
A =
    1    2    3
    0   -1    5
    7   -4    1
```

```
» expm(A)
ans =
  131.3648    -9.5601    80.6685
   97. 8030   -7. 1768   59. 9309
  123. 0245   -8. 8236   75. 4773

» expm1(A)
ans =
  131.3648    -9.5601    80.6685
   97. 8030   -7. 1768   59. 9309
  123. 0245   -8. 8236   75. 4773

» expm2(A)
ans =
  131.3648    -9.5601    80.6685
   97. 8030   -7. 1768   59. 9309
  123. 0245   -8. 8236   75. 4773

» expm3(A)
ans =
  1.0e+002 *
   1. 3136 + 0. 0000i    -0. 0956 + 0. 0000i    0. 8067 - 0. 0000i
   0. 9780 + 0. 0000i    -0. 0718 - 0. 0000i    0. 5993 - 0. 0000i
    1.2302 + 0. 0000i    -0. 0882 - 0. 0000i    0. 7548 - 0. 0000i
```

Функція ***logm***(A) робить обернену операцію - логарифмування матриці за натуральною основою, наприклад:

```
 A =
    1    2    3
    0    1    5
    7    4    1

» B = expm3(A)
B =
  1.0e+003 *
    0.9378    0.7987    0.9547
    1. 0643   0. 9074   1. 0844
    1. 5182   1. 2932   1. 5459

» logm(B)
ans =
    1. 0000   2. 0000   3. 0000
    0. 0000   1. 0000   5. 0000
    7.0000    4. 0000   1. 0000
```

Функція ***sqrtm***(A) обчислює таку матрицю Y, що Y*Y = A:

```
» Y = sqrtm(A)
Y =
    0.7884 + 0.8806i      0.6717 - 0.1795i      0.8029 - 0.4180i
   0. 8953 + 0. 6508i    0. 7628 + 0. 8620i    0. 9118 - 1. 0066i
    1.2765 - 1. 4092i      1. 0875 - 0. 5449i      1. 3000 + 1. 2525i

» Y * Y
ans =
   1. 0000 + 0. 0000i      2. 0000 - 0. 0000i      3. 0000 + 0. 0000i
   0. 0000 - 0. 0000i      1. 0000 - 0. 0000i      5. 0000 - 0. 0000i
    7.0000 + 0. 0000i      4. 0000 + 0. 0000i      1. 0000 + 0. 0000i
```

1.3.8. Завдання

Завдання 1.5. Обчисліть значення функції *f(x)* на відрізку [*a*; *b*] із кроком *h*.

Таблиця 1.3

Варіант	$f(x)$	*a*	*b*	*h*
1	$\frac{x^2}{1+0{,}25\sqrt{x}}$	1,1	3,1	0,2
2	$\frac{x^3-0{,}3x}{\sqrt{1+2x}}$	2,05	3,05	0,1
3	$\frac{2e^{-x}}{2\pi+x^3}$	0	1,6	0,16
4	$\frac{\cos \pi x^2}{\sqrt{1-3x}}$	-1	0	0,1
5	$\sqrt{1+4x}\sin \pi x$	0,1	0,8	0,07
6	$\frac{e^{x/3}}{1+x^2}$	1,4	2,4	0,1
7	$e^{-2x}+x^2-1$	0,25	2,25	0,2
8	$(e+x)\sin\left(\pi\sqrt{x-1}\right)$	1,8	2,8	0,1
9	$\sqrt{3+2x}\cdot tg\frac{\pi x^3}{2}$	0,1	0,9	0,08
10	$\sqrt{2+3x}\cdot\ln\left(1+3x^2\right)$	-0,1	0,9	0,1
11	$\sqrt[3]{x^2+3}\cdot\cos\frac{\pi x}{2}$	1	2,5	0,15
12	$(4+7x)\sin\left(\pi\sqrt[3]{1+x}\right)$	0	7	0,7
13	$e^{-x^2}\left(1+3x-x^2\right)$	0	2	0,2
14	$x^3-3x+\frac{8}{\sqrt{1+x^2}}$	0	1,7	0,17
15	$\sqrt{sh\sqrt{2\pi x}}$, $\left(sh\,x=\frac{e^x-e^{-x}}{2}\right)$	0	1,2	0,12
16	$\sqrt{ch\frac{x}{\sqrt{2\pi}}}$, $\left(ch\,x=\frac{e^x+e^{-x}}{2}\right)$	0,5	1,5	0,1

17	$\frac{x^3+2x}{\sqrt{1+e^x}}$	-0,2	0,8	0,1
18	$\sqrt{1+2x^2}\cdot\sin\frac{3x}{2}$	2	4	0,2
19	$\sqrt{3x^2+5}\cdot\cos\frac{\pi x}{2}$	0,5	1,5	0,1
20	$\arccos e^{-\sqrt[3]{3x}}$	0,2	0,5	0,03
21	$\arcsin e^{-x^2/5}$	8	13	0,5
22	$x+\ln\left(x+\sqrt{1+x^2}\right)$	-0,5	0,5	0,1
23	$\frac{1+e^{-x/2}}{\sqrt{3x^2+1}}$	3	5	0,2
24	$3x^3+\frac{1}{x}+e^{-2x^2}$	1,2	2,2	0,1
25	$x^{2x+1}+x^3-2x$	1	5	0,4

1.3.9. Запитання

1. Як уводяться вектори в мові Matlab? Якими функціями можна формувати вектори в мові Matlab?
2. Які функції Matlab дозволяють перетворювати вектор поелементно?
3. За допомогою яких засобів у Matlab здійснюються основні операції з векторами?
4. Як уводяться матриці в системі Matlab ?
5. Які функції є в Matlab для формування матриць визначеного виду?
6. Як сформувати матрицю:
   а) по заданих векторах її рядків?
   б) по заданих векторах її стовпчиків?
   в) по заданих векторах її діагоналей ?
7. Які функції поелементного перетворення матриці є в Matlab?
8. Як здійснюються в Matlab звичайні матричні операції ?
9. Як розв'язати в Matlab систему лінійних алгебричних рівнянь?

1.4. Функції прикладної чисельної математики

1.4.1. Операції з поліномами

В системі Matlab передбачені деякі додаткові можливості математичного оперування з поліномами.

Поліном (багаточлен) як функція визначається виразом :

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

В системі Matlab поліном задається й зберігається у виді вектора, елементами якого є коефіцієнти полінома від a_n до a_0 :

$$P = [a_n \ ... \ a_2 \ a_1 \ a_0].$$

Уведення полінома у Matlab здійснюється таким самим чином, як і введення вектора довжиною n+1, де n - порядок полінома.

Множення поліномів. Добутком двох поліномів степенів n і m відповідно, як відомо, називають поліном степеня n+m, коефіцієнти якого визначають простим перемножуванням цих двох поліномів. Фактично операція множення двох поліномів зводиться до побудови розширеного вектора коефіцієнтів по заданих векторах коефіцієнтів поліномів-співмножників. Цю операцію в математиці називають *згорткою векторів* (а самий вектор, одержуваний у результаті такої процедури - *вектором-згорткою двох векторів*). У Matlab її здійснює функція ***conv***(P1, P2).

Аналогічно, функція ***deconv***(P1, P2) здійснює ***ділення*** полінома P1 на поліном P2, тобто *обернену згортку векторів* P1 і P2. Вона визначає коефіцієнти полінома, що є часткою від ділення P1 на P2.

Приклад :

```
» p1 = [1,2,3];   p2 = [1,2,3,4,5,6];
» p = conv(p1,p2)
p =    1    4   10   16   22   28   27   18
» deconv(p,p1)
ans =    1    2    3    4    5    6
```

У загальному випадку ділення двох поліномів призводить до одержання двох поліномів - полінома-результату (частки) і полінома-остачі. Щоб одержати обидва ці поліноми, треба оформити звернення до функції у такий спосіб:

[Q,R] = ***deconv***(B,A) .

Тоді результат буде виданий у виді вектора Q з остачею у виді вектора R таким чином, що буде виконане співвідношення

B = conv(A,Q) + R.

Система Matlab має функцію ***roots***(P), що ***обчислює вектор, елементи якого є коренями заданого полінома*** P.

Нехай потрібно знайти корені полінома:

P(x) = x5 + 8x4 + 31x3 + 80x2 + 94x + 20.

Нижче показано, як просто це зробити :

```
» p = [1,8,31,80,94,20];
» disp(roots(p))
  -1.0000 + 3.0000i
  -1.0000 - 3.0000i
  -3.7321
```

```
  -2. 0000
  -0. 2679
```

Обернена операція - побудова вектора ***p*** коефіцієнтів полінома за заданим вектором його коренів - здійснюється функцією ***poly***:

p = poly(r).

Тут ***r -*** заданий вектор значень коренів, ***p -*** обчислений вектор коефіцієнтів полінома. Наведемо 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 =8. 0000   31. 0000   80. 0000   94. 0000   20. 0000
```

Зауважимо, що *одержуваний вектор не показує старшого коефіцієнта, який за замовчуванням покладається рівним одиниці*.

Ця ж функція у випадку, коли аргументом її є деяка квадратна матриця A розміром (n*n), *будує вектор характеристичного полінома цієї матриці*. Звернення

p = poly(A)

формує вектор ***p*** коефіцієнтів полінома

$p(s) = \det(s*E - A) = p1*sn + ... + pn*s + pn_{+1}$,

де E - позначення одиничної матриці розміром (n*n).

Розглянемо приклад:

```
» A = [1 2 3; 5 6 0; -1 2 3]
A =
    1    2    3
    5    6    0
   -1    2    3
» p = poly(A)
p =
   1. 0000  -10. 0000   20. 0000  -36. 0000
```

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

y = polyval(p,x),

де ***p*** - заданий вектор коефіцієнтів полінома, а ***x -*** задане значення аргументу. Приклад:

```
» y = polyval(p,2)
y = 936
```

Якщо як аргумент поліному зазначена матриця X, то функція ***polyval(p,X)*** обчислює матрицю Y, кожний елемент якої є значенням зазначеного полінома при значенні аргументу, рівному відповідному елементу матриці X, наприклад:

```
p =    1    8   31   80   94   20
» X = [1 2 3; 0 -1 3; 2 2 -1]
X =
    1    2    3
    0   -1    3
```

```
    2     2    -1
» disp(polyval(p,X))
       234        936       2750
        20        -18       2750
       936        936        -18
```

У цьому випадку функція обчислює значення полінома для кожного елемента матриці X, і тому розміри вхідної й вихідної матриць однакові size(Y) = size(X).

Обчислення похідної від полінома здійснюється функцією ***polyder.*** Ця функція створює вектор коефіцієнтів полінома, який є похідною від заданого полінома. Вона має три види звернень:

dp = polyder(p) за заданим поліномом ***p*** обчислює вектор ***dp***, елементи якого є коефіцієнтами полінома-похідної від заданого:

```
» dp = polyder(p)
dp =     5    32    93   160    94;
```

dp = polyder(p1,p2) обчислює вектор ***dp***, елементи якого є коефіцієнтами полінома-похідної від добутку двох поліномів ***p1*** і ***p2***:

```
»  p1 = [1,8,31,80,94,20];
»  p2 = [1,2,16];
» p = conv(p1,p2)
p =
  Columns 1 through 6
         1     10      63      270      750      1488
  Columns 7 through 8
      1544         320
» dp = polyder(p)
dp =
  Columns 1 through 6
         7     60      315      1080       2250       2976
  Column 7
      1544
» dp1 = polyder(p1,p2)
dp1 =
  Columns 1 through 6
         7     60      315      1080       2250       2976
  Column 7
      1544;
```

[***q,p***] = ***polyder(p1,p2)*** обчислює похідну від відношення (***p1/p2***) двох поліномів ***p1*** і ***p2*** і видає результат у виді відношення (***q/p***) поліномів ***q*** і ***p***:

```
»  p1 = [1,8,31,80,94,20];
» p2 = [1,2,16];
» [q,p] = polyder(p1,p2)
q =          3         24        159        636       1554       2520       1464
p =     1     4    36    64   256
» z = deconv (q,p)
z =     3    12     3
» y = deconv(p1,p2)
y =     1     6     3   -22
» z1 = polyder(y)
z1 =     3    12     3.
```

1.4.2. Обробка даних вимірів

Система Matlab дає користувачеві додаткові можливості для обробки даних, що задані у векторній або матричній формі.

Припустимо, що є деяка залежність ***y(x),*** яка задана рядом точок

x	2	4	6	8	10
y	5.5	6.3	6.8	8	8.6

Її можна задати в командному вікні Matlab як матрицю **xydata**, що містить два рядки - значення ***x*** і значення ***y***:

```
>> xydata =[2 4 6 8 10; 5.5 6.3 6.8 8 8.6]
xydata =
   2. 0000   4. 0000   6. 0000   8. 0000  10. 0000
   5. 5000   6. 3000   6. 8000   8. 0000   8. 6000
```

На прикладі цієї залежності розглянемо основні засоби для обробки даних.

Функція ***size***(***xydata***) призначена для визначення числа рядків і стовпчиків матриці ***xydata***. Вона формує вектор [n, p], який містить ці величини:

```
>> size(xydata)
ans =
    2    5
```

Звернення до неї виду

```
>> [n, p] = size(xydata);
```

дозволяє зберегти в пам'яті машини і використовувати потім при подальших обчисленнях дані про кількість рядків ***n*** і кількість стовпчиків ***p*** цієї матриці:

```
>> n, p
n =
    2
p =
    5
```

За допомогою цієї функції можна встановити довжину й тип (рядок або стовпчик) вектора :

```
» v = xydata(:)
v =
   2.0000
   5.5000
   4.0000
   6.3000
   6.0000
   6.8000
   8.0000
   8.0000
  10. 0000
   8. 6000
» n = size(v)
n =   10    1
» v1 = v'
v1 =
  Columns 1 through 7
   2. 0000   5. 5000   4. 0000   6. 3000   6. 0000   6. 8000   8. 0000
  Columns 8 through 10
   8. 0000  10. 0000   8. 6000
»  size(v')
ans =    1   10
```

Функція ***max***(V), де V - деякий вектор, видає значення максимального елемента цього вектора. Аналогічно, функція ***min***(V) витягає мінімальний елемент вектора V. Функції ***mean***(V) і ***std***(V) визначають, відповідно, середнє значення і середньоквадратичне відхилення від нього значень елементів вектора V.

