МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Воронежский государственный технический университет»

Кафедра радиоэлектронных устройств и систем

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторной работы № 1 (часть 2) по дисциплине «Цифровая обработка сигналов» для студентов специальности 11.05.01 «Радиоэлектронные системы и комплексы» очной формы обучения

Воронеж 2022

УДК 621.391.083.92
ББК 32.811.3

Составитель:
д. ф.-м.н. Кузьменко Р.В

Цифровая обработка сигналов: методические указания к выполнению лабораторных работ № 1 (часть 2) по дисциплине «Цифровая обработка сигналов» для студентов специальности 11.05.01 «Радиоэлектронные системы и комплексы» очной формы обучения/ ФГБОУ ВО «Воронежский государственный технический университет»; сост. Р.В. Кузьменко. Воронеж: Изд-во ВГТУ, 2022. – 40 с.

Основной целью указаний к выполнению лабораторных работ является поддержка выработка навыков цифровой обработки сигналов и средств их компьютерного моделирования в системе MATLAB.

Издание предназначено для проведения лабораторных работ по дисциплине «Цифровая обработка сигналов» для студентов 4-го курса.

Методические указания подготовлены в электронном виде и содержатся в файле ЦОС Лаб. работа № 1 (часть 2).docx

Библиогр.: 3 назв.

УДК 621.391.083.92
ББК 32.811.3

Рецензент: Доктор технических наук, заведующий кафедрой
«Конструирования и производства радиоаппаратуры»
Башкиров А.В.

Издается по решению редакционно-издательского совета
Воронежского государственного технического университет

Лабораторная работа №1
Знакомство с MATLAB.

3. Типы массивов

Цель работы: изучить основные типы массивов, используемых в MATLAB, и овладеть навыками их формирования.

3.1. Краткая теоретическая справка

В MATLAB тип массива (тип данных) определяется типом его элементов. По умолчанию мы имели дело с матрицами, элементы которых представлены константами в форме с плавающей точкой (ПТ) с двойной точностью. Такие массивы относятся к типу double.

Основные типы числовых и нечисловых массивов представлены в табл. 3.1.

Таблица 3.1. Типы массивов в MATLAB

Символическое обозначение типа массива	**Тип массива**	**Функция преобразования типа**
	Числовой:	
double	вещественный двойной точности	double(X)
single	вещественный одинарной точности	single(X)
int8	целый 8-битовый со знаком	int8(X)
uint8	целый 8-битовый без знака	uint8(X)
int16	целый 16-битовый со знаком	int16(X)
uint16	целый 16-битовый без знака	uint16(X)
int32	целый 32-битовый со знаком	int32(X)
uint32	целый 32-битовый без знака	uint32(X)
int64	целый 64-битовый со знаком	int64(X)
uint64	целый 64-битовый без знака	uint64(X)

Таблица 3.1 (окончание)

Символическое обозначение типа массива	**Тип массива**	**Функция преобразования типа**
logical	Логический	logical(X)
character (char)	Символьный	num2str(X)
structure (struct)	Структура (массив записей)	—
cell	Массив ячеек	—

3.1.1. Матрицы числового и логического типов

Преобразование матриц числового типа double в матрицы других числовых типов выполняется с помощью специальных встроенных функций (см. табл. 3.1).

При обработке матриц числового целого типа необходимо иметь в виду, что с ними запрещено выполнение некоторых арифметических операций и вычисление большинства встроенных функций, но разрешено выполнение операций отношения и логических операций:

```
>> A = [int8(159.7) int8(125.7) int8(-125.7)]
A       =
  127  126 -126
>> B = [uint8(159.7) uint8(125.7) uint8(-125.7)]
B       =
  160  126    0
>> C = [A<B; A==B; and(A,B)]
C       =
  1     0     1
  0     1     0
  1     1     0
```

Элементами матрицы логического типа (logical array) являются логические константы, принимающие значения 1 (true — истина) или 0 (false — ложь), например, как в матрице C, или логические выражения (см. разд. 1.1.2):

```
>>x = [sin(3)<0.5  1;0 (sin(3)<0.1)&(cos(3)<0.2)]

x =
  1     1
  0     0
```

3.1.2. Матрицы символьного типа

Матрица символьного типа (char array) — это разновидность нечисловых матриц, элементами которой являются символьные константы (см. разд. 1.1.2).

Строки и столбцы матрицы символьного типа формируются по-разному, а именно:

элементы строки представляют собой слитную запись, поэтому необходимые пробелы должен предусмотреть пользователь:

```
>> a = ['Alla ' 'Woman ' 'Russian']
a =
AllaWoman Russian
```

элементы столбца должны содержать одинаковое число символов, при этом пробел считается символом:

```
>> a = ['Alla   ';'Woman  ';'Russian']
a =
Alla
Woman
Russian
```

Автоматическое добавление пробелов в элементах столбца с выравниванием по левому краю выполняется с помощью функции: **char('<char1>','<char2>'...)**

где <char1>,<char2>... — элементы столбца с произвольным количеством символов.

Одна функция char описывает один столбец матрицы с символьными константами '<char1>','<char2>'...:

```
>> a = char('Alla','Woman','Russian')
 a =
Alla
Woman
Russian
```

Матрицу символьного типа удобно формировать по столбцам, используя для каждого из них свою функцию char и предусматривая необходимое количество пробелов для разделения столбцов:

```
>> x = [char('a','aa','aaa') char(' bb',' bbb',' b') ...
 char(' cc',' ccc',' c')]
```

$$x = \begin{matrix} a & bb & cc \\ aa & bbb & ccc \\ aaa & b & c \end{matrix}$$

Преобразование матрицы численного или логического типа в матрицу символьного типа выполняется с помощью функции num2str(x):

```
>> x = [5 7;-1 9]
 x =
    5    7
   -1    9
>> y = num2str(x)
 y =
```

```
5  7
-1  9
```
Здесь x — матрица типа double, а y — матрица символьного типа:
```
>> [size(x); size(y)]
ans =
    2     2
    2     5
```
3.1.3. Структуры (массивы записей)

Структура (массив записей — struct array) — это разновидность нечислового массива, предназначенного для описания M объектов N параметрами.

Для описания структуры потребуется ввести ряд новых терминов:

- *поле* (field) — имя параметра, описывающего объект: скаляра, вектора, матрицы[1] или нечислового массива;
- *число полей* равно количеству параметров N;
- *значение поля* — значение параметра;
- *список полей* — список имен параметров;
- *запись* — список полей, одинаковый для всех M объектов;
- *число записей* равно количеству объектов M;
- *значение записи* — список полей с их значениями для одного объекта.

Структура (массив записей) — это упорядоченная совокупность значений записей, объединенная одним именем.

Имя массива записей выбирается так же, как обычно для переменной (см. разд. 1.1.2), а размер равен числу записей M.

Значение каждой i-й записи формируется отдельно по каждому n-му полю:

<имя массива>(<i>).<имя n-го поля> = <значение n-го поля>

Таким образом, для формирования M значений записей со списком из N полей потребуется M×N операторов присваивания.

Сформируем массив записей (структуру) с именем personal для описания трех объектов (M = 3) — трех членов кафедры — четырьмя параметрами (N = 4).

Запись включает в себя следующий список полей:

- surname — фамилия — скаляр символьного типа;
- data — дата рождения (число, месяц, год) — вектор численного типа;
- position — должность — скаляр символьного типа;
- phd — наличие ученой степени — скаляр логического типа.

Сформируем значения массива записей personal по каждому полю:
```
>> personal(1).surname = 'Иванов';
>> personal(2).surname = 'Петров';
>> personal(3).surname = 'Сидоров';
```
[1] Напомним (см. разд. 1.1.2), что по умолчанию в MATLAB любая переменная — матрица, а скаляр и вектор — ее частные случаи.

```
>> personal(1).data = [1 2 1949];
>> personal(2).data = [5 7 1975];
>> personal(3).data = [5 8 1956];
>> personal(1).position = 'профессор';
>> personal(2).position = 'доцент';
>> personal(3).position = 'зав. лаб.';
>> personal(1).phd = true;
>> personal(2).phd = true;
>> personal(3).phd = false;
```

Список полей выводится по имени массива записей:

```
>> personal
 personal =
1x3 struct array with fields:
   surname
   data
   position
   phd
```

Значение i-й записи выводится по имени массива записей с указанием индекса в круглых скобках. Например, значение 2-й записи:

```
>> personal(2)
 ans =
     surname: 'Петров'
        data: [5 7 1975]
    position: 'доцент'
         phd: 1
```

Значение поля в i-й записи выводится по имени массива записей с указанием индекса в круглых скобках и имени поля. Например, значение поля surname первой записи:

```
>> personal(1).surname
ans =
Иванов
```

Значения поля во всех записях выводятся по имени массива с указанием имени поля.

