ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ
ЛЕКЦИЯ №2
«Работа с данными в Matlab»

Отделение ядерно-топливного цикла

Лектор:
Зав. каф. - руководитель ОЯТЦ ИЯТШ
Горюнов А.Г.

2020

План лекции

2.1 Операции с массивами/матрицами в Matlab.

2.2 Основные стандартные функции Matlab, математические и статистические функции.

2.3 Импорт и экспорт данных в Matlab.

2.4 m-файлы сценариев (скриптов) и функций.

2.5 Локальные и глобальные переменные.

2.6 Анонимные функции, подфункции, вложенные функции. Использование дескрипторов и имен функций.

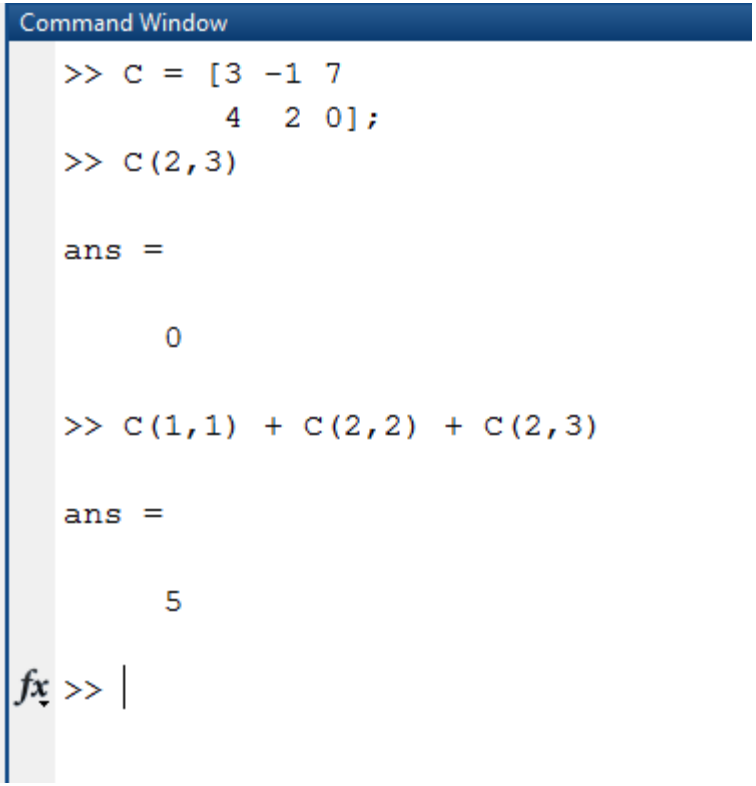
Информация по курсу:

https://portal.tpu.ru/SHARED/a/ALEX1479/study/Matmod/Tab

2.1 Операции с массивами/матрицами в Matlab.

2.1.1 Обращение к элементам матрицы

Доступ к элементам матриц осуществляется при помощи двух индексов - номеров строки и столбца, заключенных в круглые скобки



2.1.2 Сложение, вычитание, умножение, транспонирование и возведение в степень

В матричных операциях следует помнить, что для **сложения** или **вычитания** матрицы должны быть **одного размера**, а при **перемножении** число столбцов первой матрицы обязано равняться числу строк второй матрицы.

Сложение и вычитание матриц, так же как чисел и векторов, осуществляется при помощи знаков плюс и минус.

```
Command Window
>> A = [6 2 -2
        4 8  6];
>> B = [8 6 -2
        4 14 0
      -10 2  4];
>> C = [6 -2 14
        8  4  0];
>> S = A + B
Error using  +
Matrix dimensions must agree.

>> S = B + C
Error using  +
Matrix dimensions must agree.

>> S = A + C

S =

    12     0    12
    12    12     6

fx >>
```

2.1.2 Сложение, вычитание, умножение, транспонирование и возведение в степень

1. Для умножения матриц предназначена *;
2. Умножение матрицы на число также осуществляется при помощи *, умножать на число можно как справа, так и слева;
3. Транспонирование матрицы или вектора, производится при помощи .', а символ ' означает комплексное сопряжение. Для вещественных матриц эти операции приводят к одинаковым результатам.

```
Command Window
>> P = C * B

P =

  -100    36    44
    80   104   -16

>> P = A * 3

P =

    18     6    -6
    12    24    18

>> B'

ans =

     8     4   -10
     6    14     2
    -2     0     4

>> B.'

ans =

     8     4   -10
     6    14     2
    -2     0     4

fx >>
```

2.1.3 Умножение матриц и векторов

Вектор-столбец или **вектор-строка** в MatLab являются матрицами, у которых один из размеров равен единице. Поэтому все вышеописанные операции применимы и для умножения матрицы на вектор-столбец или вектор-строки на матрицу.

