Лабораторная работа №1. Программирование в MatLab

Первое знакомство с MATLAB

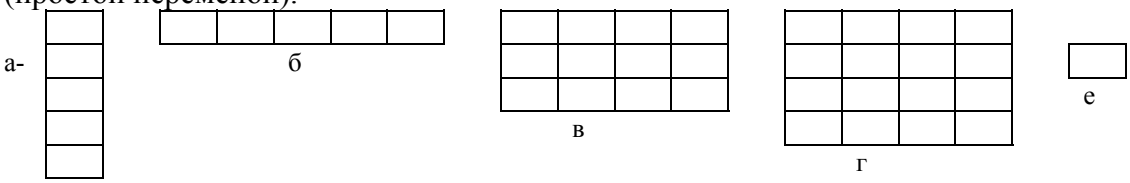
Для запуска MATLAB Вам необходимо найти на рабочем столе ярлык этой программы и запустить его на выполнение, при этом отроется рабочие окно программы. Подсказка «>>» показывает готовность системы к выполнению Ваших команд. Набрав простейшие математические выражения в естественной форме записи Вы сразу же получаете 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, Вы так же можете использовать эту переменную в расчетах. Если Вы не хотите что бы MATLAB выводил результаты промежуточных выражений на экран, то Вам необходимо поставить в конце выражения символ «;».

При наборе и редактировании команд действуют такие же команды как в любом другом оконном редакторе Windows, например в блокноте. MATLAB предоставляет пользователю возможность пролистать предыдущие команды, сделать это можно с помощью клавиш стрелка вверх и вниз. Вы легко можете внести изменения в эти команды и повторно их выполнить.

Типы данных MATLAB

Массивы

Фактически MATLAB содержит один тип данных - массив или матрица (таблица). 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 вектора - а, строки - б, прямоугольной матрицы - в, квадратной матрицы - г, матрицы единичной размерности - е (простой переменой).



Локальные и глобальные переменные

Так же как и все структурные языки программирования MATLAB различает глобальные и локальные переменные. Локальные переменные действуют только в модуле где они описаны, глобальные переменные доступны всем модулям загруженным в этот момент в MATLAB. Ниже приведена иллюстрация области действия переменных.

Глобальные переменные, действуют во всех модулях		
Модуль 1 Локальные	Модуль 2 Локальные переменные,	Модуль 3 Локальные переменные,

переменные, действуют только в этом модуле.	действуют только в этом модуле.	действуют только в этом модуле.

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

Для того чтобы узнать какие переменные есть в Вашем распоряжении Вам необходимо вызвать команду who, по которой будет выдан список всех локальных переменных. Команда who global, выводит список всех глобальных переменных созданных Вами за время этого сеанса работы с MATLAB.

Простейшие действия над матрицами

Присвоение значений матрице

Простейшей операций с матрицей является ее создание. Для создания столбца Вам необходимо указать его имя, знак равенства и в квадратных скобках через запятую или через пробел перечислить значения элементов. Например: A=[1 2 3 4 5]. В случае если Вам необходимо создать строку чисел, то в качестве разделителя выступает символ точка с запятой, например: B=[1 ; 3 ; 5 ; 7]. Для создания квадратной или прямоугольной матрицы Вам понадобится чередовать оба этих способа, например: C=[1 2 3 ; 4 5 6 ; 7 8 9].

Создание матриц специального вида

Для генерации векторов пользователю предоставляется следующая команда: <Имя вектора>=<Начальное значение>:<Шаг>:<Конечное значение>. Например: X=6 : 0. 2 : 26.

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

Единичная матрица, рис. 3.а. В единичной матрице все элементы равны нулю, кроме элементов стоящих на главной диагонали. Для создания единичной матрицы Вам необходимо подать команду <Имя матрицы>=eye(<Размер>). Матрица является квадратной.

Матрица со всеми единицами, рис. 3.б. Эта матрица содержит единицы во всех ячейках. Для создания матрицы Вам необходимо указать <Имя матрицы>=ones(<Кол-во строк>, <Кол-во столбцов>). Например: A=ones(6, 3).

Нулевая матрица, рис. 3.в. Эта матрица содержит во всех своих ячейках одни нули. Для создания Вам необходимо выполнить следующую команду: <Имя матрицы>=zeros(<Кол-во строк>, <Кол-во столбцов>). Например: A=zeros(6, 3).

Случайная матрица. Все значения этой матрицы получаются с генератора случайных чисел. Для создания такой матрицы Вам необходимо дать следующую команду: <Имя матрицы>=rand(<Кол-во строк>, <Кол-во столбцов>). Например: A=rand(6, 3).

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

а

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

б

0	0
0	0
0	0
0	0

в

Доступ к ячейкам матрицы

Для доступа к ячейкам матрицы Вам 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>).

Например A(1,2).

Простейшие действия над матрицами

Умножение матрицы на скаляр. В математики для всех матриц определена операция умножения матрицы на скаляр (число). Все значения матрицы в этом случае умножаются на это число.

r	*	A(1,1)	A(1,2)	A(1,3)	=	r*A(1,1)	r*A(1,2)	r*A(1,3)
		A(2,1)	A(2,2)	A(2,3)		r*A(2,1)	r*A(2,2)	r*A(2,3)
		A(3,1)	A(3,2)	A(3,3)		r*A(3,1)	r*A(3,2)	r*A(3,3)

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. Пример:

r	+	A(1,1)	A(1,2)	A(1,3)	=	r+A(1,1)	r+A(1,2)	r+A(1,3)
		A(2,1)	A(2,2)	A(2,3)		r+A(2,1)	r+A(2,2)	r+A(2,3)
		A(3,1)	A(3,2)	A(3,3)		r+A(3,1)	r+A(3,2)	r+A(3,3)

Сложение матриц (вычитание). Эта операция допустима только с матрицами одинакового размера. При выполнении операции действие выполняется с соответствующими друг другу ячейками. Приер:

a(1,1)	a(1,2)	a(1,3)		b(1,1)	b(1,2)	b(1,3)		a(1,1)+ b(1,1)	a(1,2)+ b(1,2)	a(1,3)+ b(1,3)
a(2,1)	a(2,2)	a(2,3)	+	b(2,1)	b(2,2)	b(2,3)	=	a(2,1)+ b(2,1)	a(2,2)+ b(2,2)	a(2,3)+ b(2,3)
a(3,1)	a(3,2)	a(3,3)		b(3,1)	b(3,2)	b(3,3)		a(3,1)+ b(3,1)	a(3,2)+ b(3,2)	a(3,3)+ b(3,3)

Произведение матриц

При выполнении операции перемножения матриц выполняется последовательное умножение строки на вектор. При этом количество столбцов в первой матрице должно равняться количеству строк во второй матрице. Матрица результата будет иметь количество 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)			

Присвоение матрице математического выражения

Организация всех переменных системы MATLAB как матрицы вызывает следующие ограничения применения их в выражениях. На пример:

```
t=1:5
t=  1 2 3 4 5
y=cos(t)
y=1 0.5 -0.4 -1 -0.6 0.2
z=y/t
z=-0.08
```

Результат - получается одно число, а ожидали функцию z(t)=cos(t)/t.

Для организации поэлементного деления одного массива на другой, в MATLAB предусмотрена специальная операция - «./» - поэлементное деление.

Результат этого выражения будет другой:

```
z=y ./ t
z= 0.5403   -0.2081   -0.3300   -0.1634    0.0567.
```

Графические средства представления результатов

Вывод одного графика

MATLAB предоставляет следующие функции для работы с графикой:

plot(<Массив>) - построение графика значений из массива X от номера отсчета.

plot(<Массив точек по оси X>,<Массив точек по оси У>) - построение графика значений из массива У от значений из массива X.

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)
```

Графический метод решения уравнений

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

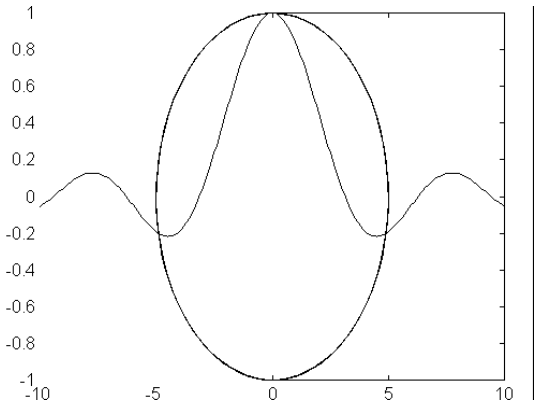


Рис. 1.1. Иллюстрация графического метода решения

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.

Поиск решения уравнения

Графическим методом можно лишь примерно оценить решение. Для более точного нахождения решения в пакете MatLAB необходимо воспользоваться функцией **fsolve(уравнение, начальное значение).** Позже мы познакомимся как с помощью данной функции решать системы уравнений. В простейшем случае решаемое уравнение можно указать можно указать в одинарных кавычках, например: **'x*x-abs(x)'.** Но данная функция имеет три решения, представленных на рис.1.2.

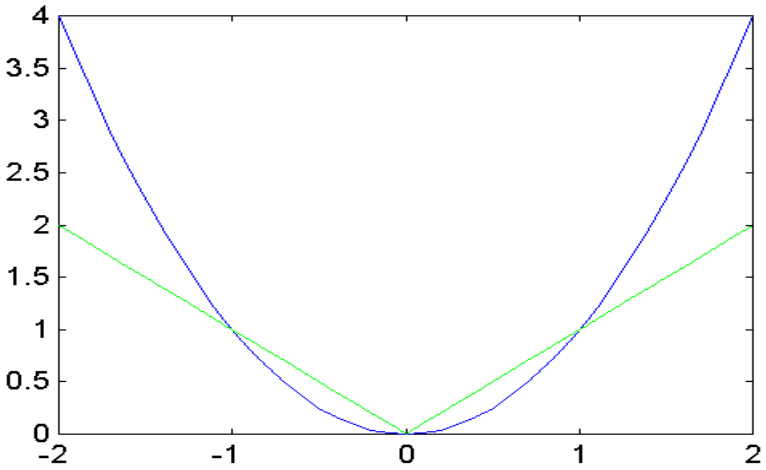


Рис.1.2. Графики функций

Решение, которое найдет в этом случае функция **fsolve** будет определяться начальным значением откуда она начнет итерационную процедуру поиска решения. Например:

fsolve('x*x-abs(x)',-2), ans = -1.0000;

fsolve('x*x-abs(x)',0.6), ans = 1.0000;

fsolve('x*x-abs(x)',0.4), ans = 7.9062e-008.

Функция **fsolve** продолжает итерационную процедуру до тех пор пока она не найдет решение с заданной точностью. По этому в нашем примере мы и получили **7.9062e-008** а не **0**.

В случае более сложных функций их удобнее представить в виде M файла. Тогда в качестве первого параметра функции **fsolve** подставляется в одинарных кавычках имя этого файла.

Трехмерные графики

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). Напоминаю что операции «**.**^» и «**.***» означают поэлементные а не матричные действия.

После этого подается команда на вывод трехмерного графика: **plot3(X,Y,Z)**. Результат исполнения для данного примера приведен на рис. 1.3.

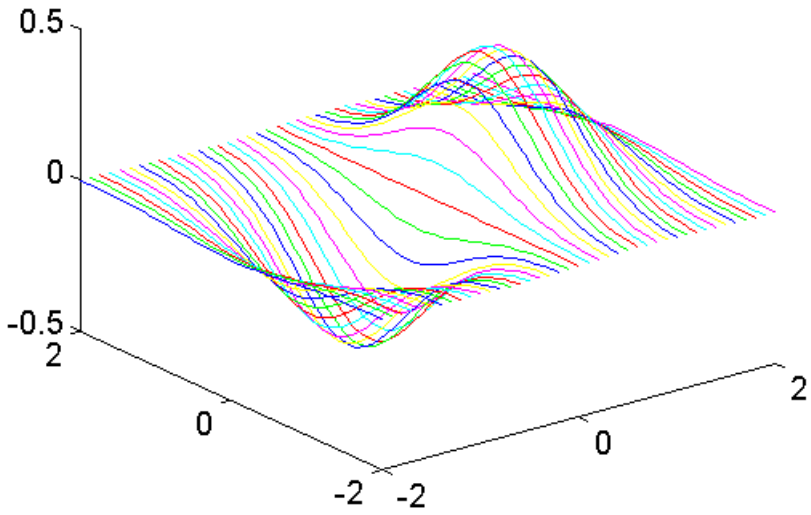


Рис. 1.3. Вывод трехмерных графиков

Элементы программирования в пакете MATLAB

Программирование в системе MatLAB очень близко к обычному программированию. Программа создается в любом текстовом редакторе. Файл должен иметь расширение *.M. Для того чтобы Вы могли вызвать его из любого места он должен размещаться в одном из каталогов перечисленных в конфигурационном файле MATLABRC.M. К строкам matlabpath =[...'путь',...,'путь'] ; необходимо по аналогии добавить путь до своего рабочего каталога.

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.1.4.

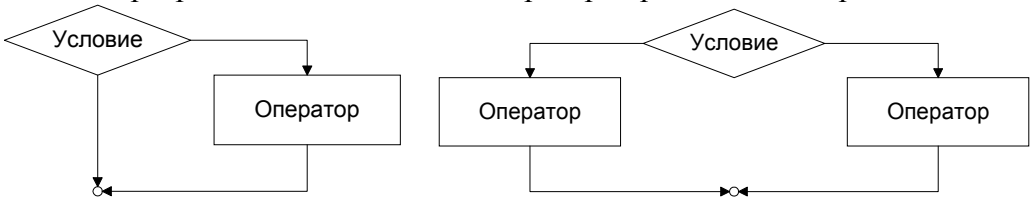


Рис. 1.4. Блок-схема условного оператора: редуцированная и полная формы

Формат записи оператора, редуцированная форма:

```
if условие
   операторы
end
```

полная форма:

```
if условие
   операторы
else
   операторы
end
```

Обращаю Ваше внимание что в отличие от современных языков программирования не используются такое понятие как составной оператор. Блок условного оператора обязательно заканчивается служебным словом **end**.

Пример:

```
d=b^2-4*a*c ;
if d<0
   error(‘ коней нет’)
else
   x1=(-b+sqrt(d))/(2*a)
   x2=(-b-sqrt(d))/(2*a)
end
```

Использованная в данном фрагменте программы функция **error** выводит на экран сообщение об ошибке.

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.

Задание на практику

Необходимо написать программу определяющую в какую область на рисунке попала точка. В качестве подготовки к выполнению задания необходимо по рисунку составить уравнения всех геометрических фигур и нарисовать блок-схему программы.

Дополнительное условие - в каждом операторе If можно работать только с одной геометрической фигурой.

Блок-схема программа и листинг на языке Си приведен на рис.1.5.

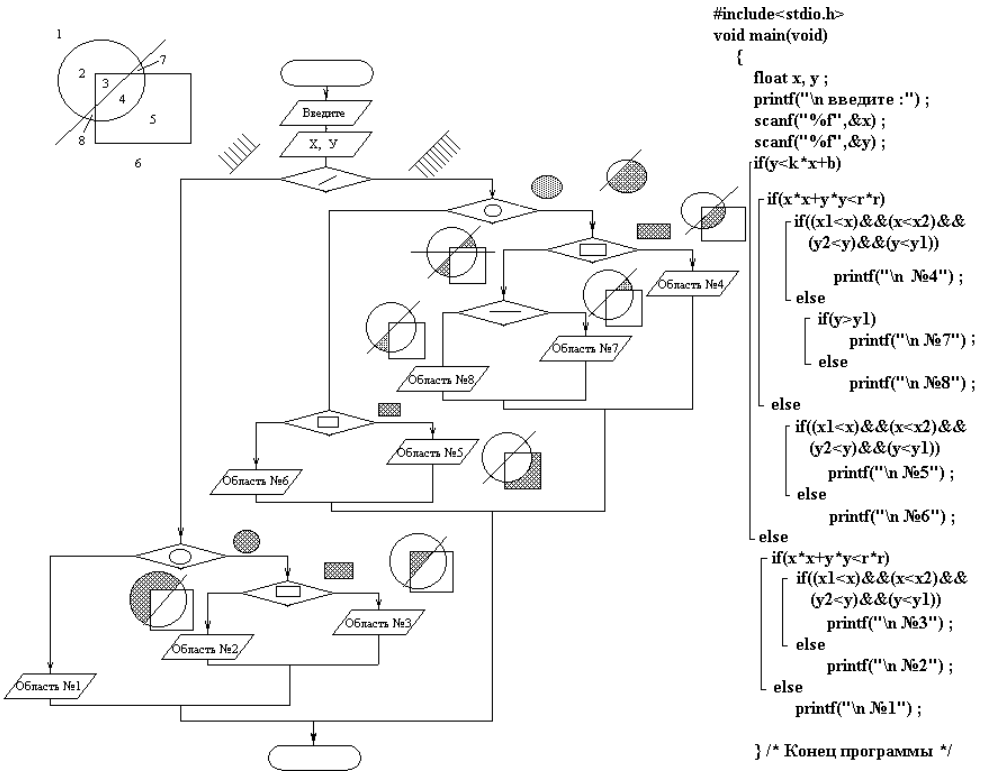


Рис. 1.5. Блок-схема программы определения попадания точки в области

Оглавление