Например, поля surname:

```
>> personal.surname
 ans =
Иванов
ans =
Петров
ans =
Сидоров
```

Удаление поля выполняется с помощью функции:

<имя массива> = rmfield(<имя массива>,'<имя поля>')Например, удалим поле data:

```
>> personal = rmfield(personal,'data')
personal =
1x3 struct array with fields:
surname
position
phd
```

Массивы ячеек

Массив ячеек (cell array) — это наиболее сложный тип массива, элементами которого являются ячейки, представляющие собой массивы любой размерности, любого размера и типа.

Элементы массива ячеек указываются в фигурных скобках.

Сформируем квадратную матрицу ячеек 3×3:

```
>> A{1,1} = pi;
>> A{1,2} = [1 2 3;4 5 6];
>> A{1,3} = char('abs','angle');
>> A{2,1} = [ones(5,1)]';
>> A{2,2} = zeros(3,1);
>> A{2,3} = 'Alla';
>> A{3,1} = 7 ;
>> A{3,2} = rand(5,1); >> A{3,3} = personal;
```

гдеpersonal — имяструктуры, сформированнойвразд. 3.1.3.

Вывод элементов массива ячеек выполняется по его имени с указанием индексов:

```
>> A{1,2}
ans =
    1    2    3
    4    5    6
>> A{3,3}
ans =
1x3 struct array with fields:
surname
age
position
phd
```

С элементами массива ячеек можно выполнять операции, разрешенные для данного типа массива с учетом согласования их размерностей и размеров, например:

```
>> B = sum(A{1,2})+A{1,1}
B =
   8.1416   10.1416   12.1416
```

Графическое представление матрицы ячеек создается с помощью функции:

cellplot(A,'legend')

Определение типа массива

Для определения типа массива служит функция: class(<имя массива>)

Например, для массива A, сформированного в разд. 3.1.4:

>> class(A) ans =

cell

Содержание лабораторной работы

Содержание работы связано с изучением типов массивов в MATLAB в режиме прямых вычислений.

Задание на лабораторную работу

Задание на лабораторную работу включает в себя следующие пункты:

1. Знакомство с матрицами числового и логического типов. Ввести матрицу A с элементами:

$$A = \begin{bmatrix} 127{,}1 & -127{,}1 & 127{,}7 \\ -127{,}7 & 0 & 128{,}4 \\ -128{,}4 & 255{,}7 & 255{,}1 \end{bmatrix}$$

и выполнить с ней следующие действия:

- преобразовать в матрицу B целых 8-битовых чисел со знаком;
- преобразовать в матрицу C целых 8-битовых чисел без знака;
- преобразовать в матрицу D логического типа;
- определить тип матриц A, B, C, D.

Пояснить:

- как преобразовать матрицу A в матрицы B, C и D;
- по какому правилу формируются элементы матрицы B;
- по какому правилу формируются элементы матрицы C; • по какому правилу формируются элементы матрицы D;
- как определяется тип матрицы.

2. Операции с матрицами числового типа.

Выполнить следующие операции:

- вычислить значения синуса всех элементов матриц A, B и C;
- вычислить суммы матриц A и B, B и C;
- проверить, являются ли элементы матрицы A числами, большими единицы, и определить вид и тип результата;
- выполнить логическую операцию "И" с матрицами B, C и определить вид и тип результата.

Сделать выводы по результатам выполнения операций.

3. Знакомство с матрицами символьного типа.

Выполнить следующие действия:

- сформировать трехэлементный вектор-строку X символьного типа с ФИО студента;

- сформировать трехэлементный вектор-столбец Y символьного типа с ФИО студента;
- сформировать матрицу F символьного типа 2×2 с элементами:

$$F = \begin{bmatrix} \text{КИХ} & R = 15 \\ FIR & Order = 15 \end{bmatrix};$$

- преобразовать матрицу A (см. п. 1) в матрицу G символьного типа;
- определить типы матриц A и G.

Пояснить:

- как обеспечить не слитный вывод элементов вектора-строки;
- как обеспечить автоматическое добавление пробелов при выводе элементов столбца;
- как преобразовать матрицу числового типа в матрицу символьного типа.

4. Знакомство с массивами записей (структурами).

Сформировать массив записей (структуру) с именем Filter для описания четырех фильтров.

Каждая запись должна содержать три поля со следующими именами и их значениями:

- Type (типизбирательности) — lowpass, highpass, bandpass, stopband;
- Order (порядок фильтра) — 10, 20, 30, 40;
- Poles (наличиеполюсов) — true, false, false, true.

Выполнить следующие действия с массивом Filter:

- вывести список полей;
- вывести значение 1-й записи;
- вывести значения поля Type во всех записях;
- вывести значение поля Type в 3-й записи;
- удалить поле Poles.

Пояснить:

- с какой целью создается массив записей;
- что собой представляет запись и значение записи;
- каков размер массива записей Filter.

5. Знакомство с матрицами ячеек.

Создать матрицу ячеек S 3×3, элементами которой являются сформированные ранее массивы:

$$S = \begin{bmatrix} A & B & C \\ D & F & G \\ X & Y & Filter \end{bmatrix}$$

Выполнить следующие действия:

- последовательно вывести элементы матрицы S и определить их тип;
- вывести графическое представление матрицы S.

Пояснить:

- из каких элементов создается матрица ячеек;
- как эти элементы вводятся;
- как выводится графическое представление матрицы ячеек;
- что оно собой представляет.

3.4. Задание на самостоятельную работу

Самостоятельное задание рекомендуется для закрепления полученных знаний и включает в себя следующие пункты: 1С. Операции с элементами массива ячеек.

Сформировать массив ячеек A, рассмотренный в разд. 3.1.4, и привести пример арифметического выражения с элементами данного массива типа double.

2С. Операции с матрицами целого типа.

Привести пример выражения с матрицами целого типа.

3С. Операции с матрицами логического типа.

Привести пример логического выражения, в котором все переменные — матрицы.

4С. Операции с матрицами символьного типа.

Сформировать матрицу символьного типа размера 3×3 посредством преобразования числовой матрицы типа double.

5С. Операции с массивом записей.

Привести пример массива записей для описания трех объектов двумя параметрами, один из которых представлен вектором, а другой — матрицей.

3.5. Отчет и контрольные вопросы

Отчет составляется в редакторе MS Word и содержит результаты выполнения каждого пункта задания, копируемые из окна Command Window (шрифт Courier New), и ответы на поставленные вопросы (шрифт Times New Roman).

Защита лабораторной работы проводится на основании представленного отчета и контрольных вопросов из следующего списка:

1. К какому типу относятся числовые массивы по умолчанию?
2. Как определить тип массива?
3. Какие типы числовых массивов используются в MATLAB?
4. Как преобразовать матрицу типа double в матрицы целых чисел разрядности 8, 16, 32 и 64 со знаком и без знака?
5. Какие операции возможны с матрицами числового целого типа?
6. Как преобразовать числовую матрицу в матрицу логического типа?
7. Что собой представляет матрица логического типа?
8. Какие типы нечисловых массивов предусмотрены в MATLAB?
9. Что собой представляет матрица символьного типа?
10. Что собой представляет массив записей?

11. В каких случаях целесообразно создавать массив записей (структуру)?

12. Что собой представляет матрица ячеек?

13. В каких случаях целесообразно создавать матрицу ячеек?

14. С какой целью выводится графическое представление матрицы ячеек?

4. Средства графики

Цель работы: изучить графический инструментарий MATLAB и овладеть навыками построения двумерных и трехмерных графиков.

4.1. Краткая теоретическая справка

Графический инструментарий MATLAB для построения и оформления двумерных и трехмерных графиков имеет свою специфику. Однако ряд следующих типовых положений является для них общим.