Command Window

```
>> a = [1 3 -2];
>>
>> B = [2 0 1
       -4 8 -1
        0 9 2];
>>
>> c = [-8
        3
        4];
>>
>> a * B * c

ans =

    74

fx >>
```

2.1.4 Удаление строк и столбцов

1. В MatLab парные квадратные скобки [ ] обозначают пустой массив, который позволяет удалять строки и столбцы матрицы. Для удаления строки следует присвоить ей пустой массив;
2. Аналогичным образом удаляются и столбцы. Для удаления нескольких идущих подряд столбцов (или строк) им нужно присвоить пустой массив.

```
Command Window
>> G = [4 0 6
        2 2 8
        6 1 3];
>>
>> G(1,:) = [];
>> G

G =

     2     2     8
     6     1     3

>> size(G)

ans =

     2     3

>> G(: , 2:3) = [];
>> G

G =

     2
     6

fx >>
```

2.1.5 Заполнение матриц при помощи индексации

Рассмотрим пример:

Сгенерируем матрицу T в три этапа:

1. Создание массива T размера пять на пять, состоящего из нулей.
2. Заполнение первой строки 1.
3. Заполнение части последней строки -1 до последнего элемента.

```
Command Window
>> T(1:5, 1:5) = 0;
>> T

T =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

>> T(1, :) = 1;
>> T

T =

     1     1     1     1     1
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

>> T(end, 3:end) = -1;
>> T

T =

     1     1     1     1     1
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0    -1    -1    -1

fx >>
```

2.1.6 Создание матриц специального вида

Заполнение прямоугольной матрицы нулями производится встроенной функцией **zeros**, аргументами которой являются число строк и столбцов матрицы.

Один аргумент функции **zeros** приводит к образованию квадратной матрицы заданного размера.

Единичная матрица инициализируется при помощи функции **eye**.

Command Window

```
>> A = zeros(3,6);
>> A

A =

     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0

>> A = zeros(3);
>> A

A =

     0     0     0
     0     0     0
     0     0     0

>> I = eye(3);
>> I

I =

     1     0     0
     0     1     0
     0     0     1
```

2.1.6 Создание матриц специального вида

Функция **eye** с двумя аргументами создает прямоугольную матрицу, у которой на главной диагонали стоят 1, а остальные 0.

Матрица, состоящая из единиц, образуется в результате вызова функции **ones**.

Использование одного аргумента в **ones** приводит к созданию квадратной матрицы, состоящей из 1.

```
Command Window
>> I = eye(4,8);
>> I

I =

     1     0     0     0     0     0     0     0
     0     1     0     0     0     0     0     0
     0     0     1     0     0     0     0     0
     0     0     0     1     0     0     0     0

>> E = ones(6,8);
>> E

E =

     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1

>> E = ones(3);
>> E

E =

     1     1     1
     1     1     1
     1     1     1

fx >>
```

2.1.6 Создание матриц специального вида

В MatLab возможно заполнения матриц случайными элементами. Результатом функции **rand** является матрица чисел, распределенных случайным образом между нулем и единицей, а функции **randn** — матрица чисел, распределенных по нормальному закону.

Один аргумент функций **rand** и **randn** приводит к формированию квадратных матриц.

Функция **diag** формирует диагональную матрицу из вектор-столбца или вектор-строки, располагая их элементы по диагонали матрицы.

Функция **diag** также служит для выделения диагонали матрицы в вектор.

Command Window

```
>> R = rand(3,4);
>> R

R =

    0.8147    0.9134    0.2785    0.9649
    0.9058    0.6324    0.5469    0.1576
    0.1270    0.0975    0.9575    0.9706

>> d = [1; 2; 3; 4];
>> D = diag(d);
>> D

D =

     1     0     0     0
     0     2     0     0
     0     0     3     0
     0     0     0     4

>> A = [10 1 2
        1 20 6
        4 5  30];
>> d = diag(A);
>> d

d =

    10
    20
    30

fx >>
```

2.1.7 Поэлементные операции с матрицами

Умножение каждого элемента одной матрицы на соответствующий элемент другой производится при помощи оператора **.***

Для деления элементов первой матрицы на соответствующие элементы второй используется оператор **./**, а для деления элементов второй матрицы на соответствующие элементы первой служит **.**

Поэлементное возведение в степень осуществляется при помощи оператора **.^** Показатель степени может быть числом или матрицей того же размера, что и матрица, возводимая в степень. Во втором случае элементы первой матрицы возводятся в степени, равные элементам второй матрицы.

```
Command Window
>> A = [2 5 -1
        3 4 9];
>> B = [-1 2 8
       7 -3 -5];
>>
>> C = A.*B;
>> C

C =

    -2    10    -8
    21   -12   -45

>> D = A./B

D =

   -2.0000    2.5000   -0.1250
    0.4286   -1.3333   -1.8000

>> E = A.\B;
>> E

E =

   -0.5000    0.4000   -8.0000
    2.3333   -0.7500   -0.5556

fx >>
```

