НАВЧАЛЬНО-МЕТОДИЧНА ЛІТЕРАТУРА

Міністерство освіти та науки України
Тернопільський національний технічний університет
імені Івана Пулюя

Кафедра біотехнічних систем

МЕТОДИЧНІ ВКАЗІВКИ

для виконання практичних робіт
з дисципліни

ОБРОБКА БІОМЕДИЧНИХ СИГНАЛІВ

для студентів спеціальності
163 «Біомедична інженерія»
та напряму підготовки
6.051402 «Біомедична інженерія»

Тернопіль, 2018

Хвостівський М.О. Методичні вказівки для виконання практичних робіт з дисципліни “Обробка біомедичних сигналів” для студентів спеціальності 163 «Біомедична інженерія» та напряму підготовки 6.051402 «Біомедична інженерія» // Хвостівський М.О. – Тернопіль: ТНТУ імені Івана Пулюя, 2018. – 52 с.

Укладач: к.т.н., доц. Хвостівський М.О.

Відповідальний за випуск: зав. кафедрою Яворська Є.Б.

Методичні вказівки розглянуто та затверджено на засіданні кафедри біотехнічних систем Тернопільського національного технічного університету імені Івана Пулюя, протокол № 3 від 23 лютого 2018 р.

Методичні вказівки схвалено та рекомендовано до друку на засіданні методичної ради факультету прикладних інформаційних технологій та електроінженерії Тернопільського національного технічного університету імені Івана Пулюя, протокол № 3 від 23 лютого 2018 р.

ЗМІСТ

Практична робота №1

Ознайомлення з роботою MATLAB

Мета роботи: Вивчити основні принципи роботи, та ознайомитись з базовими командами системи MATLAB.

ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Інтегрований програмний комплекс Matlab

Matlab - це інтерактивне середовище для виконання різноманітних наукових та інженерних розрахунків. Назва пакета походить від скорочень англійських слів Matrix Laboratory і найкраще характеризує його сутність, де матричні операції є основою більшості розрахунків.

Пакет підтримує виконання операцій з векторами, матрицями та масивами даних, реалізує сингулярні і спектральні розкладання, підтримує роботу з алгебраїчними поліномами, вирішує нелінійні рівняння і задачі оптимізації, інтегрування в квадратурах, вирішує диференціальні рівняння, будує різного виду графіки, трьохмірні поверхні та лінії рівня.

До найважливіших особливості пакета відносять:

• можливість вибору та зміни платформи - програми та дані можна переносити на різні типи комп'ютерів з різними операційними системами;

• відкрита архітектура з точки зору можливості створення спеціальних підпрограм, спрямованих на розв'язування певного класу задач. Такі підпрограми можна написати як за допомогою мови програмування самого пакета (так звані m-файли), так і мовою програмування С. Отже, кожен користувач пакета може зробити свій внесок у розширення його можливостей.

До основних областей використання Matlab відносять:

- математичні обчислення;
- розробка алгоритмів;
- обчислювальний експеримент, моделювання;
- аналіз інформації, дослідження та візуалізація результатів;
- наукова та інженерна графіка;
- розробка додатків.

Спеціальні підпрограми, пов'язані з розв'язанням певного класу задач, формують тематичні підкаталоги (toolbox). Можна виділити такі найважливіші toolbox -и:

• System Identification Toolbox призначений для аналізу сигналів у системах керування. Дозволяє використовувати параметричні та непараметричні алгоритми ідентифікації, зокрема, розрахунок і верифікацію моделі, вибір порядку моделі, демонстрацію та перетворення сигналів. Дає змогу за відомими вхідними та вихідними сигналами об'єкта, попередньо задавши порядок моделі, створити його модель у вигляді рівнянь стану.

• Control System Toolbox призначений для синтезу, аналізу та моделювання неперервних у часі та дискретних систем. Дозволяє

використовувати різні форми опису системи (передавальна функція, система рівнянь змінних стану, розкладання на прості дроби). Виконує перетворення від однієї форми запису системи до іншої, а також від неперервної до дискретної та навпаки. Дає змогу досліджувати реакцію системи на різні типи вхідних сигналів, а також синтезувати регулятори. Дозволяє аналізувати поведінку системи в частотній області.

• Signal Processing Toolbox призначений для цифрового перетворення та аналізу сигналів у часовій та частотній областях. Дозволяє проектувати цифрові та аналогові фільтри. Можливим є параметричне моделювання.

• Fuzzy Logic Toolbox включає середовище моделювання в області нечіткої логіки разом з засобами до проектування інтелектуальних систем керування.

• μ-Analysis and Synthesis Toolbox вимагає інсталяції Signal Processing Toolbox і є пакетом для аналізу та синтезу лінійних робастних систем керування (систем керування з підвищеною стійкістю). Використовується для проектування оптимальних систем керування, основна увага акцентується на питаннях стійкості системи та її вразливості до зміни параметрів.

• Heiral Network Toolbox спрощує побудову та дослідження штучних нейронних мереж. Дає змогу використовувати різні алгоритми навчання нейронних мереж. Реалізує різні типи нейронів і нейронних мереж. Кожен нейрон описується вектором ваг, значенням зміщення та видом активаційної функції. Зв'язок із Simulink-ом дає змогу використовувати штучні нейронні мережі як окремі блоки в моделях досліджуваних систем.

• Nonlinear Control Design Toolbox дає змогу виконувати оптимізацію лінійних та нелінійних систем керування.

• Optimizatioп Toolbox реалізує різні методи оптимізації лінійних і нелінійних систем та розв'язування систем нелінійних рівнянь. Функції пакету дають змогу знаходити екстремум довільної функції як за наявності, так і за відсутності обмежень, а також для випадку багатокритеріальної оптимізації.

• Robust Control Toolbox призначений для дослідження багатовимірних робастних систем керування. Виконує синтез оптимальних регуляторів.

• Spline Toolbox призначений для розв'язування задач апроксимації та інтерполяції за допомогою сплайнів, з можливістю інтегрування та диференціювання отриманого рівняння. .

• Statistics Toolbox реалізує різноманітні статистичні функції, включаючи моделювання випадкових подій та генератори випадкових сигналів.

1.2 Режими роботи Matlab

Після входу в середовище Matlab на екрані розкривається вікно (Рисунок 1.1), у якому з'являється символ ">>", що сигналізує про готовність пакета до роботи.

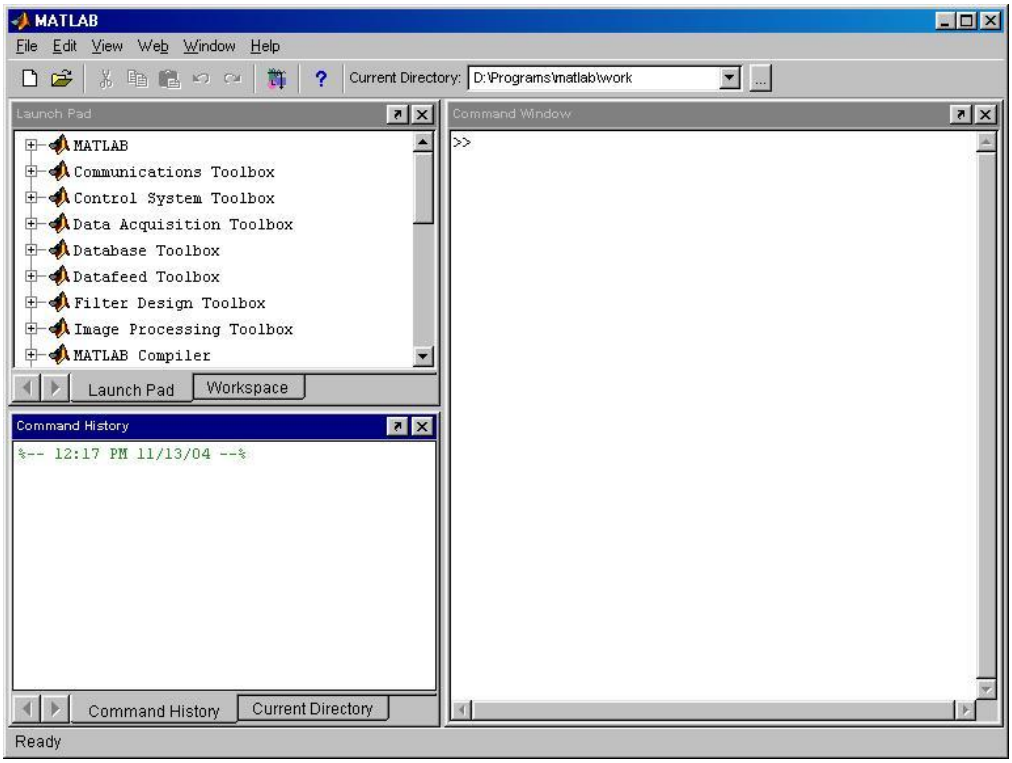

Рис. 1.1. Вигляд головного вікна середовища пакету Matlab

Починаючи з цього моменту кожна команда з клавіатури буде сприйнята, інтерпретована та виконана. Результат обчислення значення виразу є доступним як значення змінної, якій присвоєний вираз. Так, виконання команди наступного синтаксису:

>> змінна = вираз

дозволить обраховувати значення виразу, внести змінну до робочої області, а на появиться результат.

Пакет Matlab має деякі особливості:

- вбудовані функції (наприклад, sin x) записуються прописними буквами, а їх аргументи вказуються в круглих дужках;
- значення змінних записані з використанням букв різного регістру будуть різними (наприклад, A і a);
- для блокування виводу результату обчислень деякого виразу після нього треба поставити знак “;” (крапка з комою);
- у деяких випадках математичний вираз, що вводиться, може виявитися настільки довгим, що для нього не вистачить одного рядка. У цьому випадку частину виразу можна перенести на новий рядок за допомогою знаку три крапки “...”;
- пакет MatLab є насамперед програмою для роботи з матрицями, тому звичайні математичні операції *, /, +, - виконуються в матричному виді. Для поелементного виконання арифметичних операцій використовується оператор “.” (наприклад Z=V./A);

- елементи векторів і матриць записуються в квадратних дужках, та розділяються пробілами або комами (наприклад $V=[1\ \ 2\ \ 3\ \ 4]$ і $V=[1,\ 2,\ 3,\ 4]$ – ці записи є ідентичними).

1.3. Команди керування вікном командного режиму

До основних команд керування вікном командного режиму відносять наступні команди:

- **clc** – очищає екран та розміщує курсор у лівому верхньому куті порожнього екрану;
- **home** – повертає курсор у лівий верхній кут вікна;
- **echo <file_name> on** – включає режим виведення на екран тексту Script-файлу (файлу-сценарію);
- **echo <file_name> off** – відключає режим виведення на екран тексту Script-файлу;
- **echo <file_name>** – змінює режим виведення на протилежний;
- **echo on all** – включає режим виведення на екран тексту всіх m-файлів;
- **echo off all** – відключає режим виведення на екран тексту всіх m-файлів;
- **more on** – включає режим посторінкового виведення (корисний при перегляді великих m-файлів);
- **more off** – відключає режим посторінкового виведення (у цьому випадку для перегляду великих файлів треба використовувати лінійку прокручування);
- **diary file_name.txt** – веде запис на диск усіх команд у рядках введення та отриманих результатів у вигляді текстового файлу з зазначеним ім'ям;
- **diary off** – призупинити запис у файл;
- **diary on** – починає запис у файл.
- **clear <name I>, <name 2>, ...,** – використовується для знищення певних змінних чи функцій з робочої області пакету;
- **clear all** – знищення усіх змінних або функцій з робочої області пакету;
- **help** – видає назви всіх доступних файлів допомоги;
- **help <filename>** – допомога в роботі з окремим файлом з зазначеним ім’ям.

Клавіші ↑ і ↓ використовуються для підстановки після маркера рядка введення “>>” раніше введених стрічок, наприклад для їхнього виправлення, дублювання або доповнення.

У випадку великих програм, під час використання циклічних операторів доцільніше записати необхідну програму у вигляді m-файлу, а потім подати його назву в командній стрічці. При необхідності програму з файлу можна вивести на екран за допомогою команди **type**. Створення такого файлу можна здійснювати за допомогою будь-якого текстового редактора, навіть поза пакетом.

Для ілюстрації можливостей пакета та окремих його toolbox-ів служать численні демонстраційні програми, що мають потужну систему підказок та пояснень виконаних дій. Для доступу до описаних вище ілюстрацій роботи пакета необхідно в командній стрічці набрати:

>>demo

Відкривши за допомогою "мишки" необхідний каталог у дереві каталогів “Matlab Demos”, користувач отримує доступ до демонстраційних файлів, запуск яких здійснюється за допомогою кнопки **“**Run”**.**

2.4 Типи та формати даних

Matlab не вимагає декларації типу даних чи їх розміру. Ім'я змінної може складатися з довільної комбінації букв та цифр, але не більше 19 знаків, при цьому перший символ має бути буквою. Дані можуть бути занесені в робочу область пакету в скалярній та матричній формі.