Функція сортування ***sort***(V) формує вектор, елементи якого розподілені в порядку зростання їхніх значень.

Функція ***sum***(V) обчислює суму елементів вектора V.

Функція ***prod***(V) видає добуток усіх елементів вектора V.

Функція ***cumsum***(V) формує вектор того ж типу й розміру, будь-який елемент якого є сумою всіх попередніх елементів вектора V (вектор кумулятивної суми).

Функція ***cumprod***(V) створює вектор, елементи якого є добутком усіх попередніх елементів вектора V.

Функція ***diff***(V) видає вектор, що має розмір на одиницю менший за розмір вектора V, елементи якого є різницею між суміжними елементами вектора V.

Застосування описаних функцій проілюстровано нижче.

```
» v = [ 1, 0.1, 0.5, 0.1, 0.1,0.4 ];
» disp(size(v))
    1    6
» disp(max(v))
    1
» disp(min(v))
   0. 1000
» disp(mean(v))
   0. 3667
» disp(std(v))
   0. 3559
» disp(sort(v))
   0. 1000   0. 1000   0. 1000   0. 4000   0. 5000   1. 0000
» disp(sum(v))
   2. 2000
» disp(prod(v))
  2. 0000e-004
» disp(cumsum(v))
   1. 0000   1. 1000   1. 6000   1. 7000   1. 8000   2. 2000
» disp(cumprod(v))
   1. 0000   0. 1000   0. 0500   0. 0050   0. 0005   0. 0002
» disp(diff(v))
  -0. 9000   0. 4000  -0. 4000        0   0. 3000
```

Якщо вказати другий вихідний параметр, то можна одержати додаткову інформацію про індекс першого елемента, значення якого є максимальним або мінімальним:

```
>> [M,n]=max(v)
M =    1
n =    1
>> [N,m]=min(v)
N =   0.1000
m =    2
```

Інтегрування методом трапецій здійснює процедура ***trapz***. Звернення до неї вигляду ***trapz***(x,y) призводить до обчислення площі під графіком

функції y(x), у якому всі точки, задані векторами x і y, з'єднані відрізками прямих. Якщо перший вектор x не зазначений у зверненні, за умовчанням припускається, що крок інтегрування дорівнює одиниці (тобто вектор x є вектором із номерів елементів вектора y).

Приклад. Обчислимо інтеграл від функції y = sin(x) у діапазоні від 0 до π. Його точне значення дорівнює 2. Візьмемо рівномірну сітку із 100 елементів. Тоді обчислення зведуться до сукупності операцій:

```
» x = 0 : pi/100 : pi;
» y = sin(x);
» disp(trapz(x,y))
   1. 9998
```

Ті ж функції ***size, max, min, mean, std, sort, sum, prod, cumsum, cumprod, diff*** можуть бути застосовані і до матриць**.** Основною відмінністю використання як аргументів цих функцій саме матриць є те, що відповідні описані вище операції провадяться не по відношенню до рядків матриць, а до кожного зі стовпців заданої матриці. Тобто кожний стовпець матриці A розглядається як змінна, а кожний рядок - як окреме спостереження. Так, у результаті застосування функцій ***max, min, mean, std*** утворюються вектори-рядки з кількістю елементів, яка дорівнює кількості стовпців заданої матриці. Кожний елемент містить, відповідно, максимальне, мінімальне, середнє або середньоквадратичне значення елементів відповідного стовпця заданої матриці.

Наведемо приклади. Нехай маємо 3 величини y1, y2 і y3, що виміряні за деяких п'яти значень аргументу (які не зазначені). Тоді дані вимірів утворять 3 вектори по 5 елементів:

```
>> y1 = [ 5.5 6.3 6.8 8 8.6];
>>  y2 = [-1. 2 0.5 -0. 6 1 0.1];
>>  y3 = [ 3.4 5.6 0 8.4 10.3] ;  .
```

Сформуємо з них матрицю вимірів так, щоб вектори y1, y2, y3 утворювали стовпці цієї матриці:

```
» A = [ y1', y2', y3']
A =
   5.5000   -1.2000   3.4000
   6.3000    0.5000   5.6000
   6.8000   -0.6000    0
   8. 0000   1. 0000   8. 4000
   8. 6000   0. 1000  10. 3000
```

Застосуємо до цієї матриці вимірів описані вище функції. Одержимо

```
» size(A)
ans =    5    3
» max(A)
ans =   8. 6000   1. 0000  10. 3000
» min(A)
ans =   5. 5000  -1. 2000        0
» mean(A)
ans =   7. 0400  -0. 0400   5. 5400
   » std(A)
   ans =   1. 2582   0. 8735   4. 0655
```

*Якщо при зверненні до функцій **max** і **min** зазначити другий вихідний параметр, то він дасть інформацію про номер рядка, де знаходиться у відповідному стовпчику перший елемент із максимальним (або мінімальним) значенням*. Наприклад:

```
>> [M,n]=max(A)
M =    8.6000 1.0000 10.3000
n =     5     4     5
>> [N,m]=min(A)
N =    5.5000 -1.2000 0
m =     1     1     3
```

Функція ***sort*** сортує елементи кожного зі стовпців матриці. Результатом є матриця того ж розміру.

Функція ***sum*** і ***prod*** формують вектор-рядок, кожний елемент якого є сумою або добутком елементів відповідного стовпця початкової матриці.

Функції ***cumsum, cumprod*** утворять матриці того самого розміру, елементи кожного стовпця яких є сумою або добутком елементів цього ж стовпця початкової матриці, починаючи з відповідного елемента і вище.

Нарешті, функція ***diff*** створює із заданої матриці розміром (m*n) матрицю розміром ((m-1)*n), елементи якої є різницею між елементами суміжних рядків початкової матриці.

Застосовуючи ці процедури до тієї самої матриці вимірів, одержимо:

```
» sort(A)
ans =
   5.5000    -1.2000      0
   6.3000    -0.6000    3.4000
   6.8000     0.1000     5.6000
   8. 0000    0. 5000    8. 4000
   8. 6000    1. 0000   10. 3000
» sum(A)
ans =   35. 2000   -0. 2000   27. 7000
» prod(A)
ans =  1. 0e+004 *
   1. 6211    0. 0000         0
» cumsum(A)
ans =
   5.5000    -1.2000      3.4000
  11.8000    -0.7000      9.0000
  18.6000    -1.3000      9.0000
  26. 6000   -0. 3000   17. 4000
  35. 2000   -0. 2000   27. 7000
» cumprod(A)
ans =  1.0e+004 *
   0.0006    -0.0001      0.0003
   0.0035    -0.0001      0.0019
   0.0236     0.0000         0
   0. 1885    0. 0000         0
   1. 6211    0. 0000         0
» diff(A)
ans =
   0.8000     1.7000     2.2000
   0. 5000   -1. 1000   -5. 6000
   1. 2000    1. 6000    8. 4000
   0.6000    -0. 9000    1. 9000
```

Розглянемо деякі інші функції, надані користувачеві системою Matlab.

Функція ***cov***(A) обчислює *матрицю коваріацій* вимірів. При цьому утворюється квадратна симетрична матриця з кількістю рядків і стовпчиків, рівним кількості виміряних величин, тобто кількості стовпчиків матриці вимірів.

Наприклад, при застосуванні до прийнятої матриці вимірів вона дає такий результат:

```
» cov(A)
ans =
   1. 5830   0. 6845    3. 6880
   0. 6845   0. 7630    2. 3145
   3.6880    2. 3145   16. 5280
```

На діагоналі матриці коваріацій розміщені *дисперсії* виміряних величин, а поза нею - *взаємні кореляційні моменти* цих величин.

Функція ***corrcoeff***(A) обчислює *матрицю коефіцієнтів кореляції* за тих самих умов. Елементи матриці S = corrcoef(A) пов'язані з елементами матриці коваріацій C=cov(A) таким співвідношенням:

$$S(k,l) = \frac{C(k,l)}{\sqrt{C(k,k)\cdot C(l,l)}}$$

Приклад:

```
» corrcoef(A)
ans =
   1. 0000   0. 6228   0. 7210
   0. 6228   1. 0000   0. 6518
   0.7210    0. 6518   1. 0000
```

1.4.3. Функції лінійної алгебри

Традиційно до лінійної алгебри відносять такі задачі, як обернення і псевдообернення матриці, спектральне й сингулярне розкладання матриць, обчислення власних значень і векторів, сингулярних чисел матриць, обчислення функцій від матриць. Коротко ознайомимося з деякими основними функціями Matlab у цій області.

Функція ***k = cond***(A) обчислює й видає число обумовленості матриці стосовно операції обернення, яке дорівнює відношенню максимального сингулярного числа матриці до мінімального.

Функція ***k = norm***(v,p) обчислює p-норму вектора v за формулою:

k = ***sum***(***abs***(v) . ^p)^(1/p),

де p - ціле додатне число. Якщо аргумент p при зверненні до функції не зазначений, обчислюється 2-норма.

Функція ***k = norm***(A,p) обчислює p-норму матриці A за формулою:

k = ***max(sum***(***abs***(A) . ^p))^(1/p),

де p = 1,2, 'fro' або inf. Якщо аргумент p не зазначений, обчислюється 2-норма. При цьому є слушними співвідношення:

norm(A,1) = max(sum(abs(A)));

norm(A,inf) = max(sum(abs(A')));

*norm(A,'fro') = sqrt(sum(diag(A'*A)));*

norm(A) = norm(A,2) = $\sigma_{max}(A)$.

Функція ***rd = rcond***(A) обчислює величину, обернену значенню числа обумовленості матриці A щодо 1-норми. Якщо матриця A добре обумовлена,

значення ***rd*** близько до одиниці. Якщо ж вона погано обумовлена, ***rd*** наближається до нуля.

Функція ***r = rank***(A) обчислює ранг матриці, який визначається як кількість сингулярних чисел матриці, що перевищують поріг

*max(size(A))*norm(A)*eps.*

Наведемо приклади застосування цих функцій:

```
A =
    1    2    3
    0    1    5
    7    4    1
» disp(cond(A))
   13. 8032
» disp(norm(A,1))
    9
» disp(norm(A))
    8. 6950
» disp(rcond(A))
    0. 0692
» disp(rank(A))
    3
```

Процедура ***d = det***(A) обчислює *визначник квадратної матриці* на основі трикутного розкладання методом виключення Гаусса.

Функція ***t = trace***(A) обчислює *слід матриці* A, який дорівнює сумі її діагональних елементів.

Q = null(A) обчислює ортонормований *базис нуль-простору* матриці A.

Q = orth(A) видає *ортонормований базис матриці* A.

Процедура R = ***rref***(A) здійснює *приведення матриці до трикутного виду на основі методу виключення Гаусса* з частковим вибором провідного елемента.

Приклади:

```
» disp(det(A))
   30
» disp(trace(A))
    3
» disp(null(A))
» disp(orth(A))
   0. 3395    0. 4082   -0. 8474
   0. 2793    0. 8165    0. 5053
   0.8982   -0. 4082    0. 1632
» disp(rref(A))
                    1    0    0
                    0    1    0
                    0    0    1
```

Функція R=***chol***(A) здійснює ***розкладання Холецького*** для дійсних симетричних і комплексних ермітових матриць. Наприклад:

```
» A = [ 1 2 3; 2 15 8; 3 8 400]
A =
    1    2    3
    2   15    8
    3    8  400
» disp(chol(A))
   1. 0000    2. 0000    3. 0000
      0       3. 3166    0. 6030
      0          0      19. 7645
```

Функція ***lu***(A) здійснює ***LU-розкладання*** матриці A в виді добутку нижньої трикутної матриці L (можливо, із перестановками) і верхньої трикутної матриці U так, що A = L * U.

Звернення до цієї функції виду

[L, U, P] = ***lu***(A)

дозволяє одержати три складові цього розкладання - нижню трикутну матрицю L, верхню трикутну U і матрицю перестановок P такі, що

P * A =L * U.

Наведемо приклад:

```
A =
     1     2     3
     2    15     8
     3     8   400
» disp(lu(A))
   3.0000     8.0000  400.0000
  -0. 6667    9. 6667 -258. 6667
  -0. 3333    0. 0690 -148. 1724
» [ L, U, P] = lu(A);
» L
L =
   1.0000         0          0
   0. 6667    1. 0000        0
   0. 3333   -0. 0690    1. 0000
» U
U =
   3.0000  8.0000   400.0000
        0     9. 6667 -258. 6667
        0         0 -148. 1724
» P
P =
    0    0    1
    0    1    0
    1    0    0
```

З нього випливає, що в першому, спрощеному варіанті звернення функція видає комбінацію з матриць L і U.

Обернення матриці здійснюється за допомогою функції ***inv***(A):

```
» disp(inv(A))
   1. 3814   -0. 1806   -0. 0067
  -0. 1806    0. 0910   -0. 0005
  -0. 0067   -0. 0005    0. 0026
```

Процедура ***pinv***(A) знаходить матрицю, *псевдообернену* матриці A, яка має розміри матриці A' і задовольняє умови

A * P * A = A; P * A * P = P.