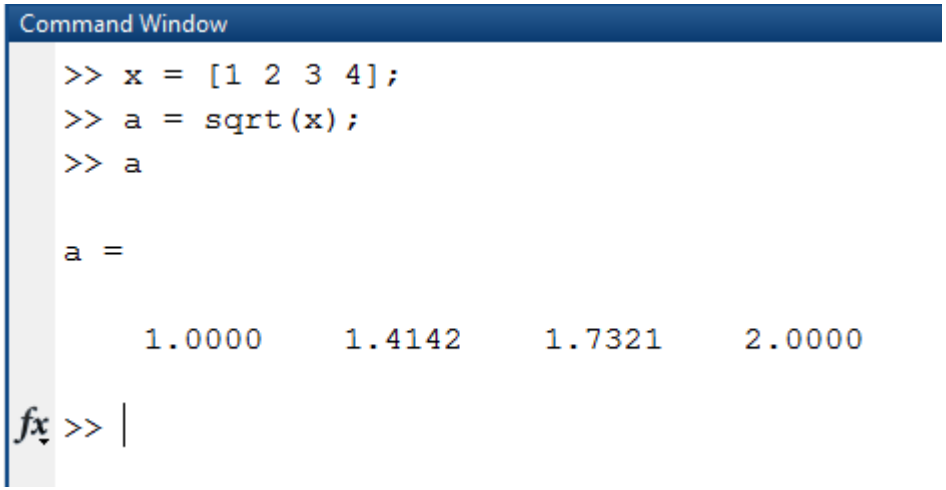
2.2 Основные стандартные функции Matlab, математические и статистические функции.

2.2.1 Основные математические функции MatLab

sqrt(x)	вычисление квадратного корня
exp(x)	возведение в степень числа e
pow2(x)	возведение в степень числа 2
log(x)	вычисление натурального логарифма
log10(x)	вычисление десятичного логарифма
log2(x)	вычисление логарифма по основанию 2
sin(x)	синус угла x, заданного в радианах
cos(x)	косинус угла x, заданного в радианах
tan(x)	тангенс угла x, заданного в радианах
cot(x)	котангенс угла x, заданного в радианах
asin(x)	арксинус
acos(x)	арккосинус
atan(x)	арктангенс
pi	число пи
round(x)	округление до ближайшего целого

2.2.1 Основные математические функции MatLab

fix(x)	усечение дробной части числа
floor(x)	округление до меньшего целого
ceil(x)	округление до большего целого
mod(x)	остаток от деления с учётом знака
sign(x)	знак числа
factor(x)	разложение числа на простые множители
isprime(x)	истинно, если число простое
rand	генерация псевдослучайного числа с равномерным законом распределения
randn	генерация псевдослучайного числа с нормальным законом распределения
abs(x)	вычисление модуля числа



2.2.2 Основные статистические функции MatLab

Законы распределения случайных величин:

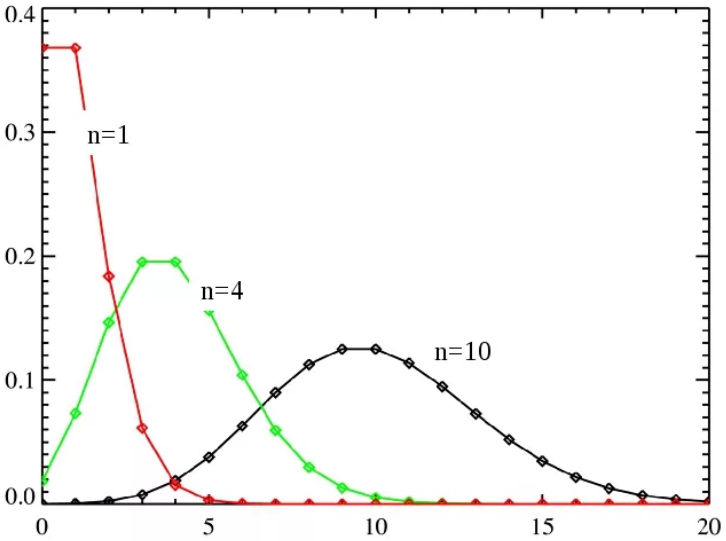
betacdf - Бета распределение
binocdf - Биномиальное распределение
cdf - Параметризованная функция распределения
chi2cdf - Функция распределения хи-квадрат
expcdf - Экспоненциальное распределение
ecdf - Эмпирическая функция распределения (на основе оценки Каплана-Мейера)
fcdf - Распределение Фишера
gamcdf - Гамма распределение
geocdf - Геометрическое распределение
hygecdf - Гипергеометрическое распределение
logncdf - Логнормальное распределение
nbincdf - Отрицательное биномиальное распределение
ncfcdf - Смещенное распределение Фишера
nctcdf - Смещенное распределение Стьюдента
ncx2cdf - Смещенное хи-квадрат распределение
normcdf - Нормальное распределение
poisscdf - Распределение Пуассона
raylcdf - Распределение Релея
tcdf - Распределение Стьюдента
unidcdf - Дискретное равномерное распределение
unifcdf - Непрерывное равномерное распределение
weibcdf - Распределение Вейбулла

