МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний гірничий університет

Кафедра
систем електропостачання

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи СПТ-4

“Вивчення командного режиму роботи і прийомів візуалізації результатів обчислень в пакеті MATLAB для застосування у вивченні курсу "Перетворювальна техніка"”

для студентів напряму 0906 – Електротехніка
денної та заочної форм навчання

Дніпропетровськ
2006

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний гірничий університет

Кафедра
систем електропостачання

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи СПТ-4

“Вивчення командного режиму роботи і прийомів візуалізації результатів обчислень в пакеті MATLAB для застосування у вивченні курсу "Перетворювальна техніка"”

для студентів напряму 0906 Електротехніка
денної та заочної форм навчання.

Затверджено на засіданні кафедри
систем електропостачання
Протокол № від 2006 р.

Дніпропетровськ
2006

Методичні вказівки до лабораторної роботи СПТ-4 **„Вивчення командного режиму роботи і прийомів візуалізації результатів обчислень в пакеті MATLAB для застосування у вивченні курсу "Перетворювальна техніка""** для студентів напряму 0906 – Електротехніка денної і заочної форм навчання / Укладачі: Ковальов О.Р. , Дибрін С.В. – Дніпропетровськ: НГУ, 2006. – 21 с.

Укладачі:
Ковальов Олександр Робертович, ст. викладач
Дибрін Сергій Володимирович, аспірант

Відповідальний за випуск – Заїка В.Т., д-р. техн. наук, професор

ЗМІСТ

Мета: ознайомитися з простими прийомами розрахунків, а також методами побудови і оформлення графіків в середовищі програмного пакету **MatLab.**

ТЕОРЕТИЧНІ ВІДОМОСТІ

РОБОТА В КОМАНДНОМУ РЕЖИМІ

Робота в середовищі MATLAB може здійснюватися у двох режимах: **командному** (калькулятора), тобто обчислення проводяться безпосередньо після набору чергового оператора, і **шляхом виклику програми**, записаної мовою MATLAB.

В обох режимах користувачеві доступні практично всі обчислювальні можливості системи.

Командне вікно

Після запуску програми MATLAB на дисплеї комп'ютера з'являється її головне вікно (рис.1) і система – готова до проведення обчислень в командному режимі (режимі калькулятора).

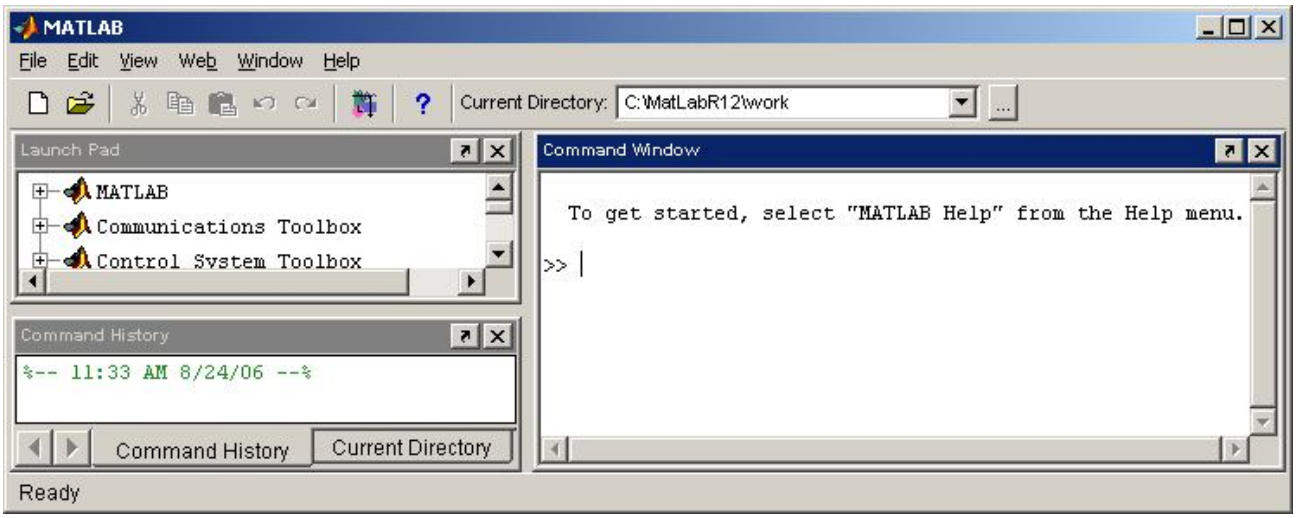


Рис. 1. *Головне вікно MATLAB*

Вікно містить меню, інструментальну лінійку з кнопками, вказівки поточної папки (**Current Directory**) і ряд підвікон, набір яких можна змінювати за складом і кількістю при допомозі меню View (**Вигляд**). Розглянемо основне з підвікон – командне вікно (**Command Window**). У ньому відображаються символи команд, що набирають на клавіатурі, результати виконання цих команд, інформація щодо помилок виконання програм і т.п. Ознакою того, що програма MATLAB готова до сприйняття і виконання чергової команди є наявність в останньому рядку текстового поля командного вікна знаку запрошення (**>>**) , після якого стоїть миготлива вертикальна межа.

У верхній частині вікна MATLAB розміщений рядок меню: **File**, **Edit**, **View**, **Web**, **Window**, **Help**. Призначення пунктів меню пояснюватиметься за необхідністю. При цьому послідовність дій указуватиметься перерахуванням пунктів меню за допомогою стрілки. Вихід з системи MATLAB здійснюється закриттям головного вікна або вибором **File → Exit MATLAB**.

Пункт меню **Help** дозволяє скористатися докладною довідковою системою і демонстраційними прикладами. На ці приклади можна також вийти, набравши в командному вікні – **DEMO**. Для швидкої довідки щодо будь-якого об'єкту треба в командному вікні ввести – **help** ***ім'я об'єкту***.

Якщо немає упевненості в правильному написанні імені об'єкту, або воно невідоме, можна використати пошук по ключовому слову або по послідовності слів. Для цього можна використовувати команду **lookfor** ***Ключове слово*** або **lookfor** '***Ключові слова***'.

Існують ще декілька команд для "швидких" довідок.

Прості дії в командному вікні

Для арифметичних операцій використовуються звичайні знаки "**+**", "**–**", "*****", "**/**". Для піднесення до степеня прийнятий символ "^", а для ділення справа наліво "****". Наприклад, якщо ввести вираз:

```
>> R = 4.5^2*7.23 - pi*10.4
R =
    113.7349
```

і натиснути **Enter**, отримаємо відповідь:

Тут **pi** — це константа **π**.

