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

5.1. Цель работы

Изучить программные средства MATLAB и овладеть навыками разработки файлов-сценариев (script-файлов) и внешних функций (function-файлов).

5.2. Краткая теоретическая справка

Режим программирования предназначен для разработки программ пользователя в среде MATLAB.

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

Различают *две* разновидности M-файлов:

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

В M-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);
- не фиксируется (оператором или служебным словом) конец программы.

5.2.1. Script-файлы

Script-файлом называют создаваемый пользователем M-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. 1.2.1)*.

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 сохранятся в Workspace и доступны для использования в любых приложениях.

5.2.2. 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 = <имя функции>(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.

5.2.3. Оформление и вывод листинга M-файлов

При оформлении M-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*>

5.2.4. 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 <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 =
hello    goodbye
```

Для вывода значения численной переменной одновременно с текстом удобно воспользоваться функцией `num2str` *(см. разд. 3.2.2)*:

```
>> 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),'-го ВАРИАНТА']))
  Коэффициент 5-го ВАРИАНТА
```

Подобный вывод удобно организовать в теле цикла с изменяющейся переменной цикла, о чем пойдет речь далее в *разд. 6.2.2*.

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

```
>> N = 3;
>> title(strcat(['Amplitude Spectrum N = ',num2str(N)]))
```

или в *обозначении осей графика*:

```
>> 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
```

5.2.5. 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 на клавиатуре нажать комбинацию клавиш <Ctrl> + <Break>.

5.2.6. Создание и хранение М-файлов

Для создания М-файла и его сохранения в папке пользователя необходимо выполнить следующую последовательность действий:

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 позволять в текущей сессии запускать М-файл, не открывая данную папку.

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

5.3. Литература

1. Солонина А. И., Арбузов С. М. Цифровая обработка сигналов. Моделирование в MATLAB. — СПб.: БХВ-Петербург, 2008, *гл. 7*.
2. Сергиенко А. Б. Цифровая обработка сигналов. 3-е издание — СПб.: БХВ-Петербург, 2010, *Приложения 1—2*.

5.4. Содержание лабораторной работы

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

5.5. 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-файл;
   - как сохранить путь к собственной папке перед запуском script-файла;
   - какие переменные сохраняются в Workspace после выполнения script-файла.

5.6. 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

$$AX = B. \tag{5.1}$$

Организовать ввод с клавиатуры матрицы A и вектора B.

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

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

с помощью функции:

```
y = polyval(a,x)
```

где `a` — вектор коэффициентов многочлена (5.2) в порядке убывания степеней, а `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 стоки соответствующих сообщений.

2С. Создание script-файла.

Создать script-файл для генерации "магической" матрицы 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 стоки 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.

5.7. Отчет и контрольные вопросы

Отчет составляется в редакторе 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).

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

1. Для чего предназначен режим программирования?
2. Что такое М-файл?
3. Какие разновидности М-файлов создаются в режиме программирования?

4. Как вывести листинг М-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 М-файл в требуемой папке?
18. Как сохранить путь к данной папке?