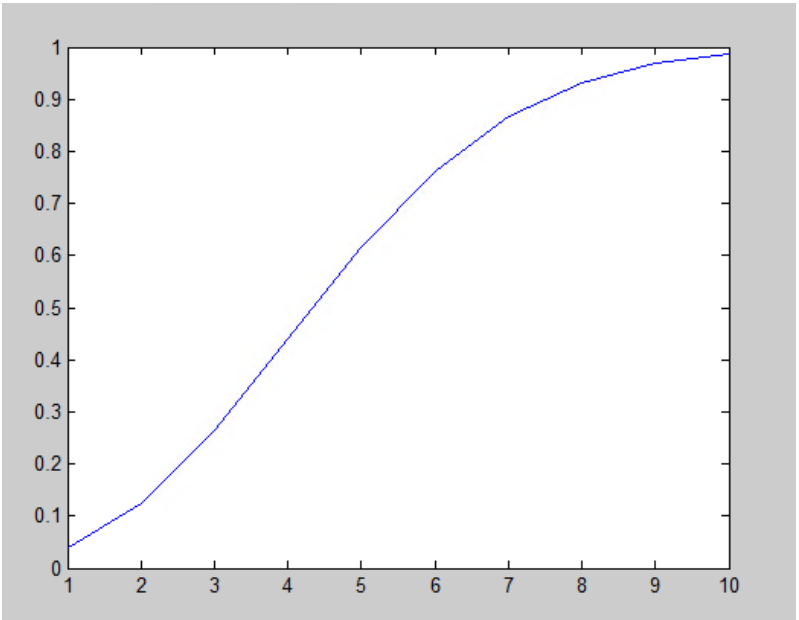
2.2.2 Основные статистические функции MatLab

Пример, распределение Пуассона:

$$p(x) = \frac{x^n}{n!}\exp(-x)$$

$0<x<\infty$, n – целочисленный параметр (n=0, 1, 2, …)





Command Window

```
>> x = 1:10,1

x =

  Columns 1 through 9

     1     2     3     4     5

  Column 10

    10


ans =

     1

>> Lamda = 5;
>> F = poisscdf(x,Lamda);
>> plot(x,F)
>>
```

2.2.2 Основные статистические функции MatLab

Оценка параметров закона распределения по экспериментальным данным

betafit - Оценка параметров бета распределения
binofit - Оценка параметров биномиального распределения
nbinfit - Оценка параметров отрицательного биномиального распределения
expfit - Оценка параметров экспоненциального распределения
gamfit - Оценка параметров гамма распределения
normfit - Оценка параметров нормального распределения
poissfit - Оценка параметров распределения Пуассона
raylfit - Оценка параметров распределения Релея
unifit - Оценка параметров равномерного распределения
weibfit - Оценка параметров распределения Вейбулла
mle - Расчет функции максимального правдоподобия

```
Command Window
>> Lambda = 1;
>> X = poissrnd(Lambda,100,1);
>>
>> Lambdahart = poissfit(X);
>> Lambdahart

Lambdahart =

    0.9500

fx >>
```

lambdahart = poissfit(X) служит для расчета точечной оценки lambdahart параметра распределения Пуассона по исходной выборке значений X. Значения оценок параметров определяются методом максимального правдоподобия.

Смотрим описание **Statistics Toolbox** пакета Matlab

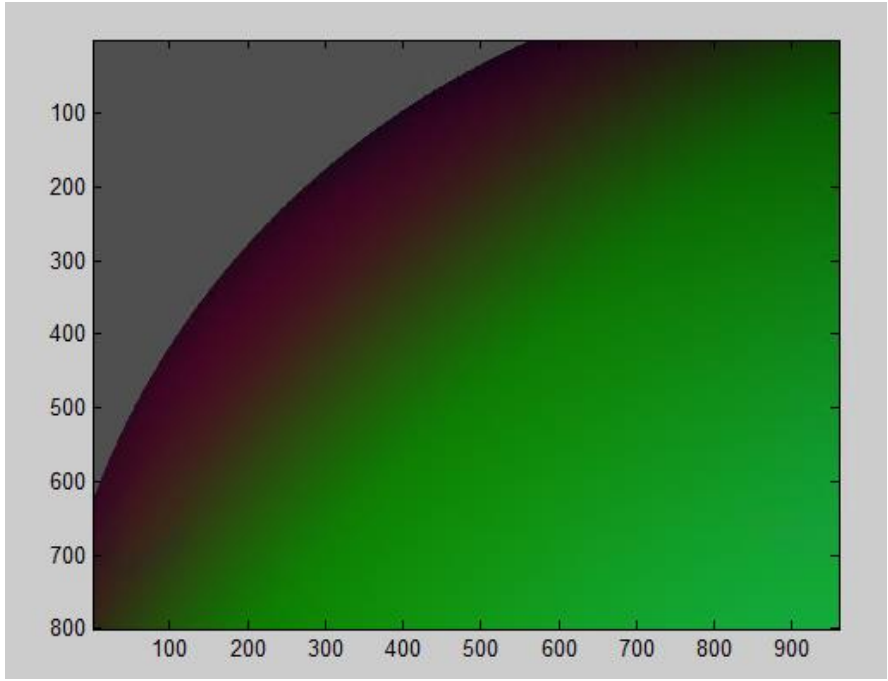
2.3 Импорт и экспорт данных в Matlab

Импорт данных в Matlab предполагает загрузку данных из внешнего файла. Функция **importdata** позволяет загружать файлы данных различных форматов.

A = importdata(имя файла)
Загружает данные в массив A из файла, обозначенного именем файла .