Наприклад:

```
A =
    1    2    3    4    5
    5   -1    4    6    0
» P = pinv(A)
P =
  -0.0423    0.0852
   0.0704   -0.0480
   0.0282    0.0372
   0. 0282   0. 0628
   0. 1408  -0. 0704
```

```
» A*P*A,                    % перевірка 1
ans =
   1. 0000   2. 0000   3. 0000   4. 0000   5. 0000
   5. 0000  -1. 0000   4. 0000   6. 0000   0. 0000
» P*A*P                     % перевірка 2
ans =
  -0.0423    0.0852
   0.0704   -0.0480
   0. 0282   0. 0372
   0. 0282   0. 0628
   0.1408  -0. 0704
```

Для квадратних матриць ця операція рівнозначна звичайному оберненню.

Процедура [Q, R, P] = ***qr***(A) здійснює розкладення матриці A на три - унітарну матрицю Q, верхню трикутну R із діагональними елементами, що зменшуються за модулем, і матрицю перестановок P такі що

A * P = Q * R.

Наприклад:

```
A =
    1    2    3    4    5
    5   -1    4    6    0
» [Q,R,P] = qr(A)
Q =
  -0.5547 -0.8321
  -0.8321 0.5547
R =
  -7.2111 -2.7735 -4.9923 -4.7150   -0.2774
        0   -4.1603 -0.2774  1.9415   -2.2188
P =
    0    0    0    1    0
    0    0    0    0    1
    0    0    1    0    0
    1    0    0    0    0
    0    1    0    0    0
```

Визначення характеристичного полінома матриці A можна здійснити за допомогою функції ***poly***(A). Звернення до неї виду *p=**poly**(A)* дає можливість знайти вектор-рядок *p* коефіцієнтів характеристичного полінома

$p(s) = det(s*E - A) = p_1*s^n + ... + p_n*s + p_{n+1}$,

де *E* - позначення одиничної матриці розміром (n*n). Наприклад :

```
» A = [1 2 3; 5 6 0; -1 2 3]
A =
    1    2    3
    5    6    0
   -1    2    3
» p = poly(A)
p =
   1. 0000  -10. 0000   20. 0000  -36. 0000
```

Обчислення власних значень і власних векторів матриці здійснює процедура ***eig***(A). Звичайне звернення до неї дозволяє одержати вектор власних значень матриці A, тобто коренів характеристичного полінома матриці. Якщо ж звернення має вид:

[R, D] = ***eig***(A),

то в результаті одержують діагональну матрицю D власних значень і матрицю R правих власних векторів, що задовольняють умові

A*R = R * D.

Ці вектори є внормованими таким чином, що норма кожного з них дорівнює одиниці. Наведемо приклад:

```
A =
    1    2    3
   -1    8   16
   -5  100    3
» disp(eig(A))
   1.2234
  45. 2658
 -34. 4893
» [ R,D] = eig(A)
R =
   0.9979 -0.0798 -0.0590
   0.0492 -0.3915 -0.3530
   0.0416 -0.9167 0.9338
D =
   1. 2234        0          0
        0     45. 2658       0
        0          0      -34. 4893
```

Сингулярне розкладання матриці робить процедура ***svd***(A). Спрощене звернення до неї дозволяє одержати сингулярні числа матриці A. Більш складне звернення виду:

[U, S, V] = ***svd***(A)

дозволяє одержати три матриці - U, що сформована з ортонормованих власних векторів, які відповідають найбільшим власним значенням матриці $A*A^T$; V - з ортонормованих власних векторів матриці A^T*A и S - діагональну матрицю, яка містить *невід'ємні значення квадратних коренів із власних значень матриці* A^T*A (їх *називають сингулярними числами*). Ці матриці задовольняють співвідношенню:

$$A = U * S * VT.$$

Розглянемо приклад:

```
» disp(svd(A))
  100.5617
   15. 9665
    1. 1896
» [U,S,V] = svd(A)
U =
  -0.0207 0.1806 -0.9833
  -0.0869 0.9795 0.1817
  -0.9960 -0.0892 0.0045
S =
  100.5617 0 0
        0 15.9665 0
        0 0 1.1896
V =
   0. 0502   -0. 0221   -0. 9985
  -0. 9978   -0. 0453   -0. 0491
  -0. 0442    0. 9987   -0. 0243
```

Приведення матриці до форми Гессенберга здійснюється процедурою ***hess***(A). Наприклад:

```
A =
    1    2    3
   -1    8   16
   -5  100    3
» disp(hess(A))
```

```
   1. 0000   -3. 3340   -1. 3728
   5. 0990   25. 5000   96. 5000
      0      12. 5000  -14. 5000
```

Більш розгорнуте звернення [P,H] = ***hess***(A) дає можливість одержати, окрім матриці H в верхній формі Гессенберга, також унітарну матрицю перетворень P, яка задовольняє умови:

A = P * H * P; P' * P = eye(size(A)).

Приклад:

```
» [P,H] = hess(A)
P =
   1.0000      0         0
      0    -0.1961   -0.9806
      0    -0.9806    0.1961
H =
   1.0000    -3.3340   -1.3728
   5. 0990   25. 5000   96. 5000
      0      12. 5000  -14. 5000
```

Процедура ***schur*** (A) призначена для ***приведення матриці до форми Шура.*** Спрощене звернення до неї призводить до одержання матриці у формі Шура.

Комплексна форма Шура *- це верхня трикутна матриця із власними значеннями на діагоналі*. ***Дійсна форма Шура*** *зберігає на діагоналі тільки дійсні власні значення, а комплексні зображуються у виді блоків (2*2)*, частково займаючи нижню піддіагональ.

Звернення [U,T] = ***schur***(A) дозволяє, крім матриці T Шура, одержати також унітарну матрицю U, що задовольняє умовам:

A = U * H * U'; U' * U = eye(size(A)).

Якщо початкова матриця A є дійсною, то результатом буде *дійсна форма Шура*, якщо ж комплексною, то результат видається у виді *комплексної форми Шура*.

Наведемо приклад:

```
» disp(schur(A))
   1.2234 -6.0905    -4.4758
      0    45. 2658   84. 0944
      0     0. 0000  -34. 4893
» [U,T] = hess(A)
U =
    1.0000      0        0
      0    -0.1961  -0.9806
      0    -0.9806   0.1961
T =
   1. 0000   -3. 3340   -1. 3728
   5. 0990   25. 5000   96. 5000
      0      12. 5000  -14. 5000
```

Функція [U,T] = ***rsf2csf***(U,T) перетворює дійсну квазитрикутну форму Шура в комплексну трикутну:

```
» [U,T] = rsf2csf(U,T)
U =
  -0.9934 -0.1147    0
  -0.0449  0.3892   -0.9201
  -0.1055  0.9140    0.3917
T =
   1. 4091   -8. 6427    10. 2938
```

```
     0      45. 1689   -83. 3695
     0            0      -34. 5780
```

Процедура [AA, BB, Q, Z, V] = ***qz***(A,B) приводить ***пару матриць*** A і B до ***узагальненої форми Шура***. При цьому AA й BB є комплексними верхніми трикутними матрицями, Q, Z - матрицями приведення, а V - вектором узагальнених власних векторів такими, що

Q * A * Z = AA; Q * B * Z = BB.

Узагальнені власні значення можуть бути знайдені, виходячи з такої умови:

A * V * diag(BB) = B * V * diag(AA).

Необхідність в одночасному приведенні пари матриць до форми Шура виникає в багатьох задачах лінійної алгебри - розв'язуванні матричних рівнянь Сильвестра і Ріккаті, змішаних систем диференціальних і лінійних алгебричних рівнянь.

Приклад.

Нехай задана система звичайних диференціальних рівнянь у неявній формі Коші з одним входом u і одним виходом y такого виду:

$$\boldsymbol{Q}\cdot\dot{\boldsymbol{x}}+\boldsymbol{R}\cdot\boldsymbol{x}=\boldsymbol{b}\cdot u;$$

$$y=\boldsymbol{c}\cdot x+d\cdot u$$

причому матриці $\boldsymbol{Q}$, $\boldsymbol{R}$ і вектори $\boldsymbol{b}$, $\boldsymbol{c}$ і $\boldsymbol{d}$ дорівнюють відповідно

```
Q =
   1. 0000        0
     0.1920     1. 0000
R =
   1. 1190   -1. 0000
     36.4800   1. 5380
b =
  31. 0960
   0. 1284
c =
         0.6299       0
d =   -0. 0723
```

Необхідно обчислити значення полюсів і нулів відповідної передатної функції.

Ця задача зводиться до відшукування власних значень λ, що задовольняють матричні рівняння:

$$R\cdot r=-\lambda\cdot Q\cdot r;$$

$$\begin{bmatrix}-R & b\\ c & d\end{bmatrix}\cdot r=\lambda\cdot\begin{bmatrix}Q & 0\\ 0 & 0\end{bmatrix}\cdot r.$$

Розв'язання першого рівняння дозволяє ***обчислити полюси передатної функції***, а другого - ***нулі.***

Нижче наведено сукупність операторів, яка призводить до *розрахунку полюсів:*

```
» [AA, BB] = qz(R,-Q)   % Приведення матриць до форми Шура
AA =
   5.5039 + 2.7975i      24.8121 -25.3646i
   0.0000 - 0.0000i       5.5158 - 2.8036i
```

```
BB =
 -0. 6457 + 0. 7622i  -0. 1337 + 0. 1378i
          0                -0. 6471 - 0. 7638i
» diag(AA) . /diag(BB)  % Розрахунок полюсів
ans =
 -1. 4245 - 6. 0143i
 -1. 4245 + 6. 0143i
```

Розрахунок нулів здійснюється в такий спосіб:

```
» A = [-R                b                    % Формування
       c        d]                  % першої матриці
A =
                -1.1190     1.0000     0.1284
               -36. 4800   -1. 5380   31. 0960
     0.6299         0      -0. 0723
» B = [ -Q               zeros(size(b))       % Формування
        zeros(size(c))      0          ]      % другої матриці
B =
  -1.0000         0          0
  -0. 1920   -1. 0000        0
       0          0          0

» [AA,BB] = qz(A,B)                  % Приведення матриць   до форми Шура
AA =
  31.0963 -0.7169 -36.5109
   0.0000  1.0647   0.9229
        0  0.0000   0.5119
BB =
        0   0.9860  -0.2574
        0   0. 0657   0. 9964
        0        0  -0. 0354
» diag(AA) . /diag(BB)               % Обчислення нулів
ans =
      Inf
  16. 2009
 -14. 4706
```

Обчислення ***власних значень матричного полінома*** здійснює процедура ***polyeig.*** Звернення

[R, d] = ***polyeig***(A0, A1,... , Ap)

дозволяє розв'язати повну проблему власних значень для матричного полінома ступеня p виду

(A0 + λ*A1 + ... + λ^p*Ap)*r = 0.

Вхідними змінними цієї процедури є p+1 квадратні матриці A0, A1, ... Ap порядку n. Вихідними змінними - матриця власних векторів R розміром (n*(n*p)) і вектор d власних значень розміром (n*p).

Функція ***polyvalm*** призначена для ***обчислення матричного полінома*** виду

$$Y(X) = p_n \cdot X^n + p_{n-1} \cdot X^{n-1} + ... + p_2 \cdot X^2 + p_1 \cdot X + p_0$$

за заданим значенням матриці X и вектора p = [pn, pn-1, ... , p0] коефіцієнтів полінома. Для цього достатньо звернутися до цієї процедури за схемою:

Y = ***polyvalm***(p, X).

Приклад:

```
p =    1    8   31   80   94   20
» X
```

```
X =
    1    2    3
    0   -1    3
    2    2   -1
» disp(polyvalm(p,X))
      2196       2214       2880
       882        864       1116
      1332       1332       1746
```

Примітка. *Слід розрізнювати процедури* ***polyval*** *і* ***polyvalm****. Перша обчислює значення поліному для кожного з елементів матриці аргументу, а друга при обчисленні полінома підносить до відповідного степеня всю матрицю аргументу.*

Процедура ***subspace***(А,У) обчислює кут між двома підпросторами, які "натягнуті на стовпчики" матриць А і В. Якщо аргументами є не матриці, а вектори А і В, обчислюється кут між цими векторами.

1.4.4. Апроксимація та інтерполяція даних

Поліноміальна апроксимація даних вимірів, що сформовані як деякий вектор Y, при деяких значеннях аргументу, які утворюють вектор X такої ж довжини, що й вектор Y, здійснюється процедурою ***polyfit***(X, Y, n). Тут n - порядок апроксимуючого полінома. Результатом дії цієї процедури є вектор довжиною (n +1) із коефіцієнтів апроксимуючого полінома.

Нехай масив значень аргументу є таким:

x = [1 2 3 4 5 6 7 8],

а масив відповідних значень виміряної величини - таким:

y = [-1. 1 0.2 0.5 0.8 0.7 0.6 0.4 0.1].

Тоді, застосовуючи зазначену функцію при різних значеннях порядку апроксимуючого полінома, одержимо:

```
» 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];
» polyfit(x,y,1)
ans =    0. 1143   -0. 2393
» polyfit(x,y,2)
ans =   -0. 1024    1. 0357   -1. 7750
» polyfit(x,y,3)
ans =    0. 0177   -0. 3410    1. 9461   -2. 6500
» polyfit(x,y,4)
ans =   -0. 0044    0. 0961   -0. 8146    3. 0326   -3. 3893.
```

Це означає, що задану залежність можна апроксимувати або прямою

$$y(x) = 0{,}1143x - 0{,}2393,$$

або квадратною параболою

$$y(x) = -0{,}1024x^2 + 1{,}0357x - 1{,}775,$$

або кубічною параболою

$$y(x) = 0{,}0177x^3 - 0{,}341x^2 + 1{,}9461x - 2{,}65,$$

або параболою четвертого степеня

$$y(x) = -0{,}0044x^4 + 0{,}0961x^3 - 0{,}8146x^2 + 3{,}0326x - 3{,}3893.$$

Побудуємо в одному графічному полі графіки заданої дискретної функції й графіки всіх отриманих при апроксимації поліномів:

```
 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];
 P1=polyfit(x,y,1) ;
 P2=polyfit(x,y,2);
 P3=polyfit(x,y,3);
 P4=polyfit(x,y,4) ;
stem(x,y);
x1 = 0.5 : 0.2 : 8.5;
y1=polyval(P1,x1);
y2=polyval(P2,x1);
y3=polyval(P3,x1);
y4=polyval(P4,x1);
hold on
plot(x1,y1,'.-',x1,y2,'x-',x1,y3,'o-',x1,y4,'^-'),
grid, set(gca, 'FontName', 'Arial Cyr', 'FontSize', 16),
title('Поліноміальна апроксимація ');
xlabel('Аргумент');
ylabel('Функція')
legend(' Задані значення','1','2','3','4',0)
```

Результат поданий на рис. 1.18.