- Текущий график выводится в текущее графическое окно **Figure**, первый — в окно **Figure 1**. По умолчанию новый график выводится в то же окно, при этом предыдущий график автоматически удаляется.
- Вывод графиков в отдельных графических окнах с автоматически присваемыми номерами **Figure 1, Figure 2**, ... выполняется с помощью функции (без аргумента):

figure

которая ставится перед новой функцией построения графика.

- Вывод графика в отдельном графическом окне, имя которого присваивается пользователем, выполняется с помощью функции:

figure('Name','<Имяграфика>','NumberTitle','off')

- Вывод в текущее графическое окно **Figure** нескольких графиков на одних координатных осях выполняется по команде:

hold on

которая ставится перед новой функцией построения графика.

- Удаление из текущего графического окна **Figure** всех предыдущих графиков перед выводом нового графика выполняется по команде:

hold off

- Разбиение текущего графического окна **Figure** на отдельные поля для вывода независимых графиков выполняется с помощью функции:

subplot(m,n,p)

где m×n — размер матрицы графического окна: m строк и n столбцов; p — порядковый номер поля выводимого графика, считая по строкам слева направо.

Функция построения графика ставится после функции subplot.

- Средства оформления графиков, представленные в виде функций MATLAB (табл. 4.1), ставятся после функции построения графика. Во избежание ошибок $<_{текст}>$ рекомендуется вводить латинскими буквами.

Таблица 4.1. Функции оформления графиков

Функция	**Назначение**
grid	Нанесение координатной сетки с автоматическим выбором шага
title('<текст>')	Заголовок графика
xlabel('<текст>') **ylabel('<текст>')** **zlabel('<текст>')**	Обозначение осей графика x, y, z
xlim([xmin xmax]) **ylim([ymin ymax])** **zlim([zmin zmax])**	Установка границ (двухэлементным вектором) по осям x, y, z при выводе графика
legend('legend1','legend2',...)	Размещение легенды на автоматически выбираемом месте. При выводе нескольких графиков на одних осях их легенда отображается в порядке вывода графиков

Двумерные графики

Система MATLAB предлагает большое разнообразие стандартных функций для построения двумерных графиков.

Полный список функций, используемых в двумерной графике, выводится по команде:

help graph2d

Основные из них с наиболее распространенными форматами приведены в табл. 4.2.

Управление свойствами двумерных графиков

Свойствами графика можно управлять с помощью <параметров управления> (см. табл. 4.2), которые условно можно разделить на две группы:

- LineSpec — свойства без стандартных имен;
- PropertyName — свойства со стандартными именами.

Таблица 4.2. Функции построения двумерных графиков

Имя функции	**Назначение и формат**
plot	Графики в линейном масштабе с линейной интерполяцией между соседними значениями: **plot(x,y[,<параметры управления>])** где x, y — аргумент и функция (векторы или матрицы), согласованные по длине; <параметры управления> — необязательные параметры, управляющие свойствами графика (см.

	разд. 4.2.2). Квадратные скобки используются для условного обозначения необязательных параметров
semilogx	Графики в логарифмическом масштабе по оси абсцисс и линейном — по оси ординат. Формат подобен функции plot. Диапазон по оси абсцисс в логарифмическом масштабе можно задавать с помощью функции: **logspace(d1,d2[,n])** где d1, d2 — начальное 10^{d1} и конечное 10^{d2} значения диапазона; n — коли- чество точек в логарифмическом масштабе, по умолчанию равно 50
semilogy	Графики в линейном масштабе по оси абсцисс и логарифмическом — по оси ординат. Формат подобен функции plot
loglog	Графики в логарифмическом масштабе по осям абсцисс и ординат. Формат подобен функции plot
stem	Графики последовательностей чисел: **stem(x,y,'fill'[,<параметры управления>])** где 'fill' — необязательный параметр, указывающий на закрашивание маркеров. Остальные входные параметры определяются как для функции plot
hist	Гистограммы: **hist(y,x)** где y, x — векторы одинаковой длины; гистограмма отображает число попаданий значений элементов вектора y в интервалы, центры которых заданы элементами вектора x. В отсутствии вектора x для значений элементов вектора y по умолчанию выбирается 10 интервалов, и гистограмма отображает число попаданий значений элементов вектора y в центры данных интервалов. Цвет столбцов выбирается с помощью функции colormap (см. разд. 4.1.4)

Параметры группы LineSpec определяют тип и цвет линии графика, а также вид маркеров. Значения параметров данной группы представлены в табл. 4.3. В функциях построения графиков значения параметров указываются в апострофах без разделяющих символов в произвольном порядке. Например: stem(x,y,'-ms')

Если параметры не указаны, то они выбираются автоматически.

Таблица 4.3. Параметры группы LineSpec

Тип линии		**Цвет линии или маркера**		**Вид маркера**	
символ	**линия**	**символ**	**цвет**	**символ**	**маркер**
-	solid (непрерывная)	**y**	yellow (желтый)	**.**	point (точка)
:	dotted (пунктир, короткий штрих)	**m**	magenta (фиолетовый)	**o**	circle (кружок)
-.	dashdot (штрихпунктир)	**c**	cyan (голубой)	**x**	x-mark (крестик)
--	dashed (пунктир, длинный штрих)	**r**	red (красный)	**+**	plus (плюс)
		g	green (зеленый)	*****	star (звездочка)
		b	blue (синий)	**s**	square (квадрат)
		w	white (белый)		
		k	black (черный)		

Параметры группы PropertyName представлены четырьмя разновидностями со следующими стандартными именами, задаваемыми в апострофах:

- LineWidth — толщина линии в пунктах (1 пункт = 1/75 дюйма ≈ 0,34 мм), задаваемая цифрой без апострофов, по умолчанию равна 0,5;
- MarkerEdgeColor — цвет маркера, задаваемый значением соответствующего параметра из табл. 4.3 в апострофах;
- MarkerFaceColor — цвет закрашивания маркера (для замкнутых маркеров типа кружок, квадрат и т. п.), задаваемый значением соответствующего параметра из табл. 4.3 в апострофах;
- MarkerSize — размер маркера в пунктах, задаваемый цифрой без апострофов, по умолчанию равен 7.

Например:

```
stem(x,y,'MarkerSize',5,'MarkerEdgeColor','g','MarkerFaceColor',...
'r','LineWidth',1)
```

4.1.3. Трехмерные графики

Трехмерная графика предназначена для построения в трехмерном пространстве графиков функций двух переменных (двух аргументов) z(x, y).

Построение трехмерных графиков начинается с формирования сетки на плоскости *xOy* с помощью вспомогательных матриц **X** и **Y** по известным векторам x и y соответственно, где **X** — матрица, строки которой — копии вектора x, а Y — матрица, столбцы которой — копии вектора y. Матрицы X и Y должны иметь одинаковые размеры: количество строк каждой из них равно длине вектора y, а столбцов — длине вектора x.

Матрицы **X** и **Y** формируются с помощью функции:

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

Если векторы x и y одинаковы, то допускается короткий формат:

[X,Y] = meshgrid(x)

Полный список функций, используемых в трехмерной графике, выводится по команде: **help graph3d**

Основные из них с наиболее распространенными форматами приведены в табл. 4.4.

Таблица 4.4. Функции построения трехмерных графиков

Имя функции	**Назначение и формат**
plot3	Трехмерные графики в виде двумерных линий: plot3(X,Y,Z[,<параметры управления>]) где X, Y — матрицы, формирующие сетку на плоскости xOy с помощью функции meshgrid; Z — функция (вектор или матрица); <параметры управления> — необязательные параметры, управляющие свойствами графика (см. разд. 4.1.4). Квадратные скобки используются для условного обозначения необязательных параметров
mesh	Трехмерные сетчатые графики (с автоматическим нанесением координатных сеток). Формат подобен функции plot3
surf	Трехмерные сетчатые графики с окрашиванием поверхности (с автоматическим нанесением координатных сеток). Формат подобен функции plot3

Пример формирования матриц **X** и **Y** для сетки на плоскости xOy и построения трехмерных графиков с помощью функций mesh и plot3:

```
>> [X,Y] = meshgrid(-5:0.25:5);
>> Z = X.^2+Y.^2;
>> mesh(X,Y,Z)
>> figure
>> plot3(X,Y,Z),grid
```

4.1.4. Управление свойствами трехмерных графиков

Свойствами трехмерного графика можно управлять с помощью $<_{\text{параметров}}$ управления>(см. табл. 4.4), рассмотренных в разд. 4.1.2. Для управления свойствами трехмерных графиков предусмотрен ряд дополнительных средств, из которых выделим следующие два:

֍ выбор палитры цветов;

Палитра цветов задается с помощью функции: colormap('<символическое имя палитры>')

Символические имена основных палитр представлены в табл. 4.5; по умолчанию установлена палитра hsv.