Видно, що введене значення привласнене змінній **R**. До речі, імена змінних та інших елементів повинні починатися з букви, вони можуть мати будь-яке число символів (запам'ятовуються перші 31), заголовні і малі літери системою розрізняються. Якщо формат висновку вас не влаштовує, його можна встановити через меню **File → Preferenses**.

Введена у вікно інформація зберігається в стеку і знову може бути викликана клавішею "↑", що зручно при коректуванні виразів або повторних обчисленнях.

Загалом форма виведення результату в командне вікно має вигляд ***ім'я змінної = вираз***. При цьому виведення в командне вікно підкоряється наступним правилам:

- якщо запис оператора **не** закінчується символом ";", то результат його дії відразу відображається у командному вікні;
- якщо запис оператора закінчується символом "**;**", то результат його дії не відображається у командному вікні;
- оператор може містити тільки <***вираз***>, тоді значення результату привласнюється спеціальній системній змінній **ans**;
- набуте значення можна використовувати в подальших обчисленнях під ім'ям **ans**, але слід пам'ятати, що значення цієї змінної зміниться при першому ж операторові, що зустрівся, без знаку привласнення.

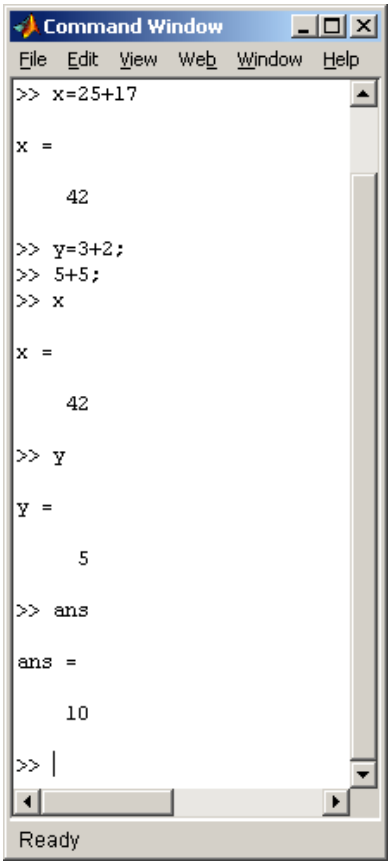


Рис. 2. *Приклад розрахунку*

Приклади (рис.2) пояснюють сказане. Нагадаємо, що уведені користувачем вирази, знаходяться в рядках, які починаються з символу "**>>**", а в решті рядків – результати.

Далі інформація представлятиметься не у формі копій екрану (як на приведених малюнках), а просто у вигляді напівжирного тексту і без пропуску рядків усередині одного прикладу. Так, дії, зображені на рис. 2, можна продовжити таким чином:

```
>> sin(pi/2)
ans =
    1
```

Окрім вже відомих імен **ans** і **pi** використовуються й інші системні імена:

- **i, j** – уявна одиниця (корінь квадратний з -1);
- **eps** – похибка для операцій над числами з плаваючою крапкою (2^{-52});
- **realmin** – найменше число з плаваючою крапкою (2^{-1022})
- **realmax** – найбільше число з плаваючою крапкою (2^{1023});
- **Inf** – машинна нескінченність;
- **NAN** – вказівка на нечисловий характер даних (Not-a-Number).

Приклади:

```
>> sqrt(-2)
ans =
    0 + 1.4142i.

>> eps
ans =
    2.2204e-016

>> realmin
ans =
    2.2251e-308
```

```
>> realmax
ans =
    1.7977e+308

» 2/0
Warning: Divide by zero.
ans =
  Inf

» 0/0
Warning: Divide by zero.
ans =
  NAN
```

Загальна форма виклику функції: ***ім'я результату = ім'я функції*** (список імен аргументів або їх значень).

Ім'я результату може бути відсутнім, тоді результат привласнюється змінній **ans**. Застосування як елементарних, так і інших вбудованих функцій пояснюватиметься по ходу їх використання.

Масиви

Рекомендується уважно розібрати цю тему, оскільки масиви грають визначальну роль у мові MATLAB. Всі дані представлені у вигляді масивів. Наприклад, навіть проста змінна подається як масив розміром 1x1.

Як і в інших мовах програмування, іменовані набори чисел називають масивами. Всьому масиву привласнюється одне ім'я, а доступ до окремих елементів масиву здійснюється по цілочисельному індексу, тобто за номером елементу в масиві. Правила використання масивів – аналогічні іншим мовам. Основна відмінність – масив наперед не визначається, а при введенні елементів допускається значна свобода дій. Ознакою масиву є прямі дужки, а елементу масиву – ім'я з індексом в круглих дужках. Якщо масив являє собою вектор-рядок, то символи розділяються пропусками або комами, а якщо вектор-стовпець – символами ";" або переведенням рядків.

Приклади:

```
>> a1 = [1 2 3]
a1 =
    1  2  3
```

```
>> a2 = [3,2,1]
a2 =
    3  2  1
```

```
>> [3;7;8]
ans =
    3
    7
    8
```

```
>> d = [4
3]
d =
    4
    3
```

Масиви можна вводити частинами, які потім об'єднуються. Наприклад, об'єднаємо визначені вище масиви **a1** і **a2**:

```
>> b = [a1 a2]
b =
    1  2  3  3  2  1
```

Можна визначати масив, прописуючи кожен елемент:

```
>> A(1)= 2;
>> A(2)= 7;
```

```
>> A(3)= 5
A =
    2  7  5
```

Або ще "цікавіше":

```
>> B(4)= 3
B =
    0  0  0  3
```

Якщо значення елементів змінюються з постійним кроком, то для формування масиву зручно використовувати оператора двокрапка – ":", наприклад:

```
>> x = 1:0.1:2
x =
Columns 1 through 7
1.0000  1.1000  1.2000  1.3000  1.4000  1.5000  1.6000
Columns 8 through 11
1.7000  1.8000  1.9000  2.0000
```

Можна будувати масиви, використовуючи спеціальні вбудовані функції. Наприклад, отримати масив з п'яти одиниць:

```
>> E = ones(1,5)
E =
    1  1  1  1  1
```

Для визначення двовимірних і багатовимірних масивів використовуються усі ті ж прийоми, а рядки відділяються один від одного записом з нового рядка або символом "**;**". Приклади:

```
>> [9 8 7
6 5 4
3 2 1]
ans =
    9  8  7
    6  5  4
    3  2  1
```

```
>> [1 2 3;4 5 6;7 8 9]
ans =
    1  2  3
    4  5  6
    7  8  9
```

