# Лабораторная работа № 2

**Матричные операции в среде MATLAB.**

**Графические построения в среде MATLAB**

**Цель работы**: знакомство с основными операциями над векторами и матрицами в MATLAB, в том числе позволяющими решать системы линейных уравнений.

## Рабочее задание

1 Присваивать значения элементам вектора в MATLAB можно поэлементно с клавиатуры, указывая их внутри квадратных скобок через пробел или запятую, в конце строк необходимо указать знак точка с запятой. Значения элементов матриц могут задаваться в виде арифметических выражений с функциями MATLAB либо в комплексной форме (рисунки 2.1, 2.2).

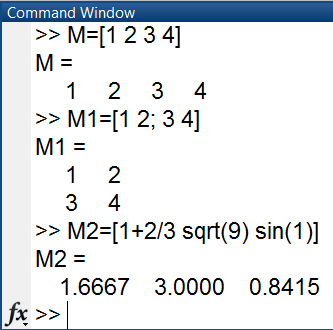
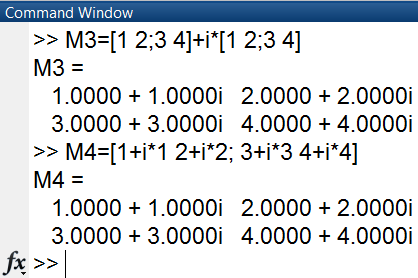


Рисунок 2.1 Рисунок 2.2

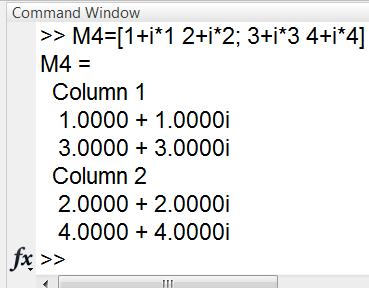
Если для отображения матрицы размера командного окна недостаточно, то она будет выведена по столбцам (рисунок 2.3, матрица **М4**).

Рисунок 2.3

**В отчет:** Ввести в командном окне значения элементов матриц **М**, **М1**, **М2**, **М3**, **М4**, указанные на рисунках 2.1, 2.2. Включить в отчет по лабораторной работе содержимое рабочего окна.

1. Для формирования упорядоченных числовых последовательностей в MATLAB используется оператор «**:**» (двоеточие) в следующем формате:

Начальное \_значение : Шаг : Конечное значение

При этом шаг может быть как положительным, так и отрицательным. Если шаг не указан, то по умолчанию значение шага принимается равным единице.

**В отчет:** Создать вектор, значения элементов которого представляют собой арифметическую прогрессию вида

*N* (*N*  1) (*N*  2) … 0,

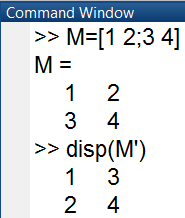
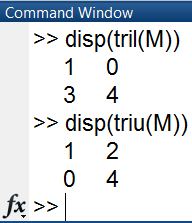
где *N* – номер варианта по указанию преподавателя. Включить в отчет по лабораторной работе содержимое рабочего окна.

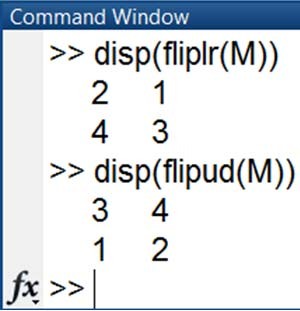
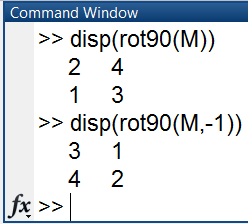
1. Для формирования векторов и матриц определенного вида в MATLAB можно использовать следующие функции, приведенные в таблице 2.1.