Функция colormap может стоять до или после функции построения графика.

Восстановление палитры hsv выполняется с помощью функции: **colormap('default')**

֍ вывод на поле графика шкалы цветов, устанавливающей соответствие со значениями функции, выполняется по команде: colorbar которая обязательно ставится последней.

Таблица 4.5. Стандартные палитры

Символическое имя	**Палитра**
bone	Серо-синяя
cool	Фиолетово-голубая
copper	Оттенки меди
flag	Чередование: красный, белый, синий, черный
gray	Оттенки серого
hot	Чередование: черный, красный, желтый, белый
hsv	Радуга
jet	Разновидность hsv
pink	Розовая
colorcube	Расширенная палитра hsv
autumn	Красно-желтая
spring	Желто-фиолетовая

Таблица 4.5 (окончание)

Символическое имя	**Палитра**
winter	Сине-зеленая
summer	Желто-зеленая
white	Белая (бесцветная)

4.2. Содержание лабораторной работы

Содержание работы связано с изучением инструментария MATLAB для построения, оформления и управления свойствами двумерных и трехмерных графиков.

4.3. Задание на лабораторную работу

Задание на лабораторную работу включает в себя следующие пункты:

1. Построение двумерного графика.

Для аргумента x, заданного на интервале

$$x \in [0; 8\pi] \text{ с шагом } \Delta x = \pi/8 \quad (4.1)$$

вычислить функцию

$$y = \sin x \quad (4.2)$$

и вывести ее график в линейном масштабе с линейной интерполяцией между соседними значениями.

Выполнить следующие действия по оформлению графика:

- нанести координатную сетку;
- обозначить ось абсцисс.

Пояснить:

- какая функция используется для вывода графика;
- в какое окно выводится график;
- какие функции используются для нанесения координатной сетки и обозначения оси абсцисс.

2. Построение нескольких двумерных графиков на одних координатных осях.

В том же окне вывести графики функций:

$$y_2 = \frac{\sin x}{x}; \quad (4.3)$$

$$y_3 = 0{,}5 \cos x, \quad (4.4)$$

для которых аргумент x задан на интервале (4.1).

При выводе графиков выбрать различный цвет линий для функций y_1, y_2 и y_3.

Выполнить следующие действия по оформлению графика:

- обозначить ось ординат как axis y;
- ввести заголовок графика в виде Functions y1 y2 y3;
- разместить легенду для графиков функций: y1 — sin(x); y2 — sin(x)/x; y_3 — 0.5cos(x).

Пояснить:

- какая команда обеспечивает вывод нескольких графиков на одних координатных осях;
- какая функция используется для вывода графиков;
- сохраняется ли координатная сетка и обозначение оси абсцисс при выводе следующих графиков в то же окно;

- какие функции используются для обозначения оси ординат, вывода заголовка и размещения легенды.

3. Построение независимых графиков в одном окне с его разбиением на отдельные поля.

В графическом окне с именем **Graph2D** вывести друг под другом графики

функций y1 (4.2), y2 (4.3) и y3 (4.4).

Выполнить следующие действия по оформлению графиков:

- нанести координатную сетку;
- обозначить оси абсцисс и ординат;
- ввести заголовки графиков.

Пояснить:

- как создается окно с заданным именем;
- какая функция позволяет строить несколько независимых графиков в одном графическом окне.

4. Построение графика последовательности чисел.

В окне **Sequence1** вывести график значений функции y2 (4.3) с нанесением координатной сетки и без закрашивания маркеров.

В новом окне **Sequence2** вывести тот же график с нанесением координатной сетки, закрашиванием маркеров и следующей установкой параметров управ- ления:

- толщина линий равна 2;
- размер маркеров равен 6;
- цвет маркеров отличается от цвета линий;
- цвет закрашивания маркеров отличается от цветов линий и маркеров.

Пояснить:

- какая функция используется для вывода графика последовательности чисел;
- какой параметр этой функции отвечает за закрашивание маркеров;
- как устанавливаются параметры управления.

5. Построение графиков в полулогарифмическом и логарифмическом масштабах. По оси абсцисс x1 задать диапазон значений [1; 10^4] с помощью функции logspace.

Вычислить функцию

$$y_4 = \sqrt{x} \qquad 4.5)$$

В окне **Logarithmsaxes** вывести друг под другом графики функции y4 (4.5) с нанесением координатной сетки и следующих масштабах по осям:

- логарифмическом — по оси абсцисс; линейном — по оси ординат;
- логарифмическом — по осям абсцисс и ординат.

Пояснить:

- как диапазон значений задается с помощью функции logspace;

• какая функция используется для вывода графика в логарифмическом масштабе по оси абсцисс;

• какая функция используется для вывода графика в логарифмическом масштабе по осям абсцисс и ординат.

6. Построение гистограмм.

В окне **Histogram** вывести гистограмму нормального белого шума (см. табл. 2.1) — вектора y5 длиной 1000. Количество интервалов выбрать по умолчанию.

Пояснить:

• какая функция используется для построения гистограммы;

• что отображает гистограмма;

• как гистограмма связана с плотностью вероятности нормального белого шума.

7. Построение трехмерного графика.

Для аргументов x и y, заданных на одинаковых интервалах:

$x \in [-\pi; \pi]$ с шагом $\Delta x = \pi/32$,

$y \in [-\pi; \pi]$ с шагом $\Delta y = \pi/32$,

вычислить функцию

$$z = \sin x + \cos y,$$

и в окне **Graph3D** вывести ее сетчатый график с автоматическим нанесением координатных сеток.

Выполнить следующие действия по оформлению графика:

• выбрать фиолетово-голубую палитру;

• обозначить оси x, y, z;

• вывести на поле графика шкалу цветов.

Пояснить:

• с чего начинается построение трехмерного графика; какая функция для этого используется;

• какая функция используется для вывода графика;

• какая функция используется для выбора палитры;

• какая команда используется для вывода шкалы цветов.

4.4. Задание на самостоятельную работу

Самостоятельное задание рекомендуется для закрепления полученных знаний и включает в себя следующие пункты:

1С. Построение двумерного графика.

Для аргумента x, заданного на интервале $x \in [-2\pi; 2\pi]$, вывести график функции

$$y = x + \sin x,$$

с помощью функции plot, вывести заголовок и легенду, обозначить оси абсцисс и ординат.

2С. Построение двумерных графиков на одних координатных осях.

Для аргумента x, заданного на интервале $x \in [-2\pi; 2\pi]$, с помощью функции plot вывести графики функций, образующих систему уравнений:

$$\begin{cases} y = 5\sin x; \\ y = 5x + 2. \end{cases}$$

Найти решение системы (приблизительное) по графику и проверить его методом подстановки в окне **Command Window.**

Для оформления графиков, включая вывод легенды, использовать программные средства.

3С. Построение двумерных графиков в одном графическом окне на отдельных полях

Для аргумента x, заданного на интервале $x \in [-2\pi; 2\pi]$, с помощью функции plot на отдельных полях вывести графики функций:

$$y_1 = \sin x;$$
$$y_2 = sin\,|x|;$$
$$y_3 = |sin|x||.$$

4С. Построение трехмерных графиков.

Привести примеры построения трехмерных графиков с помощью функций plot3, mesh и surf в отдельных графических окнах с оформлением, включая вывод шкалы цветов.

4.5. Отчет и контрольные вопросы

Отчет составляется в редакторе MS Word и содержит результаты выполнения каждого пункта задания, включая операции для вычисления функций и построения графиков, копируемые из окна Command Window (шрифт Courier New), созданные графики (копируются по команде Edit | Copy Figure в окне Figure) и ответы на поставленные вопросы (шрифт Times New Roman).