Визначимо масив з одиниць розміром 3*5

```
>> E = ones(3,5)
E =
    1  1  1  1  1
    1  1  1  1  1
    1  1  1  1  1
```

За умовчанням елементи двовимірного масиву продивляються по рядках, тобто спочатку змінюється другий індекс. У разі обробки багатовимірних масивів спочатку змінюється крайній правий індекс, потім – другий справа і т.д. Згаданий вище оператор двокрапка ("**:**") використовується і для інших цілей. Він як би здійснює перерахування всіх елементів по даному індексу. Пояснюють це приклади формування масивів **v, w, u** з визначеного вище двовимірного масиву ans:

```
>> v = ans(:)
v =
    1
    4
    7
    2
    5
    8
    3
    6
    9
```

```
>> w = ans(:,1)
w =
    1
    4
    7

>> u = ans (2:)
u =
    4  5  6
```

Порожній масив задається двома квадратними дужками без символів між ними. Порожній масив можна, зокрема, використовувати для видалення елементів масиву, наприклад, видалити другий стовпець з масиву **ans**:

```
>> ans(:,2)= []
ans =
    1  3
```

```
     4   6
     7   9
```

Видалити третій рядок з масиву **ans:**

```
>> ans(3:)= []
ans =
     1   3
     4   6
```

Аналогічно будуються і використовуються багатовимірні масиви.

Далі, разом з терміном "масив", використовуватимемо терміни "вектор", "матриця" і т. і.

Деякі функції для роботи з матрицями:

- **eye** – створення одиничної матриці;
- **ones** – створення матриці з одиниць;
- **zeros** – створення матриці з нулів;
- **linspace** – формує вектор-рядок рівновіддалених вузлів;
- **cat** – конкатенація матриць;
- **diag** – створення матриць із заданою діагоналлю;
- **fliplr –** перестановка стовпців матриці відносно вертикальної осі;
- **flipud** – перестановка рядків матриці відносно горизонтальної осі;
- **perms** – формує матрицю з перестановок елементів вектора;
- **prod** – добуток елементів масиву;
- **cumprod** – добуток елементів масиву з накопиченням;
- **sum** – сума елементів масиву;
- **cumsum** – сума елементів масиву з накопиченням;
- **repmat** – копіювання матриці;
- **reshape** – вибірки з матриці;
- **rot90** – поворот матриці на 90°;
- **tril** – виділяє нижню трикутну частину матриці;
- **triu** – виділяє верхню трикутну частину матриці;
- **length** – кількість елементів в одновимірному масиві A;
- **size** – розмір масиву (наприклад, число рядків і число стовпців);
- **ndims** – розмірність масиву, тобто кількість індексів.

Кількість функцій достатньо велика, при необхідності можна звернутися до системи **Help**.

Всі перераховані функції мають декілька варіантів застосування з різним числом аргументів. Далі ми описуватимемо лише потрібні для прикладів варіанти використання тих або інших функцій.

Наприклад, для визначених вище **v** і **ans:**

```
>> d = length(v)
d =
     9
```

```
>> s = size(ans)
s =
     2   2
```

```
>> n = ndims(ans)
n =
     2
```

Можна використовувати вкладення функцій:

```
>> d = log((cumprod(1:6))')
d =
     0
     0 6931
     1 7918
     3 1781
     4 7875
     6 5793
```

Тут (1:6) – формування масиву [1 2 3 4 5 6], функція **cumprod** обчислює факторіали елементів отриманого масиву – [1 2 6 24 120 720], апостроф позначає операцію транспонування, і нарешті, обчислюються логарифми від факторіалів, а результат привласнюється змінній **d**.

Прості операції з векторами і матрицями

Вже з назви системи виходить, що MATLAB спеціально призначена для виконання операцій над матрицями. Навіть звичайну константу MATLAB розглядає як матрицю розміром (1, 1) В цьому легко переконатися:

```
>> size (5)
ans =
    1  1
```

У простих обчисленнях користувач не помічає такого представлення даних, а в складніших – це необхідно враховувати. Надалі нам такі випадки зустрінуться.

MATLAB представляє вектор-рядок з N елементами як матрицю розміром (1, M), а аналогічний вектор-стовпець – як матрицю розміром (N, 1).

Правила дій над векторами і матрицями діляться на дві групи:

- правила, що передбачені векторним численням в математиці;
- правила по перетворенню елементів, що не передбачені математикою, але зручні для програмування.

Дії першої групи позначаються так само, як дії над звичайними математичними об'єктами. Сюди додається знак транспонування і функція з обернення матриці **inv(A)**. ***Наприклад:***

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

>> B = A'
B =
    1  4
    2  5
    3  6
```

```
>> C = [1 2
3 4];
>> D = inv(C)
D =
   -2.0000   1.0000
    1.5000  -0.5000
```

```
>> G = inv(D)
G =
    1.0000  2.0000
    3.0000  4.0000

>> E=C*D
E =
    1.0000       0
    0.0000  1.0000
```

Тут матриця **D** отримана оберненням матриці **C**, а **G** – оберненням **D**. Видно, що ми отримали матрицю, рівну результатною. Матриця **E** отримана множенням матриці **C** на зворотну матрицю $\mathbf{C^{-1}}$ (т.е. **D**). Певна річ, що **E** – одинична матриця.

Операцію обернення матриці можна записувати в природнішому вигляді:

```
>> H = C^(-1)
H =
   -2.000    1.0000
    1.5000  -0.5000
```

Видно, що матриця **H** співпала з матрицею **D**.

Для зручності роботи в MATLAB додано дві операції, яких в математиці немає: ділення матриць зліва направо /, і ділення справа наліво \.

Операція **B/A** еквівалентна діям **B*inv(A)**, а операція **A\ B** еквівалентна діям **inv(A)*B**.

Ці операції зручні при рішенні системи лінійних рівнянь. Наприклад, щоб вирішити рівняння **A*X = B**, можна помножити зліва це рівняння на зворотну матрицю $\mathbf{A^{-1}}$ і отримати рішення $\mathbf{X = A^{-1}B}$.

Так, рішення системи:

$$\begin{cases} x1 + 2x_2 + 3x_3 = 14 \\ 2x1 - x2 - 5x3 = -15 \\ x1 - x2 - x3 = -4 \end{cases}$$

на **MATLAB** можна записати в наступному вигляді:

```
>> 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 =
    1.0000
    2.0000
    3.0000
```

Відзначимо, що при рішенні систем лінійних рівнянь краще користуватися записом **X=A\ B**, а не **inv(A)*B**, оскільки перша використовує алгоритм Гауса, а друга – обернення матриці **A**.

За допомогою операції ділення можна обернення матриці **A** записати як **E/A**, де **E** – одинична матриця.

До другої групи відносяться операції поелементного множення, ділення і піднесення до степеня. Ці дії позначаються додаванням крапки перед символом операції.

Наприклад:

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

```
>> B = [2 1; 3 1]
B =
    2  1
    3  1
```

```
>> C = A .* B
З =
    2  2
    9  4
```

Порівняємо із звичайним множенням матриць:

```
>> C1=A*B
C1 =
     8  3
    18  7
```

Якщо узяти функцію від матриці, то отримаємо матрицю функцій від її елементів.

Наприклад:

```
>> D = [0 pi/6
pi/3 pi/2]
D =
         0  0.5236
    1.0472  1.5708
```

```
>> F = sin(D)
F =
         0  0.5000
    0.8660  1.0000
```

Збереження і зчитування даних сеансу роботи (сесії)

Всі значення змінних, обчислені протягом поточного сеансу роботи (його зазвичай називають сесією), зберігаються в спеціально зарезервованій області пам'яті комп'ютера, яка зветься робочим простором системи MATLAB (**Workspace**). Подивитися, що в нім знаходиться, можна командою **who** або вибором підвікна **Workspace** (якщо підвікна немає, його можна відкрити, використовуючи **View → Workspace**). Для проглядання значення будь-якої змінної із робочого простору, достатньо подвійно клацнути лівою кнопкою миші на імені змінної або набрати її ім'я в командному вікні і натиснути ***Enter***.

Робочий простір можна очистити командою **clear**, а для видалення конкретних змінних слід використовувати ***clear имя1 имя2***....

Після закінчення роботи і виходу з MATLAB всі раніше обчислені змінні втрачаються. Щоб зберегти робочий простір до наступного сеансу, необхідно вибрати в меню **File → Save Workspace As...** (**Файл → Зберегти Робочий Простір як...**). У вікні **Save Workspace Variables** (**Зберегти змінні робочого простору**), що відкрилося, в рядку **File name** (**Ім'я файлу**) набрати ім'я файлу, при необхідності вибрати потрібну папку і натиснути кнопку **Save** (**Зберегти**). Робочий простір буде збережений у файлі ***ім'я файлу*** з розширенням **.mat**.

За умовчанням робочий простір зберігається в поточній (активній) папці у файлі ***ім'я файлу*** з розширенням **.mat**. Зазвичай поточною папкою є папка **work** системи MATLAB. Щоб зробити поточною іншу папку, потрібно скористатися списком головного

вікна MATLAB, що розкривається **Current Directory** (**Поточна Директорія**) і кнопкою праворуч від списку.

При пошуку файлів користувача система починає перегляд з поточної папки, а потім проглядає всі папки, вказані в списку шляхів, відомих системі. Доступ до списку здійснюється через меню **File → SetPath**.

Завантажити робочий простір перед продовженням роботи можна через меню: **File → Open...** (**Файл → Відкрити...**).

ВІЗУАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ОБЧИСЛЕНЬ

Процедура plot

Ця процедура забезпечує побудову двовимірних графіків. Загальна форма звернення: **plot(x1, y1, s1, x2, y2, s2...)**.

Тут **x1, y1** – задані вектори – масиви значень аргументу (**x1**) і функції (**y1**), відповідних першою кривою графіка, **x2**, **y2** – масиви значень аргументу і функції другої кривої, і т.д. При цьому значення **x** відкладаються по горизонтальній осі, а **y** – по вертикальній. Змінні **s**, є символьними (їх вказівка не обов'язкова). Кожна з них може містити три спеціальні символи, які визначають тип лінії, колір і тип крапок, що проставляються. Якщо **s** не вказане, то беруться значення за умовчанням.

Графіки завжди виводяться в окремому (графічному) вікні, яке називають фігурою.

Приклад. Хай потрібно вивести графік функції $y = 3 \cdot \sin(x + \pi/3)$ на проміжку від -3π до 3π з кроком $\pi/100$.

Спочатку формуємо масив значень аргументу **x**: `x = -3*pi : pi/100 : 3*pi`.

Потім обчислюємо масив відповідних значень функції: `y = 3*sin(x+pi/3)`.

І, нарешті, будуємо графік **y(x)**.

У командному вікні ця послідовність операцій виглядатиме так:

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

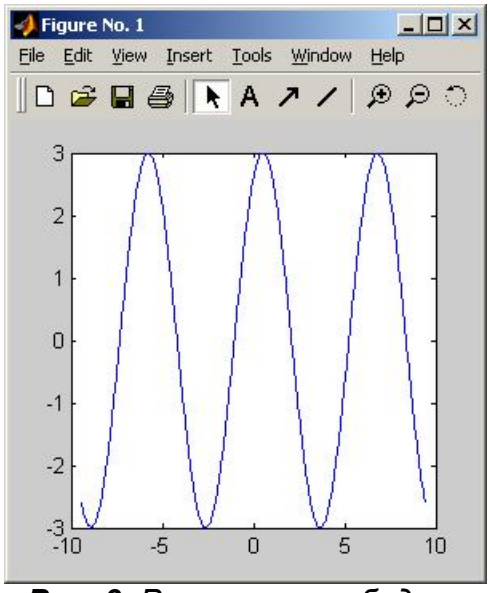


Рис. 3. *Результат побудови графіка*

В результаті на екрані з'явиться додаткове вікно з графіком (рис.3). Відзначимо, що при побудові кривої функція **plot** використовує лінійну інтерполяцію, тобто сполучає сусідні крапки відрізком прямої. Тому чим більше узято крапок, тим більш гладкою буде крива.

Якщо вказати тільки масив (`>> plot(y)`), то як аргумент, за умовчанням, будуть взяті послідовні номери крапок. Якщо необхідна координатна сітка, то потрібно додати ключове слово **grid**, а прибрати сітку можна за допомогою команди **grid off**.

Заголовок графіка можна вивести за допомогою команди **title.** Після звернення до процедури **plot** викликати **title** можна таким чином: **title('*текст*')**.

Зверху в полі фігури з'явиться текст, записаний між апострофами. Нагадаємо, що в команді **title** текст завжди повинен поміщатися в апострофи.

Аналогічно можна вивести пояснення до графіка, які розміщуються уздовж осей за допомогою функцій **xlabel** і **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('Вісь X'); ylabel('Вісь Y');
```

приведе до оформлення поля у вигляді, представленому на рис.4.