Таблица 2.1

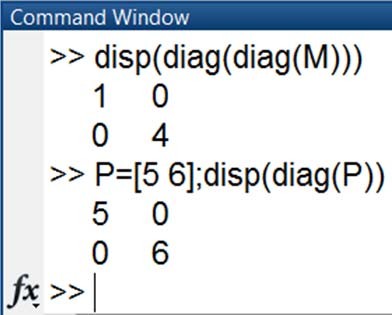
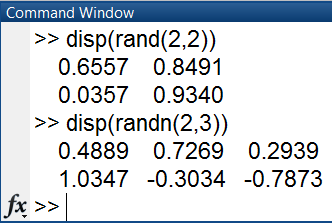
|  |  |
| --- | --- |
| Функция | Описание |
| **'** | Транспонирование |
| **tril(M)** | Создание нижней треугольной матрицы **M** |
| **triu(M)** | Создание верхней треугольной матрицы **M** |
| **fliplr(M)** | Вращение матрицы **М** относительно вертикальной оси |
| **flipud(M)** | Вращение матрицы **М** относительно горизонтальной оси |
| **rot90(M).** | Поворот матрицы **М** на 90 против часовой стрелки |
| **rot90(M,-1)** | Поворот матрицы **М** на 90 по часовой стрелке |
| **diag(diag(М))** | Создание диагональной матрицы по матрице **М** |
| **diag(Р)** | Создание диагональной матрицы из массива **Р** |
| **rand(i,j)** | Задание матрицы размерностью **i**  **j** по случайному равномерному закону |
| **randn(i,j)** | Задание матрицы размерностью **i**  **j** по случайному нормальному закону |

На рисунках 2.4,*а*-*е* приведены примеры применения некоторых функций из таблицы 2.1 для матрицы **М** и вектора-строки **Р**. Обратите внимание на команду **disp**, которая осуществляет вывод значений указанной переменной или текста в командное окно без указания имени переменной.

*а*)  *б*)

*в*) *г*)



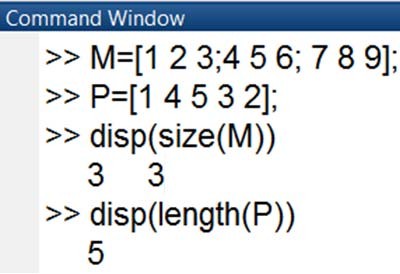
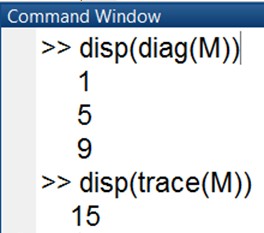
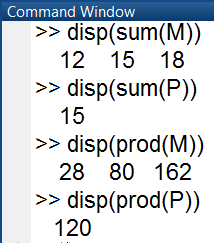
*д*) *е*)

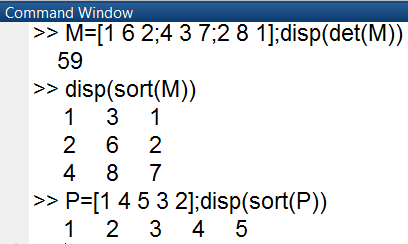
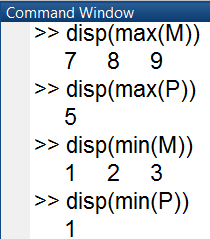
Рисунок 2.4

1. В таблице 2.2 приведены некоторые функции MATLAB, которые можно использовать для обработки векторов и матриц и получения информации о них.

Таблица 2.2

|  |  |
| --- | --- |
| Функция | Описание |
| **size(М)** | Размерность матрицы **М** |
| **length(Р)** | Длина вектора **Р** |
| **diag(М)** | Извлечение диагонали заданной матрицы **М** |
| **trace(М)** | Вычисление следа матрицы **М** (сумма элементов главной диагонали) |
| **sum(М)** | Суммирование элементов столбцов матрицы **М** |
| **sum(Р)** | Суммирование элементов вектора **Р** |
| **prod(М)** | Формирование произведения элементов столбцов матрицы **М** |
| **prod(Р)** | Формирование произведения элементов вектора **Р** |
| **max(М)** | Нахождение максимального значения матрицы **М** по столбцам |
| **max(Р)** | Нахождение максимального значения вектора **Р** |
| **min(М)** | Нахождение минимального значения матрицы **М** по столбцам |
| **min(Р)** | Нахождение минимального значения вектора **Р** |
| **det(M)** | Вычисление главного определителя матрицы **М** |
| **sort(M)** | Сортировка значений матрицы **М** по столбцам |
| **sort(Р)** | Сортировка значений вектора **Р** |

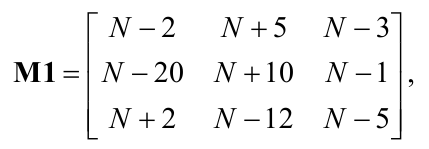
На рисунках 2.5,*а*-*д* приведены примеры применения функций из таблицы 2.2 для матрицы **М**, вектора-строки **Р**.