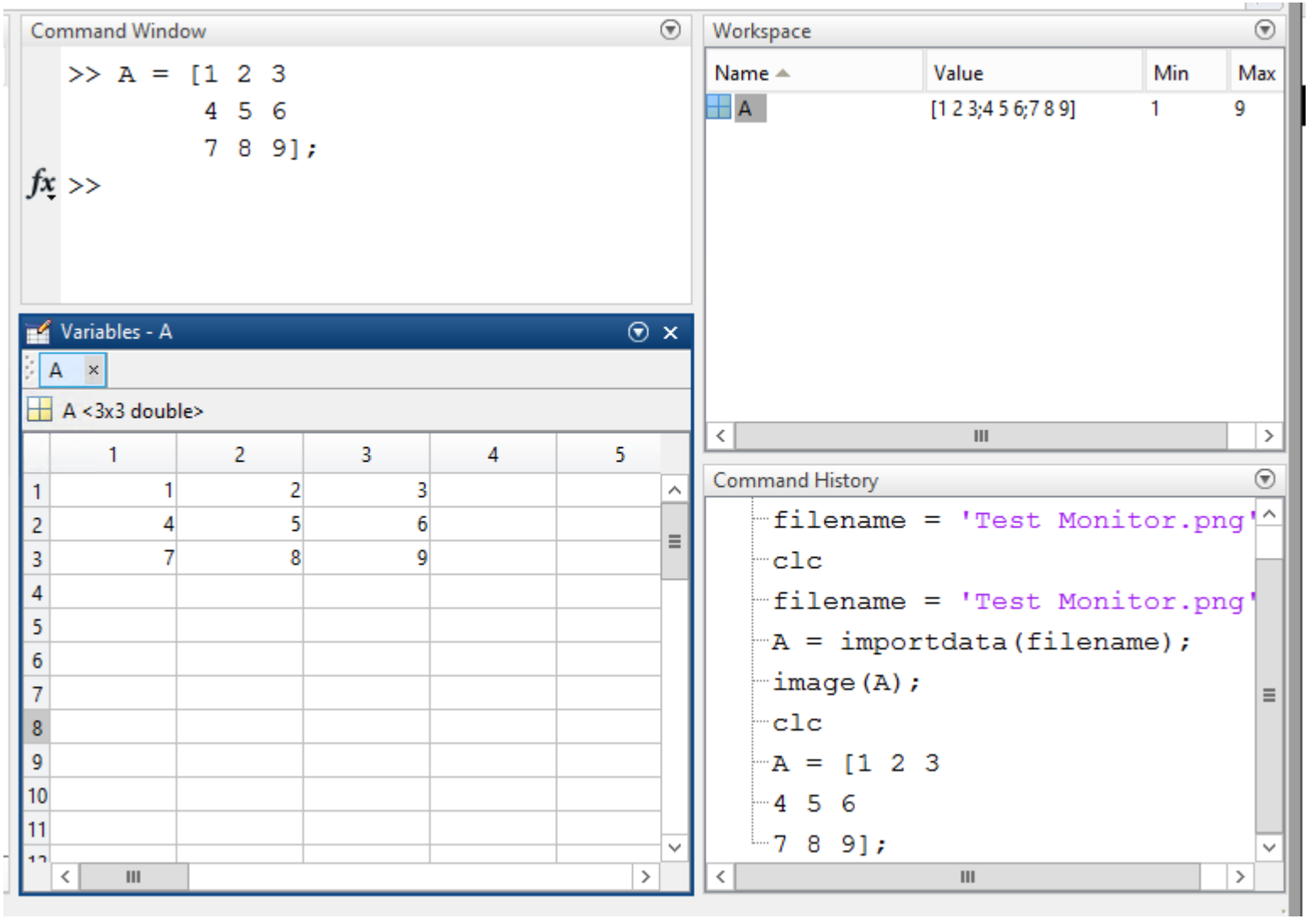
A = importdata (‘- pastespecial’)
Загружает данные из системного буфера обмена.

```
Command Window
>> filename = 'Test Monitor.png';
>> A = importdata(filename);
>> image(A);
>>
```



2.3 Импорт и экспорт данных в Matlab

Редактирование векторов и матриц возможно в табличной форме, в том числе возможно копирование данных через буфер обмена из других приложений (например Excel).



2.3 Импорт и экспорт данных в Matlab

функция	Описание
fclose	Закройте один или все открытые файлы
feof	Тест на конец файла
FERROR	Информация об ошибках файлового ввода-вывода
fgetl	Чтение строки из файла, удаление символов новой строки
fgets	Читать строку из файла, сохраняя символы новой строки
Eopen	Откройте файл или получите информацию об открытых файлах
fprintf	Записать данные в текстовый файл
Fread	Чтение данных из двоичного файла
frewind	Переместить индикатор положения файла в начало открытого файла
fscanf	Читать данные из текстового файла
FSEEK	Переместить в указанную позицию в файле
ftell	Положение в открытом файле
FWRITE	Записать данные в двоичный файл