Примітка: якщо в заголовку графіка або в поясненнях замість кирилічних символів з'являється нісенітниця, потрібно змінити шрифт відображення відповідного елементу. Для цього необхідно активізувати ікону із стрілкою (праворуч від ікони з принтером), навести покажчик миші на відповідний напис, натиснути праву кнопку миші; в меню, що з'явилося, вибрати **Properties…** → вкладка **Text**. У випадному меню **Font name** вибрати новий шрифт, що містить кирилицю, наприклад **Arial Cyr.**

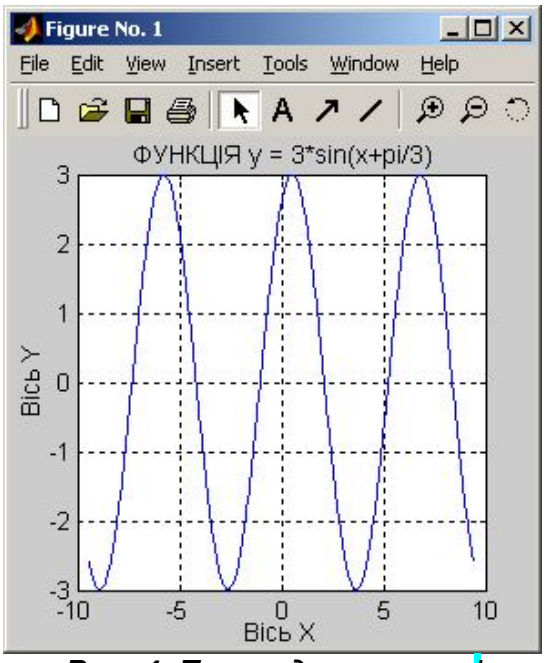


Рис. 4. *Приклад заголовка і пояснень до графіка.*

Неважко вивести функцію, задану параметрично. Хай необхідно побудувати графік функції(x), яка задана формулами:

$$x = 4e^{-0,5t} \cdot \sin(t), \qquad y = 0,2e^{-0,1t} \cdot \sin(2t).$$

Виберемо діапазон зміни параметра **t** від **0** до **50** з 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')
>> grid
```

отримаємо графік, приведений на рис. 5.

Розглянемо, як задавати стиль лінії, тобто параметр **s** в аргументах функції plot**.**

Стилі **s** задаються у вигляді набору трьох символьних маркерів, узятих в одиночні лапки (апострофи). У таблицях 1–3 приведені параметри для задання різних типів маркерів.

Є й інші маркери крапок, що проставляються.

Можна вказувати не всі три маркери. Порядок теж не має значення.

Табл. 1. *Маркер типу лінії*

Маркер	-	- -	:	-.
Тип лінії	Безперервна	Штрихова	Пунктирна	Штрихпунктирна

Табл. 2. *Маркер типу крапок, що проставляються*

Маркер	.	+	*	o	x	s	d
Тип крапки	.	+	*	°	x	□	◊

Табл. 3. *Маркер кольору*

Маркер	Колір лінії	Маркер	Колір лінії	Маркер	Колір лінії	Маркер	Колір лінії
c	Блакитний	y	Жовтий	g	Зелений	w	Білий
m	Фіолетовий	r	Червоний	b	Синій	k	Чорний

Як приклад розглянемо графіки двох функцій **y=sin(x)** і **u = cos(x):**

```
>> x = 0 : 0.3 : 2*pi;
>> y = sin(x);
>> u = cos(x);
>> plot(x,y,'ro:',x,u,'k*-.')
```

Результат представлений на рис.6.

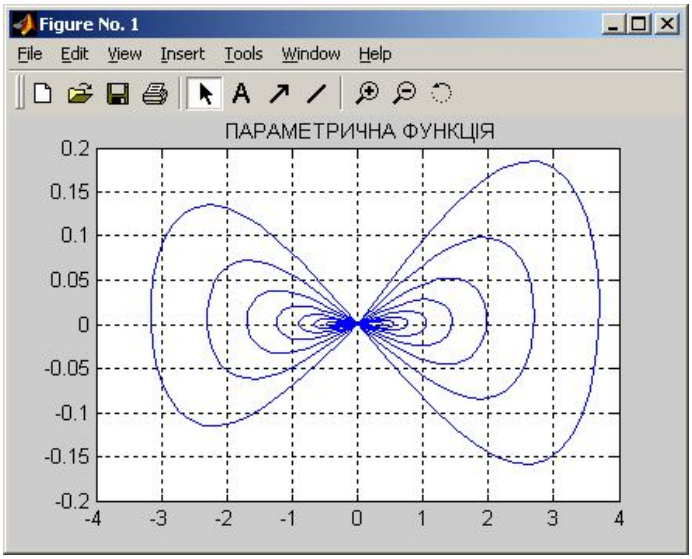


Рис. 5. *Приклад графіка параметричної функції*

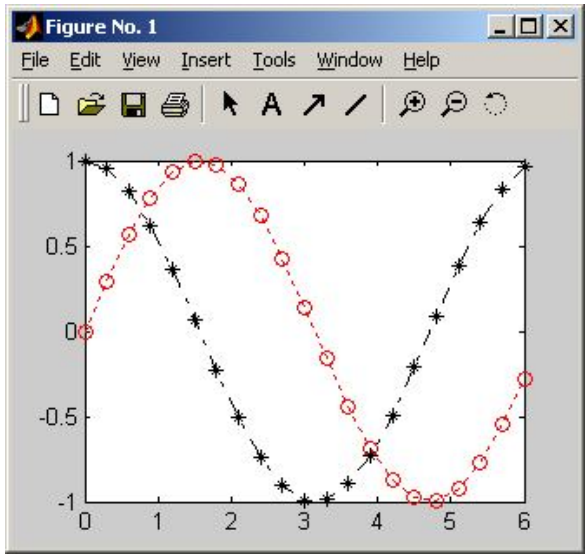


Рис. 6. *Приклади різних стилів ліній*

У чорно-білому зображенні червона лінія виглядає більш блідою. Щоб декілька послідовно обчислених графіків (тобто що виводяться різними командами **plot**) були відображені в одному графічному вікні, можна використовувати команду **hold on.**

Тоді кожен наступний графік будуватиметься в тому ж заздалегідь відкритому графічному вікні, тобто нова лінія додаватиметься до раніше побудованих. Команда **hold off** відключає цей режим.

В деяких випадках попередня інформація в графічному вікні заважає користуванню новою, тоді заздалегідь можна очистити вікно командою **clf reset** або відкрити нове вікно командою **figure.**

MATLAB має багато інших можливостей по "облагороджуванню" графіків: зміна розмірів вікна, розміщення пояснюючих написів, об'єднання декількох фігур в одне вікно і т.д. Просте редагування графіків виконується за допомогою лінійок інструментів, які викликаються в меню **View** (**Вигляд**) графічного вікна. За умовчанням встановлена лінійка **Figure Toolbar**. Позначення елементів лінійки звичайні для редакторів типа Word, за винятком стрілки праворуч від ікони принтера. Коли стрілку натиснуто, можна викликати редактор графіків. Для цього слід вибрати редаговану криву і зробити на ній подвійне клацання лівою клавішею миші. У вікні редактора можна міняти товщину, колір, тип лінії і т. п.

Процедура plot3

Для побудови тривимірних графіків використовується команда **plot3**. Правила її використання аналогічні команді plot2, тільки масивів для побудови кривої повинно бути три, наприклад:

```
>> t = 0:pi/50:10*pi;
>> x=sin(t); y = cos(t);
>> plot3(x,y,t); grid on
```

Результат побудови представлений на рис. 7. Товщина траєкторії кривої збільшена редактором графіків.

Більший інтерес представляють поверхні. Для їх побудови необхідно створити "об'ємний" масив крапок, який можна сформувати функцією **meshgrid.** Вона створює два двовимірні масиви **x**

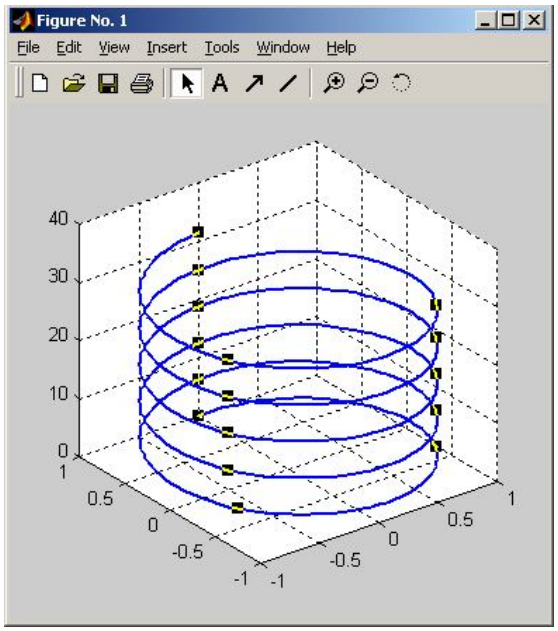


Рис. 7. *Приклад побудови тривимірного графіка*

і **y**, на яких будується сітка координат для обчислення змінної **z**. Наприклад, команда

```
>> [x,y] = meshgrid(0:3, 0:3)
```

формує масив

$$\mathbf{x} = \begin{Bmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{Bmatrix} \quad \text{і масив} \quad \mathbf{y} = \begin{Bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \end{Bmatrix}$$

Для обчислення значень змінної **z** використовуються крапки $(x_{1,1}; y_{1,1})$ $(x_{1,2}; y_{1,2})$ ... $(x_{i,k}; y_{i,k})$ ... $(x_{4,4}; y_{4,4})$.

Побудуємо тривимірну поверхню, що описується функцією $\mathbf{z(x,y)=x^2+y^2}$:

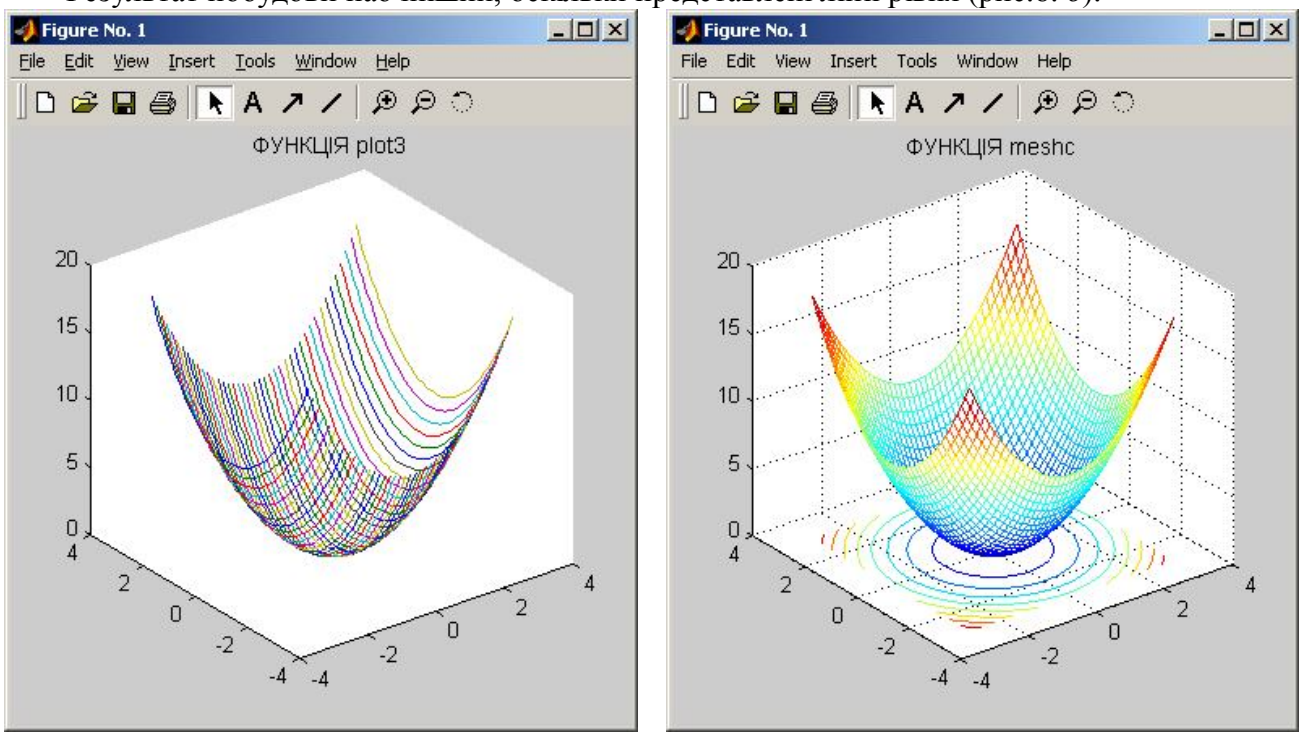
```
>> [x y] = meshgrid([-3:0.15:3]);
>> z = x.^2+y.^2;
>> plot3(x,y,z)
```

Результат представлено на рис.8,*а*.

Ми розглянули дві найпростіші команди для роботи з графікою в їх найпростіших варіантах. Є й інші команди. Наприклад, якщо в останньому прикладі замінити **plot3** на **meshc**, то отримаємо:

```
>> [x y] = meshgrid([-3:0.15:3]);
>> z = x.^2+y.^2;
>> meshc(x,y,z)
```

Результат побудови наочніший, оскільки представлені лінії рівня (рис.8. *б*).



а ***б***

Рис. 8. *Приклад побудови поверхні: а) функцією* ***plot3****; би) функцією* ***meshc***

Взагалі система MATLAB містить дуже широкий набір команд і функцій, як загальних, так і спеціальних, для роботи з графікою (аж до рухомих і озвучених кривих і поверхонь). Причому майже кожна функція має декілька варіантів застосування, що взагалі характерно для MATLAB. Для ознайомлення з усіма цими можливостями звертайтеся до системи **help** і літератури, наприклад, [1, 2] та ін. Тут ми приведемо імена лише декількох з них (це полегшить пошук споріднених команд):

- **loglog, semilogx, semilogy** – для побудови логарифмічних і напівлогарифмічних графіків;

- **feather** – проекції векторів на площину;
- **polar** – графіки в полярних координатах;
- **bar, bar3** – стовпчикові діаграми;
- **hist** – гістограми;
- **stairs** – сходові графіки;
- **contour, contours** – контурні графіки (будують лінії рівня);
- **quiver** – графіки полів градієнтів;
- **mesh** – виводить сітчасту поверхню функції z(x,y);
- **surf, surfc, surfl** – графіки з функціональним забарвленням і підсвічуванням;
- **slice** – побудова перетинів 3D-поверхонь;
- **subplot** – створення в графічному вікні декількох підвікон;
- **comet3** – рух крапки в просторі; і т.д.

Є також засоби низькорівневої дескрипторної графіки [1], які істотно розширюють можливості візуалізації результатів обчислень.

Приклад графічного рішення задачі

Графіки в **MATLAB** використовують як для ілюстрації, так і для вирішення завдань. Одним з таких прикладів є завдання знаходження коренів рівнянь. Вона ділиться на два етапи [3] – відділення коренів і уточнення коренів. У MATLAB достатньо зручно і перший, і другий етап виконати графічно.

Для прикладу знайдемо корені рівняння $e^{-2x} + 3x - 2 = 0$. Відомо: якщо побудувати графік функції, що стоїть у лівій частині рівності, то точки перетину графіка з віссю **x** відповідатимуть кореню рівняння.

"Грубо" визначимо діапазон розташування коренів:

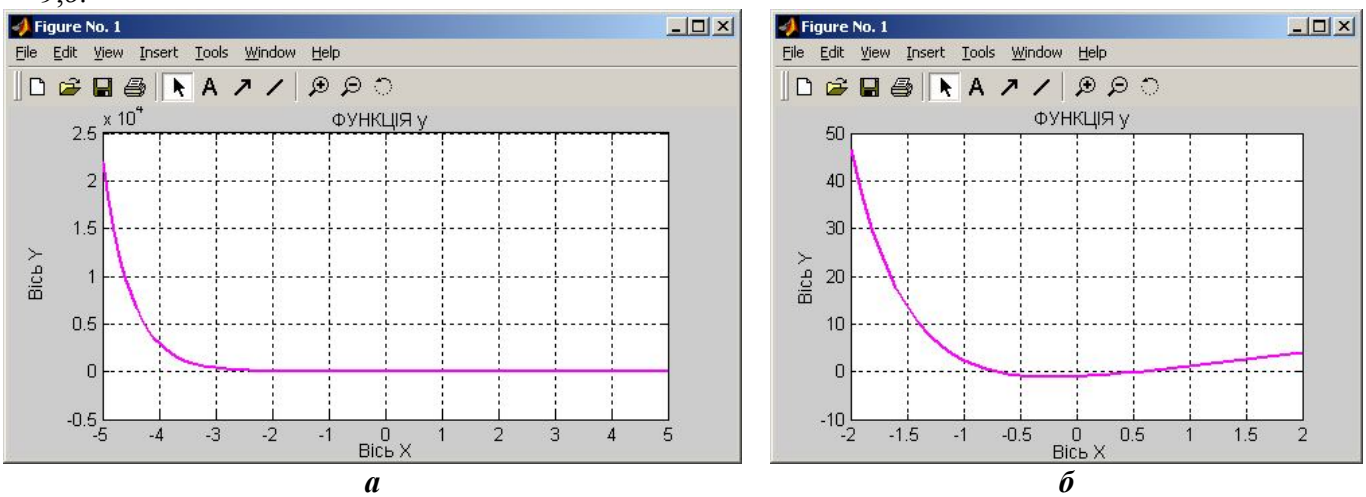
- якщо $x \to -\infty$, то $y \to \infty$, тобто все визначає член e^{-2x};
- якщо $x \to \infty$, то $y \to \infty$, тобто все визначає член $3x$.

Очевидно, що при **x < -5**, **y** завжди буде більше нуля, а при **x > 5**, **y** теж завжди буде більше нуля. Тому спершу виберемо інтервал зміни **x** – (-5; 5); колір лінії задамо фіолетовий. Відповідний набір команд:

```
>> x=-5:0.1:5;                    >> title('ФУНКЦІЯ y ');
>> y=exp(-2*x)+3*x-2;             >> xlabel('Вісь X');
>> plot(x,y,'m'), grid on         >> ylabel('Вісь Y');
```

Результат представлено на рис. 9,*а*.

Аналізуючи малюнок, можна зробити висновок, що крива стосується осі **x** в околиці (-1; 1), тобто корені десь тут. Для гарантії побудуємо новий графік на інтервалі (-2; 2) – рис. 9,*б*.



а ***б***

Рис. 9. *Приклад графічного рішення рівняння (відшукання коренів)*

Чітко видно, що існують два корені: один – в околиці **x1 = -0,7**, а інший – при **x2 = 0,5**. Визначимо перший корінь з точністю до трьох знаків. Для цього діапазон зміни **x** виберемо (-0,75; -0,65). Можна бачити, що **x1 = -0,711**. Зменшуючи діапазон, можна і далі підвищувати точність.

Аналогічно побудувавши графік для другого кореня, можна побачити, що **x2 = 0,486.**

Висновок: рівняння $\mathbf{e^{-2x} + 3x - 2 = 0}$ має два речовинні корені – **x1 = -0,711** і **x2 = 0,486**.

ПРАКТИЧНІ ЗАВДАННЯ

1. Ввести матрицю

$$\mathbf{A} = \left\{ \begin{matrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ pi & 2i & 4j & 2pi \\ 1+2i & 0 & -3 & pi/2 \end{matrix} \right\}$$

різними способами: у природному вигляді; у вигляді одного рядка; у вигляді одного стовпця; у вигляді 2x8 і 8x2.

2. Введену в п.1 матрицю **A** перетворити у вектор-стовпець, потім – у вектор-рядок.

3. Переконатися, що після виконання дій