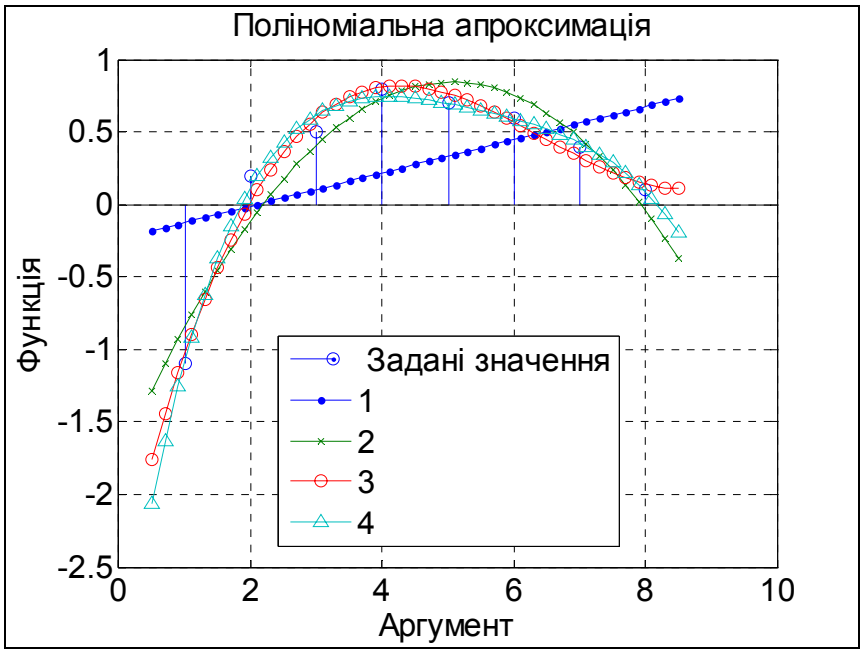


Рис. 1.18. Поліноміальна апроксимація функцією ***polyfit***

Функція ***spline***(X,Y,Xi) здійснює ***інтерполяцію кубічними сплайнами***. При зверненні

Yi = ***spline***(X,Y,Xi)

вона інтерполює значення вектора Y, заданого при значеннях аргументу, поданих у векторі X, і видає значення інтерполюючої функції у виді вектора Yi при значеннях аргументу, заданих вектором Xi. У випадку, коли вектор X не зазначений, за замовчуванням приймається, що він має довжину вектора Y і кожний його елемент дорівнює номеру цього елемента.

Як приклад розглянемо інтерполяцію вектора

```
 x = -0.5:0.1:0.2;
```

```
y = [-1.1 0.2 0.5 0.8 0.7 0.6 0.4 0.1];
x1 = -0.5:0.02:0.2;
y2 = spline(x,y,x1);
plot (x,y,x1,y2,'.-'), grid
set(gca,'FontName','Arial Cyr','FontSize',16),
title('Інтерполяція процедурою SPLINE ');
xlabel('Аргумент');
ylabel('Функція')
legend('лінійна','сплайнова',0)
```

Результат наведений на рис. 1.19.

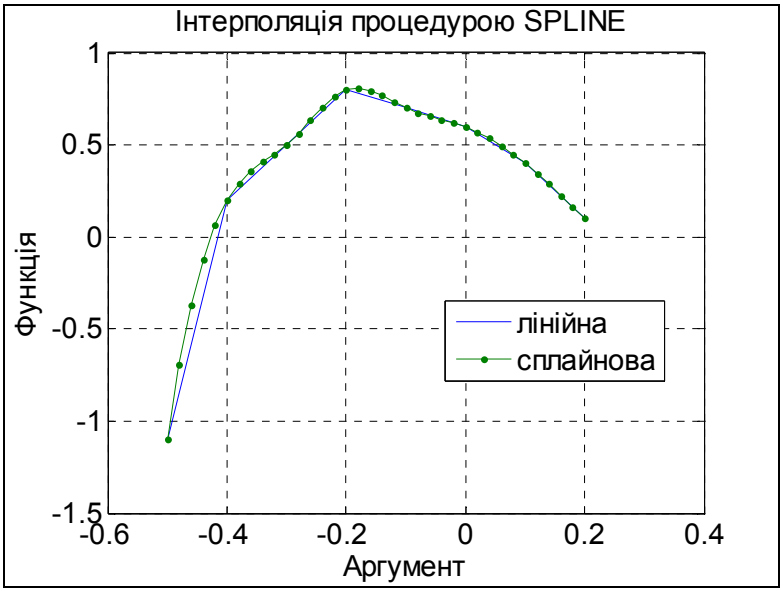


Рис. 1.19. Інтерполяція функцією ***spline***

Одновимірну табличну інтерполяцію робить процедура ***interp1***. Звернення до неї у загальному випадку має вид:

Yi = ***interp1***(X,Y,Xi,'<метод>'),

і дозволяє додатково зазначити метод інтерполяції у четвертому вхідному аргументі:

'nearest' - ступінчаста інтерполяція;

'linear' - лінійна;

'cubic' - кубічна;

'spline' - кубічними сплайнами.

Якщо метод не зазначений, здійснюється за умовчанням лінійна інтерполяція. Наприклад, (для того самого вектора):

```
x = -0.5:0.1:0.2;
y = [-1.1 0.2 0.5 0.8 0.7 0.6 0.4 0.1];
x1 = -0.5:0.02:0.2;
y1 = interp1(x,y,x1);
y4 = interp1(x,y,x1,'nearest');
y2 = interp1(x,y,x1,'cubic');
y3 = interp1(x,y,x1,'spline');
%plot (x1,y1,x1,y2,x1,y3,x1,y4), grid
plot (x1,y1,x1,y2,'.',x1,y3,x1,y4), grid
legend('лінійна','кубічна','сплайнова','ступінчаста')
set(gca,'FontName','Arial Cyr','FontSize',16),
title('Інтерполяція процедурою INTERP1 ');
xlabel('Аргумент');     ylabel('Функція')
```

Результат наведений на рис.1.20.

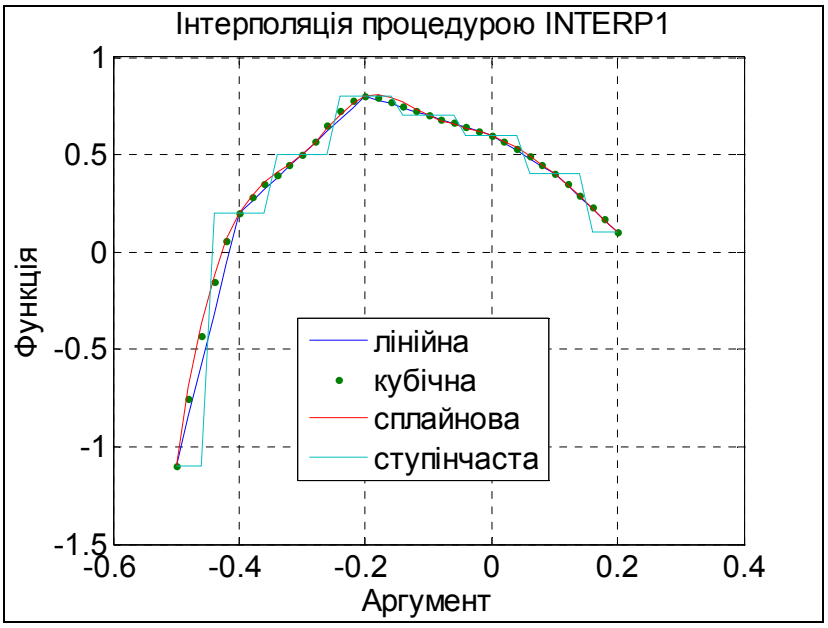


Рис. 1.20. Інтерполяція функцією ***interp1***

1.4.5. Векторна фільтрація й спектральний аналіз

У системі Matlab є декілька функцій для проведення цифрового аналізу даних спостережень (вимірів).

Так, функція ***y = filter(b,a,x)*** забезпечує формування вектора ***y*** по заданих векторах ***b, a, x*** відповідно до співвідношення:

$$y(k) = b(1)*x(k) + b(2)*x(k-1) + ... + b(nb+1)*x(k-nb) - a(2)*y(k-1) - a(3)*y(k-3) - ... - a(na+1)*y(k-na), \quad (1)$$

де вектор ***b*** має такий склад

$$\boldsymbol{b} = [\, b(1),b(2),... ,b(nb+1)],$$

а вектор ***a***

$$\boldsymbol{a} = [1,a(2),a(3),... ,a(na+1)].$$

Співвідношення (1) можна розглядати як кінцево-різницеве рівняння фільтра з дискретною передатною функцією виду раціонального дробу, коефіцієнти чисельника якого утворюють вектор ***b***, а знаменника - вектор ***a***, на вхід якого подається сигнал *x(t)*, а на виході формується сигнал *y(t)*.

Тоді вектор ***y*** буде являти собою значення вихідного сигналу цього фільтра в дискретні моменти часу, що відповідають заданим значенням вхідного сигналу *x(t)* (вектор ***x***).

Нижче наведений приклад застосування функції ***filter.***

```
» x = 0:0.1:1;
» b = [1 2];
» a = [ 1 0.1 4];
» y = filter(b,a,x)
y =
  Columns 1 through 7
        0    0.1000    0.3900    0.2610   -0.5861    0.3146    3.9129
  Columns 8 through 11
```

0.2503 -13.4768 2.8466 56.4225

Функції ***fft*** (*Fast Fourier Transformation*) і ***ifft*** (*Invers Fast Fourier Transformation*) здійснюють перетворення заданого вектора, що відповідають дискретному прямому й оберненому перетворенням Фур'є.

Звернення до цих функцій виду:

y = fft (x, n); ***x = ifft (y, n)***

призводить до формування вектора ***y*** у першому випадку і ***x*** - у другому по формулах:

$$y(k) = \sum_{m=1}^{n} x(m) \cdot e^{-j \cdot 2\pi \cdot (m-1) \cdot (k-1)/n}; \qquad (2)$$

$$x(m) = \frac{1}{n} \sum_{k=1}^{n} y(k) \cdot e^{j \cdot 2\pi \cdot (m-1) \cdot (k-1)/n}, \qquad (3)$$

де *j* - позначення уявної одиниці; *n* - число елементів заданого вектора ***x*** (воно є також розміром вихідного вектора ***y***).

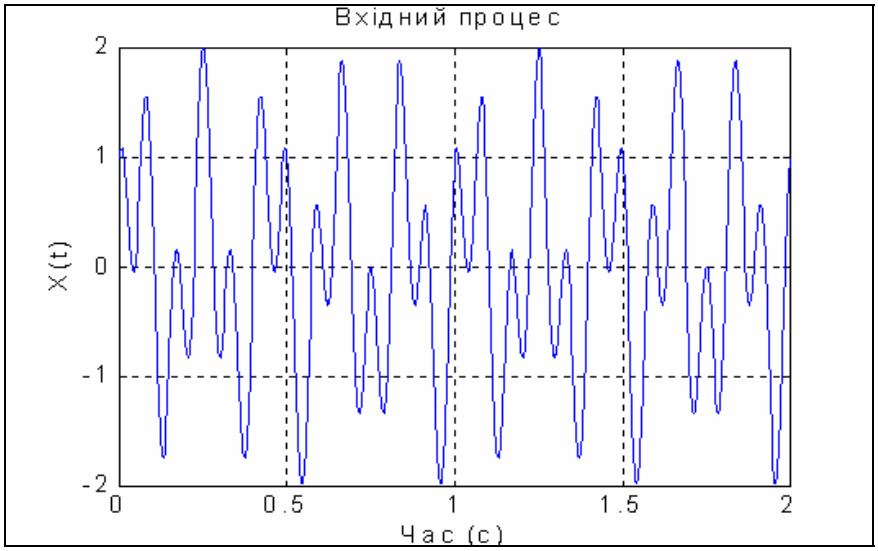


Рис. 1.21. Початковий процес

Наведемо приклад. Сформуємо вхідний сигнал у виді вектора, елементи якого дорівнюють значенням функції, що є сумою двох синусоїд із частотами 5 і 12 Гц. Знайдемо Фур'є-зображення цього сигналу і виведемо графічні подання вхідного процесу й модуля його Фур'є-зображення:

```
 t =0:0.001:2;
 x = sin(2*pi*5*t) + cos(2*pi*12*t);
plot(t, x); grid
 set(gca,'FontName','Arial Cyr','FontSize',16),
title('Вхідний процес ');
xlabel('Час (с)');
ylabel('X(t)')
 y = fft(x);
a =abs(y);
plot(a); grid
 set(gca,'FontName','Arial Cyr','FontSize',16),
title('Модуль Фурьє - зображення ');
xlabel('Номер елемента вектора');
ylabel('abs(F(X(t))')
```

Результати відображені відповідно на рис. 1.21 і 1.22.