Для ілюстрації різних форматів розглянемо вектор, що містить два елементи – числа:

$$X = [4/3 \;\; 1.2345e\text{-}6]$$

У різних форматах представлення числа будуть мати наступний вигляд:

format short	1.3333	0.0000
format short e	1.3333e+000	1.2345e-006
format long	1.33333333333333	0.00000123450000
format long e	1.33333333333333e+000	1.23450000000000e-006
format bank	1.33	0.00

Задання формату відбивається тільки на формі виведення чисел. Обчислення завжди відбуваються у формі подвійної точності, а введення чисел можливе в будь-якому зручному для користувача вигляді.

У середовищі пакету визначена змінна типу string. Змінна такого типу є довільним текстовим фрагментом, записаним з допомогою апострофів, причому розрізняють верхні та нижні символи. Текст запам'ятовується у вигляді вектора, а кожен знак тексту становить окремий елемент такого вектора. Наприклад, запис s = 'student' є текстовою змінною s = student.

1.5 Системні змінні

Основні системні змінні, що застосовувані в системі MatLab:

- **i** або **j** – уявна одиниця (квадратний корінь з -1), (наприклад >> $c = 45 + 56*i$);
- **pi** – число $\pi = 3.141592\ldots$;

- **eps** – похибка операцій над числами з плаваючою крапкою (2^{-52});
- **realmax** – найбільше число з плаваючою крапкою;
- **realmin** – найменше число з плаваючою крапкою;
- **ans** – змінна, що зберігає результат останньої операції;
- **NaN** – вказівка на нечисловий характер даних (Not-a-Number), а також на невизначеність 0/0

1.6 Елементарні математичні функції пакету

Прийнято, що всі функції пакета Matlab визначені за допомогою малих літер**,** а звертання до них з використанням великих літер буде трактовано як помилка**.** Деякі елементарні математичні функції, доступні в середовищі пакета наведені в таблиці 2.1.

Таблиця 1.1 - Основні математичні функції пакету

Тригонометричні	
sin, cos, tan, cot	тригонометричні функції
asin, acos, atan, acot	обернені тригонометричні функції
sinh, cosh, tanh, coth	гіперболічні функції
asinh, acosh, atanh, acoth	обернені гіперболічні функції
sec, csc	секанс і косеканс
asec, acsc	обернені функції секанса і косеканса
sech, csch	гіперболічний секанс і косеканс
asech, acsch	обернені гіперболічні функції
Логарифмічні	
exp	експонента
log	натуральний логарифм
log10	десятковий логарифм
log2	логарифм за основою два

Піднесення до степені	
x^y	піднесення числа x у степінь y
pow2(n)	підносить 2 у степінь n
sqrt	корінь квадратний
nextpow2	від виразу 2^n повертає степінь n
Функції комплексного аргументу	
abs	модуль комплексного числа
angle	фаза комплексного числа
conj	комплексно-спряжене число
imag	уявна частина комплексного числа
real	дійсна частина комплексного числа
cplxpair	сортування на комплексно-спряжені парі

1.7 Текстові коментарі

Оскільки MatLab використовується для досить складних обчислень, важливе значення має наочність їхнього опису. Вона досягається за допомогою

текстових коментарів. Текстові коментарі вводяться за допомогою символу % (наприклад % It is factorial function).

При введенні букви “с” російського алфавіту буде відбуватися перехід на наступну стрічку. Рекомендуємо міняти російське “с” на англійське, що на загальний вид коментарю ніяк не позначається. Так само не рекомендуємо вводити російськомовні коментарі й у тексти m-файлів, що може привести до того, що програма стане непрацездатною.

1.8. Робота з mat-файлами

Файли з розширенням *.mat є бінарними, у них можуть зберігатися значення змінних робочої області.

Для збереження використовується команда:

>> save FILENAME

У разі необхідності збереження значення тільки окремих змінних після імені файлу необхідно податі перелік цих змінних, наприклад

>>save C:\Users\mia\labor1.mat X Y Z

Другий спосіб збереження через панель інструментів Save Workspace as…

Для завантаження збережених змінних у робочу область необхідно виконати наступну команду:

>> load FILENAME

де FILENAME –ім’я файлу у якому збережені дані.

Якщо потрібно завантажити окремі змінні необхідно ввести команду:

>> load FILENAME X Y Z

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

1.9 Завдання для виконання

1.9.1 Перед виконанням роботи ознайомитись з теоретичними відомостями.

1.9.2. У відповідності до отриманого номеру варіанту вибрати значення змінних для виконання лабораторної роботи (таблиця 2.2) та вирази для обчислення (таблиця 2.3).

1.9.3. Зберегти значення змінних у mat-файл з назвою <name>, де name – ваше прізвище.

1.9.4. Очистити робочу область та вікно. Завантажити значення змінних з mat-файлу та обчислити значення виразу. Результат зберегти.

1.9.5. Створити текстові коментарі до роботи, провести запис на диск фрагменту роботи, використовуючи команди керування вікном.

1.10. Зміст звіту

1.10.1. Тема та мета роботи.

1.10.2. Завдання до роботи.

1.10.3. Відобразити отримані результати (п. 2.3-2.5) у вигляді копій екрану.

1.10.4. Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Коротка характеристика MATLAB.
2. Перелічіть основні операції, що можна виконувати засобами MATLAB.
3. Перелічіть та дайте коротку характеристику тематичним підкаталогам MatLab (toolbox).
4. Головне вікно пакету MatLab.
5. Перелічіть області використання MatLab.
6. Особливості роботи з пакетом MatLab.
7. Які ви знаєте команди керування вікном командного режиму?
8. Як подивитись демонстрації програми?
9. Для чого використовується команда format?
10. Перелічіть системні змінні Matlab.
11. Назвіть основні тригонометричні функції Matlab.
12. Як закоментувати стрічку Matlab?
13. Робота з mat-файлами.

Таблиця 1.2 – Варіанти завдання

Номер варіанту	**Значення змінних**					**Номер виразу**
	a	**b**	**R**	**W**	**L**	
1	7,1	2	1,89	2,1	3,2	1
2	0,43	2,1	1	9,8	2,45	2
3	0,12	2,3	1,1	3	1,89	3
4	0,34	4,5	2,34	1,32	5	4
5	0,45	6	4,3	3	5	5
6	0,1	1	2	2,34	2,45	6
7	2,3	1,9	2,31	2	7,6	7
8	4,3	0,9	1,78	4	3,2	8
9	2,1	0,45	1,78	1,7	2	9
10	1,4	0,12	2,9	2	2,4	10
11	2,3	1,34	8,1	7	5,1	11
12	1,4	1,98	2	2,8	0,98	12
13	8,6	1,78	1,1	3	0,56	13
14	5,4	1,4	2	1,89	2,1	14

15	3,1	0,3	0,89	2,9	2,3	15
16	0,43	3,5	0,56	3,4	4,5	16
17	9	4,3	0,54	2,89	6	17
18	1	2,1	0,67	2,5	1	18
19	2,1	3,2	9,1	1,9	1,9	19
20	2,2	1,5	4,3	4,6	0,9	20
21	3,1	2,7	4,7	0,45	0,45	21
22	1,67	2,8	8,1	0,87	0,12	22
23	1,45	5,1	3,2	0,3	1,34	23
24	1,9	0,9	1,34	0,1	1,98	24
25	1,2	8,1	2,65	0,01	1,78	25
26	1,34	1,5	1,89	1,4	1,1	26
27	1,56	0,56	1,23	2,1	2	27
28	1,67	0,9	2,31	1,98	0,89	28
29	1,89	1,1	1,78	9,7	0,56	29
30	1,78	1,78	1,9	1,76	0,54	30

Таблиця 1.3 – Математичні вирази

Номер виразу	**Значення виразу**
1	$X = tg(\frac{1}{2} arctg(\frac{2a}{1-a^2-b^2}) + \frac{b}{4} \cdot \ln(\frac{a^2+(b+1)^2}{a^2+(b-1)^2})$
2	$X = tg(\frac{\sin 2a}{\cos 2a + \sin 2b} + j\frac{2b}{\cos 2a + \cos 2b})$
3	$X = tg(\frac{tba}{\cos 2a + \cos 2b} + b\frac{\sin 2b}{2a + \cos 2b})$
4	$X = \frac{1}{2} + \frac{\sin(W^2/2)}{\pi a} \cdot (1 - \frac{3}{(\pi b^2)^2}) - \frac{\cos(\pi\pi^2/2)}{\pi^2 L^3} \cdot (1 - \frac{5}{(\pi L^2)^2})$
5	$X = \frac{1}{2} - \frac{\cos(\pi\pi^2/2)}{\pi R} \cdot (1 - \frac{3}{(\pi R^2)^2}) - \frac{\cos(\pi\pi^2/2)}{\pi^2 W^3} \cdot (1 - \frac{5}{(\pi x^2)^2})$
6	$X = \frac{RL(R+L) + R(wL - 1/wa)^2}{(R+L)^2 + (wL - 1/wa)^2} + b\frac{R(wL - 1/wa)}{(R+L)^2 + (wL - 1/wa)^2}$
7	$X = 16 \cdot \omega^2 a \left[\ln(1 + \frac{\pi}{b/a}) + \frac{1}{3.64 + 2(L/a) + 0.51(L/a)^2} \right]$
8	$X = \frac{120}{\sqrt{b}} \left[arch\frac{a}{W} - \ln\sqrt{1 + (\frac{a}{2L})^2} \right]$
9	$X = \sqrt{1 + L^2 \frac{(\pi a)^2}{2\ln(W/b)} \left[1 - (\frac{a}{b})^2 \right]}$
10	$X = 1.25 \left[\frac{5.9R}{\exp(a \cdot \sqrt{b + 1.41})/87} \right] - R \cdot \log(\frac{a}{b})$
11	$X = \frac{\sqrt{\pi \left[W(1 + (\cos 2b)/2 \right] - (3 \cdot \sin 2R)L}}{a \cdot R}$

12	$X = a(1-\frac{b^{-WL}}{\sqrt{1-a^2}}\cdot\sin(W\sqrt{1-a^2}+arctg(\frac{\sqrt{1-a^2}}{b})$
13	$X = a\sin\frac{\pi}{4}+L^2\sin 2\frac{\pi}{4}+W^L\sin b\frac{\pi}{4}$
14	$X = \frac{1}{4}\ln\frac{1+a}{1-b}+\frac{1}{2}arctgW$
15	$X = \frac{b\cos\frac{\pi}{3}}{1}+\frac{W^2\cos 2\frac{\pi}{3}}{2}+\frac{a^L\cos W\frac{\pi}{3}}{R}$
16	$X = 1.25\left[\frac{5.9R}{\exp(a\cdot\sqrt{b+1.41})/87}\right]$
17	$X = \sqrt{1-L^2\frac{a^2}{2\ln(W)}\left[1-(\frac{a}{b})^2\right]}$
18	$X = \frac{\sqrt{\pi\left[W(1+(\cos 2b)/2\right]}}{a\cdot R-(3\cdot\sin 2R)L}$
19	$X = \frac{\sqrt{\pi\left[W(1+(\cos 2b)/2\right]}}{a\cdot R}$
20	$X = \frac{1}{4}\ln\frac{1+a}{1-b}+\frac{1}{2}arctga$
21	$X = \frac{W^2\cos 2\frac{\pi}{3}}{2}+\frac{a^L\cos W\frac{\pi}{3}}{R}$
22	$X = \frac{1}{2}-\frac{\cos(\pi\pi^2/2)}{\pi R}-\frac{\cos(\pi\pi^2/2)}{\pi^2W^3}\cdot(1-\frac{5}{(\pi a^2)^2})$
23	$Z = tg(\frac{1}{2}-\frac{2a}{1-a^2-b^2})+\ln(\frac{a^2+(b+1)^2}{a^2+(b-1)^2})$
24	$X = \cos(\frac{\sin 2a}{\sin 2b}+\frac{2b}{\cos 2b})$
25	$X = \sin(\frac{a}{\cos 2a+\cos 2b}+b\frac{\sin 2b}{2a+\cos 2b})$
26	$X = (1-\frac{3}{(\pi b^2)^2})-\frac{\cos(\pi\pi^2/2)}{\pi^2L^3}\cdot(1-\frac{5}{(\pi a^2)^2})$
27	$X = \left[\frac{5.9R}{\exp(a\cdot\sqrt{b+1.41})/87}\right]-R\cdot\log(\frac{a}{b})$
28	$X = \frac{\sqrt{\pi}}{a\cdot RL}+\sin(2a)$
29	$X = \sin(W\sqrt{1-a^2}+arctg(\frac{\sqrt{1-a^2}}{b})$
30	$X = L^2\sin 2\frac{\pi}{4}+W\sin b\frac{\pi}{4}$

Практична робота №2
Оператори керування обчислювальним процесом в середовищі обробки біомедичної інформації MATLAB

Мета роботи: вивчити процедуру обчислювального процесу в процесі обробки біосигналів в середовищі MATLAB із застуванням операторів безумовного переходу, умовних переходів і операторів організації циклічних процесів.