```
>> A = [1 2; 3 4]; B = [4 3;2 1]; C = [5 6;7 8]; D = [8 7;7 6];
>> F = [A B;C D];
>> F(:,1)= [];  F(:,3)= [];  F(1,:)= []; F(3,:)= []
```

вийде наступний результат:

```
F =
     4  2
     6  8
```

4. За допомогою оператора **":"** створити масив **x = [0, 1, 2, 3, 4, 5]**, потім обчислити **cos(x)** і **sin(x)/x**. Якщо в другій функції вийде одне число – подумати, в чому справа.

5. В отриманому масиві **x** змінити порядок елементів на зворотний і обчислити суму елементів з накопиченням, тобто $x_k = \sum_{i=1}^{k} x_i$, k = 1, 2..., 6. Скористатися функціями **fliplr** і **cumsum**.

6. З масиву **x** (п.5) побудуйте матрицю Вандермонда. Нагадаємо загальний вигляд такої матриці:

$$\mathbf{W} = \left\{ \begin{matrix} 1 & x_1 & x_1 2 & x_1^3 & \ldots\ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & x_2^3 & \ldots\ldots & x_2^{n-1} \\ 1 & x_3 & x_3^2 & x_3^3 & \ldots\ldots & x_3^{n-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & x_n & x_n^2 & x_n^3 & \ldots & x_n^{n-1} \end{matrix} \right\}$$

Бажано все зробити однією командою, можна скористатися функціями **repmat**, cumprod.

7. Знайти всі речовинні корені рівнянь (табл.4) графічним методом з точністю до трьох знаків. Графіки повинні мати назву і найменування осей. Колір ліній вказаний в таблиці. Місця розташування коренів слід вказати стрілками і супроводжувати написами, наприклад, "корінь 1" і т.п. Номер варіанту – остання цифра номера за списком в журналі.

Таблиця 4

№ варіанту	Рівняння	Колір лінії
1, 6	x - 4 = cos(x)	Блакитною
2, 7	x3 - 3x + 1 =0	Фіолетовий
3, 8	x·ln(x) - 14 = 0	Червоний
4, 9	x4 + x2 - 2x - 2 = 0	Зелений
5, 0	x = sin(x)+ 0,25	Чорний

8. Побудувати поверхні функцій (табл. 5). Графіки повинні мати назву і найменування осей. Всі характерні ділянки (опуклості, угнутості, екстремуми і т.п.) повинні бути вказані і прокоментовані безпосередньо на графіку. Якщо використовуватимете функцію **plots**, то виділіть будь-яку лінію і змініть її товщину за допомогою редактора графіку. При обчисленні функцій не забувайте про операцію поелементного множення. Номер варіанту – остання цифра номера за списком в журналі.

Таблиця 5

№ варіанту	Функція	Межі зміни аргументів (x, y)
1, 9	z = sin(x)/(x2+y2+0.3)	-3 : 0.1 : 3
0, 7	$z = x*e^{(-x*x-y*y)}$	-2 : 0.1 : 2
8, 4	$z = e^{(-x*x-y*y)}$	-2 : 0.1 : 2
2, 6	z = x2-y2)	-2 : 0.1 : 2
3, 5	z = x3 + y3 - 3xy	-4 : 0.1 : 4

9. Дослідити режими роботи схеми випрямляча згідно варіанту (табл. 6). Під дослідженням розуміється: а) розрахунок основних параметрів роботи відповідної схеми; б) відображення на графіку регулювальних характеристик (змін параметрів залежно від зміни кута регулювання α). При дослідженні скористатися теоретичними відомостями лабораторної роботи СПТ-1. Номер варіанту – остання цифра номера за списком в журналі.

Таблиця 6

№ варіанту	Досліджувана схема	Напруга обмоток трансформатора		Опір навантаження
		Первинне	Вторинне	
N		U_1, В	U_2, В	R_d, Ом
6, 3, 4	Однофазна двонапівперіодна з середньою крапкою	220	U2=N10	N2
9, 7	Однофазна мостова			
5, 8, 1	Трифазна з середньою крапкою			
2, 0	Трифазна мостова			

ПИТАННЯ ДЛЯ САМОПЕРЕВІРКИ

1. Як здійснюється введення–виведення інформації в командному вікні?
2. Що таке системні змінні?
3. Як використовується оператор "двокрапка"?
4. Дві групи операцій над векторами і матрицями?
5. Як зберегти робочий простір?
6. Коли перед знаком операції ставиться крапка?
7. Який загальний вид команди plot?
8. Як вивести координатну сітку?
9. Як вивести назву графіка?
10. Як вивести назву осей?
11. Як задати типи ліній графіків, їх колір і форму крапок, за якими вони будується?
12. Як сформувати масив для побудови поверхні?

ЛИТЕРАТУРА

1. Дьяконов В. П., Абраменюва И.В. MATLAB 50/53 Система символьной математики. – М.: Нолидж, 1999 .– 640 с.

2. Потемкин В. Г. Система инженерных и научных расчетов MATLAB 5 x. В 2-х т. – М.: ДИАЛОГ-МИФИ, 1999. - 366 с. (т. 1) – 304 с. (т. 2)

3. Демидович Б.П., Марон И.А. Основы вычислительной математики. – М.: Наука, 1966. – 664 с.

Укладачі:
Ковальов Олександр Робертович
Дибрін Сергій Володимирович

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи СПТ-4
“Вивчення командного режиму роботи і прийомів візуалізації результатів обчислень в пакеті MATLAB для застосування у вивченні курсу "Перетворювальна техніка"”
для студентів напряму 0906 – Електротехніка
денної та заочної форм навчання.

Редакційно-видавничий комплекс
Друкується в обробці укладачів

Підписано до друку . .05. Формат 30х42/4.
Папір Pollux. Різографія. Условн.-печат. лист. .
Учетно-іздат. лист. . Тираж 30 прим.
Заст. № .

НГУ
49600, м. Дніпропетровськ-27, пр. К. Маркса, 19.