2.4 m-файлы сценариев (скриптов) и функций

2.4.1 m-файл сценариев

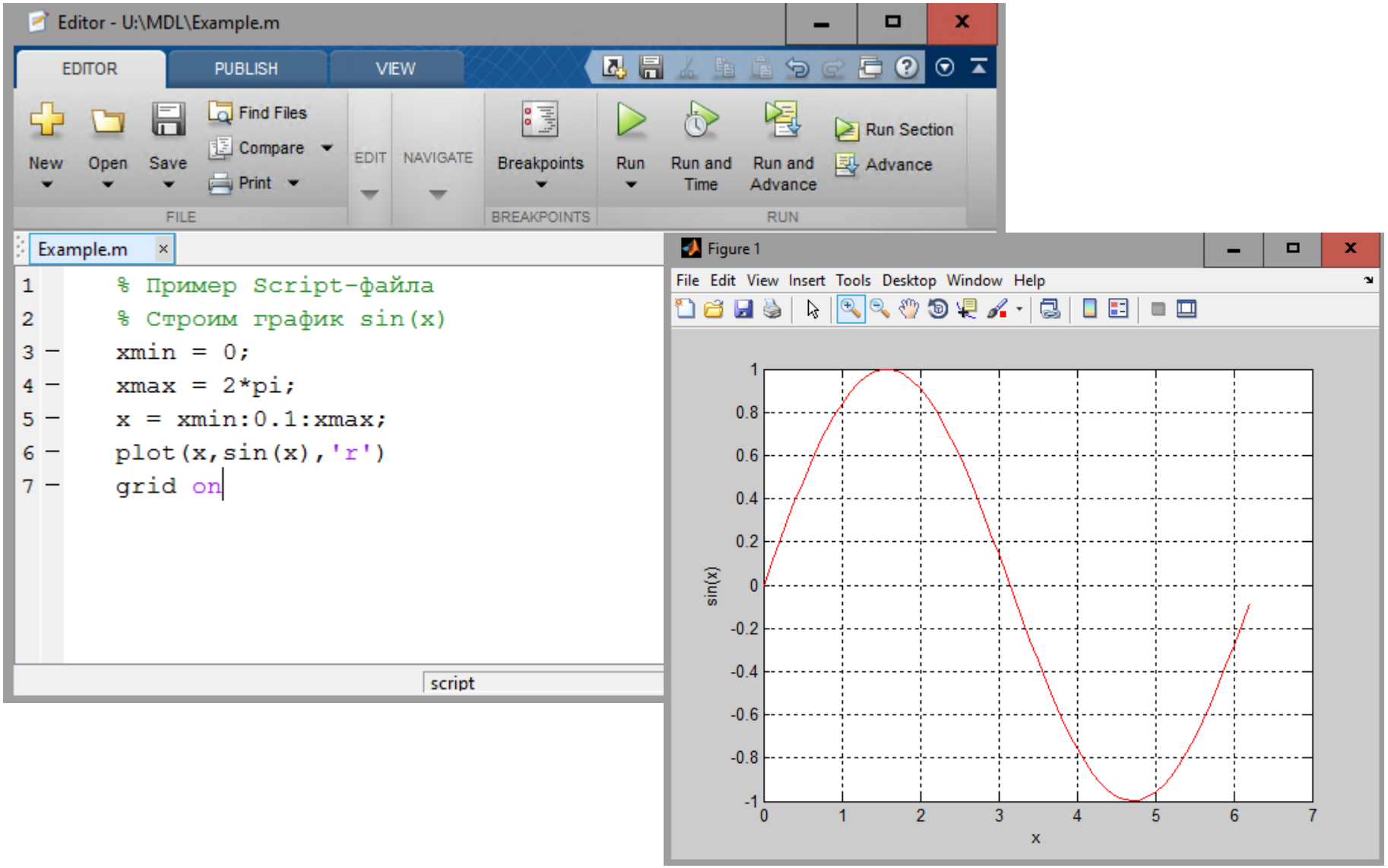
Файл-сценарий, также именуемый **Script-файлом**, является записью серии команд без входных и выходных параметров. Он имеет следующую структуру:

%Основной комментарий
%Дополнительный комментарий
Тело файла с любыми выражениями.

Важны следующие свойства файлов-сценариев:

- они не имеют входных и выходных аргументов;
- работают с данными из рабочей области;
- в процессе выполнения не компилируются;
- представляют собой зафиксированную в виде файла последовательность операций, полностью аналогичную той, что используется в сессии.

2.4.1 m-файл сценариев



2.4.2 m-файл функций

М-файл-функция является объектом языка программирования системы Matlab. Одновременно он является полноценным модулем исходя из структурного программирования, так как содержит входные и выходные параметры и использует аппарат локальных переменных. **Структура такого модуля с одним выходным параметром выглядит следующим образом**:

```
function var = f_name(список_параметров)
%Основной комментарий
%Дополнительный комментарий
Тело файла с любыми выражениями
Var = выражение
```

М-файл-функция имеет следующие свойства:

- начинается с объявления **function**, после которого указывается имя переменной **var** – выходного параметра, имя самой функции и список ее входных параметров;
- функция возвращает свое значение и может использоваться в виде **name (Список_параметров)** в математических выражениях;
- все переменные, имеющиеся в теле файла-функции, являются **локальными**, т. е. действуют только в пределах тела функции;
- файл-функция является самостоятельным программным модулем, который общается с другими модулями через свои входные и выходные параметры;
- правила вывода комментариев те же, что у файлов-сценариев;
- файл-функция служит средством расширения системы MATLAB;
- при обнаружении файла-функции он компилируется и затем исполняется, а созданные машинные коды хранятся в рабочей области системы MATLAB.