ТЕОРЕТИЧНІ ВІДОМОСТІ

2.1 Загальні відомості

Загалом кажучи, оператори керування необхідні, головним чином, для організації обчислювального процесу в процесі обробки біосигналів, який записується у виді деякого тексту програми на мові програмування високого рівня. При цьому до операторів керування обчислювальним процесом звичайно відносять оператори безумовного переходу, умовних переходів (розгалуження обчислювального процесу) і оператори організації циклічних процесів. Проте система MatLAB побудована таким чином, що ці оператори можуть бути використані і при роботі MatLAB у режимі калькулятора.

Усі оператори циклу й умовного переходу побудовані в MatLAB у виді складного оператора, що починається з одного зі службових слів **if, while, switch** або **for** і закінчується службовим словом **end.** Оператори всередині між цими словами сприймаються системою як частини одного складного оператора. Тому натискання клавіші <Enter> для переходу до наступного рядка не призводить у цьому випадку до виконання цих операторів. Виконання операторів починається лише тоді, коли введена "завершувальна дужка" складного оператора у виді мітки **end** , а потім натиснуто клавішу <Enter>. Якщо декілька складних операторів такого типу вкладені один в інший, обчислення починаються лише тоді, коли записаний кінець **end** найбільш охоплюючого (зовнішнього) складного оператора. З цього випливає можливість здійснення навіть у режимі калькулятора досить складних і об'ємних (що складаються з багатьох рядків і операторів) обчислень, якщо вони охоплені складним оператором.

2.2 Оператор умовного переходу

Конструкція оператора переходу за умовою в загальному виді є такою:

```
if <умова>
        <оператори1>
        else
        <оператори2>
end
```

Працює оператор у такий спосіб. Спочатку перевіряється, чи виконується зазначена умова. Якщо її виконано, програма виконує сукупність операторів, що записана в поділі <оператори1>. Якщо умову не виконано, виконується послідовність операторів поділу <оператори2>.

Скорочена форма умовного оператора має вид:

if <умова>
<оператори>
end

Дія оператора в цьому випадку аналогічно, за винятком того, що при невиконанні заданої умови виконується оператор, наступний за оператором **end.**

Легко помітити хиби цього оператора, що випливають із відсутності оператора безумовного переходу: усі частини програми, що виконуються в залежності від умови, повинні розміщатися усередині операторних дужок **if** і **end.**

Як умова використовується вирази типу:

<ім'я змінної 1> <операція порівнювання> <ім'я змінної2>

Операції порівнювання в мові MatLAB можуть бути такими:

< - менше;
> - більше;
<= - менше або дорівнює;
>= - більше або дорівнює;
= = - дорівнює;
~ = - не дорівнює.

Умова може бути складеною, тобто складатися з кількох простих умов, що об'єднуються знаками логічних операцій. Знаками логічних операцій у мові MatLAB є:

& - логічна операція "І" ("AND");
| - логічна операція "АБО" ("OR");
~ - логічна операція "НІ" ("NOT").

Логічна операція "Виняткове АБО" може бути реалізована за допомогою функції **хог**(А,У), де А і В - деякі умови.

Є припустимою ще одна конструкція оператора умовного переходу:

if <умова1>
<оператори1>
elseif
<умова2>
<оператори2>

```
elseif
   <умова3>
   <оператори3>
else
   <оператори>
end
```

Оператор **elseif** виконується тоді, коли <умова1> не виконана. При цьому спочатку перевіряється <умова2>. Якщо її виконано, виконуються <оператори2>, якщо ж ні, <оператори2> ігноруються, і відбувається перехід до наступного оператора **elseif,** тобто до перевірки виконання <умови3>. Аналогічно, при виконанні її виконуються <операторьі3>, у противному випадку відбувається перехід до наступного оператора **elseif.** Якщо жодну з умов в операторах **elseif** не виконано, виконуються <оператори>, що містяться за оператором **else.** У такий спосіб може бути забезпечене розгалуження програми по кількох напрямках.

2.3 Оператор переключення

Оператор переключення має таку структуру:

```
switch <вираз, скаляр або рядок символів>
   case <значення1>
      <оператори1>
   case <значення2>
      <оператори2>
   otherwise
      <оператори>
end
```

Він здійснює розгалужування обчислень у залежності від значень деякої змінної або виразу, порівнюючи значення, отримане в результаті обчислення виразу в рядку **switch,** із значеннями, зазначеними в рядках із словом **case.** Відповідна група операторів **case** виконується, якщо значення виразу збігається зі значенням, зазначеним у відповідному рядку **case.** Якщо значення виразу не збігається з жодним із значень у групах **case,** виконуються оператори, що наслідують **otherwise.**

2.4 Оператори циклу

У мові MatLAB є два різновиди операторів циклу - умовний і арифметичний.

Оператор циклу з передумовою має вид:

```
while <умова> <оператори>
end
```

Оператори усередині циклу виконуються лише в тому випадку, якщо є виконаною умова, записана після слова **while.** При цьому серед операторів усередині циклу обов'язково повинні бути такі, що змінюють значення однієї зі змінних, зазначених в умові циклу.

Наведемо приклад обчислення значення синуса при 3 значеннях аргументу від 5 до 15 із кроком 5:

```
i = 1;
while i <= 3
   x = i*5;
   si =sin(x);
   disp([x,si])
   i=i+1;
end
```

Результат:

```
5.0000   -0.9589
10.0000   -0.5440
15.0000    0.6503
```

Примітка. Зверніть увагу на те, якими засобами в зазначеному прикладі забезпечено виведення на екран значень кількох змінних одним рядком.

Для цього використовується оператор **disp,** який раніше застосовувався. Але, відповідно до правил застосування цього оператора, у ньому повинний бути тільки один аргумент (текст, змінна або матриця). Щоб обминути цю перешкоду, потрібно декілька числових змінних об'єднати в єдиний об'єкт - вектор-рядок, а останнє легко виконується за допомогою звичайної операції формування вектора-рядка з окремих елементів:

[xl, x2, ... , x].

Таким чином, за допомогою оператора виду:

disp([x1, x2, ... , x])

можна забезпечити виведення результатів обчислень у виді таблиці даних.

Арифметичний оператор циклу має вид:

```
for <ім'я> = <ПЗ> : <К> : <КЗ>
        <оператори>
end,
```

де <ім'я> - ім'я керуючої змінної циклу - "лічильника" циклу; <ПЗ> - задане початкове значення цієї змінної; <К> - значення кроку, із яким вона повинна

змінюватися; <КЗ> - кінцеве значення змінної циклу. У цьому випадку <оператори> усередині циклу виконуються кілька разів (кожного разу при новому значенні керуючої змінної) доти, поки значення керуючої змінної не вийде за межі інтервалу між <ПЗ> і <КЗ>. Якщо параметр <К> не зазначений, за замовчуванням його значення приймається рівним одиниці.

Щоб достроково вийти з циклу (наприклад, при виконанні деякої умови) застосовують оператор **break.** Якщо програма стикається з цим оператором, виконання циклу достроково припиняється, і починає виконуватися оператор, наступний за словом **end** циклу.

Для приклада використаємо попереднє завдання: »

```
a = ['     i','      x','      sin(x)'];
for i = 1:5
   x = i/5;
   si = sin(x);
   if i==1 disp(a)
   end
   disp([i,x,si])
end;
```

Результат:

i	x	sin(x)
1.0000	0.2000	0.1987
2.0000	0.4000	0.3894
3.0000	0.6000	0.5646
4.0000	0.8000	0.7174
5.0000	1.0000	0.84151

У такий спосіб можна забезпечити виведення інформації у вигляді таблиць.

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

2.5 Завдання для виконання

Відповідно до таблиці 1.1 обчислити точні (по стандартних функціях MatLAB) значення відповідної функції у діапазоні змінювання аргументу від x_1 до x_2 в m рівновіддалених точках цього діапазону, включаючи його межі.

Таблиця 1.1 – Варіанти завдань

Варіант	x1	x2	m	Функція	Критерій прийняття рішення
1	0.2	5	20	$Y = \sum (-1)^2 \frac{x^{2k-1}}{2k-1}$	$Результат = \begin{cases} Норма, & якщо \quad Y < 5 \\ Патологія, & якщо \quad Y > 5 \end{cases}$

2	1	10	30	$Y = 1 - \sum (-1)^k \frac{x^{2k}}{2k}$	$Результат = \begin{cases} Норма, & якщо \quad Y \le 3 \\ Патологія, & якщо \quad Y \ge 3 \end{cases}$
3	0.3	3	40	$Y = 1 + \sum (-1)^k \frac{x^{2k}}{2k}$	$Результат = \begin{cases} Норма, & якщо \quad 1 < Y \le 3 \\ Патологія, & якщо \quad Y > 3 \end{cases}$
4	0.4	4	50	$Y = \sum (-1)^2 \frac{x^{2k+1}}{2k}$	$Результат = \begin{cases} Норма, & якщо \quad 3 > Y > 5 \\ Патологія, & якщо \quad 4 > Y > 3 \end{cases}$
5	0.5	5	30	$Y = \sum (-1)^3 \frac{x^{2k-2}}{2k-2}$	$Результат = \begin{cases} Норма, & якщо \quad 0 < Y < 1 \\ Патологія, & якщо \quad Y < 2 \end{cases}$

За обчисленими значення функції Y прийняти рішення (результат), а саме норма чи патологія в залежності від умови таблиці 1.1

2.6 Зміст звіту

2.6.1 Тема та мета роботи.
2.6.2 Завдання до роботи.
2.6.3 Відобразити отримані результати (програму та результати її роботи)
2.6.4 Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які засоби керування перебігом обчислювального процесу передбачені в мові MatLAB?
2. Як можна організувати обчислення за циклом мовою MatLAB?
3. Як організувати виведення таблиці результатів обчислень у командне вікно MatLAB?
4. Як здійснити складні (багатооператорні) обчислення в режимі калькулятора?

Практична робота №3
Операції з векторами та матрицями

Мета роботи: вивчення способів обробки двомірних масивів в середовищі MATLAB.

ТЕОРЕТИЧНІ ВІДОМОСТІ

3.1 Загальні відомості

Кращий спосіб розпочати роботу з MATLAB – це навчитися поводитися з матрицями. В MATLAB матриця – це прямокутний масив чисел. Особливе значення надається матрицям 1x1, які є скалярами, і матрицям, що мають один стовпець або один рядок – векторам. MATLAB використовує різні способи для зберігання чисельних і не чисельних даних, проте спочатку краще всього розглядати усі дані як матриці. MATLAB влаштований так, щоб усі операції в ньому були як можна природнішими. Тоді як інші програмні мови працюють з числами як елементами мови, MATLAB дозволяє швидко і легко оперувати з цілими матрицями.

MATLAB – система, спеціально призначена для проведення складних обчислень з векторами, матрицями і масивами. При цьому вона за умовчанням припускає, що кожна задана змінна – це вектор, матриця або масив. Усе визначається конкретним значенням змінної.

Введення векторів здійснюється у виді:

V=[x1 x2 x3]

де V – ім'я вектору, x1 x2 x3 – значення елементів вектору, які розміщені в квадратних дужках і відокремлені один від одного пропусками або комами. Наприклад, запис рядка

V =[1.2 -0.3 1.2e-5]

Довгий вектор можна вводити частинами, які потім можна об’єднати за допомогою операції об’єднання векторів в рядок:

V =[v1 v2]

Наприклад,

v1 =[1 2 3]; v2 =[4 5 6]; V =[v1 v2]

Так вводяться вектори-рядки. Вектор-стовпець вводиться аналогічно, але значення елементів відділяються знаком ";".

Для формування впорядкованих числових послідовностей використовується оператор ":"

Якщо позначити: nz – початкове значення цієї прогресії (значення першого елементу вектору); kz – кінцеве значення прогресії (значення останнього елементу вектору); h – різниця прогресії (крок), то вектор можна ввести за допомогою короткого запису

V = nz:h:kz

Наприклад,

V = - 0.1:0.3 :1.4

Якщо крок прогресії не вказаний, то він за умовчанням приймається рівним одиниці.

