**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 1**

РАБОТА В КОМАНДНОМ ОКНЕ СИСТЕМЫ *MATLAB*

ЦЕЛЬ РАБОТЫ

Целью данного практического занятия является ознакомление с основами работы в среде *MatLAB*: изучение типов используемых данных, работа с массивами, построение графиков, типовых конструкций языка, таких как функции, условия и циклы.

ВВЕДЕНИЕ

После запуска системы *MatLAB* на экране появиться окно, показанное на рис. 1.1, предназначенное для ввода команд и вывода результатов.

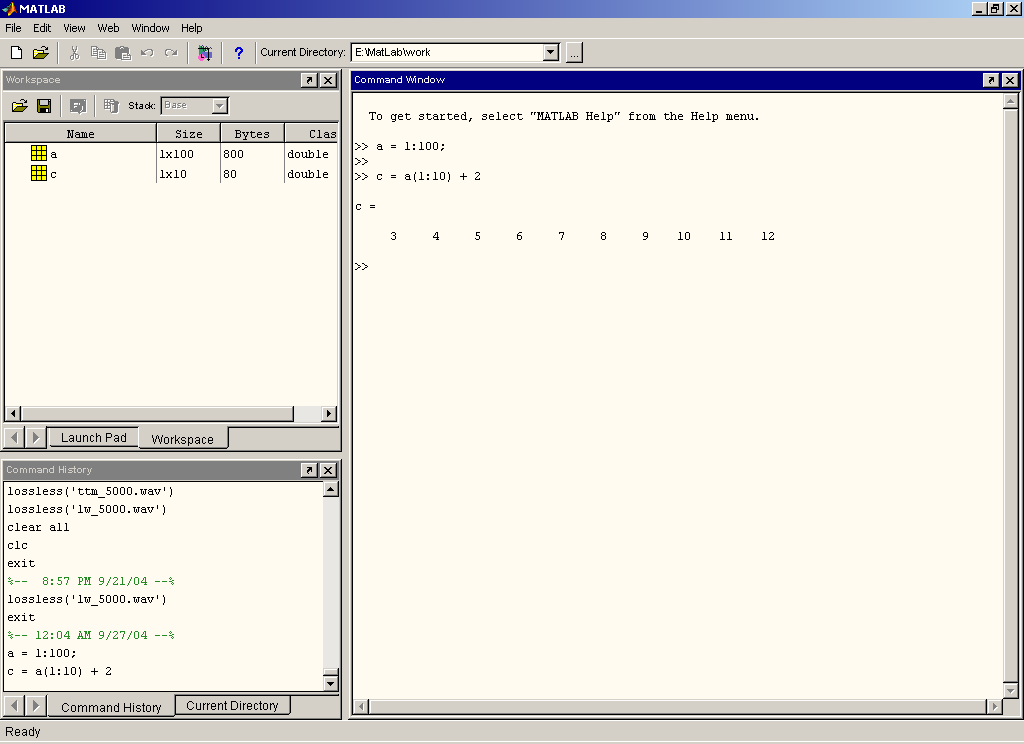


Рис. 1.1. Командное окно системы *MatLAB*

Знак «*>>*» показывает готовность системы к выполнению введенных команд. Набрав простейшие математические выражения в естественной форме записи, сразу же вычисляется результат. Это выражение может быть записано в двух видах: *<Выражение>* или *<Имя переменной> = <Выражение>*. Во втором случае результат не только вычисляется, но и присваивается указанной переменой. *MatLAB* не требует от пользователя специальных команд для объявления переменных, они создаются автоматически при первом указании пользователем их имени. В первом случае на самом деле результат выражения присваивается специальной служебной переменной имеющей имя *ans*, так же можно использовать эту переменную в расчетах. Если нет необходимости, что бы *MatLAB* выводил результаты промежуточных выражений на экран, то необходимо поставить в конце выражения символ «*;*».

**Типы данных**

Фактически *MatLAB* содержит один тип данных - массив или матрица. Массив это группа ячеек памяти, имеющие одно имя. Массивы бывают одномерные - строка или столбец, прямоугольные, квадратные (число строк рано числу столбцов). Когда Вы указываете переменную и присваиваете ей одно число, фактически *MATLAB* создает матрицу из одной строки и одного столбца.