*а*) *б*) *в*)

*г*) *д*)

Рисунок 2.5

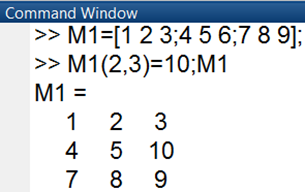
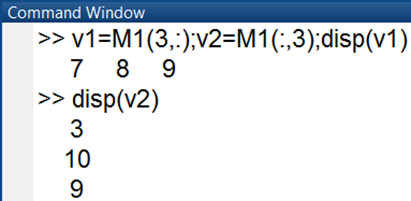
1. Создайте вектор-строку **Р**, значения элементов которой равны *N*, (*N* + 10), (*N*  20), (*N*  30), (*N* + 4), а также матрицу **М1**, значения элементов которой равны



где *N* – номер варианта по указанию преподавателя.

**В отчет:** Ввести в командном окне примеры, приведенные на рисунках 2.4 и 2.5 для созданных массивов **Р** и **М1**. Включить в отчет по лабораторной работе содержимое командного окна.

1. В системе MATLAB для обращения к любому элементу заданной матрицы **М1** необходимо указать в скобках после имени матрицы номера соответствующих ему строки и столбца через запятую. На рисунке 2.6,*а* приведен пример обращения к элементу матрицы **М1** размером 3  3, расположенному на пересечении второй строки и третьего столбца, которому присваивается значение, равное 10.

* *

*а*) *б*)

Рисунок 2.6

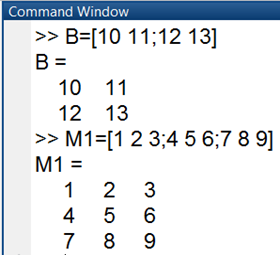
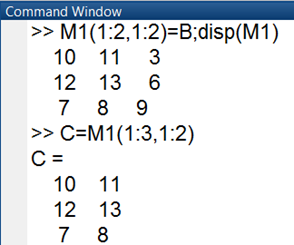
Операция обращения к конкретным строкам или столбцам матрицы выполняется в MATLAB с использованием символа двоеточие «:». На рисунке 2.6,*б* приведены примеры обращения к элементам третьей строки и третьего столбца матрицы **М1** и присвоения их значений векторам **v1** и **v2** соответственно.

**В отчет:** Сформировать в командном окне MATLAB матрицу **М2** размерностью 5  5 с помощью функции **randn**. Присвоить в зависимости от номера варианта (таблица 2.3) элементу матрицы, находящемуся на пересечении *n*-й строки и *m*-го столбца, значение, равное номеру варианта *N*; найти для матрицы **М2** минимальное значение для *n*-й строки и максимальное значение для *m*-го столбца; выполнить для четных вариантов сортировку *n*-й строки, а для нечетных вариантов – сортировку *m*-го столбца; найти главный определитель для матрицы **М2**. Включить в отчет по лабораторной работе содержимое командного окна.

Таблица 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № строки (*n*) | № варианта (*N*) | | | | |
| 1 | 1 | 2 | 3 | 4 | 5 |
| 2 | 6 | 7 | 8 | 9 | 10 |
| 3 | 11 | 12 | 13 | 14 | 15 |
| 4 | 16 | 17 | 18 | 19 | 20 |
| 5 | 21 | 22 | 23 | 24 | 25 |
|  | 1 | 2 | 3 | 4 | 5 |
| № столбца (*m*) | | | | |

1. MATLAB позволяет вставить меньшую матрицу так, чтобы она стала определенной частью матрицы большего размера и, наоборот, создать из большей матрицы вектор или другую матрицу меньшего размера. На рисунках 2.7,*а*,*б* приведены соответствующие примеры.

*а*) *б*)

Рисунок 2.7

**В отчет:** Вставить в центр созданной в п. 6 матрицы **М2** матрицу **М1**, определенную в п. 5. Включить в отчет по лабораторной работе содержимое командного окна.

1. В MATLAB могут выполняться как традиционные действия над векторами и матрицами, предусмотренные векторным вычислением в математике, так и их поэлементные преобразования.
   1. Для векторного *сложения* или *вычитания* векторов и матриц с помощью обычных знаков арифметических действий оба вектора должны быть либо вектором-столбцом, либо вектором-строкой и иметь одинаковую размерность, а обе матрицы должны иметь одинаковую размерность.

На рисунке 2.8,*а*,*б* приведены соответствующие примеры.