Защита лабораторной работы проводится на основании представленного отчета и контрольных вопросов из следующего списка:

1. В какое графическое окно выводится график по умолчанию?
2. Как вывести график в новое графическое окно?
3. Как вывести несколько графиков на одних координатных осях?
4. Как удалить графики перед выводом нового графика в то же графическое окно?
5. Как вывести несколько независимых графиков в одном графическом окне с его разбиением на отдельные поля?
6. Какие средства оформления графиков используются в MATLAB?
7. Какие средства предусмотрены для установки типа, цвета и толщины линий?
8. Какие средства предусмотрены для установки вида, размера и цвета маркеров?
9. Какая функция используется для построения двумерных графиков в линейном масштабе с линейной интерполяцией между соседними значениями?
10. Какая функция используется для построения графика последовательности чисел?

11. Какие функции используются для построения графиков в полулогарифмическом и логарифмическом масштабах?

12. Какая функция используется для построения гистограмм?

13. В чем заключается подготовка перед построением трехмерного графика?

14. Какие функции используются для построения трехмерных графиков?

15. Как выбрать палитру цветов при построении трехмерного графика?

16. Как вывести шкалу цветов на поле трехмерного графика?

5. Режим программирования: script-файлы и function-файлы

Цель работы: изучить программные средства MATLAB и овладеть навыками создания файлов-сценариев (script-файлов) и внешних функций (function-файлов).

Краткая теоретическая справка

Режим программирования предназначен для создания программ пользователя в среде MATLAB.

Все программы пользователя, создаваемые в MATLAB, сохраняются на диске и имеют расширение m, поэтому их называют М-файлами.

Различают *две* разновидности М-файлов:

- script-файл (файл-сценарий);
- function-файл (файл-функция).

В М-файлах, независимо от их вида, должны соблюдаться следующие правила языка MATLAB:

- переменные не объявляются и не описываются;
- не используются метки;
- отсутствует оператор безусловного перехода типа "go to" (т. к. нет меток); нефиксируется (оператором или служебным словом) конец программы.

Script-файлы

Script-файлом называют создаваемый пользователем М-файл, представляющий собой основную (управляющую) программу.

Термины "script-файл" и "программа" употребляют в тождественном смысле.

Программа состоит из операторов, записываемых построчно. По правилам хорошего стиля программирования рекомендуется:

- в начале программы ставить оператор-заголовок:

script

- во избежание конфликта переменных в Workspace и для очистки экрана, после заголовка разместить команды clc и clear.

Имя script-файла выбирается по тем же правилам, что и имя переменной (см. разд. 1.1.2).

Пример простейшего script-файла S1:

```
script clc
% Диапазон значений аргумента
x = 0:0.1:7;
% Вычисление значений синусоиды y
y = sin(x);
% Вычисление значений косинусоиды z
z = cos(x);
% Графики синусоиды y и косинусоиды z
subplot(2,1,1), plot(x,y,'--r'), grid
subplot(2,1,2), plot(x,z), grid
```

Обращение к script-файлу в режиме прямых вычислений осуществляется по его имени:

```
>> S1
```

После этого выполняются действия согласно программе с выводом результатов в окне **Command Window.**

Все переменные script-файла являются *глобальными,* т. е. они сохраняются в Workspace и доступны для использования в любых приложениях.

5.1.2. Function-файлы

Function-файлом называют создаваемый пользователем М-файл, представляющий собой внешнюю функцию (в отличие от встроенных функций MATLAB).

Термины "function-файл" и "внешняя функция" употребляют в тождественном смысле.

Описание function-файла начинается с оператора-заголовка function. Формат описания при нескольких выходных параметрах имеет вид:

function [Y1,Y2,...] = <имя функции>(X1,X2,...)

где <имя функции> — имя function-файла, выбираемое подобно имени переменной; X1, X2, ... — список формальных входных параметров; Y1, Y2, ... — список формальных выходных параметров.

При одном выходном параметре имеем короткий формат описания:

function Y = <имяфункции>(X1,X2,...)

После заголовка следует тело функции — записанная построчно на языке MATLAB программа определения выходных параметров Y1, Y2, ... по входным —

X1, X2, ...

Пример function-файла F1:

```
function [z,p] = F1(x,y) % Вычисление суммы кубов z
z = x.^3+y.^3;
% Вычисление квадратного корня p
p = sqrt(abs(z));
```

Пример function-файла F2 с одним выходным параметром:

```
function z = F2(x,y)
```

% Вычисление суммы кубов z

z = x.^3+y.^3;

Обращение к внешней функции подобно обращению к встроенной функции MATLAB и при нескольких выходных параметрах имеет вид:

[Y1факт,Y2факт,...] = <имя функции>(X1факт,X2факт,...)

где X1факт, X2факт, ... — список фактических входных параметров;
Y1факт, Y2факт, ... — список фактических выходных параметров.

Фактические значения входных параметров X1факт, X2факт, ... должны быть определены перед обращением к внешней функции.

Примеры обращения к function-файлу F1 с несколькими выходными параметрами:

>> [d,c] = F1(2,3); >> a = 2; b = 3;

>> [d,c] = F1(a,b);

При одном выходном параметре допускается короткий формат обращения к внешней функции:

<имя функции>(X1факт,X2факт,...)

Примеры обращения к function-файлу F2 с одним выходным параметром:

>> a = 2; b = 3;

>> d = F2(a,b)+sin(7+F2(5,7));

Разделение параметров function-файлов *на формальные* и *фактические* обусловлено тем, что *формальные* параметры являются *локальными*, т. е. они (вместе с внутренними переменными function-файла) загружаются в Workspace на время вычисления внешней функции и удаляются из Workspace по завершении вычислений. *Фактические* же параметры сохраняются в Workspace.

5.1.3. Оформление и вывод листинга М-файлов

При оформлении М-файлов рекомендуется соблюдать следующие правила:

- включать комментарии, поясняющие назначение переменных, выполняемые действия и т. п.;
- во избежание выводов нежелательных промежуточных результатов ставить точку с запятой.

Вывод листинга М-файла в окне **Command Window** выполняется по команде:

type <имя М-файла>

5.1.4. Ввод/вывод данных

Ввод данных с клавиатуры организуется с помощью функции:

<имя переменной> = input('<текст>');

приостанавливающей выполнение программы для ввода данных с клавиатуры; *точка с запятой* в конце функции input блокирует автоматический вывод вводимых данных. После ввода и нажатия клавиши <Enter> автоматически продолжается выполнение программы:

>> w0 = input('w0 = ');

w0 =

С клавиатуры следует ввести значение w0 и нажать клавишу <Enter>:

w0 = pi/16

Вывод данных в окно **Command Window** организуется следующим образом.

- Вывод значений переменной или текста выполняется с помощью соответствующей функции:

disp(<имя переменной>)

или

disp('<текст>')

Для вывода значений нескольких переменных или текстов на одной строке их следует представить в виде вектора:

```
>> x = 5; a = 3; b = 10;
>> disp([x a b])
    5     3    10
>> disp(['x     ' 'a     ' 'b'])
    x     a    b
```

- Вывод символьных переменных в виде слитного текста с игнорированием в них пробелов справа или с их учетом выполняется с помощью соответствующей функции:

strcat('<текст1>','<текст2>',...)

или

strcat(['<текст1>' '<текст2>',...])

Например:

```
>> strcat('hello    ','goodbye')
 ans = hellogoodbye
>> strcat(['hello    ' 'goodbye'])
 ans = hellogoodbye
```

Для вывода значения численной переменной одновременно с текстом удобно воспользоваться функцией num2str (см. разд. 3.1.2):

```
>>i = 5;
>>strcat(['  Коэффициент ',num2str(i),'-го ВАРИАНТА'])
ans =
  Коэффициент 5-го ВАРИАНТА
```

Вывод переменной ans можно блокировать с помощью функции disp:

```
>>i = 5;
>>disp(strcat(['  Коэффициент ',num2str(i),'-гоВАРИАНТА']))
Коэффициент 5-го ВАРИАНТА
```

Подобный вывод удобно организовать в теле цикла с изменяющейся переменной цикла, о чем пойдет речь далее в разд. 6.1.2.