Наприклад, команда >>-2.1:5 призводить до формування такого вектору

ans = - 2.100 -1.100 -0.100 0.900 1.900 2.900 3.900 4.900

Завдання матриці вимагає вказівки декількох рядків. Для розмежування рядків використовується знак «;» (крапка з комою). Цей же знак у кінці введення запобігає виведенню матриці або вектору (і взагалі результату будь-якої операції) на екран дисплея. Так, введення

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

задає квадратну матрицю, яку можна вивести:

```
>> M
M =   1 2 3
      4 5 6
      7 8 9
```

Можливе також введення елементів матриць і векторів у вигляді арифметичних виразів, що містять будь-які доступні системні функції

Приклад:

```
>> V=[2+2/(3+4) exp(5) sqrt(10)]
V = 2.2857 148.4132 3.1623
```

Деякі функцій MATLAB, що дозволяють формувати вектору і матриці певного виду:

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

ones(M, N) – створює матрицю розміром з одиничними елементами;

eye(M, N) – створює одиничну матрицю розміром (M*N), т. е. з одиницями по головній діагоналі і іншими нульовими елементами;

rand(M, N) – створює матрицю розміром (M*N) з випадкових чисел, рівномірно розподілених в діапазоні від 0 до 1;

randn(M, N) – створює матрицю розміром (M*N) з випадкових чисел розподілених за нормальним (гаусу) законом з нульовим математичним очікуванням і стандартним (середньоквадратичним) відхиленням, рівним одиниці; magic(N) – створює матрицю розміром (N*N) у якої сума усіх рядків, стовпців і діагоналей дорівнює одному і тому ж числу;

tril(A) – утворює нижню трикутну матрицю на основі матриці А шляхом обнуління її елементів вище за головну діагональ;

triu(A) – утворює верхню трикутну матрицю на основі матриці А шляхом обнуління її елементів нижче головної діагоналі;

diag(A) – формує або виймає діагональ матриці А.

Вставка та виймання частин матриць. Звернення до будь-якого елементу заданої матриці здійснюється шляхом вказівки (у дужках, через кому) після імені матриці двох цілих позитивних чисел, які визначають відповідно номери рядка і стовпця матриці, на перетині яких розташований цей елемент. Нехай маємо деяку матрицю А:

```
>> 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 отримати значення елементу цієї матриці, розташованого на перетині другого рядка з третім стовпцем, можна таким чином:

```
>> 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;
```

Якщо треба витягнути з матриці рядок або стовпець, то необхідно застосовувати оператор двокрапка " : ".

Приклад: нехай треба створити вектор V1, що складається з елементів третього стовпця матриці А. Для цього виконаємо такі дії:

```
>> V1 = A(:, 3)
V1 =
3.0000
7.0000
11.0000
```

Допустимо, що необхідно з матриці A утворити матрицю розміром (2× 2), яка складається з елементів лівого нижнього кута матриці A. Тоді роблять так:

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

Об’єднання малих матриць у велику. Описаний спосіб завдання матриць дозволяє виконати операцію конкатенації – об'єднання малих матриць у велику. Існує горизонтальна і вертикальна конкатенація.

Горизонтальна конкатенація – це об'єднання декількох матриць-блоків A1, A2,... AN з однаковою кількістю рядків.

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

Вертикальна конкатенація – це об'єднання декількох матриць-блоків за умови, що усі складені блоки-матриці мають однакову кількість стовпців. При цьому, для відокремлення блоків замість коми використовується ";".

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

Наприклад, створимо спочатку магічну матрицю розміру 3x3:

```
>> A=magic(3)
A= 8 1 6 3 5 7 4 9 2
```

Тепер можна побудувати матрицю, що містить чотири матриці:

```
>> B=[A A+16; A+32 A+16]
B =
8 1 6 24 17 22
3 5 7 19 21 23
4 9 2 20 25 18
40 33 38 24 17 22
35 37 39 19 21 23
36 41 34 20 25 18
```

Отримана матриця має вже розмір 6x6. Обчислимо суму її стовпців:

```
>>sum(B)
ans =
126 126 126 126 126 126
```

А для обчислення суми рядків використовуємо команду

```
>>sum(B')
ans = 78 78 78 174 174 174
```

Запис B' означає транспонування матриці B, тобто заміну рядків стовпцями.

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

3.2Завдання для виконання

3.2.1 Обчислити значення функції f x( ) на проміжку [a b] ; з кроком h. В табл.2.1. наведені варіанти завдань.

№ варіанту	Функція $f(x)$	Параметри		
		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	$\sqrt{1+4x}\cdot \sin \pi x$	0,1	0,8	0,05
6	$\frac{e^{x/3}}{1+x^2}$	1,4	2,4	0,1
7	$e^{-2x}+x^2-1$	0,25	2,25	0,25
8	$(e+x)\cdot \sin\left(\pi\sqrt{x-1}\right)$	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	$(4+7x)\cdot \sin\left(\pi\sqrt[3]{1+x}\right)$	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	$\sqrt{\frac{e^{\sqrt{2\pi x}} - e^{-\sqrt{2\pi x}}}{2}}$	0	1,2	0,12
16	$\sqrt{\frac{e^{x/\sqrt{2\pi}} - e^{-x/\sqrt{2\pi}}}{2}}$	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	$\arccos\left(e^{-\sqrt[3]{3x}}\right)$	0,2	0,5	0,03

3.2.2 Виконати операції з матрицями згідно власного варіанту завдання.

Варіанти 1, 7, 13, 19. Видалити з матриці А третій стовпець, а з матриці В другий рядок. В матриці А змінити значення елемента з координатами (2, 1) на 6. Розрахувати суму рядків, стовпців та діагоналі матриці В.

$$A = \begin{bmatrix} 7 & 2 & 5 \\ 10 & 4 & 6 \\ 3 & 5 & 9 \end{bmatrix}, \quad B = \begin{bmatrix} 5 & 9 & 0 \\ 3 & 12 & 4 \\ 1 & 4 & 7 \end{bmatrix}.$$

Варіанти 2, 8, 14. Задати «магічну» матрицю А розміром 4×4. Створити матрицю В кожний елемент якої є синусом відповідного елементу «магічної» матриці А. Розрахувати суму рядків, стовпців та діагоналі отриманої матриці В. Варіанти 3, 9, 15. Видалити перший рядок і третій стовпець «магічної» матриці М розміром 4×4 . Отриману матрицю скласти з матрицею А в якій заздалегідь необхідно обернути рядки в стовпці, а стовпці в рядки.

$$A = \begin{bmatrix} 2 & 0 & 1 \\ 4 & 3 & 9 \\ 21 & 3 & 2 \end{bmatrix}.$$

Варіанти 4, 10, 16, 20. Об’єднати в одну матрицю «магічну» матрицю А розміром 3×3 , матрицю В = А · 5, матрицю С і матрицю D=C+ 4 таким чином, щоб отримана матриця мала розмірність 6 × 6. В одержаній матриці замінити елемент з координатами (3, 4) на 100.

$$C = \begin{bmatrix} 5 & 2 & 7 \\ 7 & 4 & 12 \\ 1 & 3 & 9 \end{bmatrix}.$$

Варіанти 5, 11, 17. Обчислити результат добутку елементу матриці M з координатами (2, 3) на синус значень першого стовпця цієї ж матриці

$$M = \begin{bmatrix} 1 & 2 & 3 & 9 \\ 5 & 4 & 7 & 3 \\ 7 & 9 & 0 & 5 \end{bmatrix}.$$

Варіанти 6, 12, 18. Скласти матрицю X, кожний елемент якої є комплексним числом, причому дійсна складова береться з матриці A, а уявна з матриці B. Знайти модуль і фазу елемента матриці X з координатами (3, 3).

$$A = \begin{bmatrix} 1 & 2 & 7 \\ 10 & 4 & 5 \\ 3 & 8 & 9 \end{bmatrix}, \quad B = \begin{bmatrix} 12 & 7 & 4 \\ 2 & 5 & 9 \\ 1 & 4 & 7 \end{bmatrix}.$$

3.2.3 Виконати операції з матрицями згідно власного варіанту завдання.

Варіанти 1, 7, 13, 19. Створити матрицю A з нульовими елементами розміром 5 5 × та одиничну матрицю B розміром 2×3. Замінити верхню центральну частину матриці A елементами матриці B.

Варіанти 2, 8, 14, 20. Створити матрицю A з випадковими елементами розміром 4×5. Переставити стовпці відносно вертикальної осі, а рядки переставити відносно горизонтальної осі.

Варіанти 3, 9, 15. Створити матрицю A розміром 3×6 з випадковими, що розподілені по закону Гауса. Повернути цю матрицю проти годинникової стрілки на 90°.

Варіанти 4, 10, 16. Створити матрицю A розміром 2×4 з одиницями по головній діагоналі та всіма іншими елементами – нулями. Змінити розмір матриці на 8×1 .

Варіанти 5, 11, 17. На основі матриці M утворити нижню трикутну матрицю і повернути її проти годинникової стрілки на 90°.

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 5 & 4 & 7 \\ 7 & 9 & 0 \\ 0 & 4 & 5 \end{bmatrix}.$$

Варіанти 6, 12, 18. Створити матрицю A в якій головною діагоналлю є вектор V = [2 3 6 -4] . Обернути отриману матрицю.

3.3. Зміст звіту

3.3.1. Тема та мета роботи.

3.3.2. Завдання до роботи.

3.3.3. Відобразити отримані результати (п. 3.2.1-3.2.3) у вигляді копій екрану.

3.3.4. Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Як задається процедура введення векторів в середовищі MATLAB?
2. Як забезпечується процедура об’єднання векторів?
3. Як вводить вектор-стовпець?
4. Як забезпечується формування впорядкованих числових послідовностей?
5. Перерахуйте перелік функцій MATLAB, які забезпечують формування вектору і матриці певного виду.
6. Як організувати процедуру вставка та виймання частин матриць?
7. Як організувати процедуру об’єднання малих матриць у велику?
8. Як здійснюється процедура заміни рядків стовпцями?

Практична робота №4
Створення напройстіших файл-функцій (процедур) в середовищі MATLAB

Мета роботи: вивчити процедуру **створення напройстіших файл-функцій (процедур) в середовищі MATLAB**

ТЕОРЕТИЧНІ ВІДОМОСТІ

4.1 Загальні відомості

Як було зазначено раніше, файл-функція (процедура) має починатися з рядка заголовка виду:

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

Якщо перелік кінцевих (вихідних) величин (ПКВ) містить тільки один об'єкт (у загальному випадку - матрицю), то файл-функція є звичайною функцію (однієї або кількох змінних). Фактично навіть у цьому найпростішому випадку файл-функція є вже процедурою у звичайному розумінні інших мов програмування, якщо вихідна величина є вектором або матрицею. Перший рядок у цьому випадку має вид:

function <ім'я змінної> = <ім'я процедури>(<ПВВ>).

Якщо ж в результаті виконання файл-функції мають бути визначені (обчислені) кілька об'єктів (типу матриць), така файл-функція є вже більш складним об'єктом, який у програмуванні зазвичай називається або процедурою (у мові Паскаль), або підпрограмою. Загальний вид першого рядка в цьому випадку стає таким:

function [y1, y2, ... , yN] = <ім'я процедури>(<ПВВ>),

тобто перелік вихідних величин y1, y2, ... , yN має бути поданий як вектор-рядок з елементами y1, y2, ... , yN (кожний з яких може бути матрицею).

У найпростішому випадку функції однієї змінної заголовок набуває вид:

function y = **func(x)**, де **func -** ім'я функції (М-файлу).

Як приклад розглянемо складання М-файлу для функції:

$$y = f_1(x) = d^3 \cdot ctg(x)\sqrt{\sin^4(x) - \cos^4(x)}.$$

Для цього треба активізувати поділ "File" головного меню командного вікна MatLAB, вибрати в підменю, що виникло на екрані, розділ "New", а потім - команду "M-file". На екрані з'явиться вікно текстового редактора. У ньому потрібно набрати такий текст:

```
function y = F1(x,d)
% Процедура, яка обчислює значення функції
% y = (d^3)*ctg(x)*sqrt(sin(x)M-cos(x)^4).
% Звернення y = F1(x,d).
y = (d^3)*cot(x).*sqrt(sin(x).^4-cos(x).^4);
```

Після цього необхідно зберегти цей текст у файлі під ім'ям "F1.m". Необхідний М-файл створений. Тепер можна користуватися цією функцією при розрахунках. Так, якщо ввести команду

```
» y = F1(1,0.1)
```

то одержимо результат

```
y = 4.1421e-004.
```

Варто зауважити, що аналогічно можна одержати одразу вектор усіх значень зазначеної функції при різних значеннях аргументу, якщо останні зібрати в деякий вектор. Так, якщо сформувати вектор

```
» zet= 0:0. 3:1. 8;
```

і звернутися до тієї ж процедури

```
» my = F1(zet,1),
```

то одержимо:

```
Warning: Divide by
zero my =
 Columns 1 through 4
   Na + Infi 0 + 2. 9369i      0 + 0. 8799i 0.
 3783 Columns 5 through 7
  0. 3339       0. 0706      -0. 2209
```

Примітки.

1. Можливість використання сформованої процедури як для окремих чисел, так і для векторів і матриць обумовлена застосуванням у запису відповідного М-файлу замість звичайних знаків арифметичних дій їхніх аналогів із попередньою крапкою.

2. Щоб уникнути виведення на екран небажаних проміжних результатів, необхідно в тексті процедури усі обчислювальні оператори завершувати символом " ; ".

3. Як показують наведені приклади, імена змінних, зазначені в заголовку файл-функції можуть бути будь-якими (збігатися чи ні з іменами, використовуваними при зверненні до цієї файл-функції), тобто мають формальний характер. Важливіше за все лише те, щоб структура звернення цілком відповідала структурі заголовка в запису тексту М-файлу і щоб змінні в цьому зверненні мали той тип і розмір, що й в заголовку М-файлу.

Щоб одержати інформацію про створену процедуру, достатньо набрати в командному вікні команду:

» help f1,

і в командному вікні з'явиться

Процедура, яка обчислює значення функції
y =(d^3)*ctg(x)*sqrt(sin(x^4-cos(x^4). Звернення y = F1(x,d).

Інший приклад. Побудуємо графік двох функцій:

yl = 200 sin(x)/x; y2 = x2.

Для цього створимо М-файл, що обчислює значення цих функцій:

```
function y = myfun(x)
% Обчислення двох функцій
% y(1) = 200 sin(x)/x, y(2) = x2.
y(:,1)=200*sin(x)./x; y(:,2) = x.x2;
```

Тепер побудуємо графіки цих функцій:

```
» fplot('myfun', [-20 20], 50, 2), grid
» set(gcf,'color','white');
» title('Графік функції "MYFUN"')
```

Результат зображено на рис. 2.2.

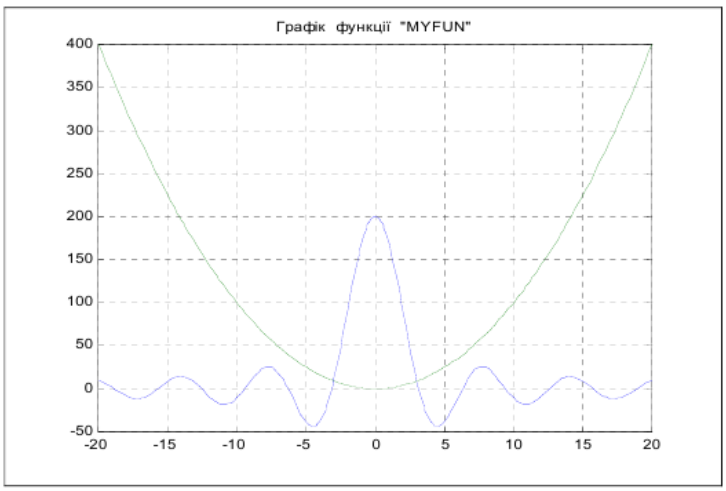

Рис. 4.2. Побудова графіків власних функцій

Третій приклад - створення файл-функції, що обчислює значення функції:

$$y(t) = k1+k2*t+k3*\sin(k4*t+k5).$$

У цьому випадку зручно об'єднати сукупність коефіцієнтів **k** у єдиний вектор К:

К = [kl k2 k3 k4 k5]

і створити М-файл: **function y = dvob(x, K)**

```
% Обчислення функції
% y = K(1)+K(2)*x+K(3)*sin(K(4)*x+K(5)),
% де К - вектор із п'яти елементів
%
% Використовується для визначення поточних значень
% параметрів руху рухомого об'єкта
y = K(1)+K(2)*x+K(3)*sin(K(4)*x+K(5));
```

Тоді розрахунок, наприклад, 11-ти значень цієї функції можна здійснити так

```
» K = ones(1,5); » t = 0:1:10;
» fi = dvob(t, K)
```

fi = 1.8415 2.9093 3.1411 3.2432 4.0411 5.7206 7.6570 8.9894 9.4560 10.0000

3.2 Типове оформлення процедури-функції

Рекомендується оформляти М-файл процедури-функції за такою схемою:

function [<Вихід>] = <ім'я функції>(<Вхід>)

% «Стисле пояснення, що робить процедура>
% Вхідні змінні
% <Детальне пояснення про зміст, типі і розміри
% кожній із змінних, перерахованих у переліку <Вхід>
% Вихідні змінні
% «Детальне пояснення про зміст, типі і розміри
% кожної із змінних переліку «Вихід>
% і величини, що використані у процедурі як глобальні>
% Використання інших функцій і процедур
% <Розділ заповнюється, якщо процедура містить звернення
% до інших процедур, крім умонтованих>
< П о р о ж н і й р я д о к >
% Автор : <Вказується автор процедури, дата створення кінцевого варіанта % процедури й організація, у якій створена програма> < Т е к с т п р о ц е д у р и >

Тут позначено: <Вихід> - перелік вихідних змінних процедури, <Вхід> - перелік вхідних змінних, розділених комами.

Примітка. При використанні команди **help** <ім'я процедури> у командне вікно виводяться рядки коментарю до першого порожнього рядка.

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

4.3 Завдання для виконання

Відповідно до таблиці 1.1 виконати обчислення точних (по стандартних функціях MatLAB) значень відповідної функції у діапазоні змінювання аргументу від x_1 до x_2 в m рівновіддалених точках цього діапазону, включаючи його межі із використанням розробленої **файл-функції (процедури).**

4.4Зміст звіту

4.4.1 Тема та мета роботи.
4.4.2 Коротко основні теоретичні відомості.
4.4.3 Відобразити отримані результати (програму та результат її роботи)
4.4.4 Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Для чого створюються програми в середовищі MatLAB?
2. Чим відрізняються файли-функції від Script-файлів? Яка сфера застосування кожного з цих видів файлів?
3. Як створити М-файл процедури або функції?
4. Які основні правила написання текстів М-файлів у мові MatLAB?

Практична робота №5
Функції функцій

Мета роботи: вивчити процедуру **розроблення функцію функцій в середовищі MATLAB**

ТЕОРЕТИЧНІ ВІДОМОСТІ

5.1 Загальні відомості

Деякі важливі універсальні процедури в MatLAB використовують як змінний параметр ім'я функції, із яким вони оперують, і тому потребують при зверненні до них указівки імені М-файлу, у якому записаний текст програми обчислення деякої іншої процедури (функції). Такі процедури називають **функціями функцій.**

Щоб скористатися такою функцією від функції, необхідно, щоб користувач попередньо створив М-файл, у якому обчислювалося б значення потрібної функції за відомим значенням її аргументу.

5.2 Стандартні функції від функцій Matlab

Перелікуємо деякі зі стандартних функцій від функцій, передбачених у MatLAB.

Обчислення інтеграла методом квадратур здійснюється процедурою

[I, cnt] = quad('<ім'я функції>',a,b).

Тут a і b - нижня й верхня межа змінювання аргументу функції; I - отримане значення інтеграла; cnt - кількість звернень до обчислення функції, поданої М-файлом із назвою, указаним у <ім'я функції>. Функція **quad** використовує квадратурні формули Ньютона-Котеса четвертого порядку.

Аналогічна процедура **quad8** використовує більш точні формули 8-го порядку.

Інтегрування звичайних диференціальних рівнянь здійснюють функції **ode23** і **ode45.** Вони можуть застосовуватися як для розв'язування простих диференціальних рівнянь, так і для моделювання складних динамічних систем, тобто систем, поводження яких можна описати сукупністю звичайних диференціальних рівнянь.

Відомо, що будь-яка система звичайних диференціальних рівнянь (ЗДР) може бути подана як система рівнянь 1-го порядку у формі Коші:

$$\frac{dy}{dt} = f(y,t),$$

де **y** - вектор змінних стану (фазових змінних системи);
t - аргумент (зазвичай -час);
f - нелінійна вектор-функція від змінних стану **y** і аргументу t.
Звернення до процедур чисельного інтегрування ЗДР має вид:

[t,y]=ode23('<ім'я функції>',tspan,y0,options)
[t,y]=ode45('<ім'я функції>',tspan,y0,options),

де використані параметри мають такий зміст: <ім'я функції> - рядок символів, що є ім'ям М-файла, у якому обчислюється вектор-функція fy,t), тобто праві частини системи ЗДР; y0 - вектор початкових значень змінних стану; t - масив кінцевих значень аргументу, що відповідають крокам інтегрування; y - матриця проінтегрованих значень фазових змінних, в якій кожний стовпчик відповідає одній зі змінних стану, а рядок містить значення змінних стану, що відповідають певному кроку інтегрування; tspan - вектор-рядок [t0 tfinal], що містить два значення: t0 - початкове значення аргументу t; tfinal - кінцеве значення аргументу; ortions - рядок із параметрів, що визначають значення припустимої відносної й абсолютної похибки інтегрування.

Параметр ortions можна не вказувати. Тоді за замовчуванням припустима відносна похибка інтегрування приймається рівною 1 • 10 , абсолютна (по кожній із змінних стану) - 1 • 10_^{6}. Якщо ж по якихось параметрах ці значення не влаштовують користувача, треба перед зверненням до процедури чисельного інтегрування встановити нові значення припустимих похибок за допомогою процедури **odeset** у такий спосіб:

options=odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-5]).

Параметр RelTol визначає відносну похибку чисельного інтегрування по усіх фазових змінних одночасно, а AbsTol є вектором-рядком, що складається з абсолютних припустимих похибок чисельного інтегрування по кожній з фазових змінних.

Функція **ode23** здійснює інтегрування чисельним методом Рунге-Кутта 2-го порядку, а за допомогою методу 3-го порядку контролює відносні й абсолютні похибки інтегрування на кожному кроці і змінює величину кроку інтегрування так, щоб забезпечити задані межі похибок інтегрування.

Для функції **ode45** основним методом інтегрування є метод Рунге-Кутта 4-го порядку, а величина кроку контролюється методом 5-го порядку.

Обчислення мінімумів і коренів функції здійснюється такими функціями
MatLAB:

fmin - відшукування мінімуму функції одного аргументу;
fmins - відшукування мінімуму функції кількох аргументів;

fzero - відшукування нулів (коренів) функції одного аргументу.
Звернення до першої з них у загальному випадку має такий вид:

Xmin = fmin ('<ім'я функції>', X1, X2).

Результатом цього звернення буде значення Xmin аргументу функції, яке відповідає локальному мінімуму в інтервалі X1<X<X2 функції, заданої М-файлом із зазначеним ім'ям.

Як приклад розглянемо відшукання значення числа π як значення локального мінімуму функції **y = cos(x)** на відрізку [3, 4]:

```
» Xmin = fmin('cos',3,4)
Xmin = 3. 1416e+000
```

Звернення до другої процедури повинно мати форму:

Xmin = fmins ('<ім'я функції>', X0),

при цьому X є вектором аргументів, а X0 означає початкове (первинне) значення цього вектора, в околі якого відшукується найближчий локальний мінімум функції, заданої М-файлом із зазначеним ім'ям. Функція **fmins** знаходить вектор аргументів Хгпіп, який відповідає знайденому локальному мінімуму.

Звернення до функції **fzero** повинно мати вид:

z = fzero ('<ім'я функції>', x0, tol, trace).

Тут позначено: x0 - початкове значення аргументу, в околі якого відшукується дійсний корінь функції, значення якої обчислюються в М-файлі із заданим ім'ям; tol - задана відносна похибка обчислення кореня; trace - позначення необхідності виводити на екран проміжні результати; z - значення шуканого кореня.

Побудова графіків функції однієї змінної може бути здійснена за допомогою процедури **fplot.** Відмінність її від процедури **plot** у тому, що для побудови графіка функції немає необхідності в попередньому обчисленні значення функції й аргументу. Звернення до неї має вид:

fplot('<ім'я функції>',[<інтервал>],n),

де **<інтервал>** - це вектор-рядок із двох чисел, що задають, відповідно, нижню й верхню межі змінювання аргументу; <ім'я функції> - ім'я М-файла з текстом процедури обчислення значення бажаної функції за заданим значенням її аргументу; n - бажане число частин розбиття зазначеного інтервалу. Якщо останню величину не задати, за замовчуванням інтервал розбивається на 25 частин. Хоча кількість частин "n", на які розбито інтервал змінювання аргументу, визначено, проте насправді кількість значень вектора "x" може бути

значно більшою за рахунок того, що функція **fplot** проводить обчислення з додатковим обмеженням, щоб збільшення кута нахилу графіка функції на кожному кроці не перевищувало 10 градусів. Якщо ж воно виявилося більшим, здійснюється роздрібнення кроку змінювання аргументу, але не більше ніж у 20 разів. Останні два числа (10 і 20) можуть бути змінені користувачем, для цього при зверненні слід додати ці нові значення в заголовок процедури в зазначеному порядку.

Якщо звернутися до цієї процедури так:

[x, Y] = fplot ('<ім'я функції>', [<інтервал>], n),

то графік зазначеної функції не відображується на екрані (у графічному вікні). Замість цього обчислюється вектор "x" аргументів і вектор (або матриця) Y відповідних значень зазначеної функції. Щоб при зверненні останнього виду побудувати графік, необхідно зробити це у подальшому за допомогою процедури **plot**(x, Y).

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

5.3 Завдання для виконання

5.3.1 Завдання 1

Створіть М-файл, що обчислює функцію із завдання 1.5. побудуйте графік цієї функції за допомогою процедури **fplot** у межах, визначених у завданні 1.5. Обчисліть інтеграл від функції у тих же межах, використовуючи процедури **quad** і **quad8.** Знайдіть точку локального мінімуму і локальний мінімум функції і найближчий корінь (нуль).

5.3.2 Завдання 2

Знайдіть точку локального мінімуму і локальний мінімум функції двох змінних, прийнявши за початкову точку із заданими координатами (таблиця 2.2). Попередньо створіть відповідну файл-функцію.

Таблиця 5.1 – Завдання до практичної роботи №5

Варіант	x_o	y_o	$f(x,y)$
1	0	1	$e^{x+y}+(x-y)^2-2x-2y$
2	0.7	-1. 2	$(x-y)^2-\cos(x-y-1)$
3	1.5	-0. 5	$e^{x+y}-2x-2y-\cos(x-y-1)$
4	0.5	1.5	$e^{x+y}+4x^2-3x-3y$
5	0	1	$4x^2+\ln(x+y)+\frac{1}{x+y}$
6	1.2	0.7	$2^{x+y}-2x-2y+2(x-y)^2$
7	0	-0. 9	$e^{x-y}+2x+2y+(x+y)^2$
8	0.8	1.3	$(x-y)^2-\cos(x+y-1)$
9	1.5	0.5	$e^{x-y}-2x+2y-\cos(x+y-1)$
10	0.5	-1. 5	$e^{x-y}-3x+3y+4x^2$
11	0	-1	$4x^2+\ln(x-y)+\frac{1}{x-y}$
12	1.2	-0. 8	$2^{x-y}-2x+2y+2(x+y)^2$

Практична робота №6
Графічне оформлення результатів обробки біомедичних сигналів у вигляді 2D-графіків

Мета роботи: вивчити процедуру графічного оформлення результатів обробки біомедичних сигналів в середовищі MATLAB

ТЕОРЕТИЧНІ ВІДОМОСТІ

6.1. Загальні вимоги до подання графічної інформації

Обчислювальна програма, утворювана інженером-розробником, призначена, по більшості, для дослідження поводження розроблювального устрою за різноманітних умов його експлуатації, при різних значеннях його конструктивних параметрів або для розрахунку певних параметрів його поводження. Інформація, одержувана в результаті виконання обчислювальної інженерної програми, як правило, має форму деякого ряду чисел, кожний з яких відповідає певному значенню деякого параметра (аргументу). Таку інформацію зручніше усього узагальнювати й подавати в графічній формі.

Вимоги до оформлення інженерної графічної інформації відрізняються від вимог до звичайних графіків у математиці. Зазвичай на основі поданої графічної інформації користувач-інженер має мати змогу прийняти якесь інженерне рішення про вибір значень певних конструктивних параметрів, що характеризують досліджуваний процес або технічний устрій, із метою, щоб прогнозоване поводження технічного устрою задовольняло певні задані умови. Тому інженерні графіки мають бути, як говорять, "читабельними", тобто мати такий вид, щоб з них легко було "відлічувати" значення функції при будь-яких значеннях аргументу і навпаки з відносною похибкою в декілька відсотків. Це стає можливим, якщо координатна сітка графіків відповідає певним цілим

числам якогось десяткового розряду. Як уже раніше відзначалося, графіки, побудовані системою MatLAB, цілком відповідають цим вимогам.

Крім цього, інженерна графічна інформація має супроводжуватися достатньо детальним описом, із якого має стати зрозуміло, який об'єкт і за якою математичною моделлю досліджується, наведені числові значення параметрів досліджуваного об'єкта і математичної моделі. Не є зайвим і вказівка імені програми, за допомогою якої отримана ця графічна інформація, а також зведень про автора програми й дослідника, щоб користувачеві було зрозуміло, до кого і куди треба звертатися для наведення довідок про отриману інформацію.

Задачею інженерної програми часто є порівняння декількох функцій, отриманих при різних сполученнях конструктивних параметрів або параметрів зовнішніх дій. Таке порівняння зручніше і наочніше проводити, якщо згадані функції подані у виді графіків.

При цьому потрібно звернути увагу на наступне:

1) якщо потрібно порівнювати графіки функцій одного аргументу, діапазони змінювання яких не надто відрізняються один від одного (не більш ніж на десятковий порядок, тобто не більш ніж у десять разів), порівняння зручніше усього здійснювати по графіках цих функцій, побудованих в одному графічному полі (тобто у загальних координатних осях); у цьому випадку варто виводити графіки за допомогою однієї функції **plot;**

2) якщо за тих самих умов діапазони змінювання функцій значно різняться, можна запропонувати два підходи:

- якщо всі порівнювані функції є значеннями величин однакової фізичної природи і ці значення усі додатні, графіки варто виводити також в одне графічне поле, але в логарифмічному масштабі (тобто використовувати процедуру **semilogy);**
- за умови, що усі функції мають різну фізичну природу, але аргумент у них загальний і змінюється в однім діапазоні, графіки потрібно будувати в одному графічному вікні (фігурі), але в різних графічних полях (користуючись для цього декількома окремими зверненнями до функції **plot** у різних підвікнах графічного вікна, що забезпечується застосуванням процедури **subplot);** при цьому зручно розміщувати окремі графіки один під одним таким чином, щоб однакові значення аргументу у всіх графіках розташовувалися на одній вертикалі;

3) крім згаданих раніше написів у кожному графічному полі, закінчене графічне оформлення будь-якого графічного вікна (фігури) обов'язково повинно містити додаткову текстову інформацію такого вмісту:

- стисле повідомлення про об'єкт дослідження;
- математична модель, закладена основу здійснених у програмі обчислень з указівкою імен параметрів і змінних;
- інформація про використані значення параметрів;
- інформація про отримані значення деяких обчислених інтегральних параметрів;
- інформація про ім'я М-файлу використаної програми, виконавця роботи й дату проведення обчислювального експерименту;

- інформація про автора використаної програми й організації, де він працює.

Остання вимога ставить перед необхідністю відокремлювати (за допомогою тієї ж процедури **subplot)** у кожній фігурі місце для виведення зазначеної текстової інформації.

6.2 Розбиття графічного вікна на підвікна

Як випливає зі сказаного, при створенні закінченого графічного інженерного документа в системі MatLAB необхідно використовувати процедуру **subplot.**

Розглянемо, як забезпечити бажане розбиття всього графічного вікна на окремі поля графіків і текстове підвікно.

Нехай потрібно розбити усе поле графічного вікна так, щоб верхня третина вікна утворила поле виведення тексту, а нижні дві третини утворили єдине поле виведення графіків. Це можна здійснити таким чином:

- перед виведенням текстової інформації у графічне вікно треба встановити команду **subplot**(3,1,1), яка показує, що весь графічний екран, розділений на три однакові частини по вертикалі, а для наступного виведення буде використана верхня з цих трьох частин (рис. 6.1);

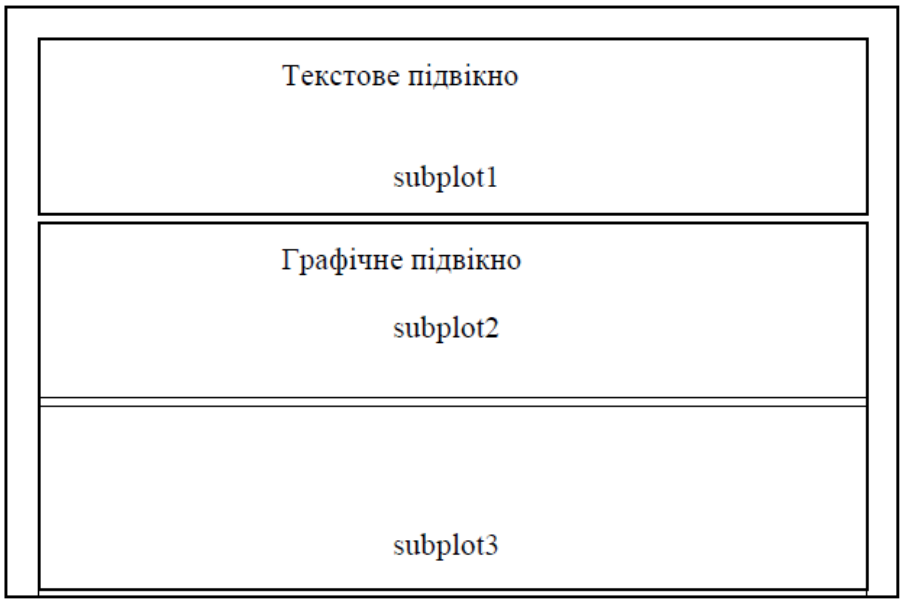

Рис. 6.1. Схема 1 розташування підвікон у графічному вікні

- виведенню графіків у графічне вікно має передувати команда **subplot(3,l,[2** 3]), відповідно до якої графічне вікно розділяється, як і раніше, на три частини по вертикалі, але тепер для виведення графічної інформації буде використаний простір, що об'єднує друге й третє зі створених підвікон (полів) (зверніть увагу, що підвікна

об'єднуються таким самим чином, як елементи вектора у вектор-рядок).

Щоб створити три окремих поля графіків один під іншим на трьох чвертях екрана по горизонталі, а текстову інформацію розмістити в останній чверті по горизонталі, то це можна зробити (рис. 6.2) так:

Рис. 6.2. Схема 2 розташування підвікон у графічному вікні

1) поділити весь простір фігури на 12 частин - на 3 частини по вертикалі і на 4 частини по горизонталі; при цьому підвікна будуть розташовані так, як показано на рис. 2.7;
2) щоб організувати виведення графіків у перше графічне підвікно, треба попередньо ввести команду **subplot**(3,4,[1 2 3]), що об'єднує підвікна spl, sp2 і sp3 у єдине графічне підвікно;
3) аналогічно, виведенню графіків у друге графічне підвікно повинно передувати звернення до команди **subplot**(3,4,[5 6 7]), а виведенню графіків у третє графічне підвікно - **subplot**(3,4,[9 10 11]);
4) нарешті, до оформлення тексту можна приступити після звернення **subplot**(3,4,[4 8 12]).

6.3 Виведення тексту в графічне вікно (підвікно)

Якщо по черзі сформувати підвікна, приміром, відповідно до останньої схеми, не здійснюючи ніяких операцій по виведенню графіків або тексту:

```
» subplot(3,4,[5 6 7])
» subplot(3,4,1:3)
» subplot(3,4,9:11)
» subplot(3,4,[4 8 12]),
```

У вікні фігури з'явиться зображення, подане на рис. 6.3.

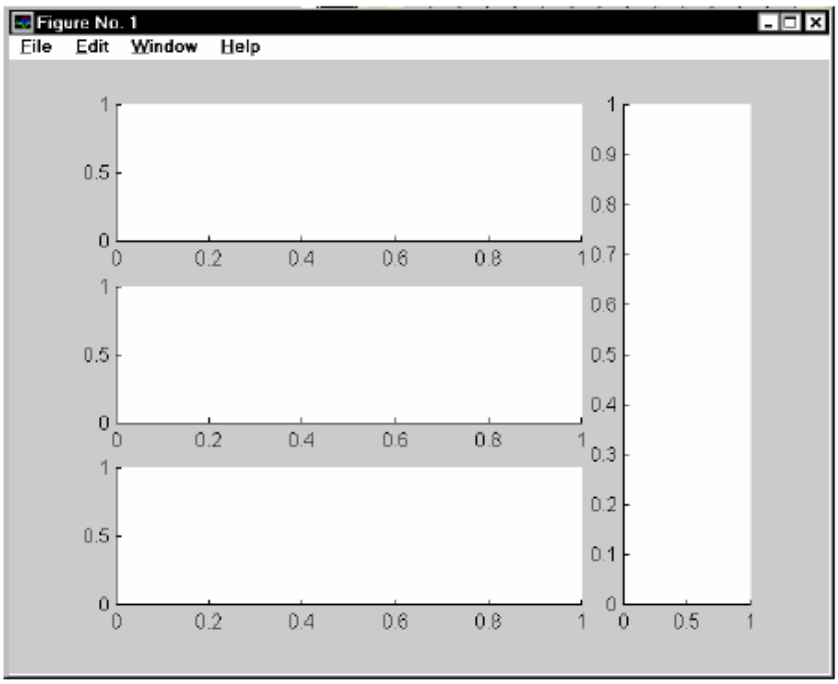

Рис. 6.3. Вид графічного вікна після розбиття його на підвікна

З його розгляду випливає:

1) після звернення до процедури **subplot** у відповідному підвікні з'являється зображення осей координат із позначенням поділків по осях;

2) початковий діапазон змінювання координат по обох осях підвікна завжди встановлюється за замовчуванням від 0 до 1;

3) поле виведення графіків займає не весь простір відповідного підвікна , - залишається деяке місце навколо поля графіка для виведення заголовка графіка, написів по осях координат та ін.

Тому для виведення тексту в одне з підвікон потрібно спочатку очистити це підвікно від зображення осей координат і написів на них. Це робиться за допомогою команди

axis('off).

Наприклад, якщо цю команду ввести після попередніх команд, у вікні фігури зникне зображення координатних осей останнього підвікна (рис. 6.4).

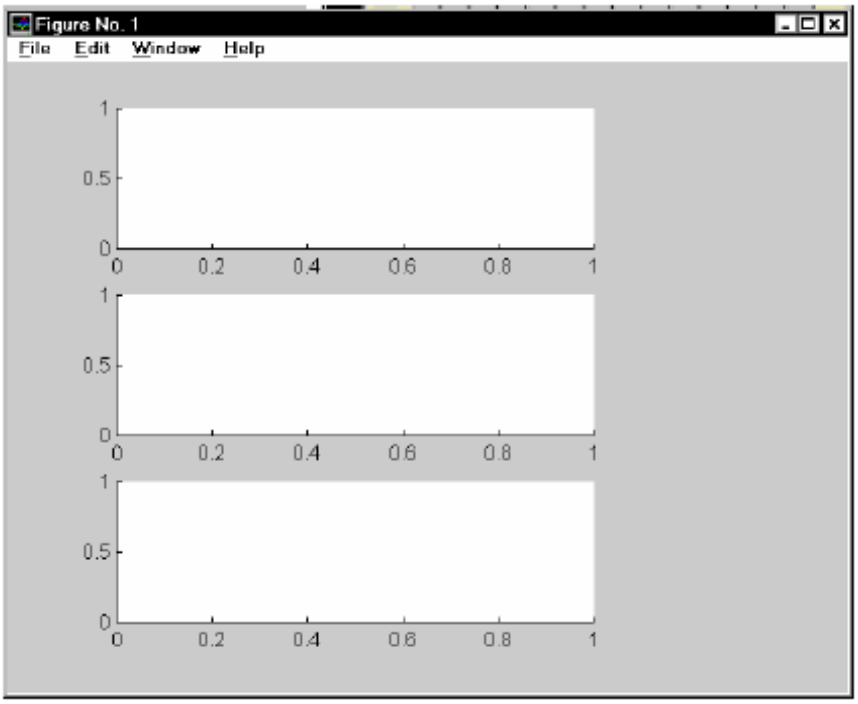

Рис. 6.4. Вид графічного вікна після команди **axis**('off').

Тепер можна починати виведення текста в це підвікно.

Основною функцією, яка забезпечує виведення тексту в графічне вікно, є функція **text.** Узагальнена форма звернення до неї має вид:

h=text (x, y, '<ТеКСТ>','FontName','<Назва шрифту>', 'FontSize',<розмір шрифту в пикселах>).

Вона здійснює виведення зазначеного тексту зазначеним шрифтом зазначеного розміру, починаючи з точки підвікна з координатами x і y відповідного поля графіка підвікна. При цьому координати x і y вимірюються в одиницях величин, які відкладаються уздовж відповідних осей графіка підвікна. Через те що, як ми переконалися, діапазон змінювання цих координат дорівнює [0...1], то для того, щоб помістити початок тексту в точку усередині поля графіка, необхідно, щоб його координати x і y були в цьому діапазоні.

Проте можна використовувати і дещо ширшій діапазон, враховуючи те, що поле підвікна більше поля його графіка.

Розглянемо приклад текстового оформлення. Нехай частина програми має вид:

```
subplot(3,4,1:3);
subplot(3,4,5:7);
subplot(3,4,9:11);
subplot(3,4,[4;8;12]);
axis('off');
% Процедура виведення даних в текстове поле графічного вікна
D1 =[2 1 300 1 50];
D2 = [ 0.1 0.02 -0.03 0 1 4 -1.5 2 0.1 -0.15 0 0];
D5 = [0.001 0.01 15 16]; sprogram = 'vsp1'; sname = 'Лазарев Ю.Ф.';
h1=text(-0.2,1,'McxoAHbie napaMeTpbi:','FontSize\12);
h1=text(0,0.95,TMpoTaxoMeTpoB','FontSize',10);
h1=text(0.2,0.9,sprintf(' = %g ',D1(3)),'FontSize',10);
```

```
h1=text(-0.2,0.85,sprintf(' = %g ',D1(4)),'FontSize',10);
h1=text(0.6,0.85,sprintf(' = %g ',D1(5)),'FontSize',10);
h1=text(-0.2,0.8,sprintf(' = %g ',D1(1)),'FontSize',10);
h1=text(0.6,0.8,sprintf('J2 = %g ',D1(2)),'FontSize',10);
h1=text(0,0.75,'ВНemНМх воздействий','FontSize',10);
h1=text(-0.2,0.7,sprintf('pst0 = %g ',D2(1)),'FontSize',10);
h1=text(0.6,0.7,sprintf(' tet0 = %g ',D2(2)),'FontSize',10);
h1=text(0.2,0.66,sprintf(' fit0 = %g ',D2(3)),'FontSize',10);
h1=text(-0.2,0.62,sprintf('psm = %g ',D2(4)),'FontSize',10);
h1=text(0.6,0.62,sprintf(' tem = %g ',D2(5)),'FontSize',10);
h1=text(0.2,0.58,sprintf(' fim = %g ',D2(6)),'FontSize',10);
h1=text(-0.2,0.54,sprintf('omps = %g ',D2(7)),'FontSize',10);
h1=text(0.6,0.54,sprintf(' omte = %g ',D2(8)),'FontSize',10);
h1=text(0.2,0.5,sprintf('omfi = %g ',D2(9)),'FontSize',10);
h1=text(-0. 2,0.46,sprintf('eps = %g ',D2(10)),'FontSize',10);
h1=text(0.6,0.46,sprintf(' ete = %g ',D2(11)),'FontSize',10);
h1=text(0.2,0.42,sprintf('efi=%g',D2(12)),'FontSize',10);
h1=text(0,0.35,'Интегрирования','FontSize',10,'FontUnderline','on');
h1=text(0,0.3,sprintf('h = %g ',D5(1)),'FontSize',10);
h1=text(0,0.25,sprintf('hpr = %g ',D5(2)),'FontSize',10);
h1=text(0,0.2,sprintf('t = %g ',D5(3)),'FontSize',10);
h1=text(0,0.15,sprintf('tfinal = %g ',D5(4)),'FontSize',10);
h1=text(-0.3,0.12,' ---------------------------------------------------------------- ','FontSize',10);
tm=fix(clock); Tv=tm(4:6);
h1=text(-0.2,0.08,['Программа ' sprogram],'FontSize',10);
h1=text(-0.3,0.04,['Рас4еТbi провел ' sname],'FontSize',10);
h1=text(-0.3,0,[sprintf(' %g :',Tv) ' ' date],'FontSize',10);
h1=text(-0.3,-0.04,' ---------------------------------------------------------------- ','FontSize',10);
h1=text(-0. 3,-0. 08,'Ukraine, KPI, cath. PSON','FontSize',10);
```

Виконання цієї програми приводить до появи у вікні фігури зображення, поданого на рис. 6.5.

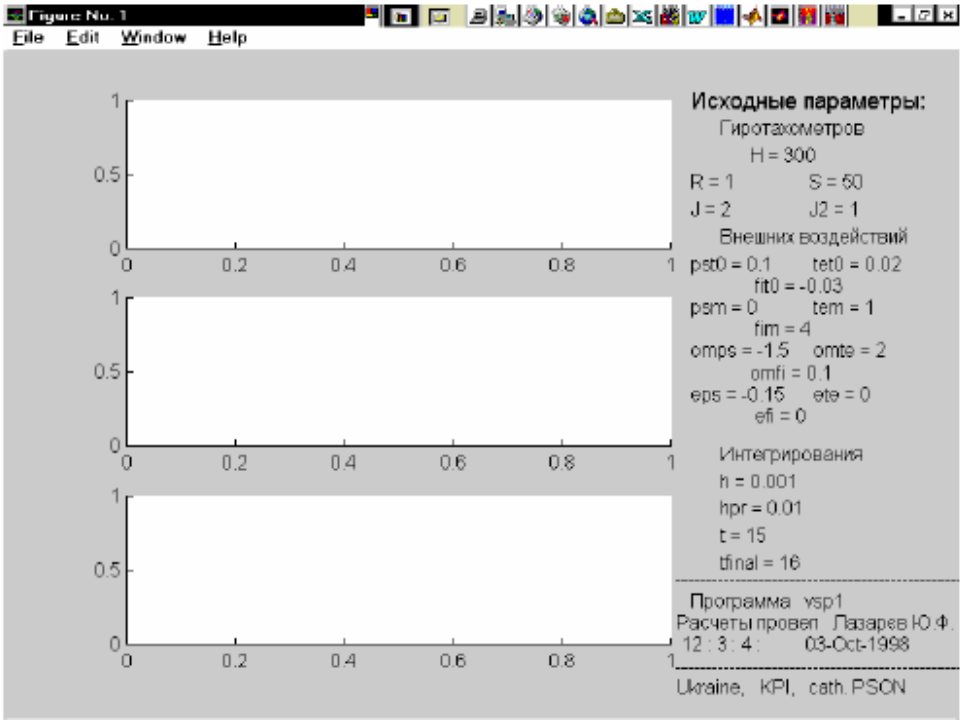

Рис.6.5. Вивід графічного вікна з оформленим текстовим підвікном

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

6.4 Завдання для виконання

Побудувати графіки функцій x(t) і y(t) в одному графічному вікні **figure** і в різних підвікнах

Таблиця 6.1 – Варіанти завдань

Варіант	A	B	f	Крок зміни t (крок дискретизації)	Функції
1	2	3	50	0.01	$x(t)=A\cdot\sin(2pi\cdot f\cdot t), t\in[0,20)$ $y(t)=B\cdot\cos(2pi\cdot f\cdot t), t\in[0,10)$
2	1	4	55	0.05	$x(t)=A\cdot\sin(2pi\cdot f\cdot t), t\in[0,20)$ $y(t)=B\cdot\cos(2pi\cdot f\cdot t), t\in[0,15)$
3	5	2	45	0.1	$x(t)=A\cdot\sin(2pi\cdot f\cdot t), t\in[0,20)$ $y(t)=B\cdot\cos(2pi\cdot f\cdot t), t\in[0,20)$
4	2	2	100	0.02	$x(t)=A\cdot\sin(2pi\cdot f\cdot t), t\in[0,20)$ $y(t)=B\cdot\cos(2pi\cdot f\cdot t), t\in[0,40)$
5	1	1	150	0.001	$x(t)=A\cdot\sin(2pi\cdot f\cdot t), t\in[0,30)$ $y(t)=B\cdot\cos(2pi\cdot f\cdot t), t\in[0,15)$

6.5. Зміст звіту

6.5.1. Тема та мета роботи.

6.5.2. Завдання до роботи.

6.5.3. Відобразити отримані результати у вигляді копій екрану (PrintScreen).

6.5.4. Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Яким чином здійснюється процедура розділення графічного вікна на підвікна?
2. Яка функція відповідає за вивід масиву даних у графічному вікні?

Практична робота №7
Графічне оформлення результатів обробки біомедичних сигналів у вигляді 3D-графіків

Мета роботи: вивчити процедуру графічного оформлення результатів обробки біомедичних сигналів у вигляді 3D-графіків в середовищі MATLAB

ТЕОРЕТИЧНІ ВІДОМОСТІ

7.1 Загальні відомості

Високорівнева графічна підсистема MATLAB автоматично реалізує тривимірну графіку без спеціальних зусиль з боку користувача. Нехай в точці з координатами x1, y1 обчислене значення функції z=f(x, y) і воно рівне z1. У деякій іншій точці (тобто при іншому значенні аргументів) x2, y2 обчислюють значення функції z2. Продовжуючи цей процес, отримують масив (набір) точок (x1, y1, z1), (x2, y2, z2), … (xN, yN, zN) у кількості N штук, розташованих в тривимірному просторі. Спеціальні функції системи MATLAB проводять через ці точки гладкі поверхні і відображають їх проекції на дисплей комп'ютера. Найчастіше точки аргументів розташовані в області визначення функції регулярно у вигляді прямокутної сітки (тобто матриці). Така сітка точок породжує дві матриці однієї і тієї ж структури: перша матриця містить значення перших координат цих точок (x - координат) а друга матриця містить значення других координат (y - координат). Позначимо першу матрицю як X, а другу – як Y. Є ще і третя матриця – матриця значень функції z=f(x, y) при цих аргументах. Цю матрицю позначимо буквою Z.

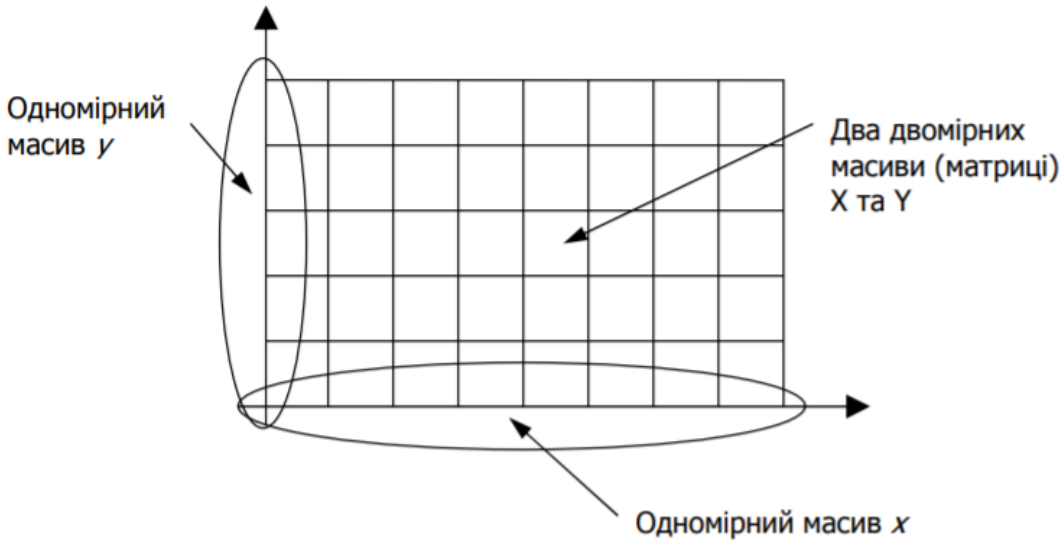

Рис.7.1. Сітка координат сформована по двом одномірним масивам

Для отримання матриць X і Y, що представляють перші і другі координати прямокутної сітки точок використовують спеціальну функцію системи MATLAB:

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

Після визначення сітки розміром X, Y, Z можна скористатися функціями для побудови тривимірних поверхонь.

mesh(X, Y, Z) – функція, що сполучає обчислені сусідні точки поверхні графіку відрізками прямих.

Приклад:

```
>> [X,Y]=meshgrid(-2:0.2:2);
>> Z=exp(-X.^2-Y.^2);
>> mesh(X,Y,Z)
```

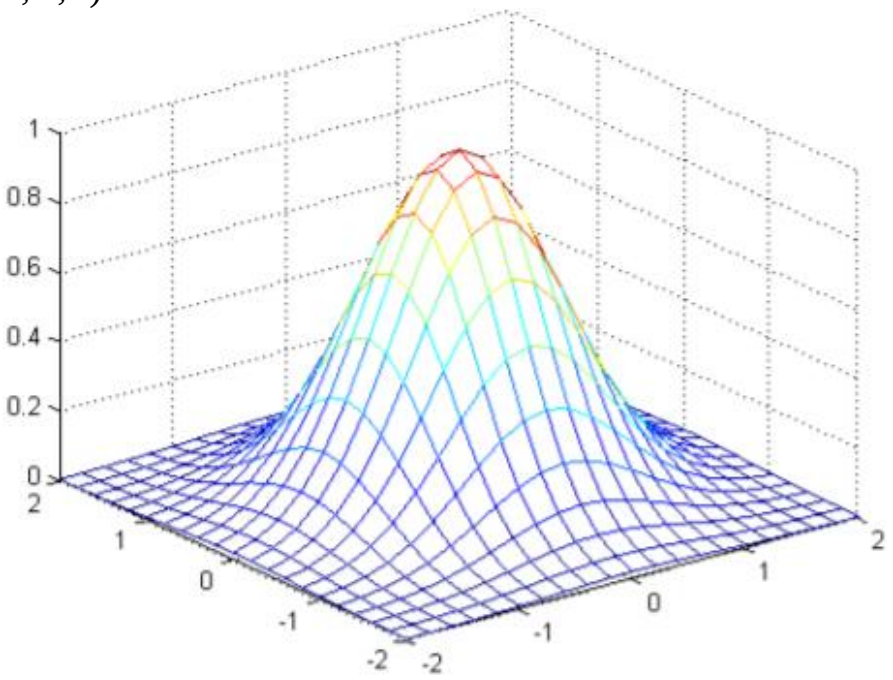

Рис.7.2. Поверхня сформована командою mesh

Подібними до команди mesh є команди surf і plot3.

surf(X, Y, Z) – будує тривимірну кольорову поверхню.

plot3(X, Y, Z) – будує масив точок представлених векторами X, Y, Z, сполучаючи їх відрізками прямих.

Приклади графіків вищенаведеної функції, які сформовані за допомогою команд surf і plot3 представлені на рис.7.3(а, б).

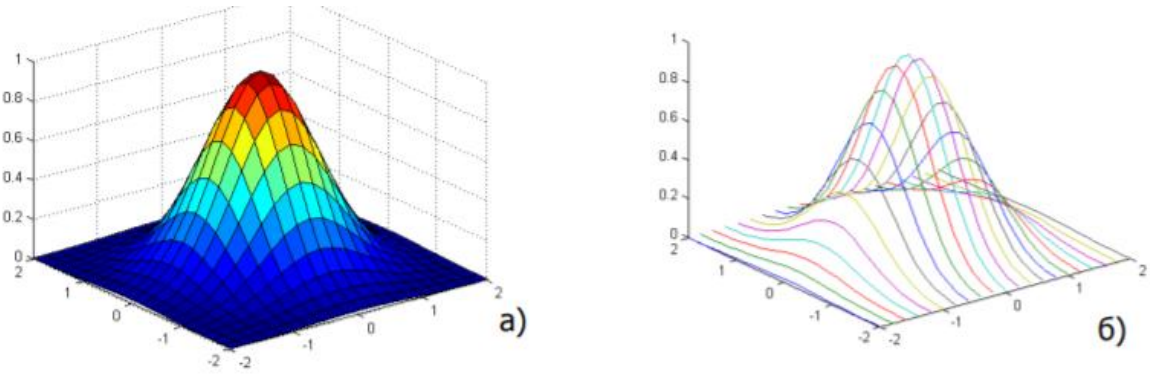

Рис.7.3. Поверхня сформована командами surf (а) і plot3 (б)

ЕКСПЕРИМЕНТАЛЬН ЧАСТИНА

Згідно власного варіанту завдання побудувати графік заданої функції Z f=(X,Y) в 3D форматі, а саме кольорову поверхню. Завдання наведені в табл.7.1.

№ варіанту	*Функція*	*Діапазон зміни значень **x** та **y***	*Крок*
1	$X \cdot e^{-X^2-Y^2}$	$[-2, 2]$	*0,1*
2	$X \cdot \sin(X+Y)$	$[-5, 5]$	*0,2*
3	X^2+Y^2	$[-3, 3]$	*0,1*
4	$X \cdot \cos(X+Y)$	$[-6, 6]$	*0,25*
5	X^3+Y^2	$[-4, 4]$	*0,2*
6	$2Y \cdot \sin(X+Y)$	$[-5, 7]$	*0,2*
7	$3Y \cdot e^{-X^2-Y^2}$	$[-3, 3]$	*0,1*
8	$Y \cdot \cos(X+2Y)$	$[-4, 4]$	*0,2*
9	X^3+Y^3	$[-2, 2]$	*0,1*
10	$2X \cdot e^{-2X^2-2Y^2}$	$[-3, 3]$	*0,1*
11	$X^2 \cdot \sin(X+Y)$	$[-5, 5]$	*0,25*
12	$X \cdot \cos(X^2+Y^2)$	$[-4, 4]$	*0,1*
13	$X \cdot (X^2+Y^2)$	$[-3, 3]$	*0,1*
14	$3X \cdot e^{-2X^2-2Y^2}$	$[-2, 2]$	*0,2*
15	$4X \cdot \cos(X+2Y)$	$[-5, 5]$	*0,25*
16	$5Y \cdot e^{-(X^2+Y^2)}$	$[-6, 6]$	*0,25*
17	$Y \cdot (X^2+Y^2)$	$[-3, 3]$	*0,1*
18	$5 \cdot e^{-X^2-Y^2}$	$[-4, 4]$	*0,2*
19	$Y \cdot \cos(4X^2+6Y^2)$	$[-5, 5]$	*0,2*
20	$\pi \cdot (X^2+Y^2)$	$[-2, 2]$	*0,1*

7.2. Зміст звіту

7.2.1. Тема та мета роботи.
7.2.2. Завдання до роботи.
7.2.3. Відобразити отримані результати у вигляді копій екрану (PrintScreen).
7.2.4. Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1 Які функції в середовищі MATLAB забезпечують побудову 3D-графіків?
2 Яка різниця між функціями, які забезпечують побудову 3D графіків?

Практична робота №8
Завантаження біосигналів та збереження результатів обробки у файл

Мета роботи: вивчити процедуру завантаження біосигналів із файлів та збереження результатів їх обробки у файл в середовищі MATLAB

ТЕОРЕТИЧНІ ВІДОМОСТІ

8.1 Загальні відомості

Розробка програм часто передбачає збереження результатів розрахунків в файли для їх подальшого аналізу, обробки, зберігання і т.д. У зв'язку з цим в MatLab реалізовані різні функції по роботі з файлами, що містять дані в самих різних форматах. У цьому розділі розглянемо найбільш корисні функції для збереження і завантаження результатів роботи алгоритмів з файлів.

У найпростішому випадку для збереження і подальшого завантаження будь-яких даних в MatLab передбачені дві функції

dlmwrite(‘директорія для збереження’, назва змінної,’ ‘) % збереження даних

load (‘<директорія для збереження > <назва змінної>’) % завантаження даних

wawread load (‘<директорія > <назва змінної>’) % завантаження звукових даних

Функція dlmwrite дозволяє зберігати довільні данні програми в файл, який буде розташовуватися в вами заданій директорії і мати розширення, яке ви задасьте (txt, dat, mat та інші).

Функція load дозволяє завантажити із зазначеного розширення файлу раніше збережені змінні.

Нижче представлено приклад використання даних функцій:

- Завантаження даних signal.txt, який розташований в директорії ‘c:/temp/’:

  x=load(‘c:/temp/signal.txt’);

- Збереження даних змінної x в директорію ‘c:/temp’:

  dlmwrite(‘c:/temp’, x,’ ‘);

- Завантаження звукових даних signal.txt (наприклад, фонокардіосигнал), який розташований в директорії ‘c:/temp/’:

  x=wawread(‘c:/temp/signal.txt’);

ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

1. Згідно власного варіанту курсової роботи завантажити дані біосигналів із файлу з використанням команди load або wavread. Завантаженні дані біосигналів відобразити на графіку із використанням команди plot.
2. Згідно власного варіанту курсової роботи зберегти завантажити дані біосигналів у файл із використанням команди dlmwrite в іншу директорію та із іншою назвою.

8.2. Зміст звіту

8.2.1. Тема та мета роботи.
8.2.2. Завдання до роботи.
8.2.3. Відобразити отримані результати у вигляді копій екрану (PrintScreen).
8.2.4. Висновки за результатами виконаної роботи.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які функції в середовищі MATLAB забезпечують завантаження даних?
2. Яка функція в середовищі MATLAB забезпечує процедуру збереження даних у файл?

Список використаних джерел

1. Верлань А.Ф. та ін. Моделирование систем управления в среде MATLAB. – К.: ЦКІС АПНУ, 2002. – 68 с.
2. Дьяконов В.П., Абраменкова И.В. Matlab 5.0/5.3. Система символьной математики. М.: "Нолидж". 1999. 633 с.
3. Муха В .С ., Птичкин В .А . Введение в MATLAB: Метод . пособие для выполнения лаб . работ по курсам "Статистические методы обработки данных " и "Теория автоматического управления " для спец . 53 01 02 "Автоматизированные системы обработки информации ". – Мн .: БГУИР , 2002. – 40 с .
4. Потемкин В.Г. Система MATLAB. Справ. пособие. Диалог-МИФИ. 1997. 350 с.
5. Потемкин В.Г. Система инженерных и научных расчетов MATLAB 5.x. В 2-х томах. Диалог-МИФИ. 1999 (т. 1. 366 с., т. 2. 304 с.).
6. Егоренков Д.Л., Фрадков А.Л., Харламов В.Ю. Основы математического моделирования с примерами на языке МАТЛАБ. Учеб. Пособие под ред. проф. Фрадкова А.Л. СПб: БГТУ. 1994. 190 с.