*Рис. 1.22. Результат застосування процедури **fft***

Тепер здійснимо зворотне перетворення за допомогою функції ***ifft*** і результат також виведемо у формі графіка:

```
 z = ifft(y);
plot(t, z); grid
 set(gca,'FontName','Arial Cyr','FontSize',16),
title('Зворотне Фур''є-перетворення ');
xlabel('Час (с)');
ylabel('Z(t)')
```

На рис. 1.23 зображений результат. Розглядаючи його, можна переконатися, що відтворений процес збігається з початковим.

*Рис. 1.23. Результат застосування процедури **ifft***

Уважно вивчаючи формулу дискретного перетворення Фур'є, можна дійти висновків:

а) номер m відповідає моменту часу t_m, у який виміряний вхідний сигнал $x(m)$; при цьому $t_1 = 0$;

б) номер k - це індекс значення частоти f_k, якому відповідає знайдений елемент *y(k)* дискретного перетворення Фур'є;

в) щоб перейти від індексів до часової й частотної областей, треба знати значення h дискрету (кроку) часу, через який виміряний вхідний сигнал *x(t)* і проміжок T часу, протягом якого він вимірюється; тоді крок (дискрет) по частоті в зображенні Фур'є визначиться співвідношенням:

$$Df = 1/T, \qquad (4)$$

а діапазон змінювання частоти - формулою

$$F = 1/h; \qquad (5)$$

так, в аналізованому прикладі (h =0. 001, T = 2, n = 21)

$$Df = 0.5; \qquad F = 1000;$$

г) із (2) випливає, що індексу k = 1 відповідає нульове значення частоти (*fo = 0)*; інакше кажучи, перший елемент вектора $y(1)$ є значенням Фур'є-зображення при нульовій частоті, тобто є просто сумою всіх заданих значень вектора ***x***; звідси одержуємо, що вектор *y(k)* містить значення Фур'є-зображення, починаючи з частоти *fo = 0* (якій відповідає k = 1) до максимальної частоти *fmax = F* (якій відповідає $k = n$); таким чином, Фур'є-зображення визначається функцією ***fft*** тільки для додатних частот у діапазоні від 0 до F; це незручно для побудови графіків Фур'є-зображення від частоти; більш зручним і звичним є перехід до вектора Фур'є-зображення, якого визначено в діапазоні частот від (-*F/2*) до *F*/2; частота $F_N = F / 2$ одержала назву *частоти Найквіста*;

д) як відомо, функція $e^{j \cdot z}$ є періодичною за z із періодом 2π; тому інформація про Фур'є-зображення при від'ємних частотах розташована в другій половині вектора *y(k)*.

Сформуємо для аналізованого прикладу масив частот, виходячи з вищезазначеного:

```
f = 0:0. 5 : 1000;
```

і виведемо графік з аргументом-частотою (рис. 1.24):

```
plot(f,a); grid
set(gca,'FontName','Arial Cyr','FontSize',16),
title('Модуль Фур'є - зображення ');
xlabel('Частота (Гц)');
ylabel('abs(F(X(t))')
```

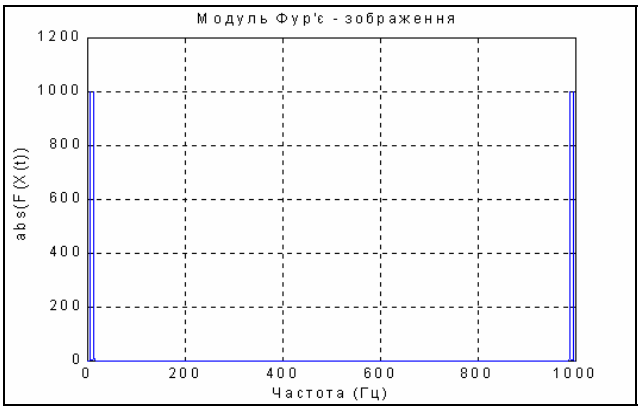


*Рис. 1.24. Результат застосування функції **fft** у частотній області*

Як випливає з розгляду рис. 1.24, за ним важко розпізнати ті частоти (5 і 12 Гц), із якими змінюється вхідний сигнал. Це - наслідок тієї обставини, яку було відзначено в примітці г). Щоб визначити справжній спектр вхідного сигналу, потрібно спочатку дещо перетворити отриманий вектор ***y*** Фур'є-зображення за допомогою процедури ***fftshift.***

Функція ***fftshift*** (звернення до неї здійснюється в такий спосіб: ***z*** = ***fftshift***(***y***)) призначена для формування нового вектора ***z*** із заданого вектора ***y*** шляхом переставлення другої половини вектора ***y*** у першу половину вектора ***z*** . При цьому друга половина вектора ***z*** складається з елементів першої половини вектора ***y***. Більш точно цю операцію можна задати співвідношеннями:

$$z(1) = y(n/2+1);\ \ldots\ ,\ z(k) = y(n/2+k);\ldots\ , z(n/2) = y(n);\ z(n/2+1) = y(1);\ldots$$
$$\ldots,\ z(n/2+k) = y(k);\ldots\ z(n) = y(n/2).$$

Примітка. Операцію ***fftshift*** зручно використовувати для *визначення масиву Фур'є-зображення з метою побудови його графіка в частотній області. Проте цей масив не може бути використаний для зворотного перетворення Фур'є.*

Проілюструємо застосування цієї функції до попереднього приклада:

```
f1 = -500 : 0.5 : 500;      % Перебудова вектора частот
v = fftshift(y); % Перебудова вектора Фур'є-зображення
a =abs(v); % Відшукання модуля
 % Відбудова графика
plot(f1(970:1030),a(970:1030)); grid
set(gca,'FontName','Arial Cyr','FontSize',16),
title('Модуль Фур'є - зображення');
xlabel('Частота (Гц)'); ylabel('abs(F(X(t))')
```

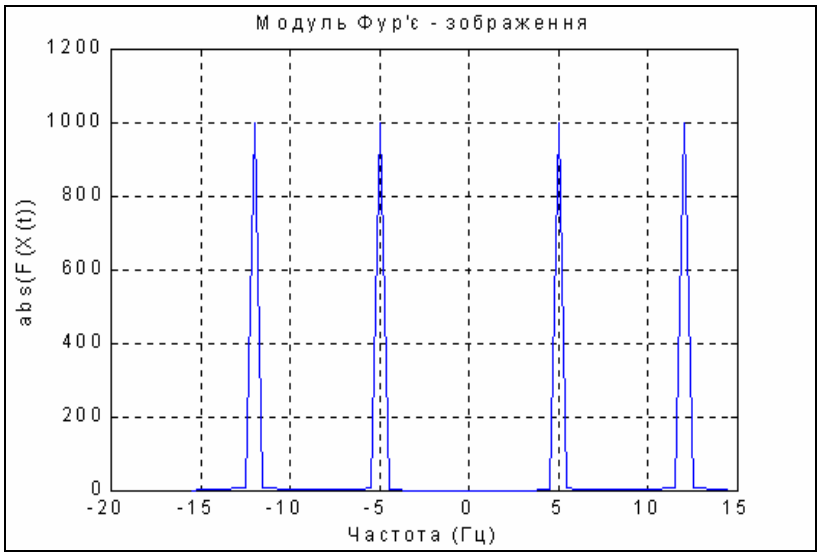


Рис. 1.25. Результат застосування процедури ***fftshift***

З графіка рис. 1.25 вже стає очевидним, що в спектрі вхідного сигналу є дві гармоніки - із частотами 5 і 12 Гц.

Залишається лише та незручність, що із графіка спектра неможливо встановити амплітуди цих гармонік. Щоб уникнути цього, потрібно весь вектор

y Фур'є-зображення розділити на число його елементів (***n***), щоб одержати вектор комплексного спектра сигналу:

```
N=length(y);    a =abs(v)/N;
plot(f1(970:1030),a(970:1030)); grid
set(gca,'FontName','Arial Cyr','FontSize',16,'Color','white'),
title('Модуль комплексного спектра');
xlabel('Частота (Гц)');
ylabel('abs(F(X(t)) / N')
```

Результат наведений на рис. 1.26.

З його розгляду випливає, що "амплітуди" усіх складових гармонік рівні 0.5. При цьому потрібно взяти до уваги, що "амплітуди" розподілені між додатними й від'ємними частотами порівну, тому вони вдвічі менше за справжню амплітуду відповідної гармоніки.

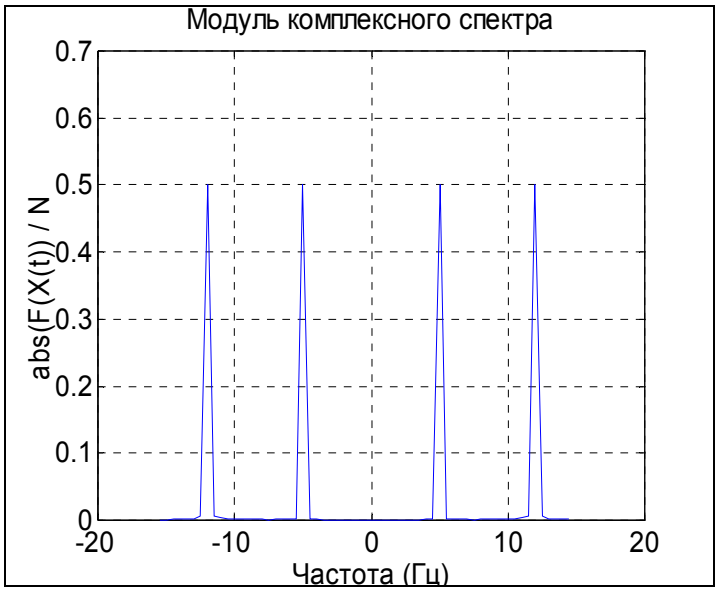


Рис. 1.26. Одержання модуля комплексного спектру

1.4.6. Завдання

Завдання 1.6.

1. Уведіть довільну матрицю розміром (4*6). Знайдіть суму найбільших елементів її рядків.

2. Уведіть квадратну матрицю (5*5) з одним найменшим елементом. Знайдіть суму елементів рядка, у якому розміщений елемент із найменшим значенням.

3. Уведіть матрицю (6*9), у якій є єдині найбільший і найменший елементи і вони розташовані в різних рядках. Поміняйте місцями рядок, що містить найбільший елемент, і рядок із найменшим елементом.

4. Уведіть матрицю (5*6) із різними значеннями елементів. У кожному рядку виберіть елемент із найменшим значенням, з отриманих чисел виберіть найбільше. Знайдіть індекси отриманих елементів.

5. Уведіть матрицю (5*6). Знайдіть вектор, елементами якого є найбільші елементи відповідного рядка матриці.

6. Уведіть матрицю (5*6). Побудуйте вектор, елементами якого є суми найбільшого й найменшого елементів відповідного рядка матриці.

7. Уведіть матрицю (5*6). Побудуйте вектор, елементами якого є середні значення елементів відповідного рядка матриці.

8. Уведіть матрицю (5*6). Побудуйте вектор, елементами якого є середньоквадратичні відхилення елементів відповідного рядка матриці від їхнього середнього значення.

9. Уведіть матрицю (5*6). Побудуйте вектор, елементами якого є середні арифметичні найбільшого й найменшого елементів відповідного рядка матриці.

10. Уведіть матрицю (6*5). Побудуйте вектор, елементами якого є суми квадратів елементів відповідного стовпчика матриці.

11. Уведіть матрицю (5*5). Побудуйте вектори, елементами яких є суми елементів стовпчиків матриці, добутки елементів стовпчиків і найменші значення елементів стовпчиків.

12. Уведіть матрицю (5*6). Знайдіть середнє арифметичне найбільшого й найменшого її елементів.

13. Уведіть матрицю (5*5). Побудуйте вектор, елементами якого є елементи головної діагоналі матриці. Знайдіть слід матриці.

14. Уведіть дві матриці (4*4). Побудуйте нову матрицю розміром (4*8), включаючи в перші 4 стовпчика рядки першої матриці, а в інші - стовпчики другої матриці.

15. Знайдіть суму всіх елементів матриці розміром (4*3).

Завдання 1.7. Обчислити вектори:

а) модуля частотної передатної функції (ЧПФ);

б) аргументу ЧПФ;

в) дійсної частини ЧПФ;

г) уявної частини ЧПФ

по заданих чисельнику і знаменнику передатної функції (таблиця 1.4).

Попередньо знайти корені знаменника передатної функції, визначити найбільшу власну частоту ω_{max} системи. Забезпечити обчислення ЧПФ при 100 значеннях частоти ω у діапазоні від 0 до $5\omega_{max}$.