Функцию strcat можно использовать для вывода значения численной переменной в заголовке графика (см. табл. 4.1):

```
>>N = 3;
```

>>title(strcat(['AmplitudeSpectrumN = ',num2str(N)]))

иливобозначенииосейграфика:

>> i = 7;

>> ylabel(strcat('S',num2str(i),'(f)'))

В М-файлах функция disp используется при выводе комментариев и сообщений:

>> disp('% Введите ИСХОДНЫЕ ДАННЫЕ')

% Введите ИСХОДНЫЕ ДАННЫЕ

>> disp('% Вывод значения СКО')

% Вывод значения СКО

5.1.5. Пауза и досрочное прерывание программы

Приостановить процесс выполнения программы на неопределенное (до нажатия любой клавиши) время можно по команде:

pause

В режиме программирования команду pause необходимо ставить в тех случаях, когда в процессе выполнения программы последовательно выводятся разные графики в текущее графическое окно; в противном случае пользователю окажется доступным только один, последний, график. Команда pause ставится перед выводом следующего графика.

В том случае, если пользователь не предполагает следить за выполнением программы, и его интересует только результат, можно выводить графики в разные графические окна по команде figure без пауз.

Командой pause удобно воспользоваться в script- или function-файле перед выводом результатов, которому предшествует сообщение:

V = var(randn(1,1000)); disp('% Для вывода ДИСПЕРСИИ ШУМА нажмите <ENTER>')

pause

disp([' V = ' num2str(V)])

в том числе при выводе графиков:

x = 0:pi/32:2*pi; y = sin(x);

disp('% Для вывода ГРАФИКА СИНУСОИДЫ нажмите <ENTER>') pause

plot(x,y), grid

Досрочное прерывание процесса выполнения программы в результате проверки тех или иных условий выполняется по команде:

return

Рекомендуется предусмотреть вывод сообщения о причине досрочного прерывания.

Для принудительного снятия script-файла с выполнения следует нажать комбинацию клавиш <Ctrl>+<Break>.

5.1.6. Создание и хранение М-файлов

Для создания М-файла и его сохранения в папке пользователя необходимо выполнить следующие действия:

1. В окне **MATLAB** выбрать в главном меню пункт **File | New** (Файл | Новый) и определить тип создаваемого М-файла.

2. В раскрывшемся окне **Editor** (Редактор) набрать текст М-файла построчно.

3. Для сохранения М-файла выбрать в главном меню команду **File | Save as** (Сохранить как).

4. В раскрывшемся окне **Save as** выбрать требуемую папку, присвоить имя новому М-файлу (без расширения) и нажать кнопку **Save** (Сохранить).

При открытом окне редактора после внесения изменений в М-файл необходимо его сохранить перед следующим запуском. Признаком несохраненного файла является символ "*" (звездочка) при его имени в окне редактора.

Создание новой папки выполняется с помощью контекстного меню в окне **Current Folder.**

Сохранение пути к требуемой папке выполняется по команде контекстного меню **Add to Path | Selected Folders** (Добавить путь | Выбранные папки). Сохранение пути к папке позволяет в текущей сессии запускать М-файл, не открывая данную папку.

При запуске М-файлов из текущей папки путь к ней можно не сохранять.

5.2. Содержание лабораторной работы

Содержание работы связано с изучением средств MATLAB для создания файловсценариев (script-файлов) и внешних функций (function-файлов).

5.3. Задание на лабораторную работу

Задание на лабораторную работу заключается в создании script- и function-файлов и их выполнении в режиме прямых вычислений и включает в себя следующие пункты:

1. Создание script-файла.

Создать script-файл, который начинается с оператора-заголовка, после чего выполняются следующие действия:

- очистка экрана;
- очистка Workspace;
- генерирование равномерного Y_uniform и нормального Y_normal белого шума длины N, равной 1000;
- вывод в графическом окне White Uniform Noise графика равномерного белого шума Y_uniform и гистограммы (друг под другом).

График шума вывести с помощью функции plot с нанесением координатной сетки и заголовком.

Гистограмму шума вывести с заголовком; количество интервалов выбрать по умолчанию;

- вывод в графическом окне White Normal Noise аналогичных графиков для нормального белого шума Y_normal.

Сохранить script-файл с именем Noise_1.

Запустить script-файл на выполнение.

Проверить содержимое Workspace после выполнения script-файла.

Пояснить:

- что такое script-файл;
- в каком окне создается script-файл;
- какие команды используются для очистки экрана и Workspace;
- как выбирается имя script-файла;
- какое расширение имеют script-файлы;
- как сохранить script-файл;
- как обратиться к script-файлу в режиме прямых вычислений;
- где хранятся переменные script-файла в процессе и по завершении его выполнения.

2. Добавление паузы и сообщения о выводе результатов.

В созданный script-файл Noise_1 (см. п. 1) добавить:

- строки с сообщением о выводе графиков с текстом:

Для вывода графика и гистограммы РАВНОМЕРНОГО БЕЛОГО ШУМА нажмите <ENTER> Для вывода графика и гистограммы НОРМАЛЬНОГО БЕЛОГО ШУМА нажмите <ENTER>

- паузу перед выводом каждого из графиков.

Пояснить, какие средства MATLAB для этого используются.

Сохранить script-файл с именем Noise_2.

Запустить script-файл на выполнение.

3. Ввод данных с клавиатуры.

В созданном script-файле Noise_2 (см. п. 2) организовать ввод длины шума N с клавиатуры с сообщением о вводе.

Сохранить script-файл с именем Noise_3.

Запустить script-файл на выполнение.

Пояснить, как организуется ввод данных с клавиатуры.

4. Создание function-файла.

Создать function-файл mean_var для вычисления среднего значения MEAN и дисперсии VAR случайной последовательности Y.

В function-файл mean_var организовать вывод:

- символьной переменной 'Mean value = ' и численного значения переменной MEAN;
- символьной переменной 'Variance value = ' и численного значения переменной VAR.

Добавить в function-файл строки комментариев.

Вычислить среднее значение и дисперсию равномерного Y_uniform и нормального Y_normal белого шума длины 5000 с помощью созданного function-файла.

Проверить содержимое Workspace после выполнения function-файла.

Пояснить:

- что такое function-файл;
- каков формат function-файла;
- назначение формальных и фактических параметров function-файла;
- в каком окне создается function-файл;
- как сохранить function-файл;
- какое расширение имеют function-файлы;
- как обратиться к function-файлу для его выполнения;
- где хранятся переменные function-файла в процессе и по завершении его выполнения.

5. Использование function-файла в script-файле.

На основе script-файла Noise_3 (см. п. 3) создать новый script-файл, в котором после вывода графиков вычислить среднее значение и дисперсию равномерного Y_uniform и нормального Y_normal белого шума с помощью внешней функции mean_var.

Добавить строки с сообщением о выводе результатов с текстом:

Вывод статистических характеристик РАВНОМЕРНОГО БЕЛОГО ШУМА

Вывод статистических характеристик НОРМАЛЬНОГО БЕЛОГО ШУМА

Сохранить script-файл с именем Noise в папке пользователя My_Folder.

Запустить script-файл на выполнение.

Проверить содержимое Workspace после выполнения script-файла.

Пояснить:

- как обратиться к function-файлу из script-файла;
- как сохранить путь к собственной папке перед запуском script-файла;
- какие переменные сохраняются в Workspace после выполнения script-файла.

5.4. Задание на самостоятельную работу

Самостоятельное задание рекомендуется для закрепления полученных знаний и включает в себя следующие пункты:

1С. Создание script-файла.

Создать script-файл для решения СЛАУ

$$\boldsymbol{Ax = b}. \qquad 5.1)$$

Организовать ввод с клавиатуры матрицы **A** и вектора **b**.

Значения элементов вектора x использовать в качестве коэффициентов a_i для вычисления значения многочлена

$$y = a_N x^N + a_{N-1} x^{N-1} + \ldots + a_1 x + a_0 \qquad 5.2)$$

с помощью функции:

y = polyval(a,x)

где a — вектор коэффициентов многочлена (5.2) в порядке убывания степеней;

x и y — значения аргумента и многочлена (скаляры, векторы или матрицы).

Организовать ввод с клавиатуры значения (значений) аргумента x.

Перед вводом исходных данных и выводом результатов добавить строки соответствующих сообщений.

2С. Создание script-файла.

Создать script-файл для генерации "магической" матрицы M с помощью функции:

M = magic(n)

где n и M — порядок и имя матрицы.

Организовать ввод с клавиатуры порядка матрицы.