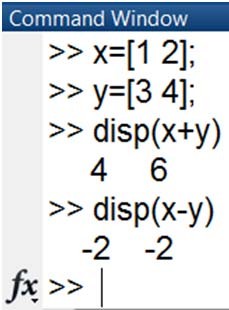
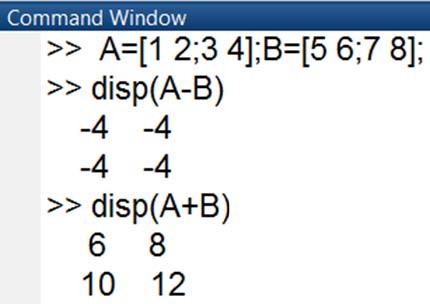
 *а*) *б*)

Рисунок 2.8

Для векторного *умножения* векторов и матриц количество столбцов первого вектора или матрицы должно совпадать с количеством строк второго вектора или матрицы. На рисунке 2.9,*а*,*б* приведены соответствующие примеры, а также пример транспонирования и обращения матрицы **В**.

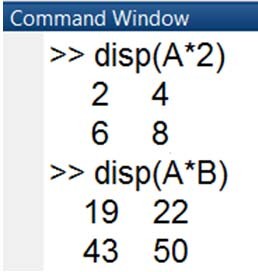
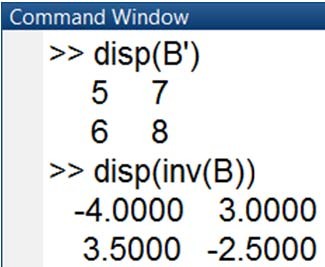
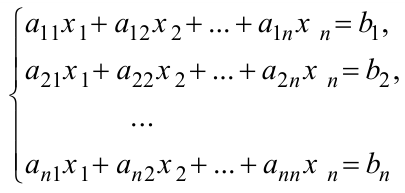
 *а*) *б*)

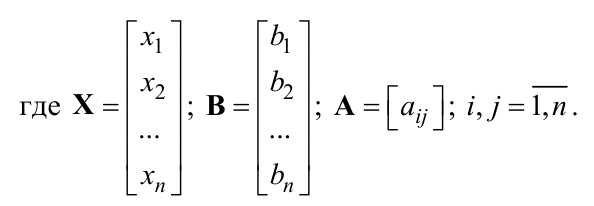
Рисунок 2.9

Обращение матрицы используется для решения системы линейных уравнений



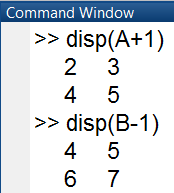
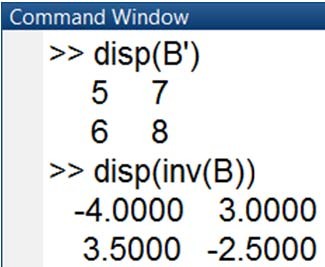
при задании ее в матричной форме как

**AX = B** (2.1)



Система (2.1) имеет единственное решение, равное **X** = **A****1B** , если ее главный определитель отличен от нуля det(**A**)  0 .. Получить обратную матрицу можно с помощью функции **inv(A)**, либо с помощью оператора **A**^1, что означает **A** в степени 1.

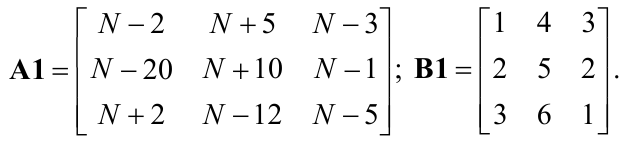
* 1. Для поэлементного преобразования векторов и матриц в MATLAB можно использовать все арифметические операции и алгебраические функции, приведенные в лабораторной работе № 1 таблицах 1.2 и 1.5 соответственно). Они формируют матрицу того же размера, что и исходная матрица, у которой каждый элемент вычисляется как значение указанной функции от соответствующего элемента заданной матрицы. На рисунке 2.10 приведены примеры поэлементного преобразования матриц. Обратите внимание, что знак операции «поэлементное сложение и вычитание» не содержит точки.



*а*) *б*)

Рисунок 2.10

Создайте два вектора-строки **x1** = (*N*, *N* + 1) и **y1** = (*N*, *N* + 2), а также матрицу **A1** и **В1**, значения элементов которых равны



**В отчет:** Ввести в командном окне примеры, приведенные на рисунках 2.8 -2.10 для массивов **х1**, **y1**, **A1** и **В1**. Включить в отчет по лабораторной работе содержимое рабочего окна.