Таблиця 1.4

Варіант	Чисельник	Знаменник
1	$1.82p+67.56$	$p^4+2.65p^3+3.09p^2+7.04p+34.05$
2	$4.61p^2+1.82p+67.56$	$p^4+3.65p^3+45p^2+7.04p+125$
3	$p^2+4p+23$	$p^4+2p^3+39p^2+2p+45$
4	$3p^2+1.82p+67.56$	$p^2+7.04p+34.05$
5	$p+6$	$p^2+0.7p+48$
6	$p^3+4.61p^2+1.82p$	$2.65p^3+3p^2+4p+87$
7	$p^3+4.61p^2+1.82p+67.56$	$p^4+2.65p^3+68p^2+5p+34$

8	$4.61p^2+68$	$p^4+2.65p^3+3.09p^2+7.04p+34.05$
9	7.56	$p^4+2.65p^3+3.09p^2+7.04p+34.05$
10	$p^3+1.8p+7$	$p^4+6.5p^3+39p^2+7p+45$
11	$p^3+4.61p^2+1.82p+67.56$	$p^3+3.09p^2+70p+34$
12	$p^2+1.8p+78$	$2.65p^3+3.09p^2+7.04p+34.05$
13	$p^3+1.82p+67.56$	$p^4+2.6p^3+3p^2+4p+34$
14	$p^3+4.61p^2+1.82p+67.56$	$7p^2+7p+34$
15	$4.61p^2+1.82p+67.56$	$p^2+7.04p+560$
16	$1.82p+67.56$	$3.09p^2+7.04p+34.05$
17	p^3	$3.09p^2+7.8p+125$
18	$1.82p$	$p^3+3.09p^2+7.04p+34.05$
19	$4.61p^2$	$p^2+7.04p+34.05$
20	$p^3+67.56$	$p^4+2.65p^3+3.09p^2+7.04p+34.05$
21	p^3	$p^4+2p^3+3p^2+12p+100$
22	$p^3+4.61p^{2+}1.82p+67.56$	$p^4+5p^3+30p^2+7p+305$
23	$p^2+1.82p+67.56$	$p^4+2p^3+9p^2+4p+35$
24	$p^{3+}61p^2+182p+67$	$p^4+3p^3+9p^2+0.04p+39$
25	$p^2+1.82p+67.56$	$p^4+5p^3+20p^2+7p+34$

Указівка. *Частотною передатною функцією називають передатну функцію системи при суто уявних значеннях її аргументу* ($p = j \cdot \omega$). *Власні частоти системи - це значення модулів уявних частин коренів характеристичного рівняння системи (яке утворюється дорівнюванням нулю знаменника передатної функції).*

Завдання 1.8. Уведіть довільну матрицю розміром (5*5). Знайдіть:

1) визначник матриці; якщо визначник дорівнює нулю, або занадто малий, змініть деякі елементи матриці і повторіть обчислення;
2) обернену матрицю; перевірте слушність шляхом обернення оберненої матриці;
3) характеристичний поліном матриці;
4) корені характеристичного полінома матриці; упорядкуйте корені по комплексно-спряжених парах і за величиною;
5) власні значення матриці; порівняйте з раніше знайденими коренями характеристичного полінома;
6) LU-розкладання матриці; перевірте його слушність;
7) QR-розкладання матриці; перевірте його слушність;
8) сингулярні числа матриці; порівняєте їх з одержуваними при ***svd-*** розкладанні;
9) слід матриці;
10) число обумовленості матриці;
11) експоненту від матриці;

12) логарифм від експоненти матриці; порівняйте з вихідною матрицею.

1.4.7. Запитання

1. Який об'єкт у Matlab називається поліномом ?
2. Як у Matlab здійснюється перемножування й ділення поліномів ?
3. За допомогою яких функцій можна знайти корені заданого полінома, значення полінома за відомим значенням аргументу ?
4. Які функції дозволяють знайти похідну від полінома ?
5. Як знайти характеристичний поліном матриці ?

1.5. Побудова графіків

1.5.1. Процедура plot

Виведення графіків у системі Matlab є настільки простою і зручною процедурою, що нею можна користуватися навіть при обчисленнях у режимі калькулятора.

Основною функцією, що забезпечує побудову графіків на екрані дисплея, є функція ***plot***. Загальна форма звернення до цієї процедури така:

plot(x1,y1,s1,x2,y2,s2,...).

Тут x1, y1 - задані вектори, елементами яких є масиви значень аргументу (x1) і функції (y1), що відповідають першої кривої графіка; x2, y2 - масиви значень аргументу й функції другої кривої і т.д. При цьому передбачається, що значення аргументу відкладаються уздовж горизонтальної осі графіка, а значення функції - уздовж вертикальної осі. Змінні s1, s2,... є символьними (їхня вказівка не є обов'язковою). Кожна з них може містити до трьох спеціальних символів, що визначають відповідно: а) тип лінії, що з'єднує окремі точки графіка; б) тип точки графіка; в) колір лінії. Якщо змінні s не зазначені, то тип лінії за умовчанням - відрізок прямої, тип точки - пиксел, а колір установлюється (у версії 5) за такою черговістю: - *синій, зелений, червоний, блакитний, фіолетовий, жовтий, чорний і білий* - у залежності від того, яка по черзі лінія виводиться на графік. Наприклад, звернення виду ***plot***(x1,y1,x2,y2,...) призведе до побудови графіка, у якому перша крива буде лінією з відрізків прямих синього кольору, друга крива - такого ж типу зеленою лінією і т.д.

Графіки в Matlab завжди *виводяться в окреме (графічне) вікно*, яку називають *фігурою*.

Наведемо приклад. Нехай потрібно вивести графік функції

$y = 3\sin(x + \pi/3)$

на проміжку від -3π до $+3\pi$ із кроком $\pi/100$.

Спочатку треба сформувати масив значень аргументу x:

```
x = -3*pi : pi/100 : 3*pi,
```

потім обчислити масив відповідних значень функції:

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

і, нарешті, побудувати графік залежності y(x).

У цілому в командному вікні ця послідовність операцій буде виглядати так:

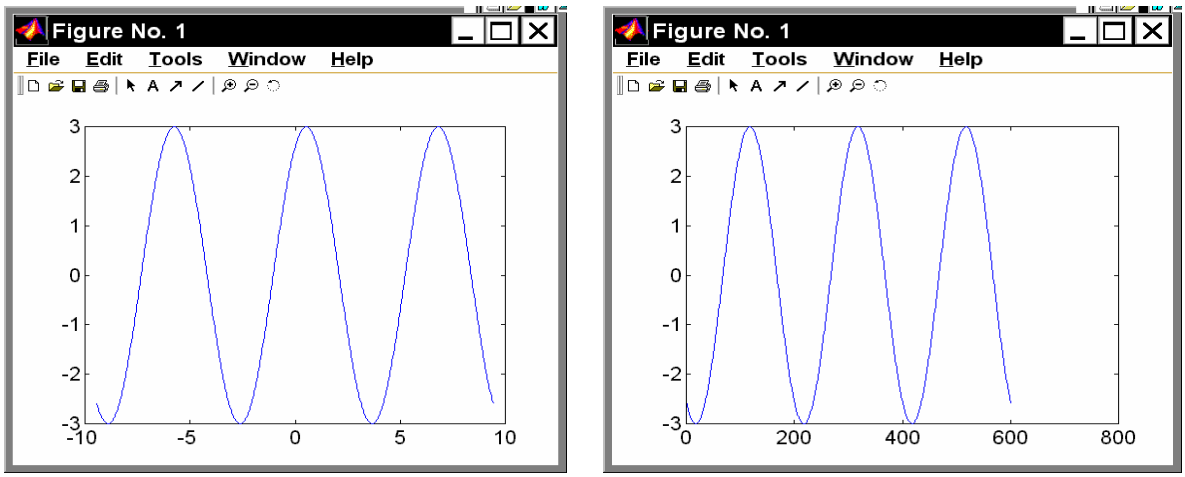
```
» x = -3*pi:pi/100:3*pi;
» y = 3*sin(x+pi/3);
» plot(x,y)
```

У результаті на екрані з'явиться додаткове вікно із графіком (див. рис. 1.27а).

Якщо вектор аргументу при зверненні до функції ***plot*** *не зазначено явно, то система обирає за умовчанням як аргумент номер елемента вектора функції*. Наприклад, якщо ввести команду

```
» plot(y),
```

то результатом буде поява графіка у виді, наведеному на рис. 1.27б.



а) *б)*

Рис. 1.27. Виведення графіка синусоїди

Графіки, наведені на рис. 1.27, мають декілька хиб:

- на них не нанесено сітку з координатних ліній, що утруднює "зчитування" графіків;
- немає загальної інформації про криві графіка (заголовка);
- невідомо, які величини відкладені по осях графіка.

Перший недолік усувається за допомогою функції ***grid***. Якщо цю функцію записати одразу після звернення до функції ***plot***:

```
» x = -3*pi:pi/100:3*pi;
» y = 3*sin(x+pi/3);
» plot(x,y),  grid,
```

то графік буде споряджений координатною сіткою (рис. 1.28).

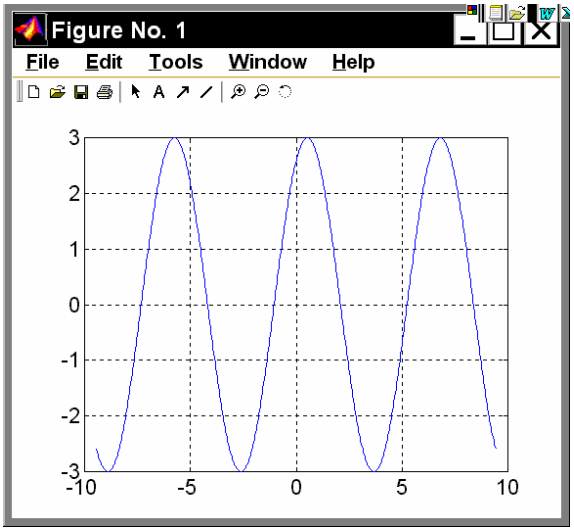


*Рис. 1.28. Застосування **grid***

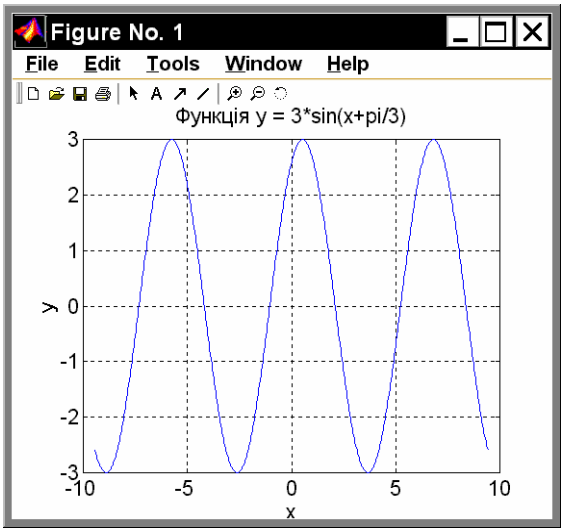


*Рис. 1.29. Застосування **title, xlabel, ylabel***

Цінною особливістю графіків, побудованих у системі Matlab, є те, що *сітка координат завжди відповідає "цілим" крокам змінювання*, що робить графіки "читабельними", тобто за графіком можна робити "відлік" значення функ-

ції при будь-якому заданому значенні аргументу і навпаки. Такої властивості не має жодний із графічних пакетів-додатків до мов програмування високого рівня.

Заголовок графіка виводиться за допомогою процедури ***title.*** Якщо після звернення до процедури ***plot*** викликати ***title*** у такий спосіб:

title(‘<текст>’),

то понад графіком з'явиться текст, записаний між апострофами у дужках. При цьому варто пам'ятати, що *текст завжди має міститися в апострофах*.

Аналогічно можна вивести пояснення до графіка, що розміщаються уздовж горизонтальної осі (функція ***xlabel***) і уздовж вертикальної осі (функція ***ylabel***).

Наприклад, сукупність операторів

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

призведе до оформлення поля фігури у виді, поданому на рис. 1.29. Очевидно, така форма вже цілком задовольняє вимоги, що висуваються до інженерних графіків.

Не більш складним є виведення у середовищі Matlab графіків функцій, заданих *параметрично*. Нехай, наприклад, необхідно побудувати графік функції y(x), що задана параметричними формулами:

$$x = 4\ e\text{-}0^{05\ tsin}\ t; \qquad y = 0{,}2\ e\text{-}0^{1\ tsin}\ 2t.$$

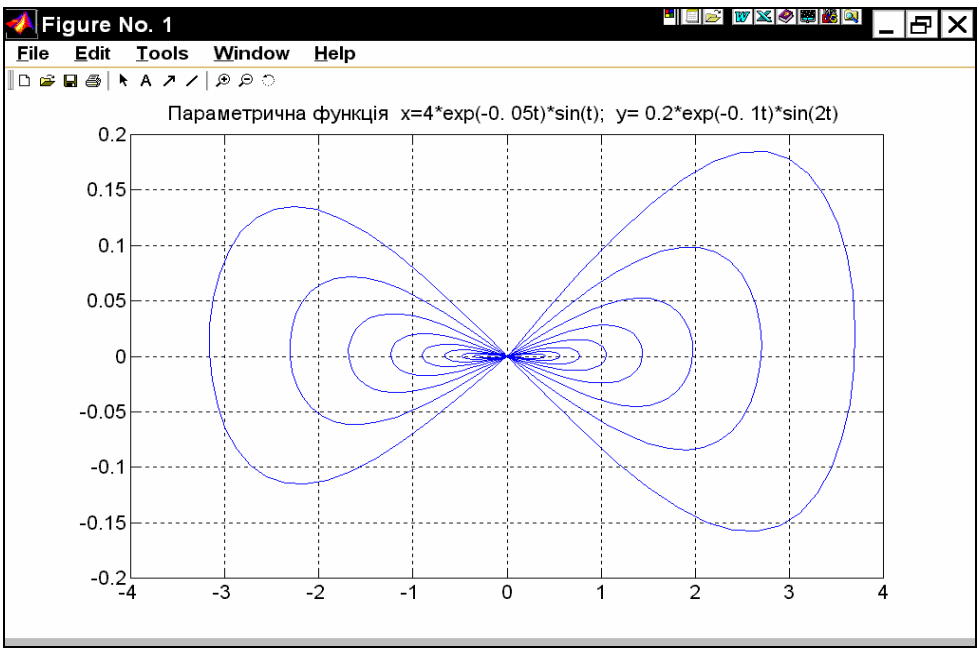


Рис. 1. 30. Графік параметрично заданої функції

Виберемо діапазон змінювання параметра ***t*** від 0 до 50 із кроком 0.1. Тоді, набираючи сукупність операторів

```
» t = 0:0.1:50;
» x = 4*exp(-0.05*t).*sin(t);
```

```
» y = 0.2*exp(-0. 1*t). *sin(2*t);
» plot(x,y)
» title('Параметрична функція  x=4*exp(-0. 05t)*sin(t);  y= 0.2*exp(-0. 1t)*sin(2t) ')
» grid,
```

одержимо графік рис. 1.30.