Ниже приведены примеры столбца – а, строки – б, прямоугольной матрицы – в, квадратной матрицы – г, матрицы единичной размерности – е (простой переменой).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| а |  |  |  |  | б |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | е |
|  |  |  |  |  |  |  |  |  |  | в |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | г |  |  |  |  |

Рис. 1.2. Виды матриц

Помимо одномерных и двумерных матриц *MatLAB* поддерживает ряд других типов данных. К ним относятся многомерные массивы, строки, структуры, массивы ячеек и объекты.

**Действия над матрицами**

***Создание матриц***

Простейшей операций с матрицей является ее создание. Для создания строки необходимо указать его имя, знак равенства и в квадратных скобках через запятую или через пробел перечислить значения элементов:

>> А=[1 2 3 4 5];

В случае если необходимо создать столбец чисел, то в качестве разделителя выступает символ точка с запятой:

>> B=[1 ; 3 ; 5 ; 7];

Для создания квадратной или прямоугольной матрицы понадобится чередовать оба этих способа.

>> С=[1 2 3 ; 4 5 6 ; 7 8 9];

Для генерации векторов пользователю предоставляется следующая команда: *<Имя вектора>=<Начальное значение>:<Шаг>:<Конечное значение>*

>> Х=6 : 0.2 : 26;

В результате получиться вектор *X* следующего вида:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 6.2 | 6.4 | 6.6 | … | 25.6 | 25.8 | 26 |

В математике часто встречаются матрицы специального вида. Ниже приведен ряд из них:

Единичная матрица, рис. 1.3.а. В единичной матрице все элементы равны нулю, кроме элементов стоящих на главной диагонали (матрица является квадратной). Для создания единичной матрицы Вам необходимо подать команду *<Имя матрицы>= eye* (*<Размер>*);

>> a = eye(4);

Матрица со всеми единицами, рис. 1.3.б. Эта матрица содержит единицы во всех ячейках. Для создания матрицы необходимо указать *<Имя матрицы>=ones* (*<Кол-во строк>, <Кол-во столбцов>*)*;*

>> b = ones(3, 4);

Нулевая матрица, рис. 1.3в. Эта матрица содержит во всех своих ячейках одни нули. Для создания необходимо выполнить следующую команду: *<Имя матрицы>=zeros* (*<Кол-во строк>, <Кол-во столбцов>*)*;*

>> c =zeros(4, 2);

Случайная матрица, рис. 1.3.г. Все значения этой матрицы получаются с генератора случайных чисел. Для создания такой матрицы необходимо дать следующую команду: *<Имя матрицы>=rand* (*<Кол-во строк>, <Кол-во столбцов>*)*;*

>> d =rand(3, 4);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 |  | 1 | 1 | 1 | 1 |  | 0 | 0 |  | 5 | 6 |  |  |  |  |  |
| 0 | 1 | 0 | 0 |  | 1 | 1 | 1 | 1 |  | 0 | 0 |  | 4 | 1 |  |  |  |  |  |
| 0 | 0 | 1 | 0 |  | 1 | 1 | 1 | 1 |  | 0 | 0 |  | 0 | 3 |  |  |  |  |  |
| 0 | 0 | 0 | 1 |  |  |  |  |  |  | 0 | 0 |  |  |  |  |  |  |  |  |
|  | а |  |  |  |  | б |  |  |  |  | в |  |  | г |  |  |  |  |  |

Рис. 1.3. Специальные матрицы

***Доступ к ячейкам матрицы***

Для доступа к ячейкам матрицы необходимо указать имя матрицы, номер строки и номер столбца. Нумерация строк и столбцов ведется с ЕДИНИЦЫ! Номера пишутся в круглых скобках. Общий формат записи: *<Имя массива>* (*<Номер строки>, <Номер столбца>*)*;*

>> d(3,2)

ans =

2

>> d(3,1) = 9;