**Графические построения в среде MATLAB**

1 Для построения в системе MATLAB графика функции одной переменной у(х) в декартовой (прямоугольной) системе координат используется команда

##### plot (X1, Y1, S1, X2, Y2, S2, …)

где **X*i***, **Y*i*** – заданные векторы одинакового размера, элементы которых являются координатами точек *i*-й кривой; **S*i*** – необязательные символьные переменные, относящиеся к *i*-й кривой. Любая из **S*i*** может содержать до трех специальных символов, определяющих тип линии, соединяющей отдельные точки графика (таблица 3.1), тип точки графика (таблица 3.2) и цвет линии (таблица 3.3).

Таблица 3.1

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Тип линии | Обозначение | Тип линии |
| **–** | Сплошная | **– .** | Штрих-пунктир |
| **:** | Двойной пунктир | **– –** | Штриховая |

Таблица 3.2

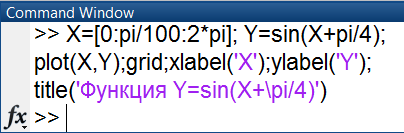
|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Тип точки | Обозначение | Тип точки |
| **.** | Точка | **V** | Треугольник (вниз) |
| **0** | Окружность | **А** | Треугольник (вверх) |
| **X** | Крест | **<** | Треугольник (влево) |
| **+** | Плюс | **>** | Треугольник (вправо) |
| **\*** | Звездочка | **Р** | Пятиугольник |
| **S** | Квадрат | **H** | Шестиугольник |
| **D** | Ромб |  |  |

Таблица 3.3

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Цвет линии | Обозначение | Цвет линии |
| **Y** | Желтый | **G** | Зеленый |
| **М** | Фиолетовый | **В** | Синий |
| **С** | Голубой | **W** | Белый |
| **R** | Красный | **К** | Черный |

В случае, если значения переменных **S** не указано, то, по умолчанию, принимаются следующие параметры графика: тип линии графика – сплошная; тип точки графика – пиксель; цвет каждого последующего графика устанавливается в следующем порядке: синий, зеленый, красный, голубой, фиолетовый, желтый, черный.

Рассмотрим пример построения функции *Y* = 10sin(*Х* + /4) для значений аргумента, изменяющихся от нуля до 2 с шагом, равным /100. Соответствующий программный код приведен на рисунке 3.1.

Рисунок 3.1

Сначала сформируем вектор значений аргумента **Х** (при этом греческую букву «» зададим как «pi»), затем вычислим вектор соответствующих значений функции **Y** и, наконец, построим график зависимости *Y*(*X*) с помощью функции **plot**.

Функция **grid** позволяет добавить на график сетку из координатных линий, названия осей задаются с помощью функции **xlabel** и **ylabel**, а заголовок графика выводится с помощью процедуры **title**. Обратите внимание на то, что для вывода в заголовке графика греческой буквы «» перед символами «**pi**» добавлена обратная косая черта «**\**». Аналогично можно вывести и другие прописные и строчные греческие буквы \alpha, \beta, \gamma, \Gamma, \Sigma, \Pi и т.д.

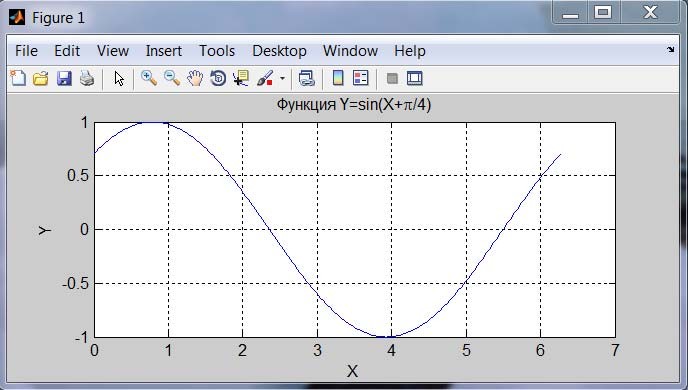
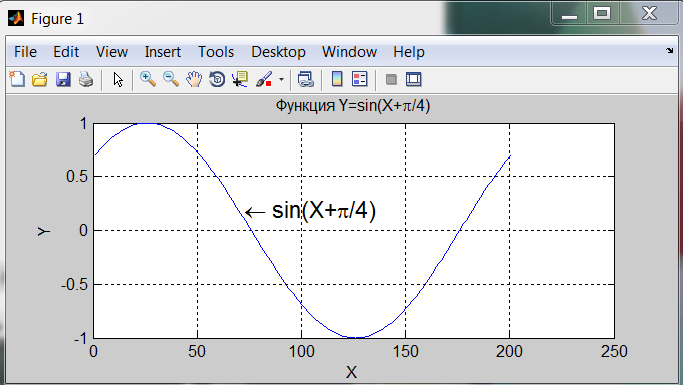
В результате на экране появится окно, в котором будет построен заданный график (рисунок 3.2).