1.5.2. Спеціальні графіки

Великою зручністю, наданою системою Matlab, є зазначена раніше можливість не вказувати аргумент функції при побудові її графіка. У цьому випадку як аргумент система приймає номер елемента вектора, графік якого будується. Користуючись цим, наприклад, можна побудувати "графік вектора":

```
» x = [ 1 3 2 9 6 8 4 6];
» plot (x)
» grid
» title('Графік вектора X')
» ylabel('Значення елементів')
» xlabel(' Номер елемента').
```

Результат поданий на рис. 1.32.

Ще більш наочним є подання вектора у виді *стовпцевої діаграми* за допомогою функції ***bar*** (див. рис. 1.33):

```
» bar(x)
» title('Графік  вектора X')
» xlabel(' Номер елемента')
» ylabel('Значення елементів')
```

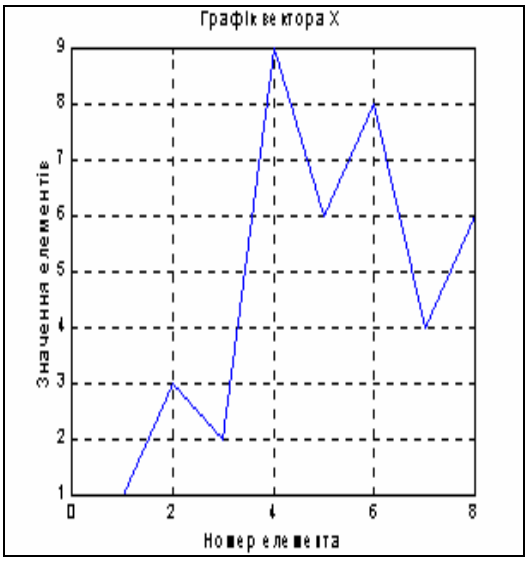


*Рис. 1. 31. Застосування **plot***

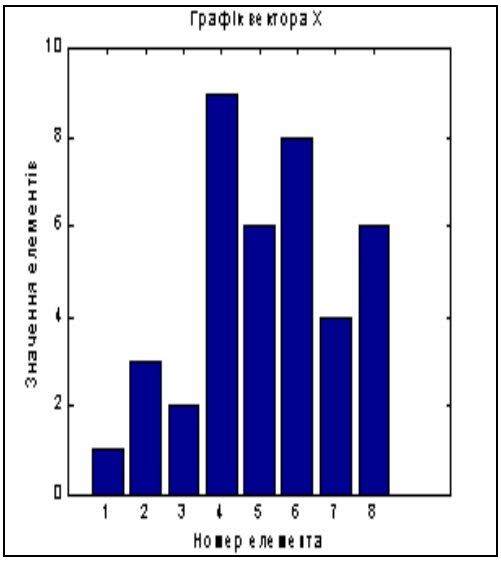


*Рис 1.32. Застосування **bar***

Якщо функція задана своїми значеннями при дискретних значеннях аргументу, і невідомо, як вона може змінюватися в проміжках між значеннями аргументу, зручніше подавати графік такої функції у виді окремих вертикальних ліній для кожного із заданих значень аргументу. Це можна зробити, застосовуючи процедуру ***stem***, звернення до якої цілком аналогічно зверненню до процедури ***plot***:

```
x = [ 1 3 2 9 6 8 4 6];
stem(x,'k')
```

```
grid
set(gca,'FontName','Arial','FontSize',14),
title('Графік вектори X')
ylabel('Значення елементів')
xlabel(' Номер елемента')
```

На рис. 1.34 зображено одержаний при цьому графік.

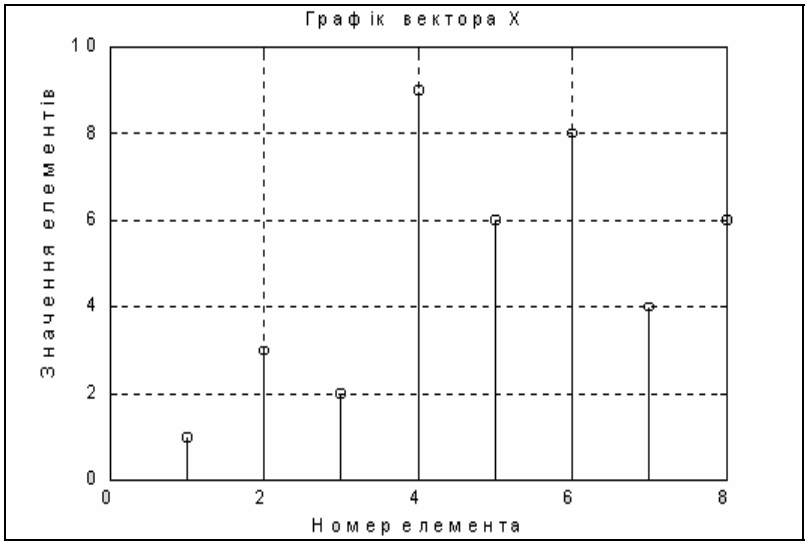


Рис. 1.33. Застосування ***stem***

Інший приклад - побудова графіка функції $y = e^{-x^2}$ у виді стовпцевої діаграми (рис. 1.35):

```
» x = - 2.9 : 0.2 : 2.9;
» bar(x, exp(-x . * x))
» title('Стовпцева діаграма функції y = exp(-x^2)')
» xlabel (' Аргумент x')
» ylabel (' Значення функції y')
```

Ще одна корисна інженеру функція - ***hist*** (побудова графіка *гістограми* заданого вектора). Стандартне звернення до неї має вид:

hist(y,x),

де y - вектор, гістограму якого потрібно побудувати; x - вектор, що визначає інтервали змінювання першого вектора, усередині яких підраховується кількість елементів вектора “y”.

Ця функція робить дві операції

- підраховує кількість елементів вектора “y”, значення яких потрапляють усередину відповідного діапазону, зазначеного вектором “x”;

- будує стовпцеву діаграму підрахованих чисел елементів вектора “y” як функцію діапазонів, зазначених вектором “x” .

Як приклад роздивимося побудову гістограми випадкових величин, що формуються вмонтованою функцією ***randn***. Візьмемо загальну кількість елементів вектора цих випадкових величин 10 000. Побудуємо гістограму для діапазону змінювання цих величин від -2,9 до +2,9. Інтервали змінювання нехай будуть рівні 0,1. Тоді графік гістограми можна побудувати за допомогою сукупності таких операторів:

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

```
» hist(y,x)
» ylabel('Кількість з 10000')
» xlabel( 'Аргумент')
» title('Гістограма нормального розподілу')
```

Результат поданий на рис. 1.36. З нього, зокрема, випливає, що вмонтована функція ***randn*** достатньо вірно відображає нормальний гауссовий закон розподілу випадкової величини.

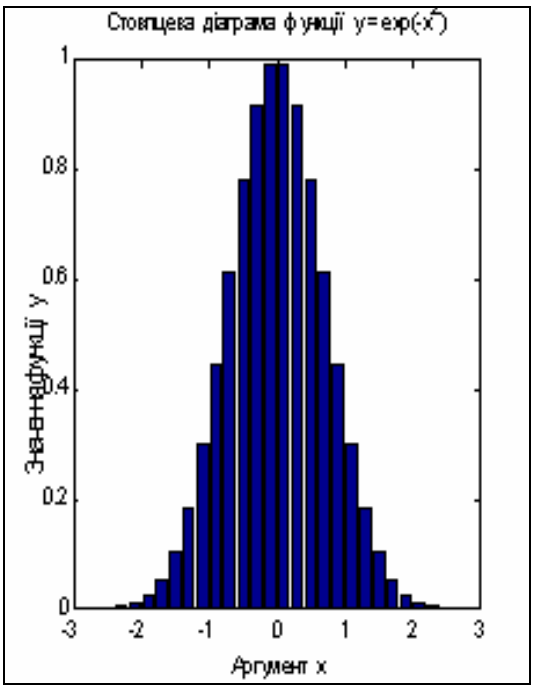


*Рис. 1. 34. Процедура **bar***

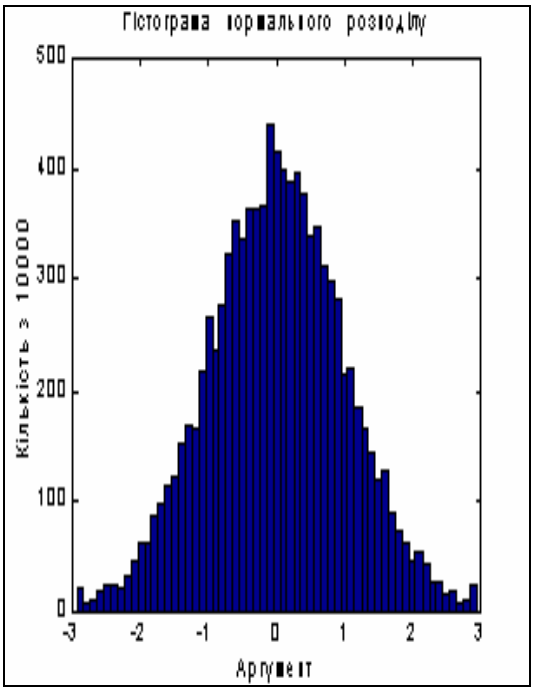


*Рис. 1.35. Процедура **hist***

Процедура ***comet***(x,y) ("комета") будує графік залежності y(x) поступово у часі у виді траєкторії комети. При цьому зображуюча точка на графіку має вид маленької комети (із голівкою й хвостиком), що плавно переміщується від однієї точки до іншої. Наприклад, якщо ввести сукупність операторів:

```
» t = 0:0.1:50;
» x = 4 * exp(-0.05*t) .* sin(t);
» y = 0.2 * exp(-0. 1*t) . * sin(2*t);
» comet(x,y),
```

то графік, наведений на рис. 1.31, буде побудований як траєкторія послідовного руху комети. Ця обставина може бути корисною при побудові просторових траєкторій для виявлення характеру змінювання траєкторії з часом.

Matlab має декілька функцій, що дозволяють будувати графіки в логарифмічному масштабі.

Функція ***logspace*** із зверненням

x = ***logspace***(d1, d2, n)

формує вектор-рядок "x", що містить "n" рівновіддалених у логарифмічному масштабі одна від одної точок, що покривають діапазон від 10^{d1} до 10^{d2}.

Функція ***loglog*** цілком аналогічна функції ***plot***, але графіки по обох осях будуються в логарифмічному масштабі.

Для побудови графіків, які використовують логарифмічний масштаб тільки по однієї з координатних осей, користуються процедурами ***semilogx*** і ***semilogy***. Перша процедура будує графіки з логарифмічним масштабом уздовж горизонтальної осі, друга - уздовж вертикальної осі.

Звернення до останніх трьох процедур цілком аналогічно зверненню до функції ***plot***.

Як приклад розглянемо побудову графіків амплітудно-частотної й фазо-частотної характеристик ланки, що описується передатною функцією:

$$W(p) = \frac{p+4}{p^2 + 4 \cdot p + 100}.$$

Для цього треба, по-перше, створити поліном чисельника Pc = [1 4] і знаменника передатної функції Pz = [1 4 100]. По-друге, визначити корені цих двох поліномів:

```
» P1 = [1 4];     P2 = [1 4 100];
» roots(P1)
ans =    -4
» roots(P2)
ans =
-2. 0000e+000 +9. 7980e+000i
-2. 0000e+000 -9. 7980e+000i
```

По-третє, задати діапазон змінювання частоти так, щоб він охоплював усі знайдені корені:

om0 = 1e-2; omk = 1e2.

Тепер потрібно задатися кількістю точок майбутнього графіка (наприклад, n = 41), і сформувати масив точок по частоті

OM = logspace(-2,2,41),

де значення -2 і +2 відповідають десятковим порядкам початкового om0 і кінцевого omk значень частоти.

Користуючись функцією ***polyval***, можна обчислити спочатку вектор "ch" комплексних значень чисельника частотної передатної функції, що відповідають заданій передатній функції за Лапласом, якщо як аргумент функції ***polyval*** використати сформований вектор частот OM, елементи якого помножені на уявну одиницю (див. визначення Частотної Передатної Функції). Аналогічно обчислюється комплекснозначний вектор "zn" знаменника ЧПФ.

Вектор значень АЧХ (амплітудно-частотної характеристики) можна знайти, вираховуючи модулі векторів чисельника і знаменника ЧПФ і поелементно ділячи отримані вектори. Щоб знайти вектор значень ФЧХ (фазочастотної характеристики) треба розділити поелементно комплекснозначні вектори чисельника й знаменника ЧПФ і визначити вектор аргументів елементів отриманого вектора. Для того щоб фази подати в градусах, отримані результати варто помножить на 180 і розділити на π.

Нарешті, для побудови графіка АЧХ у логарифмічному масштабі, достатньо застосувати функцію ***loglog***, а для побудови ФЧХ зручніше скористатися функцією ***semilogx.***

У цілому послідовність дій може бути такою:

```
» OM = logspace(-2,2,40)
```

```
» ch = polyval(P1,i*OM);
» zn = polyval(P2,i*OM);
» ACH = abs(ch). /abs(zn);
» loglog(OM,ACH);
>> grid;
>> title('Графік Амплітудно-Частотної Характеристики')
>>       xlabel('Частота  (рад/с)');
>>       ylabel('Відношення  амплітуд')
» FCH = angle(ch. /zn)*180/pi;
» semilogx(OM,FCH);
>>       grid,
>>       title('Фазо-Частотна Характеристика'),
>> xlabel('Частота  (рад/с)'),
>>       ylabel('Фаза  (градуси)')
```

В результаті утворюються графіки, зображені на рис. 1.36 і 1.37.