Вычислить определитель матрицы.

Вычислить суммы элементов столбцов, строк и главной диагонали матрицы.

Растянуть матрицу в вектор-столбец, выполнить сортировку его элементов по возрастанию и вычислить сумму элементов, деленную на порядок матрицы.

Перед вводом порядка матрицы и выводом результатов добавить строки соответствующих сообщений.

3С. Создание function-файла.

Создать function-файл для генерации двух матриц одинакового порядка:

- теплицевой матрицы **T** с произвольными целыми значениями элементов первого столбца;
- "магической" матрицы **M**.

В качестве входных параметров выбрать порядок матриц и элементы первого столбца теплицевой матрицы.

4С. Создание script-файла с использованием function-файла.

Создать script-файл для решения СЛАУ (5.1).

В качестве матрицы коэффициентов A использовать теплицеву матрицу **T**, а свободных членов **b** — "магическую" матрицу **M**.

Для генерации матриц использовать function-файл, созданный в п. 3С.

Вычислить определитель матрицы коэффициентов.

Определить количество одновременно решаемых СЛАУ.

Перед выводом результатов добавить строки соответствующих сообщений.

5.5. Отчет и контрольные вопросы

Отчет составляется в редакторе MS Word и содержит результаты выполнения каждого пункта задания, включая листинги M-файлов (шрифт Courier New), результаты их выполнения, копируемые из окна **Command Window** (шрифт Courier New), созданные графики (копируются по команде

Edit | Copy Figure в окне **Figure**) и ответы на поставленные вопросы (шрифт Times New Roman).

При защите лабораторной работы набор контрольных вопросов формируется из следующего списка:

1. Для чего предназначен режим программирования?
2. Что такое M-файл?
3. Какие разновидности M-файлов создаются в режиме программирования?
4. Как вывести листинг M-файла?
5. Что такое script-файл и как к нему обратиться в режиме прямых вычислений?
6. Что такое function-файл и как к нему обратиться в режиме прямых вычислений и в script-файле?
7. Каков формат описания function-файла?
8. Какие переменные function-файла называют формальными и фактическими?
9. Какие переменные сохраняются в Workspace после выполнения script-файла?
10. Какие переменные сохраняются в Workspace после выполнения function-файла?
11. Какие переменные называют локальными и глобальными?
12. В каком окне создаются script- и function-файлы?
13. Как организовать ввод данных с клавиатуры в режиме программирования?
14. Как организовать вывод данных в окно Command Window в режиме программирования?
15. Как вывести на одной строке значение численной переменной одновременно с текстом?
16. В каких случаях целесообразно предусмотреть паузу?
17. Как сохранить M-файл в требуемой папке?
18. Как сохранить путь к данной папке?

6. Режим программирования: организация разветвлений и циклов

Цель работы: изучить средства организации разветвлений и циклов в MATLAB и овладеть навыками их использования при разработке M-файлов.

6.1. Краткая теоретическая справка

Для организации разветвлений и циклов в M-файлах используются операторы языка MATLAB, рассматриваемые в следующих разделах.

6.1.1. Операторы организации разветвлений

Имеются две основные разновидности разветвлений, реализуемые двумя операторами MATLAB.

Разветвление по условию выполняется с помощью оператора if, простейший формат которого с одним условием имеет вид:

if <условие>
<фрагмент>
end

где <фрагмент> — фрагмент программы.

Действие оператора: если значение <условия> истинно (выполняется), то управление передается <фрагменту>, в противном случае управление передается части программы, следующей за end.

Условие представляет собой логическое выражение — простое, с одной операцией отношения (см. табл. 1.7), или более сложное, включающее логические операции (см. табл. 1.8).

Пример использования оператора if с простым условием:

```
if i==j
 a(i,j) = 1;
end
```

и с более сложным условием:

```
if (i==j)&((i+j)>50)
 a(i,j) = 10;
 end
```

Расширенный формат оператора if с одним условием имеет вид:

if <условие><фрагмент1>
else
<фрагмент2>
end

Действие оператора: если значение <условия> истинно, то управление передается <фрагменту1>, если значение <условия> ложно, то выполняется <фрагмент2>; после этого управление передается части программы, следующей за end.

Пример использования оператора if расширенного формата:

```
if i==j
  a(i,j) = 1;
else
 a(i,j) = -1;
end
```

Формат оператора if с несколькими условиями имеет вид:

if <условие1><фрагмент1>
elseif <условие2><фрагмент2> ...
elseif <условиеN-1><фрагментN-1> ...
else
<фрагментN>
end

Действие оператора: если значение <условия1> истинно, то управление передается <фрагменту1>, если значение <условия2> истинно, то управление передается <фрагменту2> и т. д. вплоть до <условияN-1>; если значения всех условий ложно, то управление передается <фрагментуN>; после этого оно передается части программы, следующей за end.

Пример использования оператора if с несколькими условиями:

```
if i>j    a(i,j) = 1; elseif i==j    a(i,j) = -1; else    a(i,j) = 0; end
```

Разветвление в зависимости от значения выражения (арифметического, символьного или логического) выполняется с помощью оператора switch следующего формата:

switch <выражение>
case <значение1><фрагмент1>
case <значение2><фрагмент2>
...
otherwise
<фрагментN>
end

Действие оператора: в зависимости от значения <выражения> управление передается соответствующему <фрагменту>; если значение выражения не равно ни одному из указанных, то управление передается <фрагментуN> (который может отсутствовать); после этого управление передается части программы, следующей за end.

Пример использования оператора switch:

```
x = [pi/6 pi/8 pi/16];
a = input('a = ');
b = input('b = ');
switch (a+b)
case 0
y = sin(x);
case 1
y = cos(x);
otherwise
 y = tan(x)
end
```

6.1.2. Операторы организации циклов

Имеются две основные разновидности циклов, реализуемые двумя операторами MATLAB.

Арифметический цикл с заранее известным (фиксированным) числом повторений организуется с помощью оператора for одного из следующих форматов:

- с простой переменной цикла:

for <переменная> = <нач.значение>:[<шаг>:]<кон.значение>
<тело цикла>

end

где <переменная> — имя простой переменной цикла; <нач.значение>, <кон.значение>, <шаг> — соответственно начальное и конечное значения переменной цикла и шаг ее изменения; если шаг равен 1, то его можно не указывать; <тело цикла> — повторяющийся фрагмент программы.

Действие оператора: при изменении значений <переменной> от <нач.значения> до <кон.значения> с заданным <шагом> повторяется <тело цикла>, каждый раз с новым значением <переменной>; после этого управление передается части программы, следующей за end.

Пример использования оператора for с простой переменной цикла (полужирным шрифтом выделены элементы, вычисляемые в цикле):

```
x = [2 3 5];
for i = 1:3
x(i) = i^2
end
x =
     1    3    5
x =
     1    4    5
x =
     1    4    9
```

- с переменной цикла — вектором:

for <переменная> = <вектор>
<тело цикла>
end

Действие оператора: при изменении значений <переменной>, которой последовательно присваиваются значения элементов <вектора>,повторяется <тело цикла>, каждый раз с новым значением <переменной>; после этого управление передается части программы, следующей за end.

Пример использования оператора for с переменной цикла — вектором:

```
a = [-1 0 15];
for i = a
x = i+a
end
x =
     -2    -1    14
x =
     -1     0     15
x =
     14    15    30
```

с переменной цикла — матрицей:

for <переменная> = <матрица>
<тело цикла>
end

Действие оператора: при изменении значений <переменной>, которой последовательно присваиваются значения столбцов <матрицы>,повторяется <тело цикла>, каждый раз с новым значением <переменной>; после этого управление передается части программы, следующей за end.

Пример использования оператора for с переменной цикла — матрицей:

```
x = [1 2 3;4 5 6;7 8 9];
for i = a
x = 1'
end
x =
     1    4    7
x =
     2    5    8
x =
     3    6    9
```

Итерационный цикл с заранее неизвестным (не фиксированным) числом повторений организуется с помощью оператора while следующего формата:

while <условие>
<тело цикла>
end

где <условие> — логическое выражение, в котором хотя бы одна из переменных встречается в <теле цикла>.

Действие оператора: <тело цикла> повторяется до тех пор, пока <условие> истинно, после чего управление передается части программы, следующей за end.