>> d

ans =

5 6

4 1

9 3

**Умножение, деление матрицы на скаляр**

В математики для всех матриц определена операция умножения (деления) матрицы на скаляр (число) – «*.\**» («./»). Все значения матрицы в этом случае умножаются (делятся) на это число.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *A*(1,1) | *A*(1,2) | *A*(1,3) |  |  |  | *A*(1,1).\**r* | *A*(1,2).\**r* | *A*(1,3).\**r* |
| *A*(2,1) | *A*(2,2) | *A*(2,3) | .\* | *r* | = | *A*(2,1).\**r* | *A*(2,2).\**r* | *A*(2,3).\**r* |
| *A*(3,1) | *A*(3,2) | *A*(3,3) |  |  |  | *A*(3,1).\**r* | *A*(3,2).\**r* | *A*(3,3).\**r* |

### *Сложение, вычитание скаляра из матрицы*

Кроме операции умножения матрицы на скаляр, для матрицы и скаляра определены операции сложение «+» и вычитания «–». Действия так же выполняются с каждой ячейкой матрицы отдельно.

### *Сложение, вычитание матриц*

Эта операция допустима только с матрицами одинакового размера. При выполнении операции действие выполняется с соответствующими друг другу ячейками. Пример:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *а*(1,1) | *а*(1,2) | *а*(1,3) |  | *b*(1,1) | *b*(1,2) | *b*(1,3) |  | *а*(1,1)+*b*(1,1) | *а*(1,2)+*b*(1,2) | *а*(1,3)+*b*(1,3) |
| *а*(2,1) | *а*(2,2) | *а*(2,3) | + | *b*(2,1) | *b*(2,2) | *b*(2,3) | = | *а*(2,1)+*b*(2,1) | *а*(2,2)+*b*(2,2) | *а*(2,3)+*b*(2,3) |
| *а*(3,1) | *а*(3,2) | *а*(3,3) |  | *b*(3,1) | *b*(3,2) | *b*(3,3) |  | *а*(3,1)+*b*(3,1) | *а*(3,2)+*b*(3,2) | *а*(3,3)+*b*(3,3) |

### *Произведение матриц*

При выполнении операции перемножения матриц выполняется последовательное умножение строки на столбец. При этом количество столбцов в первой матрице должно равняться количеству строк во второй матрице. Матрица результата будет иметь столько же строк, сколько и в первой матрице, и количество столбцов равное количеству столбцов во второй матрице.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *A*(1,1) | *A*(1,2) | *A*(1,3) |  | *B*(1,1) | *B*(1,2) |  | *A*(1,1)\**B*(1,1)+*A*(1,2)\**B*(2,1)+*A*(1,3)\**B*(3,1) | *A*(1,1)\**B*(1,2)+*A*(1,2)\**B*(2,2)+*A*(1,3)\**B*(3,2) |
| *A*(2,1) | *A*(2,2) | *A*(2,3) | \* | *B*(2,1) | *B*(2,2) | = | *A*(2,1)\**B*(1,1)+*A*(2,2)\**B*(2,1)+*A*(2,3)\**B*(3,1) | *A*(2,1)\**B*(1,2)+*A*(2,2)\**B*(2,2)+*A*(2,3)\**B*(3,2) |
|  |  |  |  | *B*(3,1) | *B*(3,1) |  |  |  |

## Построение двухмерных графиков

### *Вывод одного графика*

*MatLAB* предоставляет следующие функции для работы с графикой:

*plot*(*<Массив>*) – построение графика значений из массива *Х* от номера отсчета.

*plot*(*<Массив точек по оси Х>,<Массив точек по оси Y>*) – построение графика значений из массива *Y* от значений из массива *Х*.

При вызове команды создается окно с указанным графиком.

### *Вывод нескольких графиков*

Для вывода нескольких графиков на одном окне Вам необходимо указать их последовательно, например:

>> t=-10:0.1:10;

>> x1=sin(t);

>> x2=cos(t)./t;

>> plot(t,x1,t,x2)

или

>> t=-10:0.1:10;

>> x1=sin(t);

>> x2=cos(t)./t;

>> plot(t,x1)

>> hold on;

>> plot(t,x2)

Вывод на экран сразу нескольких графиков предоставляет простейший способ найти приблизительное значение решения.