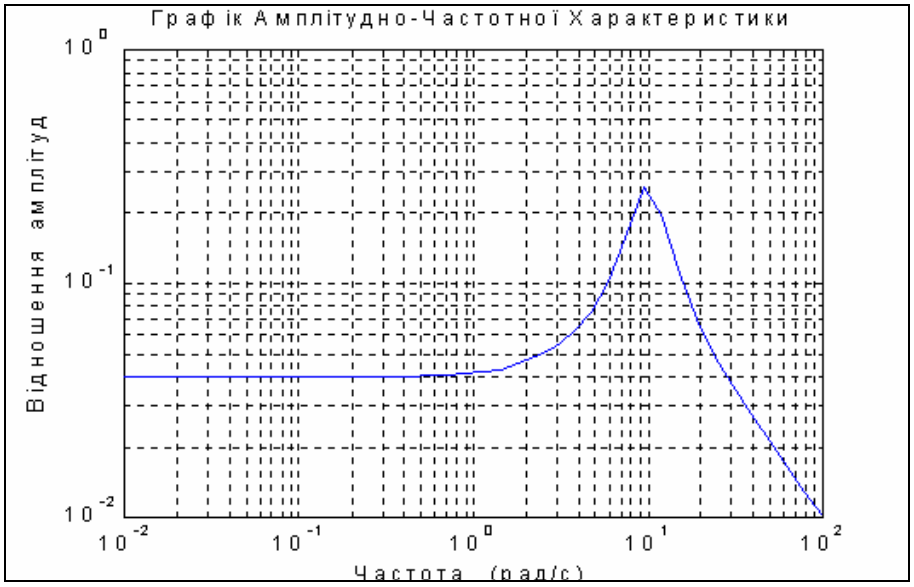


Рис. 1.36. Графік АЧХ

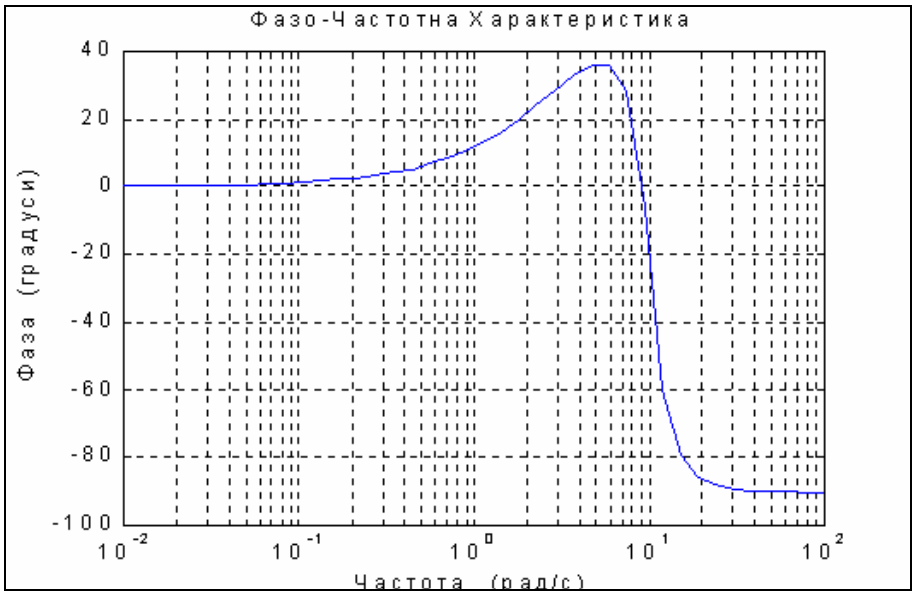


Рис. 1.37. Графік ФЧХ

1.5.3. Додаткові функції графічного вікна

Зазвичай графіки, одержувані за допомогою процедур ***plot***, ***loglog, semilogx*** і ***semilogy***, автоматично будуються в таких масштабах по осях, щоб у поле графіка помістилися всі обчислені точки графіка, включаючи максимальні і мінімальні значення аргументу й функції. Проте Matlab має можливості встановлення й інших режимів масштабування. Це досягається за рахунок використання процедури ***axis***.

Команда

axis([xmin xmax ymin ymax])

установлює жорсткі межі поля графіка в одиницях величин, що відкладаються по осях.

Команда ***axis***(‘auto') повертає масштаби по осях до їх штатних значень (прийнятих за замовчуванням).

Команда ***axis***(‘ij') переміщує початок відліку в лівий верхній ріг і реалізує відлік від верхнього лівого рогу (матрична система координат).

Команда ***axis***(‘xy') повертає декартову систему координат із початком відліку в лівому нижньому рогові.

Команда ***axis***(‘square') установлює однаковий діапазон змінювання змінних по осях графіка.

Команда ***axis***(‘equal') забезпечує однаковий масштаб по обох осях графіка.

В однім графічному вікні, але на окремих графічних полях можна побудувати декілька графіків, використовуючи процедуру ***subplot***. Звернення до цієї процедури повинно передувати зверненню до процедур ***plot***, ***loglog, semilogx*** і ***semilogy*** і мати такий вид:

subplot(m,n,p).

Тут m - указує, на скільки частин розділяється графічне вікно по вертикалі, n - по горизонталі, а p - номер підвікна, у якому буде будуватися графік. При цьому підвікна нумеруються зліва праворуч порядково зверху униз (так, як по рядках читається текст книги).

Наприклад, два попередні графіки можна помістити в одне графічне вікно в такий спосіб:

```
subplot(2,1,1);
loglog(OM,ACH,'k'); grid;
set(gca,'FontName','Arial','FontSize',14),
title('Амплітудно-Частотна Характеристика')
ylabel('Амплітуда')
subplot(2,1,2);
semilogx(OM,FCH,'k'); grid
set(gca,'FontName','Arial','FontSize',14),
title('Фазо-Частотна Характеристика')
xlabel('Частота  (рад/с)'),     ylabel('Фаза (гр.)')
```

Результат поданий на рис. 1.38.

Команда ***text***(x, y, ‘<текст>’) дозволяє розмістити зазначений текст на поле графіка, при цьому початок тексту поміщається в точку з координатами x і y. Значення зазначених координат повинні бути подані в одиницях величин, що

відкладаються по осях графіка, і знаходитися усередині діапазону змінювання цих величин. Часто це незручно, бо потребує попереднього знання цього діапазону, що рідко коли є можливим.

Більш зручним для розміщення тексту усередині поля графіка є використання команди ***gtext***(‘<текст>’), яка висвічує в активному графічному вікні перехрестя, переміщення якого за допомогою "мишки" дозволяє вказати місце початку виведення зазначеного тексту. Після цього натисканням лівої клавіші "мишки" або будь-якої клавіші текст вводиться в зазначене місце.

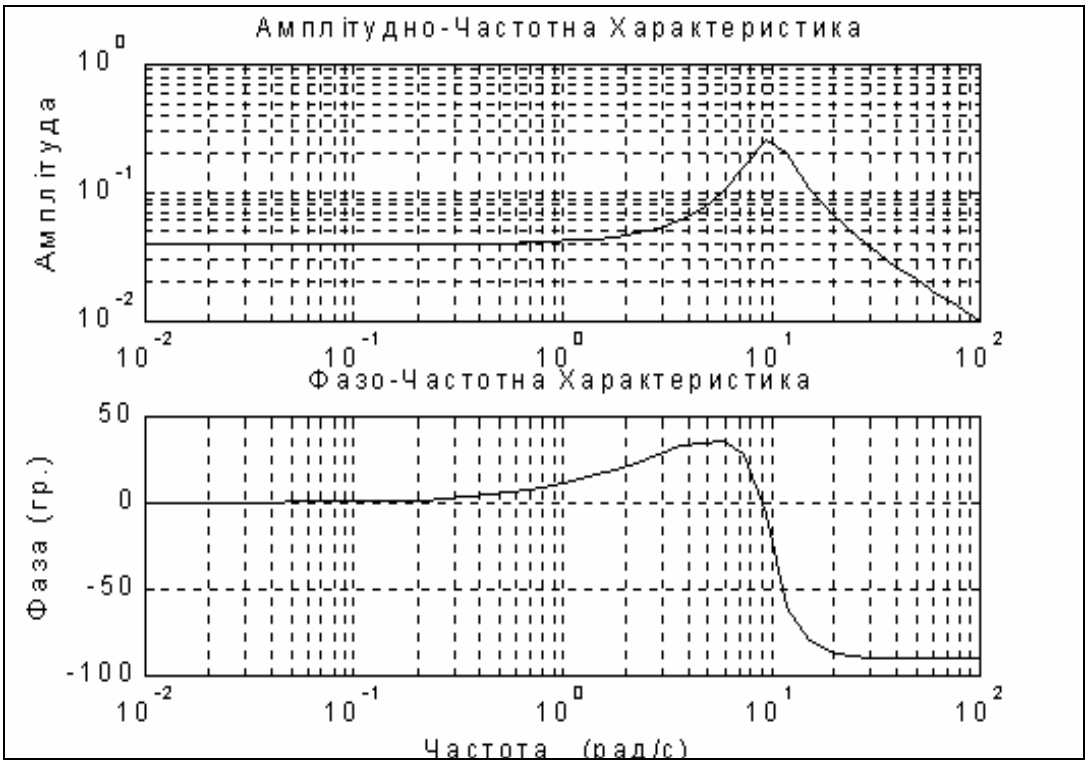


Рис. 1.38. Використання функції ***subplot***

Щоб створити декілька графічних вікон, у кожному з яких розташовані відповідні графіки, можна скористатися командою ***figure***, яка створить таке графічне вікно, залишаючи попередні.

Нарешті, для того, щоб декілька послідовно обчислюваних графіків були зображені в однім графічному вікні в одному стилі, можна використати команду ***hold on***, тоді кожний такий графік буде будуватися в тому ж попередньо відкритому графічному вікні, тобто кожна нова лінія буде додаватися до раніше побудованих. Команда ***hold off*** виключає режим зберігання графічного вікна, встановленого попередньою командою.

1.5.4. Вставлення графіків до документу

Щоб вставити графік із графічного вікна (фігури) до документу Word, слід скористатися командами меню, розташованого у верхній частині вікна фігури. У меню ***Edit*** оберіть команду ***Copy Figure***. Перейдіть у вікно документу і

натисніть клавіші <Ctrl>+<C>. Вміст графічного вікна виникне у тому місці документу, де містився курсор.

1.5.5. Завдання

Завдання 1.9. Побудуйте в графічному вікні Matlab графік функції із *завдання 1.5*. Роздрукуйте цей графік на аркуші паперу.

Завдання 1.10. Побудуйте в графічному вікні Matlab графіки амплітудно-частотної (модуля ЧПФ) і фазочастотної (аргументу ЧПФ) характеристик функції із *завдання 1.7*. Роздрукуйте отриманий графік на аркуші паперу.

1.5.6. Запитання

1. Які функції Matlab здійснюють виведення графіків на екран?
2. Якими функціями забезпечується супровід графіка координатними лініями й написами?
3. Що таке "графік вектора" і як його побудувати?
4. Як вивести графік у виді стовпцевої діаграми?
5. Як побудувати гістограму?
6. Чи можна побудувати декілька графіків в одній системі координат і в однім графічному вікні ?
7. Як вивести декілька окремих графіків у різних графічних вікнах?
8. Як побудувати декілька окремих графіків в одному графічному вікні?

1.6. Література

1. Барановская Г. Г., Любченко И. Н. Микрокалькуляторы в курсе высшей математики: Практикум. - К.: Вища шк., 1987. - 288 с.
2. Гультяев А. К. MatLAB 5.2. Имитационное моделирование в среде Windows: Практич. пособие. - СПб.: КОРОНА-принт, 1999. - 288 с.
3. Дьяконов В. П. Справочник по применению системы PC MatLAB. - М.: Физматлит, 1993. - 113с.
4. Дьяконов В. П., Абраменкова И. В. MATLAB 5.0/5.3. Система символьной математики. - М.: Нолидж, 1999. - 640с
5. Лазарєв Ю. Ф. Початки програмування у середовищі MatLAB: Навч. посібник. - К.: "Корнійчук", 1999. - 160 с.
6. Лазарев Ю. Ф. MatLAB 5.x. – К.: Издатедьская группа BHV, 2000. - 384 с.
7. Лазарев Ю. Ф. Моделирование процессов и систем в MATLAB. Учебный курс. – СПб.: Питер; Киев: Издат. группа BHV, 2005. – 512 с.
8. Лазарєв Ю. Ф. Моделювання на ЕОМ. Навч. посібник. – К.: Корнійчук, 2007. - 290 с.
9. Мартынов Г. Г., Иванов А. П. MATLAB 5.x, вычисления, визуализация, программирование. - М.: "Кудиц-образ", 2000. - 332 с.
10. Потемкин В. Г. Система MatLAB: Справ. пособие. - М.: ДИАЛОГ-МИФИ, 1997. - 350 с.
11. Потемкин В. Г. MatLAB 5 для студентов: Справ. пособие. - М.: ДИАЛОГ-МИФИ, 1998. - 314 с.
12. Потемкин В. Г., Рудаков П. И. MatLAB 5 для студентов. - 2-е изд., испр. и дополн. - М.: ДИАЛОГ-МИФИ, 1999. - 448 с.
13. Потемкин В. Г. Система инженерных и научных расчетов MatLAB 5.x: - В 2-х т.Том 1. - М.: ДИАЛОГ-МИФИ, 1999. - 366 с.
14. Потемкин В. Г. Система инженерных и научных расчетов MatLAB 5.x: - В 2-х т. Том 2. - М.: ДИАЛОГ-МИФИ, 1999. - 304 с.
15. Сулима И. М., Гавриленко С. И., Радчик И. А., Юдицкий Я. А. Основные численные методы и их реализация на микрокалькуляторах. - К.: Вища шк., 1987. - 312 с.