Рисунок 3.2

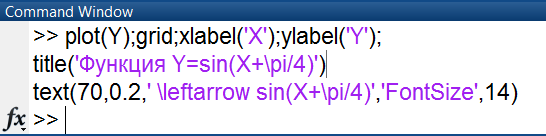
Для получения на графике гладких кривых необходимо выбрать достаточно большое количество точек на графике при формировании вектора аргумента. Если при обращении к функции **plot** вектор аргумента не указан явно, то MATLAB, по умолчанию, в качестве аргумента принимает номера элементов вектора функции (рисунок 3.3).

Рисунок 3.3

Для добавления текста в определенное место графика, например для обозначения кривой графика, используется команда text(*x*,*y*,'string')

где ***x***, ***y*** определяют координаты точки начала текста; **'string'** –строковая константа, определяющая вводимый в график текст.

На рисунке 3.3 приведен пример текста, вставленного на графике в точку с координатами 70; 0,2. Соответствующий программный код приведен на рисунке 3.4.

Рисунок 3.4

Обратите внимание на то, что строковая константа начинается с обратной косой черты «**\**» и команды **leftarrow**, которая задает стрелку, направленную влево. Аналогично можно добавить и другие виды стрелок (команды **\uparrow**, **\rightarrow** и т.д.), а также знаки «больше или равно», «меньше или равно», «не равно» (команды **\geq**,**\leq**, **\neq**).

В программе рисунок 3.4 для задания требуемого размера шрифта в пунктах используется строковая константа **'FontSize'**. В MATLAB имеется возможность задавать и требуемый тип с помощью команд **\fontname**{имя шрифта}, а также начертание шрифта с помощью следующих команд:

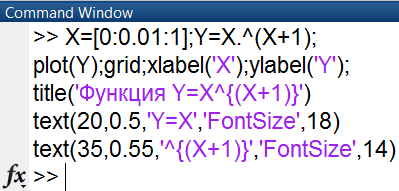
**\bf** – для жирного текста;

**\it** – для курсива;

**\rm** – для обычного текста.

MATLAB позволяет выводить с помощью команды **text** достаточно сложные математические формулы, включающие в себя индексы, дроби, корни, интегралы, суммы и т.д. Подробнее информацию о функции **text** можно посмотреть с помощью команды **doc text**.

Для задания индексов используется символ «**^**» перед верхним индексом или символ «**\_**» перед нижним индексом. Так как в программе рисунка 3.5 верхний индекс содержит больше одного символа, то он должен быть полностью заключен в фигурные скобки.

Рисунок 3.5

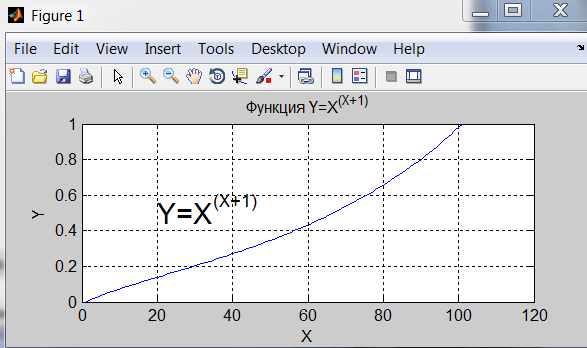
Результаты выполнения данного программного кода приведены на рисунке 3.6.

Рисунок 3.6

Координаты точки ввода текста в график удобно задавать с помощью мыши, используя команду gtext('string'). При исполнении этой команды на графике появляется большое перекрестие, установив которое в нужное место графика, достаточно нажать любую клавишу или любую кнопку мыши, и на этом месте появится надпись.

Если при неявном указании аргументов команды **plot** вектор функции **Y** содержит комплексные элементы (рисунок 3.7), то при выполнении команды **plot(Y)** в качестве аргументов функции **Y** берется вещественная часть ее элементов, а в качестве значений функции - мнимая часть ее элементов. Во всех других случаях мнимая часть данных игнорируется.