Пример использования оператора while для вычисления суммы геометрической прогрессии

$$s = \sum_{n=0}^{\infty} (-0{,}5)^n$$

с точностью до $\varepsilon = 10^{-4}(e)$ выводом после выхода из цикла значения суммы и погрешности ее вычисления (вектор [s e]):

```
n = 0; s0 = 0; e = 100;
 while e>1e-4
s = s0+(-0.5).^n;
 e = abs(s-s0);
 s0 = s;
 n = n+1;
 end [s e]
```

ans =

0.6667 0.0001

Принудительный выход из цикла for или while реализуется оператором:

break

после которого управление передается части программы, следующей за end.

6.2. Содержание лабораторной работы

Содержание работы связано с изучением средств MATLAB для организации разветвлений и циклов при разработке script-файлов и function-файлов.

6.3. Задание на лабораторную работу

Задание на лабораторную работу включает в себя следующие пункты:

1. Организация разветвлений с одним условием.

Создать function-файл y1 для вычисления функции

$$y_1(x) = \begin{cases} a \sin bx, \text{ если } a \neq 0 \text{ и } b \neq 0; \\ (a+2)x + b, \text{иначе}, \end{cases} \quad 6.1)$$

где аргумент x задан на интервале $x\epsilon[-4;4]$ с шагом $\Delta x = 0{,}1$; a, b — произвольные вещественные константы (скаляры).

Вывести график функции y_1(x) .

Обратиться к function-файлу y1 в режиме прямых вычислений для проверки разветвления по условию в (6.1).

Пояснить:

- какой оператор использован для организации разветвления;
- какие параметры function-файла y1 являются входными и выходными.

2. Организация разветвлений с несколькими условиями.

Создать function-файл y2 для вычисления функции

$$y_2(x) = \begin{cases} a \sin bx, \text{ если } a \neq 0 \text{ и } b \neq 0; \\ (a+2)x + b, \text{если } a > -2 \text{ и } b > 0; \\ (2-a)x^2 + b, \text{иначе}, \end{cases} \quad 6.2)$$

где аргумент x и скаляры a и b определены в п. 1.

Вывести график функции $y_2(x)$.

Обратиться к function-файлу y2 в режиме прямых вычислений для проверки разветвления по условиям в (6.2).

Пояснить, какой оператор использован для организации разветвления.

3. Организация цикла с заранее известным числом повторений.

Создать function-файл Fibonacci для формирования ряда Фибоначчи — вектора F из M членов, где каждый следующий член равен сумме двух предыдущих:

$$F_i = F_{i-1} + F_{i-2} = 3{,}4, \dots, M \quad 6.3)$$

Задать начальные значения $F_1 = 0$ и $F_2 = 1$.

Обратиться к function-файлу Fibonacci в режиме прямых вычислений для вывода ряда Фибоначчи.

Пояснить:

- какой оператор использован для организации цикла;
- какие параметры function-файла Fibonacci являются входными и выходными.

4. Организация цикла с заранее неизвестным числом повторений.

Создать function-файл GeomProgression для вычисления в цикле суммы бесконечной геометрической прогрессии:

$$S = \sum_{n=0}^{\infty} q^n \qquad 6.4)$$

с заданной точностью ε и значением q, при котором выполняется условие абсолютной сходимости ряда.

После выхода из цикла вычислить точное значение суммы (6.4) по формуле:

$$S_{true} = \frac{1}{1-q}$$

и погрешность вычисления суммы:

$$\Delta S = |S - S_{true}|$$

Вывести значения S , S_{true} , ε и ΔS .

Обратиться к function-файлу GeomProgression в режиме прямых вычислений для вывода требуемых значений.

Пояснить:

- какой оператор использован для организации цикла;
- какие параметры function-файла GeomProgression являются входными и выходными;
- как сопоставить выведенные значения.

5. Организация разветвления в зависимости от значения выражения.

Создать script-файл DifferentFunctions для выполнения одного из functionфайлов: y1, y2, Fibonacci или GeomProgression, в зависимости от значения переменной variant.

В script-файле организовать:

- вывод сообщения о соответствии значения переменной variant functionфайлу;
- ввод значения переменной variant с клавиатуры;
- разветвление в зависимости от значения переменной variant с выводом сообщения об исполняемом function-файле;
- вывод сообщения для непредусмотренного значения переменной variant, не соответствующего ни одному из function-файлов.

Обратиться к script-файлу DifferentFunctions в режиме прямых вычислений для проверки требуемого разветвления.

Пояснить:

- какой оператор использован для организации разветвления;
- что проверяется при организации разветвления;
- к какому типу данных может принадлежать переменная variant.

6.4. Задание на самостоятельную работу

Самостоятельное задание рекомендуется для закрепления полученных знаний и включает в себя следующие пункты: 1С. Организация разветвления по условию.

Создать function-файл для решения квадратного уравнения

$ax^2 + bx + c = 0$

двумя способами:

- используя известную алгебраическую формулу;
- с помощью функции вычисления корней многочлена производного порядка:

$$x = roots(a)$$

где a — вектор коэффициентов в порядке убывания степеней, x — корни многочлена (вектор).

В том случае, если корни оказались комплексно сопряженными, вычислить и вывести их модуль и аргументы.

2С. Организация разветвления в зависимости от значения выражения.

Создать function-файл для вычисления значения одной из следующих функций:

$$y(x) = \begin{cases} ax - b, & (a+b) = 0{,}8; \\ ax^2 + b, & (a+b) = 2{,}5; \\ \sin bx - a, & (a+b) = -3{,}4; \\ \cos ax + b, & \text{иначе.} \end{cases}$$

Построить график функции y(x) на выбранном интервале по оси x с помощью функции plot.

Обратиться к function-файлу в режиме прямых вычислений, задавая значения a и b, при которых будут выведены графики различных функций.

3С. Организация цикла с заранее известным числом повторений.

Создать function-файл для вычисления суммы конечного ряда при $0 < |x| < 1$

$$S = \sum_{n=0}^{N-1} \frac{(-1)^n \sqrt{(n+1)x}}{n+1}$$

4С. Организация цикла с заранее неизвестным числом повторений.

Создать function-файл для вычисления суммы бесконечного ряда с заданной точностью ε при $0 < |x| < 1$

$$S = \sum_{n=0}^{\infty} \frac{(-1)^n \sqrt{(n+1)x}}{n+1}$$

Определить количество циклов, требуемое для вычисления суммы при заданной точности.

6.5. Отчет и контрольные вопросы

Отчет составляется в редакторе MS Word и содержит результаты выполнения каждого пункта задания, включая листинги М-файлов (шрифт Courier New), результаты их выполнения, копируемые из окна **Command Window** (шрифт Courier New), созданные графики (копируются по команде **Edit | Copy Figure** в окне **Figure**) и ответы на поставленные вопросы (шрифт Times New Roman).

Защита лабораторной работы проводится на основании представленного отчета и контрольных вопросов из следующего списка:

1. Поясните назначение и формат оператора if.
2. Поясните назначение и формат оператора switch.
3. Поясните назначение и формат оператора for.
4. Поясните назначение и формат оператора while.
5. Как выполнить принудительный выход из цикла? Какой части программы передается управление в этом случае?

Использованные источники:

1. Цифровая обработка сигналов и MATLAB: учеб. пособие / А. И. Солонина, Д. М. Клионский, Т. В. Меркучева, С. Н. Перов. — СПб.: БХВ-Петербург, 2013. — 512 с.
2. Воробьев С.Н. Цифровая обработка сигналов : учебник для студ. учреждений высш. проф. образования / С.Н. Воробьев. - М. : Академия, 2013. - 320 с.
3. Голубинский А.Н. Теория цифровой обработки сигналов : учеб, пособие / А.Н. Голубинский, С.В. Ролдугин, И.В. Лазарев. - Воронеж : Воронежский институт МВД России, 2009. - 132 с.

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ № 1 (часть 2) по дисциплине «Цифровая обработка сигналов» для студентов специальности 11.05.01 «Радиоэлектронные системы и комплексы» очной формы обучения

Составитель:
д. ф.-м.н. Кузьменко Р.В.

Компьютерный набор Р.В. Кузьменко

Подписано к изданию_____
Уч-изд. л._____

ФГБОУ ВО «Воронежский государственный
технический университет»
394026 Воронеж, Московский просп., 14