Последняя конструкция var = выражение вводится, если требуется, чтобы функция возвращала результат вычислений.

2.4.2 m-файл функций

Editor - U:\MDL\Example.m

EDITOR | PUBLISH | VIEW

New | Open | Save | Find Files | Compare | Print | EDIT | NAVIGATE | Breakpoints | Run | Run and Time | Run and Advance | Run Section | Advance

FILE | BREAKPOINTS | RUN

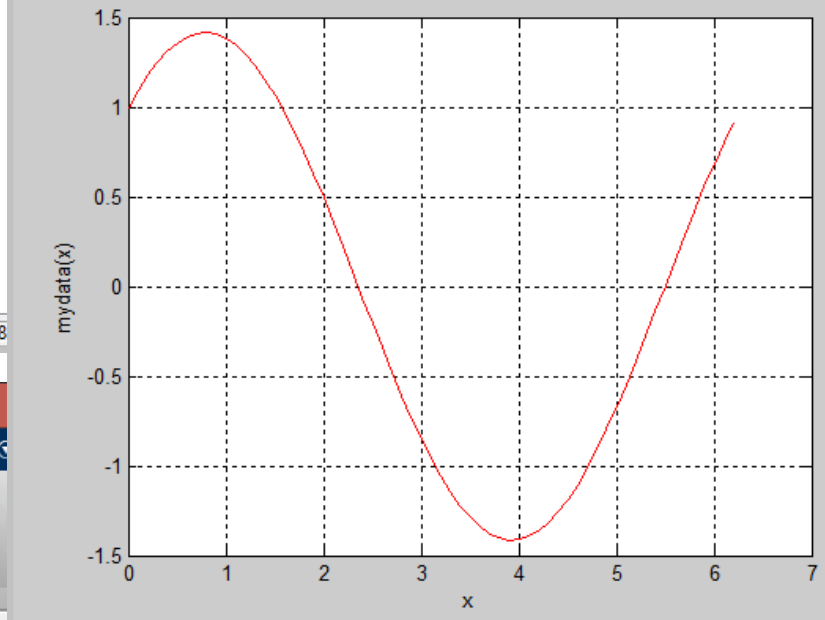
Example.m | mydata.m

```
1    % Пример Script-файла
2    % Строим график sin(x)
3 -  xmin = 0;
4 -  xmax = 2*pi;
5 -  x = xmin:0.1:xmax;
6 -  plot(x,mydata(x),'r')
7 -  grid on
8
```

script Ln 7 Col 8

Figure 1

File Edit View Insert Tools Desktop Window Help



Editor - U:\MDL\mydata.m

EDITOR | PUBLISH | VIEW

New | Open | Save | Find Files | Compare | Print | EDIT | NAVIGATE | Breakpoints | Run | Run and Time | Run and Advance | Run Section | Advance

FILE | BREAKPOINTS | RUN

Example.m | mydata.m

```
1    function [ y ] = mydata( x )
2 -  y = sin(x) + cos(x);
3 -  end
4
5
```

mydata Ln 3 Col 1

2.5 Локальные и глобальные переменные

В Matlab различает **глобальные** и **локальные** переменные. Локальные переменные действуют только в модуле, где они описаны, глобальные переменные доступны всем модулям загруженным в Matlab.

Ниже приведена иллюстрация области действия переменных.

Глобальные переменные, действуют во всех модулях		
Модуль 1	Модуль 2	Модуль 3
Локальные переменные действуют только в этом модуле	Локальные переменные действуют только в этом модуле	Локальные переменные действуют только в этом модуле

Для указания что переменная является глобальной необходимо указать перед ее именем описание **global**. Например **global X**.

Команда **who global**, выводит список всех глобальных переменных.

2.6 Анонимные функции, подфункции, вложенные функции. Использование дескрипторов и имен функций.

2.6.1 Анонимные функции

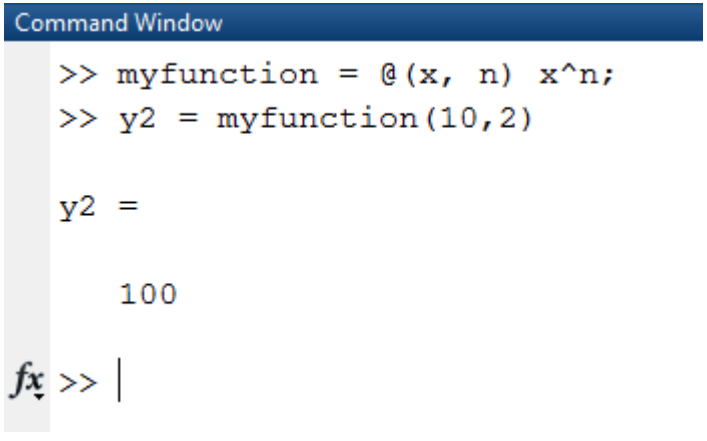
Анонимная функция аналогична встроенной функции в традиционных языках программирования, состоит из одного выражения Matlab и любого количества входных и выходных аргументов.