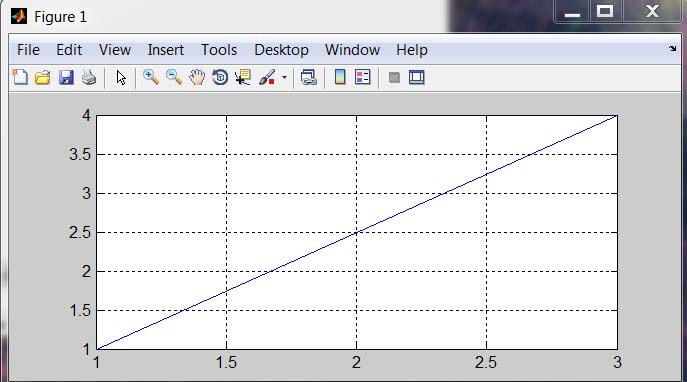
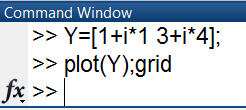
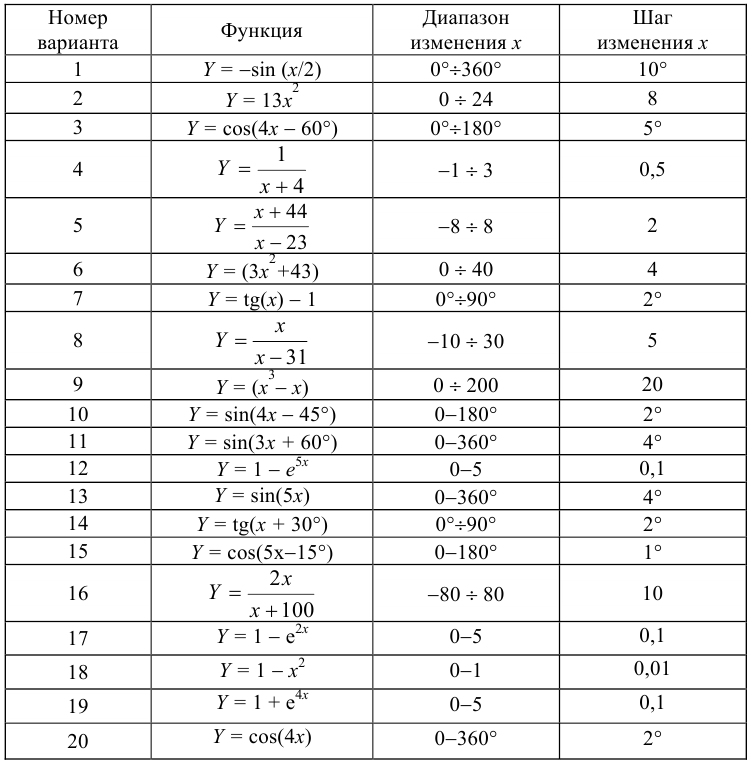
Результаты выполнения данного программного кода (см. рисунок 3.7) приведены на рисунке 3.8.

Рисунок 3.7 Рисунок 3.8

**В отчет:** Построить график функции для заданного варианта задания (таблица 3.4). Добавить к графику сетку из координатных линий, названия осей и заголовок. Включить в отчет по лабораторной работе содержимое командного окна и полученный график функции.

Таблица 3.4



2. Для сохранения и открытия графиков в MATLAB используются стандартные команды **Save As** и **Open** из меню **File** графического окна. Изучите самостоятельно эти команды и сохраните все свои графики в файле с расширением .fig.

Закройте графическое окно и снова загрузите его с помощью команды **Home****Open** (для интерфейса с меню из главного менюм **File** системы MATLAB).

3. При построении в одних осях двух графиков одного аргумента значения функции можно задать либо в виде соответствующих векторов (**Y1** и **Y2** – на рисунке 3.9), либо в виде соответствующих строк матрицы (**Y** – на рисунке 3.10).

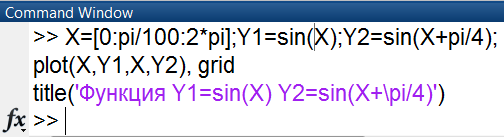


Рисунок 3.9

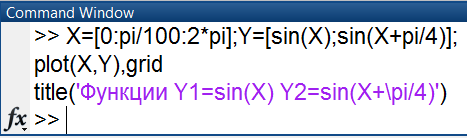


Рисунок 3.10

В результате выполнения обеих программ (см. рисунки 3.9 и 3.10) на экране появится окно с графиками двух функций в одних осях (рисунок 3.11).

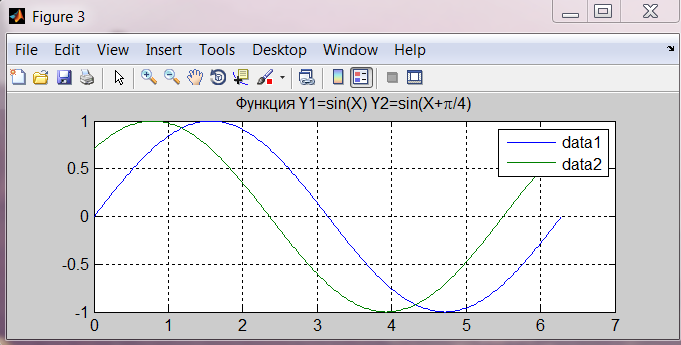


Рисунок 3.11

В правом верхнем углу графического окна добавлены пояснения в виде разноцветных отрезков линий со справочными надписями, называемые легендой. Легенду можно добавить путем нажатия на кнопку «**Insert Legend**» на панели инструментов окна графиков. Команда **legend(stringl, string2, ...)** добавляет к текущему графику легенду в виде строк, указанных в списке параметров. Для переноса легенды в другое место достаточно установить на нее курсор, нажать левую кнопку мыши и перетащить легенду в необходимую позицию.

Задайте для графиков функций **Y1** и **Y2** (рисунок 3.12) различный стиль представления: для **Y1** – тип линии сплошная, тип точки звездочка, а цвет по умолчанию синий; для **Y2** – тип линии штрихпунктир, тип точки квадрат, а цвет линии красный. В результате выполнения программного кода (см. рисунок 3.12) на экране появится окно с графиками (рисунок 3.13). Обратите внимание, что количество точек на этом графике в десять раз меньше, чем было на графике рисунка 3.3, так как был задан в десять раз больший шаг.

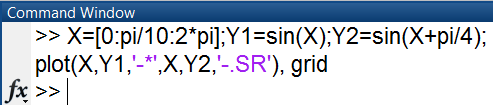
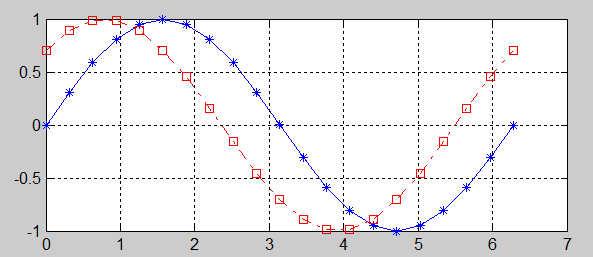


Рисунок 3.12

Рисунок 3.13

Постройте в одних осях два графика функции для одного аргумента. Значения аргумента и заданной функции первого графика взять в соответствии с вариантом задания из таблицы 3.1, а значения функции второго графика должны быть в два раза больше соответствующих значений первой функции. Задайте для графиков различный стиль, добавьте к графикам сетку из координатных линий, название оси «*Х*», легенду и заголовок.

**В отчет:** Включить в отчет по лабораторной работе содержимое командного окна и полученные графики функции.

## Контрольные вопросы

1. Каким образом в MATLAB могут задаваться значения элементов матриц?
2. Каким образом в MATLAB формируются упорядоченные матрицы?
3. Какие функции можно использовать для формирования векторов и матриц в MATLAB?
4. Какие функции MATLAB можно использовать для обработки векторов и матриц и получения информации о них?
5. Какие способы обращения к элементам, строкам и столбцам заданной матрицы существуют в системе MATLAB?
6. Как в MATLAB вставить меньшую матрицу так, чтобы она стала определенной частью матрицы большего размера и, наоборот, создать из большей матрицы вектор или другую матрицу меньшего размера?
7. Как в MATLAB могут выполняться как традиционные действия над векторами и матрицами, предусмотренные векторным вычислением в математике, так и поэлементные преобразования векторов и матриц?
8. Как в MATLAB решается система линейных уравнений, заданная в матричной форме?
9. Какие команды используются в MATLAB для построения одного и нескольких графиков в одних осях?
10. Как задаются различные стили графиков?
11. Как добавить к графикам сетку из координатных линий, названия осей, легенду и заголовок?
12. Каким образом в MATLAB можно вводить верхние и нижние индексы с помощью команды **text**?
13. Каким образом в MATLAB можно сохранить график в файле? Как можно открыть график из файла в MATLAB?