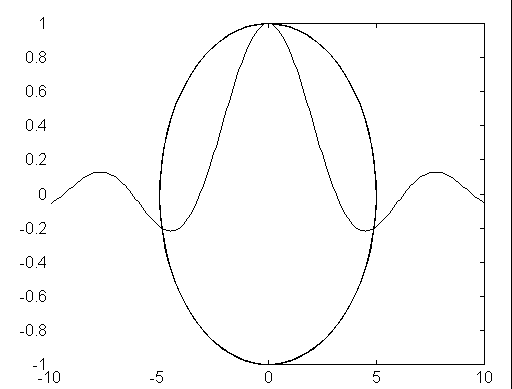


Рис. 1.4. Иллюстрация графического метода решения

На данном графике представлены графики функций *y=*sin(*t*)*/t* и (*x/*5)2*+y*2*=*1. Как не трудно заметить данные функции имеют три точки пересечения.

## Построение трехмерных графиков

Для построения трехмерных графиков, во-первых, понадобится создать сетку координат на плоскости. Выполняет это функция[*X,Y*]=*meshgrid*(*x,y*), где *x* и *y* - одномерные массивы, а *X* и *Y* - полученные в результате двухмерные массивы. Если массивы *x* и *y* одинаковые, то достаточно указать [*X,Y*]=*meshgrid*(*x*).

Например:

>> [X,Y]=meshgrid([-2:0.1:2]);

После этого описывается сама функция, например

>> Z=X.\*exp(-X.^2-Y.^2);

После этого подается команда на вывод трехмерного графика:

>> plot3(X,Y,Z);

Результат исполнения для данного примера приведен на рис. 1.5.

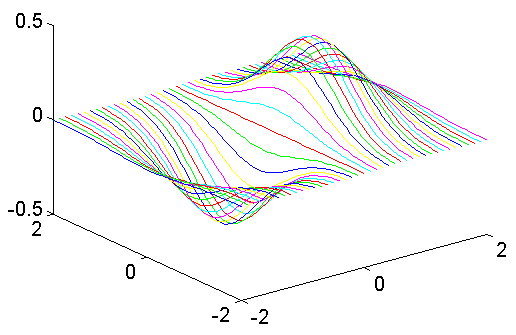


Рис. 1.5. Вывод трехмерных графиков

Помимо этой простейшей функции система *MatLAB* располагает ещё рядом функций, позволяющих добиваться большей реалистичности в изображении трехмерных графиков. Это функции *mesh, surf* и *surfl*.

К примеру, вместо ранее показанного при помощи функции *plot*3 графика, можно вызовом функции *mesh* получить следующий график (рис. 1.6).

>> mesh(X, Y, Z);

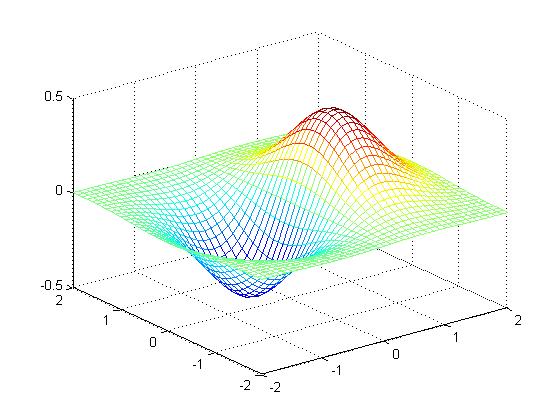


Рис. 1.6 Вывод трехмерных графиков.

## Поиск решения уравнения

Графическим методом можно лишь примерно оценить решение. Для решения уравнений или систем уравнений в среде *MATLAB* используются процедуры *fzero* и *fsolve*. Процедура *fzero* служит для нахождения нулей функции одного переменного *y = f*(*x*). Рассмотрим решение такой задачи: необходимо найти значение *x*, при котором функция *f*(*x*)*=x*2 – 5 равна 0 вблизи точки *x=*2.

>> x0=fzero('x^2-5',2)

x0 =

2.2361

Для решения систем нелинейных уравнений в среде программирования *MATLAB* используется процедура *fsolve*.

Процедура *fsolve* решает уравнения и системы уравнений следующего вида: *F*(*X*) = 0.

Рассмотрим случай определения нулей функции одного переменного:

>> x=fsolve('sin(4\*x)',1,optimset('Display','off'))

x =

0.7854

Рассмотрим подробней формат вызова *fsolve:*

* первый параметр задает функцию;
* второй, что решение будет найдено возле точки 1;
* *optimset('Display'*,'*off'*) – отключить вывод на экран сообщения о результатах оптимизации.

## Программирование в системе *MatLAB*

Программирование в системе *MatLAB* очень близко к обычному программированию. Программа создается в любом текстовом редакторе. Файл должен иметь расширение \*.*m*. Текст *М*-файла функции должен начинаться с заголовка *function*,имеющего следующий вид:

*function* [*y*1*, y*2, …] *= fname*(*x*1*, x*2, …).

Здесь *fname* – имя функции, *х*1, *х*2 и т.д. – входные параметры, *у*1, *у*2 и т.д. – выходные параметры. Входные и/или выходные параметры могут отсутствовать.

На самом деле имя функции определяется не строкой *fname*, а именем, под которым сохранен *М*-файл, оно должно совпадать с именем функции.

В качестве примера создадим функцию *myfunc*, которая будет строить трехмерных график предыдущего примера и принимать три входных параметра: точку начала построения графика, точку окончания построения, шаг. Для этого, открыв окно редактора командой меню *File -> New -> M-file*, вводим текст:

function myfunc(x1, x2, step)

[X,Y]=meshgrid([x1: step : x2]);

Z=X.\*exp(-X.^2-Y.^2);

mesh(X, Y, Z);

Введя текст, необходимо сохранить файл под именем *myfunc*.

Для того чтобы функция была доступна из системы *MatLAB*, система должна быть способна найти соответствующий *М*-файл. Поиск файлов осуществляется следующим образом: сначала просматривается текущий рабочий каталог (его имя показано в панели инструментов главного окна, рис 1.2), а затем каталоги, входящие в путь поиска (*MatLAB search path*).

Для вызова *M* файла необходимо набрать его имя в командной строке *MatLAB*, и если необходимо его аргументы. Важным элементом, облегчающим программирование, являются комментарии. Строка комментария начинается в *MatLAB* символом ‘%’.

### *Проверка условия*

Оператор проверки условия позволяет организовать разветвление исполнения программы. Внешний вид оператора представлен на рис.1.7.

Рис. 1.7. Блок-схема условного оператора: редуцированная и полная формы

Формат записи оператора, редуцированная форма:

if условие

операторы

end

полная форма:

if условие

операторы

elseif условие

операторы

else

операторы

end

### *Ввод с клавиатуры*

x=input(‘строка подсказки’)

x=input(‘строка подсказки’, ‘s’)

Функция *input* выводит на экран строку подсказки и ждет ввода переменной. Функция *x=input*(*‘строка подсказки’, ‘s’*) возвращает введенную пользователем строку. При вводе переменных допустимо пользоваться стандартными функциями.

### *Циклы*

MatLAB предоставляет пользователю два способа организации цикла. Первый из них цикл с известным количеством повторений. Блок-схема этого цикла представлена на рис. 1.8.



Рис. 1.8. Блок-схема цикла *for*

Формат оператора:

for переменная = начальное значение : [шаг :] конечное значение

операторы

end

Поле шаг в конструкции оператора не является обязательным. Например:

for i=1:100

x(i) = sin(2\*PI\*i/100);

end

Тело цикла обязательно заканчивается служебным словом *end*.

При работе с циклом *for* допустимо использование оператора прерывания цикла *break*. При выполнении оператора *break* работа цикла завершается, и управление передается на следующий после конца цикла оператор. Блок-схема программы, иллюстрирующий использование оператора *break*, приведена на рис. 1.9.

Рис. 1.9. Иллюстрация использования оператора *break* Рис. 1.10. Блок-схема цикла *while*

Текст программы соответствующий рис. 1.9.

for i=1:100

операторы

if a(i) == 0

break

end

операторы

end

Кроме цикла for в программировании на *MatLAB* используется цикл *while*. В отличие от цикла *for*в структуре цикла *while* не предусмотрены элементы для изменения переменной, по которой организован цикл. Эти элементы должен предусмотреть сам пользователь в операторах цикла. Блок-схема цикла представлена на рис. 1.10.

Формат оператора цикла с неизвестным количеством повторений:

while условие

операторы

end

Тело цикла обязательно заканчивается служебным словом *end*.

Например, программа определения точности вычислений:

a=1;

while a+1 != 1

a=a/2;

end

a

Последняя строчка программы выведет на экран значение переменной *а*.

**ЗАДАНИЕ К ПРАКТИЧЕСКОМУ ЗАНЯТИЮ**

***Найти и изобразить.***

1. Пересечение двух кривых: *y* = *x*2 – 5*x* + 3 и *y* = 7*x* – 4.

2. Точку экстремума функции *y* = (*x*+2)(*x*-4);

3. Координаты оснований перпендикуляров, опущенных из точки *B* на стороны *AC* и *A*1*С* треугольников *ABC* и *A*1*BC.* Координаты вершин: *A*(-1;-2), *A*1(2;2), *B*(-1;4), *C*(5;4).

4. Основание перпендикуляра, опущенного из точки *A*(3;10) на окружность радиуса 4, с центром в начале координат.

5. На графике функции *y =* (*5+x*)(*5-x*) точки, касательная в которых имеет угол наклона.

***Написать программу* (*m-файл*)*, которая***

6. Строит график прозрачной сферы или квадратной пирамиды по выбору пользователя. Выбор фигуры и длины запрашиваются у пользователя в процессе работы функции:

для сферы – это идентификатор и радиус – *‘сфера’*, *R*.

для пирамиды – это идентификатор, высота, длина стороны основания – *‘пирамида’, h, l.*

7. Строит график прозрачного куба или конуса. Выбор фигуры задается входными параметрами функции:

для куба – это идентификатор и длина ребра – *‘куб’, r.*

для конуса – это идентификатор, высота, радиус основания – *‘конус’, h, r.*

8. Раскрашивает черно-белый рисунок красным, зеленым или синим цветом. Цвет зависит от входного параметра: *id = 'красный'; id = 'зеленый'; id = 'синий'.*

9. Инвертирует цвета черно-белого изображения и поворачивает его на угол, кратный 90°. Угол поворота зависит от входного параметра: *id* = 90 – поворот на 90°; *id* = 180 – поворот на 180°; *id* = 270 – поворот на 270°; *id* = 0 – нет поворота.

**Пояснение к заданию 8 и 9:**

Изображения представляют собой трёхмерный массив [*x, y, z*], где *x, y* – размеры изображения, *z* – значение цвета. Черный цвет = 0, белый = 255. Для черно-белого изображения размерность *z* = 1, для цветного размерность *z* = 3.

Цвет состоит из трёх цветов: красного, зеленого, синего. Смешивание этих цветов и определяет различные оттенки.

**Содержание отчета**

1. Титульный лист
2. Результаты выполнения заданий
3. Выводы по работе[[1]](#footnote-1).

**ЛИТЕРАТУРА К ПРАКТИЧЕСКОМУ ЗАНЯТИЮ № 1**

1. Мартынов Н.Н. Иванов А.П. *MatLAB* 5.x Вычисления. Визуализация. Программирование – М.: КУДИЦ-ОБРАЗ, 2000. – 366 с.
2. Сергиенко А.Б. Цифровая обработка сигналов – СПб.: Питер, 2002. – 608 с.: ил.
3. [www.matlab.ru](http://www.matlab.ru)
4. [www.exponenta.ru](http://www.exponenta.ru)

1. Выводы должны включать ТОЛЬКО оценку результатов работы и рекомендации по использованию ценных для выполнения работы сведений. [↑](#footnote-ref-1)