Анонимную функцию можно определить прямо в командной строке Matlab или в функции или в скрипте.

Таким образом, можно создавать простые функции без необходимости создавать файл для них.

Синтаксис для создания анонимной функции из выражения:

f = @(arglist)expression



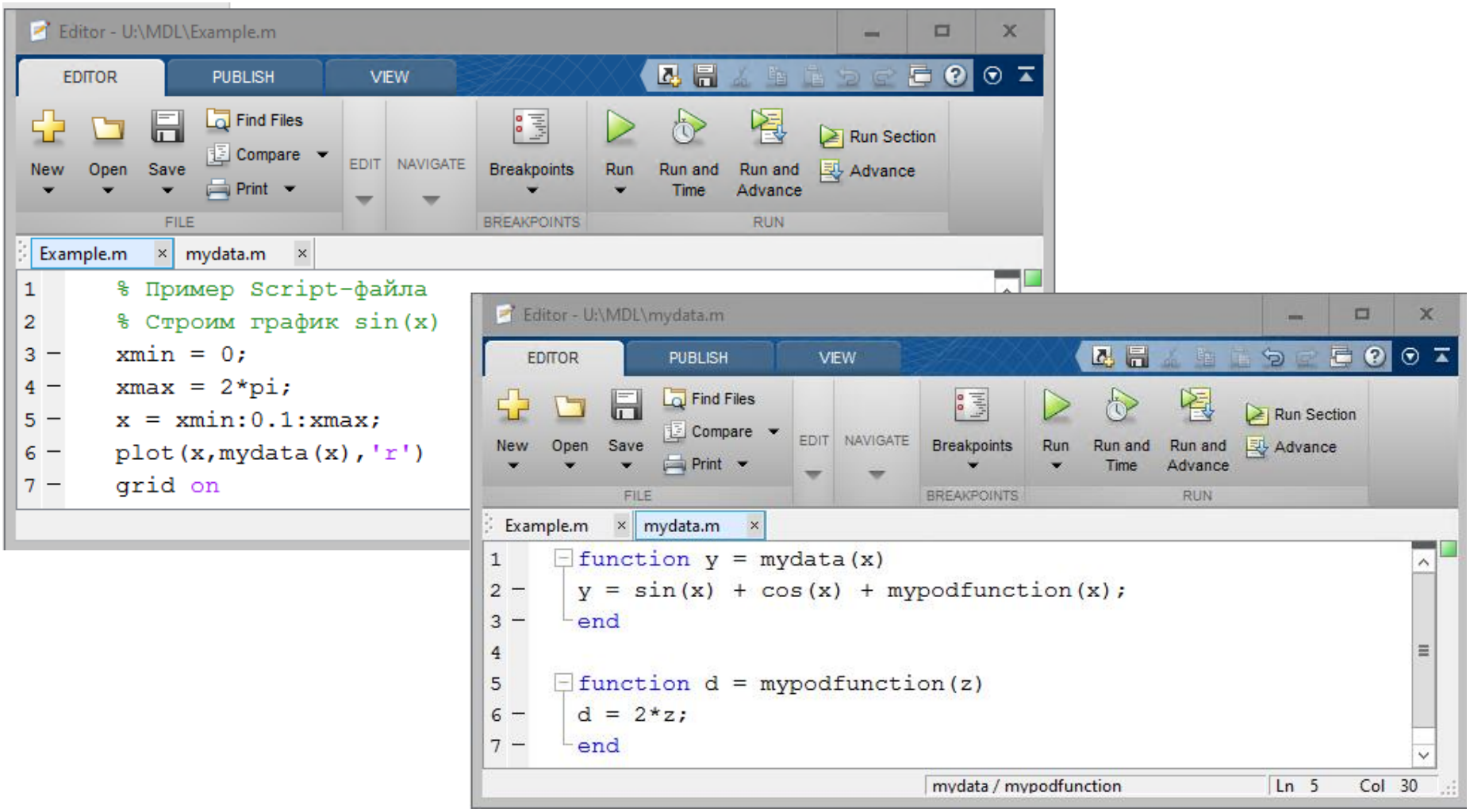
2.6.2 Основные функции и подфункции в Matlab

Любая функция, кроме анонимной, **должна быть определена в файле**. Каждый файл функции содержит требуемую первичную функцию, которая появляется первой, и любое количество необязательных подфункций, которые идут после основной функции и используются ею.

Первичные функции могут быть вызваны из-за пределов файла, который их определяет, либо из командной строки, либо из других функций, но подфункции не могут быть вызваны из командной строки или других функций вне файла функции.

Подфункции видны только основной функции и другим подфункциям в файле функций, который их определяет.

2.6.2 Основные функции и подфункции в Matlab



2.6.3 Оператор дескриптора функции

Оператор **дескриптора функции** в Matlab действует как указатель на конкретный экземпляр функции.

Например, следующая функция инициализирует count значений, а затем возвращает дескриптор функции в increment вложенной функции: