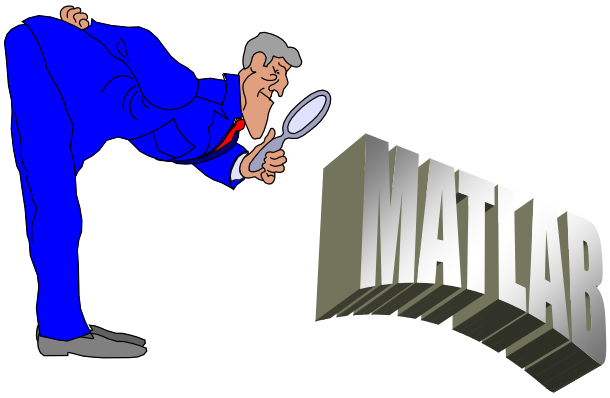
КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет вычислительной математики и кибернетики

Научно-технические расчеты в системе MATLAB

Учебное пособие
для студентов и аспирантов естественнонаучных факультетов



КАЗАНЬ – 2007

УДК 5519.6

ББК 32.81

З-15

Бахтиева Л.У.
Научно-технические расчеты в системе MATLAB. Учебное пособие для студентов и аспирантов естественнонаучных факультетов. – Казань, Изд-во КГУ, 2007, 44 стр.

Рецензент: Столов Е.Л.

Предлагаемое пособие содержит краткое описание основных возможностей системы MATLAB. Приведены примеры использования системы и задания для практических занятий.

© Казанский государственный университет, 2007.

СОДЕРЖАНИЕ

Введение.

MATLAB (Matrix Laboratory – матричная лаборатория) – мощное средство автоматизации математических расчетов, отличающееся, прежде всего, широким применением матричных операций. Одна из основных задач системы MATLAB – предоставление языка программирования, ориентированного на технические и математические расчеты, способного превзойти возможности традиционных языков программирования как по скорости вычислений, так и по адаптации к решению самых разнообразных задач.

Немаловажно, что с системой MATLAB могут интегрироваться такие популярные системы как Mathcad, Maple и Mathematica. Средство последних версий Matlab Notebook позволяет готовить документы в текстовом процессоре Word со вставками в виде результатов вычислений MATLAB, представленных в численном, табличном или графическом виде, что позволяет создавать «живые» электронные книги.

Таким образом, система MATLAB может стать отличным помощником в научных исследованиях. Однако, широкому применению системы препятствует недостаток необходимой литературы, изданной на русском языке. Документация по системе и ее приложениям содержит многостраничную информацию, разобраться в которой довольно сложно.

Цель настоящего пособия - облегчить знакомство с возможностями системы и особенностями их реализации. При подготовке пособия использованы материалы монографий и учебников, опубликованных в последние годы [1] - [4].

Часть 1. Основные сведения о системе MATLAB.

1. Режим прямых вычислений.

Систему MATLAB можно использовать, прежде всего, как мощный калькулятор, способный производить, помимо обычных вычислений, операции с векторами и матрицами, комплексными числами, рядами и полиномами. Можно выводить графики различных функций – от простой синусоиды до сложной трехмерной фигуры.

Работа в режиме прямых вычислений (командном режиме) носит диалоговый характер. Пользователь набирает выражение, редактирует его и после нажатия клавиши ENTER получает ответ. Например, команда

```
>>  v= [1 3 4];
```

задает трехэлементный вектор v (возможна также запись не через пробел, а через запятую v=[1,3,4]). Знак **;** (точка с запятой) в конце выражения блокирует немедленный вывод результата вычислений. Далее можно подать команду

```
>> sin(v)
```

и получить ответ

```
ans= 0.8415    0.1411    -0.7568
```

Как видим, система вычисляет функцию **sin** от векторного аргумента, при этом встроенные функции (см. Приложение) записываются строчными буквами, их аргументы указываются в круглых скобках. Если пользователем не указана переменная для значения результата, то MATLAB назначает переменную с именем **ans**.

Выражения записываются в соответствии с правилами языка MATLAB (п.3). Если выражение оказывается длинным, то его часть можно перенести на новую строку с помощью знака **...** (многоточие). Текстовые комментарии вводятся с помощью знака % перед текстом строки.

Полезно усвоить команды управления окном сессии:
clc – очистка экрана, home – возвращение курсора в левый верхний угол окна. Размеры и положение командного окна можно менять. Размер окна оптимален, если оно занимает нижнюю четверть экрана, а верхняя часть занята графическим окном, тогда вычисления выполняются одновременно с просмотром графиков.

С помощью клавиш ↑ и ↓ можно вернуться к выполнению любой из предыдущих команд.

2. Операции с рабочей областью.

Переменные и определения новых функций хранятся в рабочей области, которая при длительной работе в командном режиме перестает быть непрерывной и засоряется. Команда **save** позволяет сохранять значения переменных в бинарных файлах с расширением **.mat**. Команда может быть записана в одной из форм:
save fname – все переменные записываются в файл с именем fname.mat
save fname X – переменная X записывается в файл с именем fname.mat
save fname X Y Z – переменные X,Y,Z записываются в файл с именем fname.mat

После параметров команды могут быть указаны ключи, уточняющие формат записи:
mat (двоичный по умолчанию), ascii (8 цифр), ascii-double (16 цифр), append (добавление в уже существующий файл).

Для загрузки рабочей области служит команда **load** fname в аналогичных формах.

В системе MATLAB есть возможность сохранить текст уже набранных команд с помощью дневника сессии. Для этого служат следующие команды:
diary file_name – записывает весь текст сессии в файл с расширением **.txt**, diary off – приостанавливает запись, diary on – вновь начинает запись, type file_name – позволяет просмотреть записанный текстовый файл.

3. Переменные и функции системы MATLAB.

Простейший объект языка MATLAB – число. Примеры представления чисел: 15, -4, 3.205, 12.35e-24 (пробелы между символами в записи числа не допускаются!), 2+3.00i (мнимая часть комплексного числа имеет множитель i или j). Все операции над числами производятся в формате с двойной точностью. Выдача результата производится в нормализованной форме с четырьмя цифрами после десятичной точки и одной до нее. Для смены формата вывода чисел служит команда

>> format name

где name: 1) short (5 знаков), 2) short e (5 знаков мантиссы и 3 знака порядка), 3) long (15 знаков), 4) long e (15 и 3), 5) hex (в 16-тиричном формате), 6) bank (для денежных единиц).

Система обладает большим набором встроенных функций (математических и др.), позволяющих производить над числами, помимо обычных арифметических операций, самые разнообразные действия. Так, например, функции real(z), imag(z), abs(z), angle(z) возвращают соответственно действительную и мнимую части комплексного числа, его модуль и аргумент.

В MATLAB имеются системные константы, которые могут переопределяться, например: i или j (мнимая единица), pi (число π), eps (2^{-52}), realmin ((2^{-1022}), realmax (2^{1023}), inf (∞), ans, NaN (Not-a-Number – нечисловой характер данных или неопределенность), computer.

Еще один объект языка - символьная константа, т.е. цепочка символов, заключенных в апострофы, например: ‘MATLAB – наш помощник’.

Переменные в MATLAB могут иметь имя, состоящее из любого количества символов (идентифицируются первые 31), имя должно начинаться с буквы, но может содержать цифры и символ подчеркивания. Строчные и прописные буквы различаются! Значения переменных хранятся в рабочей области, для очистки которой используется команда **clear** в разных формах: clear X (стирается одна переменная X), clear a b c (несколько переменных), clear all (все переменные). Команда **who** перечисляет все переменные, хранящиеся в рабочей области на данный момент. Более подробную информацию выдает команда **whos.**

Все переменные в MATLAB рассматриваются как матрицы, вектора или массивы. Тип переменной явно не указывается. Размер самых простых переменных (чисел) 1x1, они допускают основные арифметические действия +, -, *, / с числовой переменной, такие действия выполняются поэлементно. Для матриц и векторов поэлементные операции (они обозначаются добавлением знака **.** (точка) перед соответствующей операцией, например, .* и ./) возможны лишь для операндов с одинаковыми размерами, например:

```
>> v1=[3 6 9 12];
>> v2=[1,2,3,4];
>> v1/v2
ans=3
>> v1.*v2
ans=3 12 27 48
>> v1./v2
ans=3 3 3 3
```

Операция обычного умножения (без знака **.**) выполняется по правилам линейной алгебры. Используются также операции \ и .\ (левое матричное деление), ^ и .^ (возведение в степень и поэлементное возведение в степень).

Специфическая операция **:** (двоеточие) применяется для формирования упорядоченных последовательностей (начальное значение **:** шаг **:** конечное значение, по умолчанию шаг=1), например:

```
>> 2:7
ans =
     2 3 4 5 6 7
```

Операция **:** широко используется при работе со средствами построения графиков, а также при формировании матриц и векторов. Выражения с операцией **:** могут использоваться в качестве аргументов функций, например:

```
>> %построение графика функции cos, аргумент меняется от 0 до 5 с шагом 0.2
>> plot(cos(0:0.2:5))
>> % вычисление функции Бесселя порядка от 0 до 5 с аргументом  0.5
>> bessel (0:5,1/2)
ans =
    0.9385   0.2423   0.0306    0.0026   0.0002   0.0000
```

Помимо арифметических, в языке MATLAB существуют операции отношения (= =, ~=, <, >, <= , >=), логические операции (&, and, or, not, ~, xor, any, all), а также операции = (присваиваивание), **.'** (простое транспонирование), **'** (комплексно-сопряженное транспонирование), [,] (горизонтальное соединение), [;] (вертикальное соединение).

MATLAB содержит большой набор (более тысячи) встроенных функций, элементарных и специальных, от одного или нескольких аргументов. Наиболее значимые функции перечислены в Приложении (для краткости приведены лишь названия фунций без указания аргументов). Более подробную информацию о конкретной функции можно получить, подав команду: help имя функции. Представление о полном наборе операторов, конструкций языка и системных функций можно получить с помощью команды: help имя, где имя – название соответствующей подпапки в папке MATLAB/TOOLBOX/MATLAB, а именно:

general – команды общего назначения;
ops - операторы и специальные символы;
lang – конструкции языка;
strfun – строковые функции;
iofun – функции ввода-вывода;
timefun – функции времени и дат;
datatypes – типы и структуры данных;
elmat – операции с элементарными матрицами;
elfun – элементарные математические функции;
specfun – специальные математические функции;
matfun – матричные функции линейной алгебры;
datafun – анализ данных и преобразования Фурье;
polyfun – полиномиальные функции и функции интерполяции;
funfun – функции функций и функции решения ОДУ;
soarfun – функции разреженных матриц;
graph2d – команды двумерной графики;
graph3d – команды трехмерной графики;
specgraph – команды специальной графики;
graphics – команды дескрипторной графики;
uitools – графика пользовательского интерфейса.

Язык MATLAB предоставляет возможность задать функцию пользователя. Для этого используется функция inline, аргументом которой является символьная строка, задающая функцию одной или нескольких переменных, например:

```
>> f=inline('1+sin(x)^2');
>> y=f(pi/2)
  y=2
```

Ту же задачу можно решить с помощью оператора eval(s), который вычисляет значение выражения, заданного в виде символьной переменной s:

```
>> x=pi/2;
>> y=eval('1+sin(x)^2')
  y=2
```

Функцию можно задать также в виде m-файла (см. п. 5) или с помощью оператора @ (так называемая дескрипторная или handle-функция).

4. Формирование векторов и матриц.

Если надо задать вектор из n элементов, то их значения следует перечислить в квадратных скобках через пробел или через запятую, например:

```
>> B=[2 –3 4 6];
```

При задании матрицы используется символ ; (точка с запятой) для разграничения строк:

```
>> A=[1 3; 0 –.3];
```

Можно вводить элементы векторов и матриц в виде арифметических выражений, содержащих доступные системе функции. Элементы матриц могут быть комплексными. Для указания отдельного элемента используется выражение вида B(k) или A(m,n), например:

```
>> A(1,2)
ans=3
```

Выражение A(k) дает доступ к элементам матрицы, развернутым в один столбец:

```
>> A(4)
ans=-0.3
```

Индекс k может быть логическим выражением:

```
>> A(A>0)
ans= 1 3
```

В качестве индекса может быть вектор. Например, запись A([1,2],2) обозначает первый и второй элементы второго столбца. Для указания на m-ю строку или n-й столбец матрицы используются выражения вида A(m,:) или A(:,n). Запись A(k:m;n) обозначает вектор-столбец, сформированный из нескольких элементов n-го столбца матрицы A (с k по m-й).

Имеется ряд функций для задания особых векторов и матриц, например magic(n) (матрица размером nxn, у которой сумма строк, сумма столбцов и сумма диагоналей одинаковы), hadamard(n) (матрица Адамара), hilb(n) (матрица Гильберта), zeros(m,...,q) (многомерный массив из нулей), ones(m,...,q) (массив из единиц), rand(m,...,q) (массив из случайных чисел), eye(n) (единичная матрица), rempat(A,m,n) (m*n дублей матрицы A, m вниз, n вправо), linespace(a,b,n) (вектор-строка из n чисел, равномерно расположенных на отрезке [a,b], по умолчанию n=100), logspace(a,b,n) (узлы логарифмической сетки по десятичному логарифму, по умолчанию n=50) и др.

Элементами матрицы могут быть вектора и матрицы, что позволяет формировать большие матрицы, объединяя малые. Для удаления строк или столбцов удобно использовать пустые квадратные скобки [ ]:

```
>> A(1,:)=[ ]
A=  0 –0.3
```

Основную информацию о векторах и матрицах можно получить с помощью функций size(A) (вектор-строка с размерами матрицы A), length(B) (число компонентов вектора B), ndims(X) (размерность массива X), disp(X) (массив X без его имени), isempty(X) (=1, если X=[]), isequal(X,Y) (=1, если X=Y), isnumeric(X) (=1, если X – числовая переменная), islogical(X) (=1, если X – логическая переменная).

При формировании матриц можно использовать следующие функции:

reshape(X,m,n,) – формирует матрицу m*n из элементов массива X;
diag(A,k) – формирует вектор-столбец из k-й диагонали A, по умолчанию – из главной;
tril(A) – формирует нижнюю треугольную матрицу для A;
triu(A) – формирует верхнюю треугольную матрицу для A;
fliplr(A) – меняет на обратный порядок столбцов;
flipud(A) – меняет на обратный порядок строк;
rot90(A,k) – поворачивает A на 90 градусов k раз;
find – определяет индексы ненулевых элементов;
end – на месте индекса задает его последнее значение.

Определитель квадратной матрицы вычисляется с помощью функции det(A). Для вычисления обратной матрицы методом Гаусса-Жордана используется функция inv(A).

5. Создание и отладка программных файлов.

Программами в системе MATLAB являются m-файлы текстового формата, содержащие запись программ в виде программных кодов. Основными средствами программирования являются:

- данные различных типов,
- операторы,
- функции пользователя,
- управляющие структуры,
- системные операторы и функции,
- средства расширения языка.

В MATLAB определены следующие основные типы данных: single, double, char, logical, sparse, cell, struct, function_handle, Java classes и др. Кроме того, предусмотрен тип данных UserObject (тип данных, определяемый пользователем). Каждому типу данных можно соотнести характерные для него операции.

Язык программирования системы MATLAB ориентирован на структурное программирование, в нем нет номеров строк и оператора безусловного перехода GO TO. Имеются управляющие структуры следующих типов:

- условный оператор **if...elseif...else...end,** простейший вариант: **if** условие инструкции **end**;
- цикл типа **for...end,** конструкция цикла имеет вид: **for** var=b:s:f , инструкции, **end,** где b – начальное значение переменной цикла var, s – шаг (по умолчанию s=1), f – конечное значение переменной var;
- цикл типа **while...end,** конструкция цикла имеет вид: **while** условие инструкции **end,** досрочный выход из цикла может быть реализован с помощью операторов **break** и **continue**;
- переключатель **switch...case...otherwise...end**;
- конструкция **try...catch...end**.

Имеются также функции диалогового ввода и вывода:

- r=**input** (‘String’) или **input** (‘String’,’s’);
- **disp**(X).

Функция **pause** (или **pause**(N)) позволяет приостановить выполнение программы, N – количество секунд. Функции **tic** и **toc** позволяют включить и выключить таймер.

Программы вызываются по имени, под которым записаны в виде файла с расширением .m. Имя программы начинается с латинской буквы и состоит из таких букв и цифр. При первом обращении программа ищется на диске. Рекомендуется записать программу в текущую директорию. Поиск m-файла по ключевому слову осуществляется командой **lookfor**. Путь к программе можно указать командой **path**.

Программы бывают двух видов: сценарии (scripts) и функции (functions). Программа-скрипт представляет собой набор команд без входных и выходных параметров. Она может использовать все переменные рабочей области, а не только те, что создаются в ней. Собственные переменные скрипта располагаются в рабочей области без защиты от внешнего доступа, что позволяет передать управление из скрипта в командную строку с помощью команды **keyboard** (без параметров)**,** при этом работа в командной строке заканчивается командой **return.** Таким образом, скрипт удобен для работы в интерактивном режиме, но требует повышенного внимания при использовании в нем своих переменных.

Пример программы-скрипта (определение локальных максимумов вещественного вектора):

```
m=length(x);  %  вычисление длины вектора  x
v=2:m-1;    % формирование  вектора  индексов
i=find(x(v-1)<x(v)&x(v)>x(v+1))+1;  %  поиск индексов  локальных максимумов
y=x(i);  % значения локальных максимумов
```

После набора программы ее следует сохранить, например, под именем lokmax.m. Теперь возможен ее вызов из командной строки:

```
>> x=[1,2,-3,3,-2,4]; lokmax, [y;i]
 y =2 3
 i =2 4
```

Комментарий, включенный в программу, выводится командой:

>> help имя файла

Первая строка программы-функции имеет вид: **function** [y1,y2,...yj]=ff(x1,x2,...xi), где ff – имя функции, x1,x2,...xi – входные формальные параметры, y1,y2,...yj – выходные формальные параметры. Обращение к функции записывается так: [v1,v2,...vj]=ff(u1,u2,...ui), где u1,u2,...ui,v1,v2,...vj – фактические параметры. Входных аргументов у функции может и не быть. Все переменные программы-функции имеют локальный характер и недоступны для их использования извне. Если при создании файла-функции желательно применение глобальных переменных, то следует подать команду: **global** var1 var2... . Таким образом, программа-функция неудобна для использования в интерактивном режиме, но любая ее переменная может быть открыта для использования в командной строке.

Пример программы-функции (та же задача):

```
function [y,i]=lokmax(x)
m=length(x);
v=2:m-1;
i=find(x(v-1)<x(v)&x(v)>x(v+1))+1;
y=x(i);
```

После сохранения функции к ней можно обратиться из командной строки:

```
>> x=[1,2,-3,3,-2,4]; [y,i]= lokmax(x)
```

и получить тот же ответ.

Файлы-функции могут вызываться так же из других m-файлов. Начиная с версии 5.0 появилась возможность включать в тело основной функции подфункции. Они имеют идентичную конструкцию.

При создании программ-функций часто бывают полезны две функции:

- nargin – возвращает число входных параметров данной функции;
- nargout - возвращает число выходных параметров данной функции;

Иногда в программе-функции бывает необходимо выдать сообщение об ошибке, для этого служит команда error (‘сообщение’).

Важным этапом программирования является отладка программы. MATLAB располагает большими возможностями, облегчающими процесс отладки. В частности, полный листинг программного файла можно вывести с помощью команды:

>> type имя файла (или для вывода листинга с пронумерованными строками dbtype)

Для установки в тестируемый m-файл точек прерывания используются команды:

>> dbstop in M-file at lineno – установить точку прерывания в заданной строке;

>> dbstop in M-file at subfun – установить точку прерывания в подфункции;

>> dbstop if error – установить точку прерывания при сообщении об ошибке.

Команда dbstatus (без параметров) выводит список установленных в данной сессии точек прерывания. Для удаления точек прерывания используется команда dbclear с тем же синтаксисом, что и dbstop, например

>> dbclear M-file at lineno.

Для пошагового выполнения программы используется команда dbstep (или dbstep nlines для выполнения заданного числа строк).

Для редактирования и отладки m-файлов в MATLAB существует специальный многооконный редактор, выполненный как типичное приложение Windows. Он позволяет нумеровать строки, устанавливать контрольные точки, вычислять переменные по месту нахождения, получать справку и прочее (вход в редактор осуществляется из главного меню: (File – M-file).

6. Работа с файлами.

Открыть файл в системе MATLAB можно с помощью специального Мастера импорта Import Wizard (Import Data в меню File). Возможности Мастера достаточно очевидны, они зависят от того, какие данные импортируются. Можно также использовать обычные файловые операции чтения и записи файлов в тех или иных форматах. Операции импорта и экспорта файлов открывают обширные возможности по обмену данными между системой MATLAB и другими программами. Вот некоторые из них:

open имя – открывает файл в зависимости от анализа параметра имя и его расширения;

fid=**fopen**(filename,permission)-открывает указанный файл под управлением permission (этот параметр принимает значение ‘r’, если файл открывается для чтения, ‘w’ – для записи, ‘a’ – для присоединения), при открытии текстового файла к содержимом permission добавляется t (например, ‘rt’ или ‘wt’), fid содержит идентификатор файла;

status=**fclose**(fid) – закрывает файл с указанным идентификатором, status=0, если операция прошла успешно и –1 при ошибке;

[A,count]=**fread**(fid,size,precision) – читает двоичные данные из файла с указанным идентификатором и записывает их в матрицу A, параметр count содержит число успешно прочитанных элементов (не обязателен), если параметр size не задан, то читается весь файл;

count=**fwrite**(fid,A,precision) – записывает двоичные данные из матрицы A в файл;

fscanf - читает форматируемые данные из файла;

fprintf(fid,format,A,...) – записывает форматируемые данные в файл, fid=1 для стандартного вывода (экран по умолчанию), fid=2 для стандартной ошибки, format - строка, одержащая спецификаторы %, *, \, символы преобразования f, d, i, o,u, x и пр. (см. Language Reference Guide), оператор подобен одноименному оператору языка C;

r=**input** (‘запрос’) - вводит выражение с клавиатуры, результат заносится в r;

imread - читает образ из графического файла;

imwrite – записывает образ в графический файл;

iminfo – возвращает информацию о графическом файле;

auread (или **wavread)**– считывает заданный аудиофайл;

auwrite (или **wavwrite)-** записывает заданную инфомацию в виде аудиофайла;

matlabroot– возвращает имя директории, в которой установлено программное обеспечение MATLAB.

Для позиционирования файлов можно использовать следующие команды:

frewind(fid**)-** устанавливает начальное значение указателя положения в файле с указанным идентификатором;

ftell(fid**)** – возвращает значение указателя положения;

fseek(fid,offset,origin) – устанавливает положение указателя на байт с заданным смещением offset относительно положения origin (=-1 для начала файла, =0 для текущей позиции указателя, =1 для конца файла);

feof(fid) – указывает, является ли считываемый символ признаком конца файла;

ferror(fid) – возвращает сообщение об ошибке в файле.

Для записи файлов на диск и считывания файлов с диска служат команды **load** и **save,** используемые в очевидных формах. В этих командах имя файла указывается по правилам, принятым в операционных системах класса MS-DOS.

Для запуска Мастера импорта можно использовать также команды:

uiimport (fname) – открывает файл и запускает Мастер импорта;

uiimport (‘-file’) – вначале выводит диалог выбора файла;

uisave – управляемое пользователем сохранение;

Команда **delete**(‘имя файла’) удаляет файл из текущей папки.

7. Средства создания баз данных.

Одним из типов данных, используемых системой MATLAB, являются структуры. Каждая структура состоит из ряда полей, имеющих имена. Поля могут содержать данные любого типа - от пустого поля [ ] до массивов. MATLAB поддерживает и массивы структур, что позволяет создавать мощные базы данных.

Для задания структуры используется оператор присваивания, например:

```
>> man.name='Петр';
>> man.fam='Иванов';
>> man.dr=1976;
>> man.mr='Казань';
```

Чтобы создать массив структур, надо ввести индексацию имени структуры, например:

```
>> man(1).fam='Иванов';
>> man(3).dr=1972;
```

Количество структур в массиве можно определить с помощью функции length(имя):

```
>> length(man)
ans=3
```

Для просмотра структуры или отдельного поля следует указать соответствующее имя:

```
>> man(3).dr
ans=1972
```

Структуру можно создать также с помощью функции **struct**('field1',value1,...), например:

```
>> A=struct('gorod','Казань','ulica','Московская','dom',12,'kvart',52)
A=
gorod:  'Казань'
ulica:  'Московская'
dom:  12
kvart:  52
```

С помощью функции deal возможно множественное присваивание входных данных:

```
>> [A,B,C,...]=deal(X,Y,Z,...).
```

Проверка имен полей и структур осуществляется с помощью функций isfield(S,'field') и isstruct(S).

Для работы с полями структуры можно использовать функции:

```
>> fieldname(S) – возвращает имена полей заданной структуры
>> getfield(S,'field') – возвращает содержимое заданного поля
>> setfield(S,'field',V) – присваивает заданному полю значение переменной V
>> rmfield(S,'field') – удаляет заданное поле.
```

Наиболее сложный тип данных в системе MATLAB – массив ячеек. Это массив, элементами которого являются ячейки, содержащие любые типы массивов, включая массивы ячеек. Содержимое ячеек задается в фигурных скобках {}.

Существует два способа присваивания данных отдельным ячейкам – индексация ячеек или индексация содержимого. Для создания массива ячеек можно использовать также функцию:

```
>> cell(N)  - создание массива ячеек из NxN пустых матриц
>> cell(size(A)) - создание массива ячеек из пустых матриц того же размера, что A.
```

Для отображения массива ячеек C служит команда celldisp(C).

Для более наглядного графического представления можно использовать команду:

```
>> cellplot(C,'legend').
```

Для создания строкового массива ячеек может использоваться функция cellstr(S), например:

```
>> S={'Я''живу''в Казани'}
>> C=cellstr(S)
C='Я' 'живу' 'в Казани'.
```

8. Графика, анимация.

Система MATLAB обладает большими возможностями для графической визуализации вычислений – от построения простых графиков функций до трехмерных комбинированных и презентационных графиков с элементами анимации, а также средствами проектирования графического интерфейса (GUI).

Для **работы с графическим окном** можно воспользоваться командами:

figure(i) – делает активным i–е окно при i<=k (если открыто k окон) или открывает окно с номером i при i>k

clf – очищает текущее графическое окно (по умолчанию окно очищается при построении очередного графика)

hold on – запрещает очистку окна до подачи команды hold off

zoom on – включает режим лупы (zoom(N) изменяет масштаб в N раз)

whitebg – открывает графическое окно, меняя цвет фона на противоположный

subplot(m,n,p) – разбивает окно на m*n подокон, p – номер текущего подокна.

Выдачу содержимого графического окна на печать можно осуществить с помощью опции Print в меню File. Копирование содержимого текущего графического окна осуществляется опцией Copy Figure в меню Edit.

Для **построения графика функции одной переменной в декартовых координатах** служит команда

plot(X,Y) – строит график функции Y(X), где Y и X – вектора одинаковой длины.

Если X и Y – матрицы, то строится семейство графиков по данным, содержащимся в колонках матриц. Команда plot(Y) строит график Y(i), где i –индекс элемента в векторе Y, команда plot(X,Y,S) позволяет задать тип и цвет линии графика с помощью строковой константы S (Y - желтый, M – фиолетовый, C - голубой, R - красный, G - зеленый, B - синий, W - белый, K – черный, **-** сплошная линия, **-.** штрихпунктир, **:** двойной пунктир, **_** штриховая линия, **.** линия строится точками, * звездочками, + плюсами, O окружностями, S квадратами, X крестами, V треугольниками вниз, ^ треугольниками вверх, D ромбами, < треугольниками влево, > треугольниками вправо, P пятиугольниками, H шести-угольниками). Для построения нескольких линий на одном графике возможна подача команды plot(X1,Y1,S1,X2,Y2,S2,...), например:

```
>> x=-2*pi:0.1*pi:2*pi;
>> y1=cos(x);
>> y2=cos(x).^2;
>> y3=cos(x).^3;
>> plot(x,y1,'-Y',x,y2,'-.+R',x,y3,'_OK')
```

Команда plot(Y,X) позволяет построить график обратной функции, а plot(X(t),Y(t)) – график функции, заданной параметрически.

Разметкой осей и линиями сетки можно управлять с помощью команд:

axis off – убирает с осей их обозначения и маркеры

axis on – восстанавливает введенные ранее обозначения

axis([xmin xmax ymin ymax]) – устанавливает диапазон координат по осям

axis auto – устанавливает параметры осей по умолчанию

axis equal - устанавливает параметры осей с одинаковым расстоянием между метками

axis manual – «замораживает» масштаб осей в текущем состоянии

grid on – добавляет сетку к текущему графику

grid off – отключает сетку.

Для **текстовых сообщений** на графике служат команды:

title('string') – титульная надпись

xlabel('string') или ylabel('string') – названия осей

text(X,Y,'string') – текст, начало которого находится в точке (X,Y)

gtext('string') – позиционирование текста с помощью мышки

legend(string1,string2,...,pos) – легенда, размещенная в зависимости от параметра pos (0 – лучшее место, 1 – верхний правый угол, 2 – верхний левый угол, 3 – нижний левый угол, 4 – нижний правый угол, -1 – справа от графика).

Для **построения графиков в полярной системе координат** служит команда polar(theta,ro) или polar(theta,ro,S) с синтаксисом, аналогичным plot, например:

```
>> x=1:.01:2;
>> nx=length(x);
>> ro=x.^3;
>> fi=linespace(0,3*pi,nx);
>> polar(fi,ro)
```

Команды loglog(...), semilogx(...) и semilogy(...) позволяют построить графики функций **в логарифмическом и полулогарифмическом масштабах**.

Для **построения диаграмм** служат команды:

bar(Y) – столбцовый график элементов вектора Y (или bar(X,Y), где положение столбцов задается вектором X, или bar(X,Y,width,S) с заданной шириной столбцов, параметр S аналогичен рассмотренному ранее параметру для команды plot);

hist(...) – гистограмма, параметры те же, что у команды bar;

stairs(...) – лестничный график, параметры те же, что у команды bar;

rose(theta,N) – угловая гистограмма, N – число интервалов от 0 до 2*pi (по умолчанию 20).

MATLAB обладает большим набором средств **трехмерной графики.** Прежде всего, это различные средства **построения поверхностей**, описываемых функцией двух переменных например:

meshgrid(X,Y) –задание опорной области для построения трехмерной поверхности (команда meshgrid(X,X) эквивалентна команде meshgrid(X));

plot3(X,Y,Z) – трехмерный аналог команды plot, где X, Y, Z – матрицы одинакового размера (возможен вариант с дополнительным параметром S или plot3(X1,Y1,Z1,...)), команда строит трехмерные точки и соединяет их отрезками прямых.

Пример:

```
% параболоид, изображенный методом сеток
>> [X,Y]=meshgrid([-2:.1:2]);
>> Z=X.^2+Y.^2;
>> plot3(X,Y,Z,'-r',Y,X,Z,'-r')
```

Более наглядное изображение обеспечивают команды:

mesh(X,Y,Z,C) – сетчатое изображение поверхности (C – цвет узловых точек, по умолчанию C=Z, т.е. цвет линий зависит от их высоты);

meshc(...) - изображение поверхности и ее проекции в виде линий равного уровня;

meshz(...) – построение поверхности столбцами;

surf(X,Y,Z,C) – цветная параметрическая поверхность, цвет задается массивом C;

surfc(...) – построение поверхности и ее проекции;

surfl(...) – построение поверхности с подсветкой от источника света, возможен параметр S – вектор координат источника (S=[Sx,Sy,Sz] в декартовых координатах и S=[AZ,EL] в сферических координатах), можно также задать цвет подсветки дополнительным параметром 'light'.

Окраску поверхности и различные эффекты можно задать с помощью функций:

colormap(S) – задает окраску тонами указанного цвета, например colormap('gray');

shading interp – устраняет изображение сетки и задает интерполяцию для оттенков цвета;

colorbar – выдает на экран шкалу оттенков с соответствующими координатами;

diffuse – задает эффект диффузионного рассеяния;

lighting – управляет подсветкой;

material – имитирует свойства рассеивания света различными материалами;

specular – задает эффект зеркального отражения;

view – задает положение точки просмотра;

rotate3d – задает поворот трехмерной фигуры.

Существуют специальные команды для построения цилиндра и сферы, а также для построения объемных фигур с помощью плоских треугольников:
[X,Y,Z]=cylinder(R,N) – создает массивы для дальнейшего построения с помощью функции surf цилиндрической поверхности с радиусом R и числом узловых точек N;
[X,Y,Z]=sphere(N) – генерирует матрицы размера (N+1)*(N+1) для последующего построения с помощью команды surf**;**
trimesh(TRI,X,Y,Z,C) – строит объемную каркасную фигуру с треугольниками, заданными матрицей поверхности TRI, каждая строка которой содержит три элемента и определяет треугольную грань путем указания индексов, по которым выбираются координаты из векторов X,Y,Z;
trisurf(...) – аналогична предыдущей команде, но строит закрашенные треугольники.

MATLAB обладает большим набором функций, обеспечивающих **анимацию,** например:
comet(X,Y) – отображает движение точки по заданной векторами X,Y траектории;
comet3(X,Y,Z) – движение трехмерной точки;
capture – захват с экрана текущего графического окна;
getframe – формирование вектор-столбца, определяющего один кадр для анимации;
m=moviein(N) – формирование матрицы, рассчитанной на N анимационных кадров;
movie(m) – отображение кадров, содержащихся в матрице m;
rotate – вращение фигуры вокруг заданной точки в заданном направлении;
frame2im – преобразование кадра в графический образ;
im2frame – преобразование графического образа в кадр.

Центральным понятием **дескрипторной** (handle) графики, на которой базируются графические средства MATLAB, является графический объект. Используются следующие типы объектов: root (первичный – экран компьютера), figure (объект создания графического окна), uicontrol (создание пользовательского интерфейса), axis (объект, задающий область расположения графика), uimenu (объект создания меню), uicontextmenu (создание контекстного меню), image (создание растровой графики), line (создание линии), path (создание закрашенных фигур), rectangle (закрашенный прямоугольник), surface (поверхность), text (текст), light (свет). Пример создания объекта:
% построение отрезка прямой
>> h=line([0 2 5],[1 4 –1],’color’,’blue’);

К объектам применяются операции:
get(h) – вывод свойств графического объекта;
set(h,’Name’,Value,...) – установка свойств объекта, например: set(h,’color’,’red’);
reset(h) – восстановление свойства объекта;
delete(h) – удаление объекта;
drawnow – выполнение очереди задержанных графических команд;
findobj – поиск объекта с заданными свойствами;
copyobj – копирование объекта с заданными свойствами и др.

К свойствам, задающим вид изображения, относятся, например:
Clipping – отсечение по границам осей;
SelectionHighlight – подсветка с выводом дескрипторов (описателей);
Visible – видимость (значение on (по умолчанию) или off);
EraseMode – свойство для создания анимации (normal (по умолчанию), none (не стирает изображение при перемещении или изменении), xor (стирание изображения с использованием исключающего «или»), background (стирание изображения путем прорисовывания цветом фона).

Получение изображения из графического объекта и вывод объекта на экран можно осуществить с помощью функций getimage(h) и image(C), где C – матрица, элементы которой точно указывают цвет и прямоугольный сегмент изображения.

Основные средства по обработке изображений входят в пакет Image Processing Toolbox.

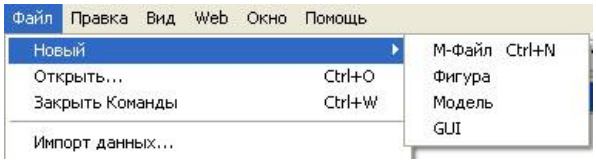
9. Графический интерфейс пользователя (GUI).

Дескрипторная графика MATLAB позволяет конструировать детали графического пользовательского интерфейса (Graphic User Interface) для создания собственных приложений. При этом различные функции и m-файлы вызываются из графического окна стандартного вида, но элементы интерфейса (кнопки, меню, слайдеры, надписи и пр.) задаются пользователем. Для этой цели используются функции:

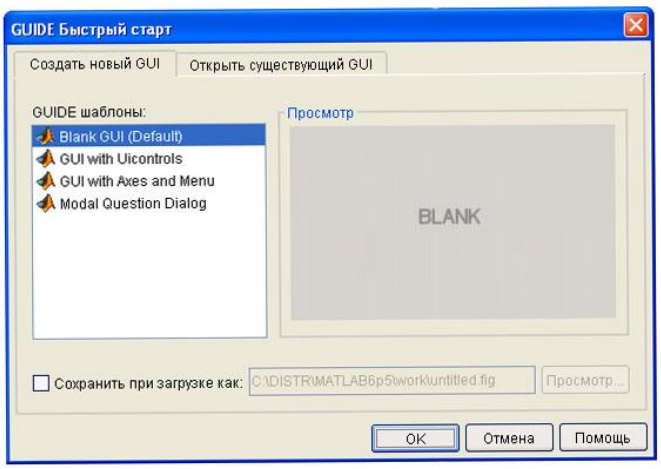
- uicontrol – управление пользовательским интерфейсом;
- uimenu – создание меню пользователя;
- ginput – графический ввод с помощью мыши (работа команды заканчивается нажатием клавиши Enter или устанавливается число обращений ginput(n));
- dragrect – создание и перетаскивание прямоугольника с помощью мыши;
- rbbox – растягивание прямоугольника мышью;
- selectmoveresize – выделение, копирование, перемещение и изменение размеров объекта;
- waitforbuttonpress – задержка выполнения программы до нажатия кнопки мыши;
- waitfor – блокировка исполнения и ожидание события;
- uiwait – прекращение выполнения программы до команды uiresume или закрытия окна;
- uiresume – возобновление работы программы;
- uisuspend – запрет интерактивного состояния графического окна;
- uirestore – восстановление интерактивного состояния;
- guide – создание интерфейса в интерактивном режиме;
- align - выравнивание объектов интерфейса;
- cbedit - редактирование объектов интерфейса в интерактивном режиме;
- menuedit – изменение меню;
- propedit – изменение свойств объекта;
- dialog – создание диалогового окна;
- axlimdlg – вызов окна для изменения координатных осей графического окна;
- errordlg – создание окна с сообщением об ошибке;
- helpdlg – создание справочного окна;
- inputdlg – создание окна для ввода;
- listdlg – создание окна для выбора вариантов значений параметра из списка;
- menu – создание меню диалогового ввода (choice=menu(header, item1, item2, ...);
- msgbox – создание окна сообщений;
- questdlg - создание окна запроса;
- warndlg - создание окна предупреждения;
- uigetfile - создание окна открытия файла;
- uiputfile - создание окна сохранения файла;
- uisetcolor - создание окна выбора цвета;
- uisetfont - создание окна выбора шрифта;
- pagedlg - создание окна парамеров страницы;
- printdlg - создание окна для вывода на печать;
- waitbar - создание “панели ожидания”;
- makemenu - создание структуры меню;
- menubar – установка типовых свойств для объекта MenuBar;
- btngroup - создание инструментальной панели с группой кнопок;
- btnstate – запрос статуса кнопки;
- btnpress – управление кнопкой;
- btndoun – нажатие кнопки;
- btnup – “поднятие” кнопки.

Полный список функций GUI можно изучить с помощью команды help uitools.

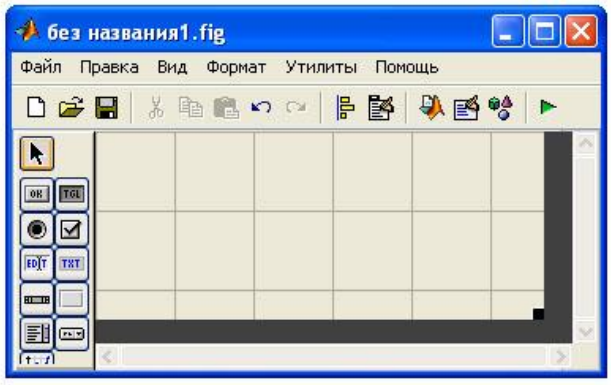
При создании GUI удобно пользоваться встроенным редактором, который вызывается из основного меню:



После выбора пункта GUI появляется окно:



Кнопка **ok** открывает конструктор GUI:



Далее можно выбрать объекты интерфейса: Static Text – статический текст, Edit Text– редактируемый текст, Popupmenu – всплывающее меню, Chekbox – флажок, Axes – график, Push Button – командная кнопка и др.

Свойства выбранных объектов можно изменить с помощью Инспектора, который появляется после двойного щелчка мышью по объекту.

Созданный интерфейс следует сохранить в файле с расширением .fig. При этом автоматически создается m-файл (программа-функция), который можно отредактировать с помощью кнопки **f** из основного меню. В частности, указав курсором в открывшемся меню на имя нужного объекта, можно изменить его программный код.

Примеры разработки GUI приведены в части 3 (тема 10).

10. Сообщения об ошибках, справочная система MATLAB.

При ошибочном написании математических выражений или команд MATLAB выдает соответствующие сообщения об ошибке или предупреждения, например:

??? Undefined function or variable ‘sqr’
Warning: Devide by Zero

Предупреждения не останавливают вычисления, а лишь предупреждают о том, что ошибка способна повлиять на ход вычислений, при этом результат иногда выдается в виде сокращения NaN, которое означает неопределенность (например, вида 0/0 или Inf/Inf). Сообщение об ошибке (после ???) останавливает вычисления.

Для исправления ошибки можно нажать клавишу Tab, предварительно поставив курсор в конец исправляемого выражения. Система введет подсказку, анализируя введенные символы. Можно также воспользоваться справочной системой MATLAB.

Основной доступ к справочной информации обеспечивает меню Help, окно которого имеет 5 вкладок:
Contents – поиск информации по оглавлению;
Index – поиск информации по алфавитному каталогу;
Search – поиск информации по заданной справке или отдельному слову;
Demos – доступ к демонстрационным примерам;
Favorites – доступ к специальным возможностям справочной системы (например, печать справочной документации).

Недостатком справочной системы является ее громоздкость (система поставляется на трех CD-ROM). Во многом она дублируется другими справочными подсистемами, например, справками, вызываемыми из командной строки и имеющимися в виде PDF-файлов.

Команда help (без параметров) вызывает список разделов интерактивной справки (список папок, содержащих m-файлы с определениями операторов, функций и др. объектов). Для получения справки по конкретному объекту служат команды:
help имя (в качестве имени может быть константа, функция, оператор и пр.)
doc имя (для вывода более полной информации)

Можно получить справку по группе объектов, например, команда:
help timefun
выдаст информацию об имеющихся в MATLAB функциях времени и дат.

Важное значение имеет поиск m-функций по ключевым словам. Для этого служит команда:
lookfor ‘ключевые слова’
она осуществляет поиск всех файлов, в заголовках которых встречаются данные ключевые слова (или слово).

Система MATLAB содержит множество демонстрационных примеров (практически на каждый оператор или функцию), изучение которых позволяет ближе познакомиться с системой. Список примеров можно получить, исполнив команду:
help demos

Просмотреть демонстрационные примеры можно также, подав команду:
demo

Важно, что демонстрационные примеры сопровождаются соответствующими листингами программ-файлов, реализующих поставленную задачу.

Полезными также могут оказаться команды:
helpwin (открывает окно справок Matlab Help Window), helpdesk (открывает в отдельном окне доступ к документации о командах системы), ver (выводит справку о версиях MATLAB и приложений), whatsnew (выводит информацию о новом в указанном разделе) и readme.

11. Интерфейс системы MATLAB.

Наиболее простой и удобный способ работы с системой MATLAB – работа с помощью панели инструментов, при этом основные команды вводятся нажатием левой клавиши мыши на нужную кнопку:

- New M-file – выводит окно редактора m-файлов;
- Open file – открывает окно для загрузки m-файла;
- Cut – вырезает в буфер выделенный фрагмент;
- Copy – копирует в буфер выделенный фрагмент;
- Paste – переносит фрагмент из буфера в текущую строку;
- Undo – отменяет последнюю операцию;
- Redo – восстанавливает последнюю операцию;
- Simulink – открывает браузер бибилиотек Simulink;
- Help – открывает окно справки.

Можно также использовать контекстное меню, появляющееся при нажатии правой кнопки мыши на выделенный фрагмент и отображающее доступные в данный момент команды.

В левой части окна системы имеется браузер рабочей области - Workspace Browser, который дает наглядную визуализацию рабочей области, позволяет редактировать содержимое находящихся в памяти объектов и удалять их. Для работы с конкретным объектом необходимо открыть его двойным щелчком по имени в списке.

Просмотр файловой структуры осуществляется специальным браузером (Path Browser), для запуска которого используется окно Current Directory (Текущая папка).

Основное меню последних версий системы MATLAB 6.* содержит шесть пунктов:

- File
- Edit
- View
- Web
- Window
- Help.

Меню **File** содержит команды для работы с файлами: New, M-file (открывает окно редактора/отладчика m-файлов, где используются цветовые выделения – синий цвет для ключевых слов, черный для операторов, констант и переменных, зеленый для комментариев (после знака %) и символьных переменных, красный для синтаксических ошибок), Figure (открывает пустое окно графики), Model (открывает окно для создания Simulink-модели), GUI (открывает окно для создания элементов графического интерфейса), Open, Close Command Windows, Import Data, Save Workspace As..., Set Path, Preferences..., Print..., Print Selection..., Exit.

Меню **Edit** содержит команды редактирования, типичные для приложений Windows: Undo, Redo, Cut, Copy, Paste, Clear, Select All, Delete, Clear Command Windows, Clear Command History, Clear Workspace.

Меню **View** позволяет управлять окнами интерфейса. Кроме командного, можно вывести окно command history с дневником сессии, окно рабочей области и др.

В основном меню новейшей версии системы MATLAB 7 появился пункт **Graphics,** содержащий команды New Figure, Plot Tools и More Plots..., удобные для построения графиков. Например, исполнив команду More Plots..., можно, указав имя переменной-массива, задать категорию и тип графика из предложенного каталога, в котором представлены все типы графиков, включая трехмерные и особую handle- графику. Команда Plot Tools открывает окно редактора графики, что позволяет выводить данные по месту установки курсора, вставлять в рисунок панель цветов и легенду, выбирать средства украшения графиков (стрелки, текстовые надписи, окружности, эллипсы, прямоугольники и пр.), задавать титульную надпись, установку для осей, а также выбирать из списка переменных тот массив, по данным которого строится очередной график.

Часть 2. Обзор расширений MATLAB.

Система MATLAB поставляется совместно с главным расширением Simulink. Это самое большое приложение системы, которое входит в блок расширений Blockset. Simulink, представляет собой систему программирования в геометрической форме, т.е. в виде блок-диаграмм, пакет предназначен для визуально-ориентированной подготовки имитационных моделей и анализа широкого класса динамических систем. Поддерживаются линейные, нелинейные, непрерывные, дискретные и гибридные системы. Модели могут образовывать иерархии, упрощая тем самым анализ подсистем. Моделирование может быть интерактивным или выполняться из командной строки MATLABa. В сочетании с такими пакетами, как Real-Time Workshop и Stateflow, Simulink образует семейство программ для решения задач цифровой обработки сигналов, разработки коммуникационных систем, систем управления и энергетических систем. Большие возможности по увеличению мощности среды моделирования предоставляют специализированные приложения SP Blockset, Fixed-Point Blockset, Power System Blockset и др. О пакете Simulink и особенностях его использования подробно написано в [1].

Другие приложения (программные дополнения к ядру системы) входят в состав большого «инструментального ящика» Toolboxes. Состав и версии этих пакетов постоянно обновляются, краткие аннотации к последним версиям можно найти в интернете по адресу: www.softline.ru. Для установки приложения его следует отметить в общем списке при инсталяции системы (иногда требуется задать некоторую дополнительную информацию). Приложения можно условно разбить на классы:

1. **Пакеты математических вычислений.**

   а) Symbolic Math Toolbox – пакет прикладных программ, представляющий возможность решения задач в символьном (аналитическом) виде. Пакет базируется на применении ядра символьной математики одной из самых мощных систем компьютерной алгебры Maple. Обеспечивает выполнение символьного дифференцирования и интегрирования, вычисление сумм и произведений, разложение в ряды Тейлора и Маклорена, операции с полиномами, вычисление корней полиномов и др. Имеет команды прямого доступа к ядру системы.

   б) NAG Foundation Toolbox – одна из мощнейших библиотек математических функций, созданная совместно с группой The Numerical Algorithms Group, Ltd. Пакет содержит более 240 функций, обеспечивающих решение самых разнообразных задач, включая оптимизацию, решение ОДУ и уравнений в частных производных, интерполяцию, вычисление собственных значений и векторов, сингулярных чисел, аппроксимацию кривых и поверхностей полиномами, кубическими сплайнами и полиномами Чебышева, минимизацию и максимизацию функций, статистические расчеты, корреляционный и регрессионный анализ и пр.

   в) Spline Toolbox – пакет прикладных программ для работы со сплайнами. Включает функции для создания, отображения, интерполяции, аппроксимации и обработки сплайнов в B-форме и в кусочно-полиномиальной форме.

   г) Statistics Toolbox – пакет прикладных программ по статистике, расширяющий возможности системы MATLAB в области реализации статистических вычислений и статистической обработки данных. К возможностям пакета относятся описательная статистика, распределения вероятностей (20 различных распределений), оценка параметров и аппроксимация, проверка гипотез, множественная регрессия, интерактивная пошаговая регрессия, моделирование Монте-Карло, статистическое управление процессами, планирование эксперимента, статистические графики и пр. Предусмотрено множество интерактивных инструментов для динамической визуализации и анализа данных.

   д) Optimization Toolbox – пакет программ, реализующих широко известные методы минимизации и максимизации линейных и нелинейных функций: безусловную

оптимизацию нелинейных функций, метод наименьших квадратов и нелинейную интерполяцию, решение нелинейных уравнений, линейное программирование, квадратичное программирование, условную минимизацию нелинейных функций, метод минимакса, многокритериальную оптимизацию и пр. В пакет включены версии традиционных и новейших алгоритмов оптимизации.

е) Partial Differential Equations Toolbox – пакет программ, содержащий множество функций для решения систем дифференциальных уравнений в частных производных. Используется метод конечных элементов. Команды и графический интерфейс пакета могут быть использованы для решения широкого класса инженерных задач, включая задачи сопротивления материалов, задачи тепломассопереноса, расчеты электромагнитных устройств.

ж) Fuzzy Logic Toolbox – пакет программ, относящихся к теории нечетких (размытых) множеств. Пакетом обеспечивается поддержка современных методов нечеткой кластеризации и адаптивных нечетких нейронных сетей. Графические средства пакета позволяют интерактивно отслеживать особенности поведения системы.

з) Neural Networks Toolbox – пакет программ, обеспечивающих всестороннюю поддержку проектирования, обучения и моделирования множества известных сетевых парадигм. Пакет может быть использован для исследования и применения нейронных сетей к таким задачам, как обработка сигналов, нелинейное управление и финансовое моделирование. В пакет включено более 15 типов сетей и обучающих правил.

и) Financial Toolbox – пакет прикладных программ по финансово-экономическим расчетам. Пакет содержит множество функций по расчету сложных процентов, операций по банковским вкладам, вычислению прибыли и др. К основным возможностям пакета относятся обработка данных, дисперсионный анализ эффективности портфеля инвестиций, анализ временных рядов, расчет доходности ценных бумаг и оценка курсов, статистический анализ, анализ чувствительности рынка, калькуляция ежегодного дохода и расчет денежных потоков, начисление износа и амортизационных отчислений. В пакет включены функции для манипулирования датами и временем в различных форматах, а также средства для презентации данных и результатов в виде графиков и диаграмм. До установки пакета на компьютер следует установить Excel, к которому должен быть указан путь при инсталляции системного приложения Excel Link.

2. Пакеты для обработки сигналов и изображений.

а) Signal Processing Toolbox – пакет, обеспечивающий широкие возможности создания программ обработки сигналов. Используется разнообразная техника фильтрации и алгоритмы спектрального анализа. Пакет можно использовать в таких областях, как обработка аудио и видеоинформации, телекоммуникации, геофизика, задачи управления в реальном режиме времени, экономика, финансы и медицина.

б) Image Processing Toolbox – пакет программ для цифровой обработки и анализа изображений. К возможностям пакета относятся восстановление и выделение деталей изображения, фильтрация сигналов изображения, работа с выделенным участком, анализ изображения, увеличение контрастности деталей, цветовые и геометрические преобразования, преобразование типов изображений. Пакет обеспечивает гибкий интерфейс для визуализации набора данных и обработки графических объектов (в том числе и трехмерных).

в) Wavelet Toolbox – пакет программ для исследования многомерных нестационарных явлений с помощью вейвлетов. Пакет позволяет анализировать такие особенности, которые упускают другие методы анализа сигналов: тренды, выбросы, разрывы в производных высоких порядков. Он позволяет сжимать и фильтровать сигналы без явных потерь. Программы сжатия выделяют минимальное число коэффициентов, представляющих исходную информацию наиболее точно. Новейшие методы пакета расширяют возможности в тех областях, где применяется техника Фурье-разложения.

г) Mapping Toolbox – пакет программ для работы с научными географическими данными. Пакет предоставляет графический и командный интерфейс для анализа данных, отображения карт и доступа к внешним источникам данных. Пакет включает более 60 проекций карт (прямые и инверсные), позволяет проектировать и отображать векторные, матричные и составные карты, трехмерные карты со встроенными средствами подсветки и затенения, глобальные и региональные атласы данных. Содержатся функции графической статистики и навигации, полезные при решении задач перемещения, таких как позиционирование и планирование маршрутов. В состав пакета входят широко известные атласы мира, США, астрономические атласы. Географическая структура данных упрощает извлечение данных из карт и их обработку. Используются конверторы для популярных форматов географических данных: DCW, TIGER, ETOPO5.

д) Virtual Reality Toolbox – пакет виртуальной реальности (доступен начиная с версии MATLAB 6.1). Позволяет осуществлять трехмерную анимацию и мультипликацию, в том числе моделей Simulink.

е) Curve Fitting Toolbox – новый математический пакет для подгонки (аппроксимации) кривых в графическом окне. Для одного и того же набора данных возможно проведение нескольких видов подгонки (полиномами от первой до девятой степени). Пакет полезен при решении типовых задач приближения данных.

ж) Dials & Gauges Blockset – новый пакет для разработки сложных моделей устройств и систем с повышенной степенью визуализации, таких, как приборные доски автомобилей и самолетов, диспетчерские пульты электростанций и аэропортов, пульты управления оборудованием и пр. Отличительная черта пакета – повышенная реалистичность отображения блоков (например, вольтметр изображается не простым прямоугольником, а прибором с четкой шкалой и движущейся стрелкой).

з) Higher-Order Spectral Analysis – пакет, содержащий специальные алгоритмы самых передовых методов для анализа сигналов с использованием моментов высшего порядка. Пакет позволяет анализировать сигналы, поврежденные негауссовым шумом, и процессы, происходящие в нелинейных системах.

и) Frequency Domain System Identification – пакет, предоставляющий специализированные средства для идентификации линейных динамических систем по их временному или частотному отклику. Частотные методы направлены на идентификацию непрерывных систем, что является мощным дополнением к традиционной дискретной методике. Методы пакета применимы к таким задачам, как моделирование электрических, механических и акустических систем. Функции для тестирования модели включают вычисление невязок, передаточных функций и нулей/полюсов, прогонку модели с использованием тестовых данных.

3. Пакеты анализа и синтеза систем управления.

а) Control System Toolbox – пакет программ для моделирования, анализа и проектирования систем автоматического управления, как непрерывных, так и дискретных. Пакет реализует традиционные методы передаточных функций и современные методы пространства состояний. Содержит полный набор средств для анализа MIMO-систем, временные характеристики, частотные характеристики (диаграммы Боде, Николса, Найквиста и др.), характеристики моделей (управляемость, наблюдаемость, понижение порядка), реализует проектирование LQR/LQE-регуляторов, разработку обратных связей, поддержку систем с запаздыванием. Пакет обладает настраиваемым окружением и позволяет создавать собственные m-файлы.

б) Robust Control Toolbox – пакет программ для проектирования и анализа многопараметрических устойчивых систем управления. К функциям пакета относятся синтез LQG-регуляторов на основе минимизации равномерной и интегральной нормы, многопараметрический частотный отклик, построение модели пространства состояний, преобразование моделей на основе сингулярных чисел, понижение порядка модели, спектральная факторизация.

в) Model Predictive Control Toolbox – пакет программ для реализации стратегии предиктивного (упреждающего) управления. Пакет обеспечивает моделирование, идентификацию и диагностику систем, поддержку MISO, MIMO, переходных характеристик, моделей пространства состояний, системный анализ, конвертирование моделей в различные формы представления (пространство состояний, передаточные функции), предоставление учебников и демонстрационных примеров. Пакет включает две функции Simulink, позволяющие тестировать нелинейные модели.

г) Communications Toolbox – пакет прикладных программ для построения и реализации телекоммуникационных устройств. Пакет содержит более 100 функций MATLAB-типа и 150 блоков SIMULINK-типа, обеспечивающих кодирование и оцифровку, контроль ошибок при кодировании, модуляцию и демодуляцию, фильтрацию при передаче и приеме, синхронизацию, коллективный доступ, вычисление в полях Галуа, генерацию сигналов, анализ и построение графиков.

д) μ-Analysis and Synthesis – пакет программ для проектирования устойчивых систем управления. Пакет использует оптимизацию в равномерной норме и сингулярный параметр μ. Полноценный графический интерфейс упрощает операции с блоками при проектировании оптимальных ресурсов, предоставляя возможность устанавливать диапазон вводимых данных и редактировать D-K-итерации для приближенного μ-синтеза. Содержатся средства понижения порядка модели. Пакет поддерживает работу с непрерывными и дискретными моделями.

е) Quantitative Feedback Theory Toolbox – пакет функций и программ для создания устойчивых систем с обратной связью. Пакет базируется на методах количественной теории обратной связи (QFT), использующей частотное представление моделей для удовлетворения различных требований к качеству при наличии неопределенных характеристик объекта. Графический интерфейс пользователя оптимизирует процесс нахождения параметров обратной связи. Пакет позволяет вычислять различные параметры обратных связей и фильтров, проводить тестирование регуляторов как в непрерывном, так и в дискретном пространстве.

ж) LMI Control Toolbox – пакет, предоставляющий интегрированную среду для постановки и решения задач линейного программирования, таких, как задачи совместности ограничений, минимизация линейных целей при наличии линейных ограничений, минимизация собственных значений. К возможностям пакета относятся задание ограничений в символьном виде, графическое редактирование задач линейного программирования, многокритериальное проектирование регуляторов, проверка устойчивости (квадратичная устойчивость линейных систем, устойчивость по Ляпунову, проверка критерия Попова для нелинейных систем). Пакет является мощным средством для решения выпуклых задач оптимизации в различных областях (управление, идентификация, фильтрация, структурное проектирование, теория графов и пр.).

з) System Identification Toolbox – пакет, содержащий средства для создания математических моделей динамических систем на основе наблюдаемых входных и выходных данных. Методы идентификации, применяемые в пакете, пригодны для решения широкого класса задач, от проектирования систем управления и обработки сигналов до анализа временных рядов и вибрации. Пакет предусматривает предварительную обработку данных (фильтрацию, удаление трендов и смещений), выбор диапазона данных для анализа, анализ отклика во временной и частотной области, отображение нулей и полюсов передаточной функции системы, анализ невязок при тестировании модели, построение сложных диаграмм (диаграмма Найквиста и др.). Пакет поддерживает все традиционные структуры моделей (включая авторегрессию и структуру Бокса-Дженкинса), линейные модели пространства состояний, которые могут быть определены как в дискретном, так и в непрерывном пространствах. Число входов и выходов произвольно.

4. Некоторые другие пакеты.

а) MATLAB Compiler – компилятор программ, создаваемых на языке программирования системы MATLAB. Транслирует коды этих программ в программы на языке Си++. Использование пакета существенно повышает скорость выполнения программ и предоставляет возможность скрытия кодов для защиты запатентованных алгоритмов.

б) Excel Link – пакет, позволяющий использовать табличный процессор Microsoft Excel как процессор ввода-вывода MATLAB. Для этого необходимо установить в Excel файл excllink.xla как add-in функцию. Тогда при каждом запуске Excel будет появляться командное окно MATLAB, а на панели управления – дополнительные кнопки getmatrix, putmatrix и evalstring. Для ввода матрицы выделяется массив ячеек Excel, нажимается кнопка getmatrix и набирается имя переменной MATLAB. Для вывода используется кнопка putmatrix.

в) Data Acquisition Toolbox – пакет сбора данных через блоки, подключаемые к внутренней шине компьютера. Содержит средства для создания функциональных генераторов, виртуальных осциллографов, анализаторов спектра и пр. Новый блок Instrument Control Toolbox позволяет подключать приборы и устройства с последовательным интерфейсом.

г) Database Toolbox – пакет, осуществляющий обмен информацией с другими системами управления базами данных через драйверы ODBC или JDBC: Oracle 7.3.3, Access 95, Access 97, Microsoft SQL Server 6.5 или 7.0, Sybase Adaptive Server 11, Sybase SQL Server Anywhere 5.0, IBM DB2 Universal 5.0, Informix 7.2.2, Computer Associates Ingres (все версии). Все данные предварительно преобразуются в массив ячеек. Пакет позволяет, находясь в среде MATLAB, использовать команды языка запросов SQL для чтения и записи данных, выполнения простых и сложных запросов.

д) Report Generations Toolbox – пакет, позволяющий создавать стандартные и пользовательские отчеты из MATLAB и Simulink в различных форматах, включая HTML, RTF, XML и SGML. Отчеты могут содержать данные, переменные, функции, программы, модели и диаграммы, а также рабочие экраны или графики, сгенерированные с помощью М-файлов.

е) Notebook Toolbox - эффективный инструмент для создания интерактивных реальных документов в текстовом редакторе Word, соединяющий в себе возможности текстового процессора с вычислительными и графическими возможностями MATLABa. Документы, получаемые с помощью данного пакета (их называют М-книгами), включают текст, команды системы MATLAB и результаты их выполнения, которые вставляются в документ Word. Для работы с пакетом необходим, помимо редактора Word, шаблон M-book..dot, с помощью которого поддерживается разработка М-книги. При установке Notebook Toolbox в Word добавляется специальный пункт меню для операций с ячейками ввода команд MATLAB.

ж) MATLAB Runtime Server – вычислительный процессор MATLAB для поддержки функционирования системы в специальном режиме. Сервер поддерживает все вычислительные и графические возможности MATLAB, но не поддерживает интерфейс для ввода команд. Команда ядра MATLAB pcode переводит М-файлы в Р-файлы (файлы с защищенными кодами), которые специально предназначены для Runtime Server. Сервер позволяет превратить любое созданное пользователем приложение в независимый продукт и распространять его, не заботясь о сохранности кодов.

з) C Math Library (или C++ Math Library) – библиотека, позволяющая вставить мощные программы для численных расчетов MATLAB в приложения, написанные на языке С (или С++). Библиотека предназначена для создания автономных приложений, которые требуют использования эффективных математических функций MATLAB.

Часть 3. Примеры расчетов, задания для практических занятий.

Тема 1. Работа в режиме прямых вычислений.

Примеры: 1.1. Вычислить произведение полиномов $P(x) = x^5+x^3+1$ и $Q(x) = x^2+2x+3$.

```
>> P=[1 0 1 0 0 1];                          %вектор коэффициентов полинома P(x)
>> Q=[1 2 3];                                %вектор коэффициентов полинома Q(x)
>> pr=conv(P,Q)                                                    %произведение
pr = 1   2   4   2   3   1   2   3
```

1.2. Найти минимум функции $y = x^2-3x+2$ на отрезке [-5, 5].

```
>> y=inline('x.^2-3*x+2');                                    %функция пользователя
>> x=-5:0.1:5;                                                  %вектор значений x
>> x1=fminbnd(y,-5,5);                    %определение абсциссы x1 точки минимума
>> %форматированный вывод на экран значений x1 и y(x1)
>> fprintf(' ymin = %3.1f ',y(x1)); fprintf(' при x = %2.1f ',x1);
 ymin = -0.3  при  x = 1.5
```

1.3. Найти частичную сумму S_n ряда $\sum(-1)^{k+1}/k^2$. Сравнить ответ с точным значением суммы $S = \pi^2/12$.

```
>> n=input(' n=');                                %интерактивный ввод параметра n
 n=100
>> for k=1:n  a(k)= (-1)^(k+1)*k^(-2);  end          %вычисление слагаемых ряда
>> s=sum(a)                                                        %суммирование
s = 0.8224
>> pi^2/12
ans = 0.8225
```

1.4. Определить, находится ли точка M(1.8; 1.8) внутри области D, заданной неравенствами: $x^2+y^2 \geq 4$, $|x| \leq 2$, $|y| \leq 2$.

```
>> x=1.8;  y=1.8;
>> z=and(x^2+y^2>=4,and(abs(x)<=2,abs(y)<=2));
>> if  z = =1  disp('да');   else  disp('нет');  end
да
```

Задания: 1. Вычислить гамма-функцию $\Gamma(x)$ для x = `0.1,0.2,...,10.`

2. Вычислить все значения корня 6-й степени из комплексного числа z= – 64, используя формулу: $z^{1/n} = r^{1/n}[\cos((\varphi+2k\pi)/n) + i\sin((\varphi+2k\pi)/n)]$, где $k = 0, \ldots, n-1$; r, φ - модуль и аргумент числа z. Сравнить полученный результат с результатом выполнения команды x=z^(1/n).

Тема 2. Программирование, работа с файлами.

Примеры: 2.1. Вычислить среднее значение и дисперсию для элементов вектора x.

Первый шаг – создание функции stat:

```
function [mx,dx]=stat(x)                                  %объявление имени функции
n=length(x);                                          %вычисление длины вектора x
mx=sr(x,n);              %вычисление среднего значения  с помощью подфункции sr
dx=sum((x-sr(x,n)).^2)/n;                                  %вычисление дисперсии
function s=sr(x,n)                          %подфункция в теле основной функции
s=sum(x)/n;
```

Второй шаг – сохранение функции в виде m-файла под именем stat.m.
Третий шаг - использование функции stat в командной строке:

```
>> x=[-1,2.2,3.1,-4,5];
>> [mx,dx]=stat(x)
mx = 1.060
dx = 10.167
```

2.2. Построить таблицу значений функции y на отрезке [x1; x2] с шагом dx, записать ее в файл tab.txt и вывести таблицу на экран. Использовать программу для вычисления значений функции y = ln x на отрезке [1; 2] с шагом 0.1.

1 шаг - создание файла – функции:

```
function []=table(func,x1,x2,dx)
f=inline(func);          %функция пользователя, строка func задается в командном окне
x=x1:dx:x2;                                          %вектор значений аргумента
y=f(x);                                               %вектор значений функции
fid=fopen('tab.txt','wt');                        %открытие файла tab.txt для записи
if fid ~ = -1
  fprintf(fid,'------------------------\n')        %линия (можно также начертить линию с...
                                                    помощью функций blanks и strrep)
  fprintf(fid,'| %9s | %9s |\n','x','y');          %вывод названий столбцов
  fprintf(fid,'------------------------\n');                                  %линия
  fprintf(fid,'| %1.5f  | %1.5f  |\n',[x y]);                    % вывод чисел
  fprintf(fid,'------------------------\n');                                  %линия
  fclose(fid);                                                          %закрытие файла
  fid= fopen('tab.txt','rt');          % открытие файла tab.txt для чтения
  while~feof(fid)                                                    % до конца файла
    stroka=fgetl(fid);                                  %считывание строк таблицы
    disp(stroka) %вывод на экран
  end
  fclose(fid);                                                          %закрытие файла
else fprintf('Ошибка при открытии файла!')
end
```

2 шаг – сохранение файла под именем table.m.

3 шаг - построение таблицы для функции y=ln x:

```
>> x1=1; x2=2; dx=.1;
>> func='log(x)';
>> table(func,x1,x2,dx)
```

x	y
1.00000	0.00000
1.10000	0.09531
1.20000	0.18232
1.30000	0.26236
1.40000	0.33647
1.50000	0.40547
1.60000	0.47000
1.70000	0.53063
1.80000	0.58779
1.90000	0.64185
2.00000	0.69315

2.3. Подсчитать количество положительных чисел в произвольном числовом массиве, вывести положительные числа на экран.

1 шаг - создание файла – сценария:

```
x=input('Исходный массив чисел:');          %ввод исходного числового массива
y=x(:)>0;                                   % логический массив
if any(y)
     s=sum(y);
     string=['В массиве содержится  ', num2str(s), '  положительных чисел'];
     disp(string)
     x(y')
 else
     string=['В массиве нет положительных чисел'];
     disp(string)
 end
```

2 шаг – сохранение файла под именем prim2_3.m.

3 шаг - проверка работы программы:

```
>> prim2_3
Исходный массив чисел: [-1,2.2,3.1,-4,5,8,9.3]
В массиве содержится  5 положительных чисел
ans = 2.2  3.1  5  8  9.3
```

Задания: 1. Написать программу, считывающую из некоторого текстового файла заданное число строк и выводящую эти данные в командное окно.

2. Написать программу - сценарий, преобразующую массив чисел из десятичной системы счисления в двоичную (использовать функцию dec2bin( )).

3. Написать программу - функцию, вычисляющую направляющие косинусы заданного трехмерного вектора.

Тема 3. Вычисление корней полинома и нулей функции.

Примеры: 3.1. Вычислить все корни полинома $P(x) = x^7 + 3.2\,x^5 - 5.2\,x^4 + 0.5\,x^2 + x - 3$. Для одного из корней сделать проверку.

```
>> p=[1 0 3.2 –5.2 0 0.5 1 -3];                % коэффициенты полинома
>> r=roots(p)                                  % вычисление корней
r=
   -0.5668 + 2.0698i
   -0.5668 - 2.0698i
   -0.6305 + 0.5534i
   -0.6305 - 0.5534i
    1.2149
    0.5898 + 0.6435i
    0.5898 - 0.6435i
>> polyval(p,1.2149)          % вычисление значения полинома при x=1.2149
 ans =
    0.0000
```

3.2. Найти нули функции $y = 0.25x + \sin x - 1$ на отрезке [0; 10].

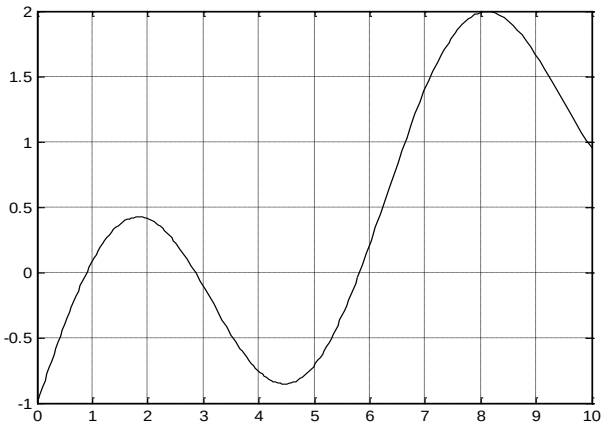
1 шаг – создание функции:

```
function f=fun1(x)
f=0.25*x+sin(x)-1;
```

2 шаг – сохранение файла fun1.m

3 шаг – работа в командном окне:

```
>> fplot(@fun1,[0,10]);   %построение графика функции fun1 на отрезке [0; 10]...
                                  для приближенного определения нулей функции
>> grid on;                                    % координатная сетка
```

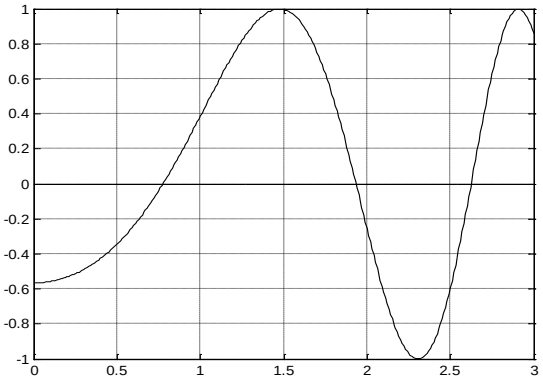


```
>> x1=fzero(@fun1,[0 1])          %вычисление нуля функции на отрезке [0; 1]
x1 =
   0.8905
>> x2=fzero(@fun1,[2 3])          %вычисление нуля функции на отрезке [2; 3]
x2 =
   2.8500
>> x3=fzero(@fun1,5,0.001)        %вычисление нуля функции при x ≥ 5,...
                                                              точность=0.0001
x3 =
   5.8128
```

3.3. Решить уравнение: $\sin(x^2 - 0.6) = 0$ на отрезке [0; 3].

1. Графический способ:

```
>> x=0:.01:3;                                          %значения аргумента
>> f=sin(x.^2-0.6);                                    %значения функции
>> plot(x,[f;0*f])                                     % графики функций y=f и y=0
>> grid on;                                            %координатная сетка
```



```
>> x1=ginput        %интерактивный вывод на экран координат точки (щелчок...
                                  мыши по нужной точке и нажатие клавиши enter)
 x1  =
      0.7746    -0.0012          %второе число соответствует координате y1≈0
 >> g2=ginput
 x2 =
      1.9343     0.0023
 >> x3= fzero(f,[2 3])
 x3  =
      2.6326
```

2. Аналитический способ:

```
>> x=0:.01:3;                                          %значения аргумента
>> n=length(x);                                        %вычисление длины вектора x
```

```
>> ind=1:n-1;                                  %вектор индексов
>> f=sin(x.^2-0.6);                            % значения функции
>> korni=x(f(ind).*f(ind+1)<=0)                %сравниваются знаки функции на концах...
                                               отрезка, вектор korni попадают только те...
                                               значения x_i , для которых f(x_i) и f(x_i+1)...
                                               имеют разные знаки
 korni =
         0.7746   1.9300   2.6200
```

Задание: Составить программу для решения алгебраического уравнения $f(x) = 0$ методом Ньютона (итерационная процедура по формуле: $x_{k+1} = x_k - f(x_k) / f'(x_k)$, начальное значение x_0,функция f и ее производная f ' задаются в командном окне). Применить программу к решению уравнения из примера 3.3.

Тема 4. Работа с матрицами, системы линейных алгебраических уравнений.

Примеры: 4.1. Вычислить алгебраические дополнения к элементам матрицы A и обратную к этой матрице (A – квадратная матрица третьего порядка). Результат сравнить с функциями inv(A) и A^(-1).

```
function [C,C1,C2]=obr(A)     % объявление функции, сохраняемой по именем obr.m
d=det(A)                                    %вычисление определителя матрицы
if d==0 disp('матрица вырожденная');
else
        A11=det(A([2,3],[2,3]));
        A12=-det(A([2,3],[1,3]));
        A13=det(A([2,3],[1,2]));
        A21=-det(A([1,3],[2,3]));
        A22=det(A([1,3],[1,3]));
        A23=-det(A([1,3],[1,2]));
        A31=det(A([1,2],[2,3]));
        A32=-det(A([1,2],[1,3]));
        A33=det(A([1,2],[1,2]));
        B=[A11,A12,A13;A21,A22,A23;A31,A32,A33];
        C=B'/d;
        C1=inv(A);
        C2=A^(-1);
end                          %конец оператора if, далее сохраняем функцию в файле obr.m
```

Работа в командном окне:

```
>> A=[1 2 1;2 11 1;2 1 2]                   %исходная матрица
>> [C,C1,C2]=obr(A)                         %вызов функции
C =
   -7.0000    1.0000    3.0000
    0.6667    0.0000   -0.3333
    6.6667   -1.0000   -2.3333
C1 =
   -7.0000    1.0000    3.0000
    0.6667    0.0000   -0.3333
    6.6667   -1.0000   -2.3333
C2 = C1
```

4.2. Решить систему уравнений:

$$\begin{aligned} 2x + y - 5z + t &= 8,\\ x - 3y - 6t &= 9,\\ 2y - z + 2t &= -5,\\ x + 4y - 7z + 6t &= 0. \end{aligned}$$

```
>> A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];                    %матрица системы
>> B=[8;9;-5;0];          %вектор – стобец правых частей  (может быть матрицей)
>> A1=[A,B];                                                  %расширенная марица системы
>> if  and(rank(A)=rank(A1),rank(A)=4)                        %проверка ранга матрицы
       disp('система имеет единственное решение:');
       x=A\B;              % обратный слэш или деление слева – решение  линейной...
                                                                  системы методом Гаусса
       x1=x';
   end
x1 =
   3.0000   -4.0000   -1.0000   1.0000
>> x=A^(-1)*B;  x2=x'                                         %второй  вариант  записи A\B
x2 =
   3.0000   -4.0000   -1.0000   1.0000
>> x=inv(A)*B;  x3=x'                                         %третий  вариант  записи A\B
x3 =
   3.0000   -4.0000   -1.0000   1.0000
```

4.3. Решить систему из примера 4.2 методом наименьших квадратов.

```
>> A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];                    %матрица системы
>> B=[8;9;-5;0];                                  % вектор – стобец правых частей
>> x=lsqr(A,B)              % встроенная функция решения  линейной  системы...
                                                        (метод  наименьших  квадратов)
x =
   3.0000   -4.0000   -1.0000   1.0000
```

Задания: **1.** Решить систему из примера 4.2 с помощью формул Крамера.

2. Решить систему нелинейных уравнений:

$$\begin{aligned} 2x - y^2 + z &= 1,\\ y^2 - z^2 &= 0,\\ 2y - 3z &= 1, \end{aligned}$$

для решения применить метод последовательных приближений (итерационная процедура $x_{k+1} = B\backslash A(x_k)$).

Тема 5. Численное дифференцирование и интегрирование.

Примеры: **5.1.** Вычислить производную полинома $P(x) = x^5 + x^3 + 1$.

```
>> P=[1 0  1 0 0 1];                                  %вектор коэффициентов исходного
полинома
>> P1=polyder(P)                                %вектор коэффициентов  производной P' (x)
P1 =
     5   0   3   0   0
```

5.2. Вычислить приближенное значение производной функции $y = \sin x$ и убедиться, что $y'(a) \approx \cos a$.

```
>> x=0:.05:10;                                                %вектор аргументов
>> y=sin(x);                                                  %вектор значений функции
>> d=diff(y);       %вектор разностей соседних элементов: d=[y(2)-y(1),...,y(n)-y(n-1)]
>> pr=d/0.05;                                                 %вектор значений производной
>> pr(5)                                          %значение производной при x=x(5)=0.2
0.9747
 >> cos(x(5))                                                 %сравнение с точным результатом
0.9801
```

5.3. Найти приближенное значение производной функции $y = \ln(1-x)$ в виде полинома 5-й степени. Вычислить $y'(0.6)$.

```
>> h=.01;  x=0:h:0.8;                                  %значения аргумента
>> y=log(1-x);                                          %значения функции
>> y1=diff(y)/h;                                        %значения производной
>> p=polyfit(x(1:length(n)-1,y1,5)  %коэффициенты аппроксимирующего полинома
p =
 -52.0782   75.5962   -43.1952   8.8005   -1.8620   -0.9894
>> c=polyval(p,0.6)                                    %значение полинома p при x=0.6
c =
     -2.5209                                              %точное значение  c= - 2.5
```

5.4. Вычислить интеграл двумя способами: $\int_0^1 (e^{2x} - 1)\, dx$.

```
1 способ – метод трапеций:
>> h=0.001;     x=0:h:1;
>> y=exp(2*x)-1;
>> int=trapz(y)*h
int =
    2.1945
2 способ – метод Симпсона:
>> int=quad('exp(2*x)-1',0,1,1.0e-5)
int = 2.1945
```

Задания: **1.** Вычислить приближенное значение второй производной функции y = tg x в точке x = π/4, результат сравнить с точным значением.

2. Вычислить различными способами: $\int_0^{\pi} dx \int_0^{2\pi} (2y\sin x + x\cos y)\, dy$.

а) методом повторного интегрирования;

б) с помощью функции dblquad( );

в) с помощью приближенной формулы $\text{int} \approx \sum\sum f(x_i, y_k)\, \Delta x_i\, \Delta y_k$.

`

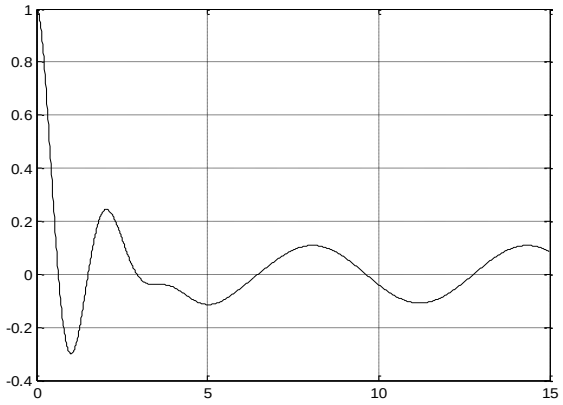
Тема 6. Численное решение дифференциальных уравнений.

Примеры: 6.1. Решить задачу Коши: y'' + 2y' +10y = sin x; y(a) = c, y'(a) = d, x ∈[a,b], к решению применить метод Эйлера, построить график функции y(x).
Указание: записать уравнение в виде системы: y' = z,
z' = -2z -10y + sin x,
разбить отрезок [a, b] на n частей $[x_k, x_{k+1}]$ и применить формулу $V_{k+1}=V_k + h*F(x_k, y_k)$, где V - вектор неизвестных, F - вектор правых частей, h - длина интервала, V_0 = [c; d].

```
function y=solvejler(a,b,y0,h)                          % начало  файла solvejler.m
xt=a;  yt=y0;   k=1;              %начальные значения текущих переменных xt, yt
while xt<b;
        x(k)=xt;
        y(k)=yt(1);
        k=k+1;
        c=[yt(2);-2*yt(2)-10*yt(1)+sin(xt)];              %вектор правых частей
        yt=yt+h*c;
        xt=xt+h;
end
plot(x,y); grid                          %график решения, сетка;  конец m-файла
```

Работа в командном окне:

```
>> a=0; b=15; y0=[1;0]; h=0.01;
>> solvejler(a,b,y0,h)                                  %вызов функии
```

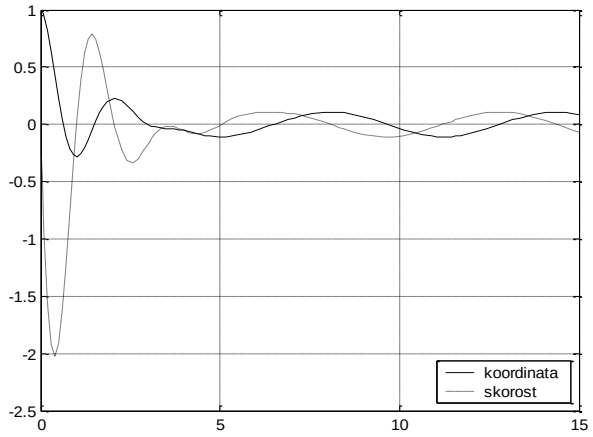


6.2. Решить задачу примера 6.1 методом Адамса с помощью стандартной функции ode113( ). Построить графики функций y(x) и y'(x).

```
function   F=primer(x,y)                    %m-файл, задающий вектор правых частей
F=[y(2);-2*y(2)-10*y(1)+sin(x)];            %конец m-файла
```

Работа в командном окне:

```
>> y0=[1;0];
>> [X,Y]=ode113(@primer,[0 15],y0);          %решение задачи Коши
>> plot(X,Y(:,1),'-')                        %график функции y(x)
>> hold on                                   %режим добавления графиков в текущее окно
>> plot(X,Y(:,2),':')                        %график функции y' (x)
>> grid                                      %координатная сетка
>> legend('koordinata','skorost',4)          %легенда
```

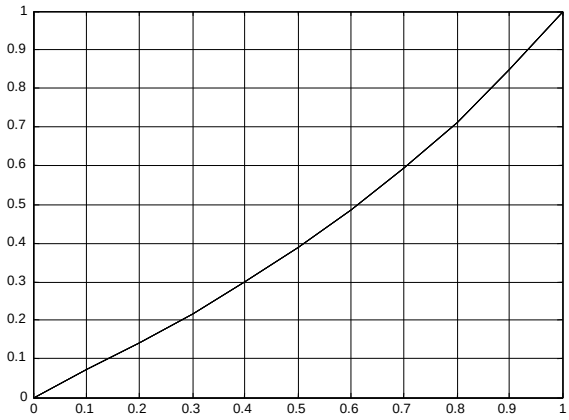


6.3. Решить краевую задачу: $y'' - y = x$; $y(0) = 0$, $y(1) = 1$.

Указание: разбить отрезок [0; 1] на n равных частей. Производные $y''(x_k)$ заменить конечными разностями $y''_k = (y_{k-1}-2y_k+y_{k+1}) / h^2$, $h = x_{k+1} - x_k$, $k \geq 2$, решить полученную линейную систему Ay=B. Построить график решения и сравнить его с графиком точного решения y = c1*exp(x) + c2*exp(-x) – x, где $c1 = 2e/(e^2-1)$, c2 = - c1.

```
>> n=11; h=1/(n-1);   x=0:h:1;                          %разбиение отрезка
>> B=[0,h*h*x(2:n-1),1]';                               %вектор правых частей системы
>> k=-2-h^2;
>> A=[1 0 0 0 0 0 0 0 0 0 0;1 k 1 0 0 0 0 0 0 0 0;..   %матрица линейной системы
 0 1 k 1 0 0 0 0 0 0 0 ;0 0 1 k 1 0 0 0 0 0 0;...
 0 0 0 1 k 1 0 0 0 0 0;0 0 0 0 1 k 1 0 0 0 0;...
 0 0 0 0 0 1 k 1 0 0 0;0 0 0 0 0 0 1 k 1 0 0;...
 0 0 0 0 0 0 0 1 k 1 0;0 0 0 0 0 0 0 0 1 k 1;...
 0 0 0 0 0 0 0 0 0 0 1];
```

```
>> y1=A\B;                                  %решение линейной системы
>> plot(x,y1);  grid                        %график приближенного решения
>> hold on                                  %режим добавления графиков
>> c1=2*exp(1)/(exp(1)^2-1);  c2=-c1;       %коэффициенты точного решения
>> y2=c1*exp(x)+c2*exp(-x)-x;               %точное решение
>> plot(x,y2)          %график точного решения (совпадает с графиком y1)
```



Другой вариант формирования матрицы A:

```
>> v=[k,1,zeros(1,n-2)];
>> A=toeplitz(v);          %матрица Тёплица с одним аргументом (см. [2])
>> A(1,1)=1;  A(1,2)=0;
>> A(n,n-1)=0;  A(n,n)=1;
```

Задание: Решить краевую задачу из примера 6.3 двумя способами:

1) с помощью стандартной функции bvp4c( );

2) с помощью метода начальных параметров, т.е. сведением краевой задачи V' (x)= A(x)V(x)+B(x), V=[y(x), y'(x)]; y(a)=y1, y(b)=y2, к решению трех задач Коши. Решение ищется в виде суммы $V(x) = V_0(x) + c1 * V_1(x) + c2*V_2(x)$, где V_0 – решение уравнения V'=AV+B с нулевыми начальными условиями $V_0(0) = [0, 0]$; V_1 и V_2 – решения уравнения V'=AV, $V_1(0) = [1, 0]$, $V_2(0) = [0, 1]$. Коэффициенты c1, c2 определяются из граничных условий.

Тема 7. Использование прикладных пакетов.

Примеры: 7.1. Финансовая задача. Величина кредита составляет $1000. Он должен быть погашен за 12 месяцев тремя платежами. Заемная процентная ставка составляет 10%. Вычислить величину одного платежа. При решении использовать функцию payadv (rate, nper, pv, fv, adv) пакета Financial Toolbox, где rate – величина заемной ставки, nper - количество месяцев, pv - текущая стоимость финансового документа, fv – целевая стоимость, adv – количество платежей.

```
>>  vel= payadv(0.1/12,12, 1000, 0,3);
>>  string=['Вы должны выплачивать по', vel, '  долларов ', adv, '  раза!'];
>> disp(string)
Вы должны выплачивать по  85.9389   долларов  3 раза!
```

Графический интерфейс для работы с этой функцией в диалоговом режиме разработан в примере 10.1.

7.2. Вычислить в символьном виде, используя функции пакета Symbolic Toobox:

а) производную от функции $y = x^3$

```
>> sym  x;
>> diff('x^3')
ans =
       3*x^2
```

б) первообразную от функции z = 2sin y

```
>> sym y;
>> int('2*sin(y)')
ans =
       -2*cos(y)
```

в) решения уравнения x^2=b

```
>> syms x b
>> solve(x^2-b)
ans =
      [ b^1/2]
      [-b^1/2]
```

г) решения квадратного уравнения $ax^2 + bx + c = 0$

```
>> syms a b c x
>> x=solve ('a*x^2+b*x+c=0')
x =
    [1/2*a*(-b+(b^2-4*a*c)^(1/2))]
    [1/2*a*(-b- (b^2-4*a*c)^(1/2))]
```

Задания: 1. С помощью функции nchoosek(n,k)=C_n^k построить разложение бинома $(x+y)^n$ для n=10. Ответ записать в символьном виде.

2. Даны два числовых массива x и y. Построить график интерполирующей функции y(x), используя функции пакета Spline Toolbox (см. [3]).

Тема 8. Структуры.

Примеры: 8.1. Создать структуру **exam,** состоящую из пяти полей: fi (фамилия имя), ng (номер группы), phys (оценка по физике), chem (оценка по химии), math (оценка по математике).

```
>> exam.fi='Петров  Петр';
>> exam.ng='921';
>> exam.phys= 5;
>> exam.chem= 3;
>> exam.math= 2;
```

8.2. К созданной в примере 8.1. структуре добавить еще две записи.

```
>> exam(2).fi='Иванов  Иван';
>> exam(2).ng='922';
>> exam(2).phys= 4;
>> exam(2).chem= 3;
>> exam(2).math= 3;
>> exam(3).fi='Сидоров  Сергей';
>> exam(3).ng='921';
>> exam(3).phys= 3;
>> exam(3).chem= 4;
>> exam(3).math= 5;
```

8.3. Подсчитать среднюю оценку одного из студентов.

```
>>  k=input('Номер студента в базе:  ');
Номер студента в базе:  3
>> s1=exam(k).phys;  s2=exam(k).chem;  s3=exam(k).math;
>> sred=num2str((s1+s2+s3)/3);          %вычисление среднего и перевод в...
                                                          символьный вид
>> f=exam(k).fi;                        %значение поля fi для записи с номером k
>> g=getfield(exam(k),'n                %значение поля ng для записи с номером k
>> string=[f,':  ',sred];
>> disp(string)
Сидоров  Сергей:  4.0000
```

8.4. Упорядочить поле fi структуры **exam** по алфавиту.

```
>> s=exam.fi;
>> s1=oderfields(s);
>> exam.fi=s1
ans =  Иванов  Иван
       Петров  Петр
       Сидоров  Сергей
```

Задания: **1.** Дополнить структуру **exam** до десяти записей, по пять в каждой из учебных групп 921 и 922, отсортировать структуру по возрастанию номера группы.

2. Подсчитать среднюю оценку группы 921.

Тема 9. Графика.

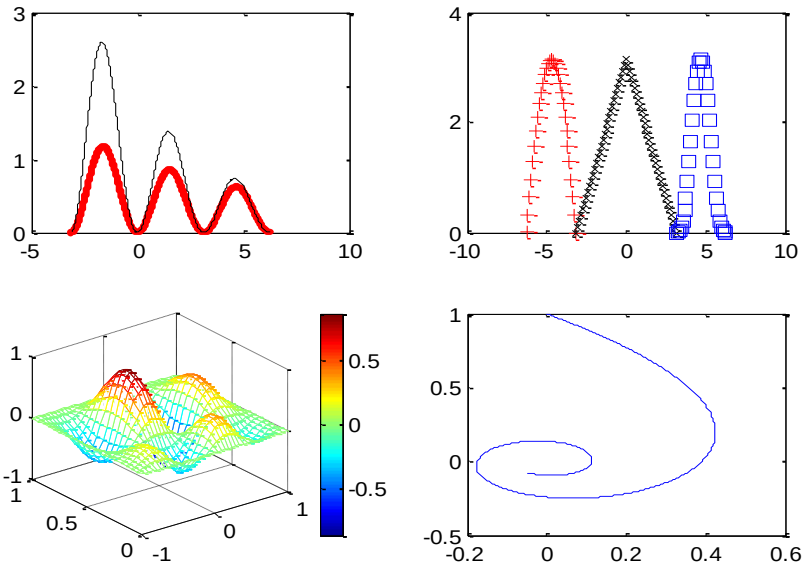
Примеры: 9.1. Разбить текущее графическое окно на четыре подокна и в каждом из них построить график: 1 – два графика функции одной переменной в одном графическом пространстве, 2 – график кусочно - заданной функции, 3 – график поверхности, 4 – график с эффектом анимации.

1.
```
>> subplot(2,2,1)          %открытие первого подокна (всего подокон...
                                              четыре - две строки и два столбца)
>> x=-pi:0.01:2*pi;
>> f=exp(-0.1*x).*sin(x).^2;
>> y=-2*pi:0.01:pi;
>> g=exp(-0.2*y).*sin(y).^2;
>> plot(x,f,'r',x,g,'k-')
```

2.
```
>> subplot(2,2,2);
>> x1=-2*pi:pi/30:-pi;
>> y1=pi*sin(x1);
>> x2=-pi:pi/30:pi;
>> y2=pi-abs(x2);
>> x3=pi:pi/30:2*pi;
>> y3=pi*sin(x1).^3;
>> x=[x1 x2 x3];
>> y=[y1 y2 y3];
>> plot(x1,y1,'r+',x2,y2,'kx',x3,y3,'bs')
```

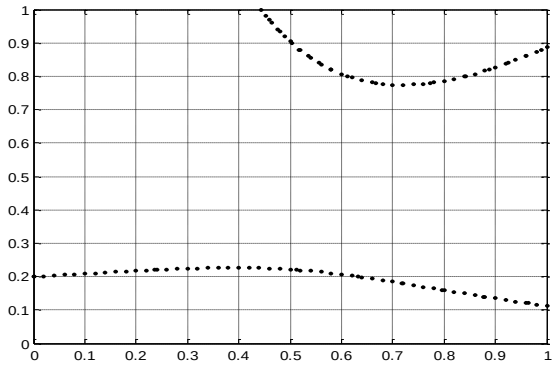
3.
```
>> subplot(2,2,3);
>> [X,Y]=meshgrid(-1:0.05:1,0:0.05:1);          %проекции узловых точек...
                                                                  поверхности
>> Z=4*sin(2*pi*X).*cos(1.5*pi*Y).*(1-X.^2).*Y.*(1-Y);
>> mesh(X,Y,Z);
>> colorbar                                                    %шкала цветов
```

4. >> subplot(2,2,4);
>> t=[0:0.001:10];
>> x=sin(t)./(t+1);
>> y=cos(t)./(t+1);
>> comet(x,y); %эффект кометы



9.2. Построить график функции, заданной неявно: $x^3y - 2xy^2 + y - 0.2 = 0$ в квадрате $|x| \le 1, |y| \le 1$ (см. [2]).

>> h=0.02; x=0:h:1;
>> [X,Y]=meshgrid(x);
>> f=X.^3.*Y-2*X.*Y.^2.+Y-.2;
>> v=[0 0]; contour(x,x,f,v); grid %линия уровня поверхности... z=f(x,y) при z=0



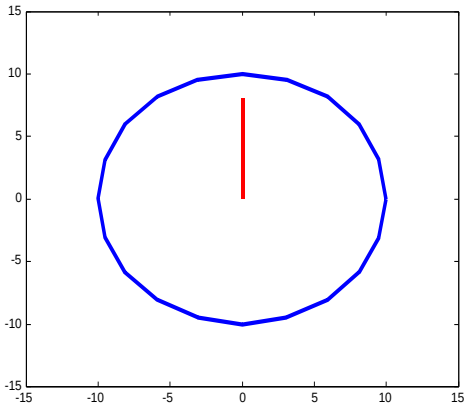
9.3. Построить график поверхности z = f(x,y) (f(x,y) – левая часть уравнения из примера 9.2) с нанесенным на нем рисунком из файла uzor.jpg. В соседнем подокне изобразить исходный (заранее подготовленный) рисунок.

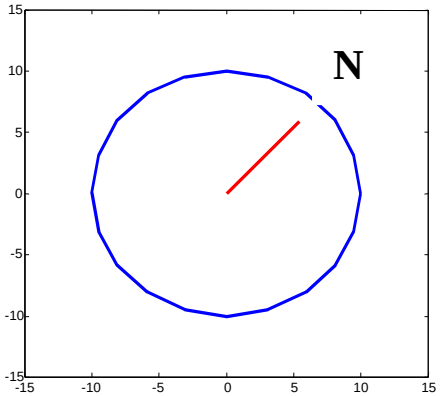
>> h=0.02; x=0:h:1;
>> [X,Y]=meshgrid(x);
>> Z=X.^3.*Y-2*X.*Y.^2.+Y-.2;
>> A=imread('uzor','jpg'); %считывание рисунка из файла, занесение... данных в графический объект A
>> subplot(1,2,1);
>> image(A) %вывод рисунка
>> axes off
>> subplot(1,2,2);

```
>> surf(X,Y,Z,'CData',A,'FaceColor','texture')        %график поверхности
>> axes off
```

9.4. Изобразить движение стрелки компаса при изменении положения точки N (север). Точку N вывести в диалоговом режиме с помощью функции gtext( ).

```
>> f=figure('Name','Kompas');                    %открытие графического окна
>> %рисование окружности
>> plot(10*cos(0:pi/10:2*pi),10*sin(0:pi/10:2*pi),'LineWidth',3);
>> hold on                                       %режим добавления графиков
>> axis([-15,15,-15,15]);                        %границы осей
>> x1=[0 0]; y1=[0 8];                           %координаты стрелки
>> h=plot(x1,y1,'LineWidth',3,'Color','r','EraseMode','xor');   %стрелка
>> a=gtext('N');                                 %ввести точку N
>> b=get(a);                %вывод графических характеристик объекта a
>> v=b.Position;
>> xn=v(1);    yn=v(2);                          %координаты точки N
>> R=sqrt(xn^2+yn^2);          %вычисление расстояния до точки N
>> alfa=-asin(xn/R);                             %угол поворота
>> xh=x1*cos(alfa)-y1*sin(alfa);                 %преобразование координат
>> yh=x1*sin(alfa)+y1*cos(alfa);
>> set(h,'XData',xh,'YData',yh)                  %новая стрелка
```



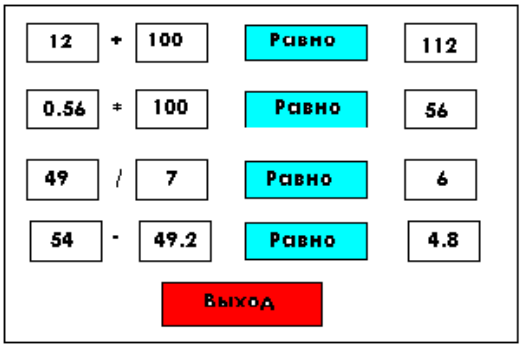


Задания: **1.** Изобразить циферблат часов с движущимися стрелками (см. [4]).

2. Построить график поверхности $z = 1 - x^2 - y^2$ при $0 < x < 1$, $0 < y < 1$ с использованием функций shading interp, diffuse, colormap( ) и вычислить объем, заключенный между указанной поверхностью и плоскостью $z = 0$ ($V \approx \sum\sum z(x_i, y_k)\,\Delta x_i\,\Delta y_k$).

Тема 10. Графический интерфейс пользователя (GUI).

Примеры: 10.1. Разработать графическое окно, реализующее функции калькулятора:



Разработка: 1 шаг – создание бланка (File-GUI-Blank)
для ввода цифр – объекты типа Edit (свойство Tag – a, b или rez);
для надписей + - * / – объекты типа Text ;
для кнопок равно и выход – объекты типа Pushbutton (Tag - ravno и Close).

2 шаг – сохранение бланка в файле под именем calcullyator.fig.

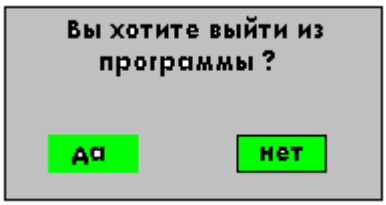
3 шаг – изменение программы (файл calculyator.m):

а) изменение кода кнопки равно для + (клавиша f – ravno_Callback):

```
function ravno_Callback(hObject,evendata,handles)
a=get(handles.a,'String');          %считывание первого слагаемого
a=str2double(a);                    %перевод из символьного вида в числовой
b=get(handles.b,'String');          %считывание второго слагаемого
b=str2double(b);                    %перевод из символьного вида в числовой
rez=a+b;                                                %суммирование
set(handles.rez,'String',rez);      %запись результата в объект rez
```

б) изменение кода кнопок равно для * / - :
по аналогии с пунктом а), но вместо a, b и rez соответственно a1, b1, rez1, a2, b2, rez2, a3,b3, rez3.

4 шаг – создание вспомогательного диалогового окна выход (File-GUI-Blank)



для надписи – объект типа Text ;
для кнопок да и нет – объекты типа Push Button (Tag - da и net).

5 шаг – сохранение вспомогательного окна в файле под именем danet.fig.
6 шаг – изменение программы диалогового окна (файл danet.m) :

а) изменение кода кнопки да (клавиша f – da_Callback):

```
function da_Callback(hObject,evendata,handles)
close;                                  %закрытие диалогового окна выход
Close_Callback=close;                                   %выход из программы
```

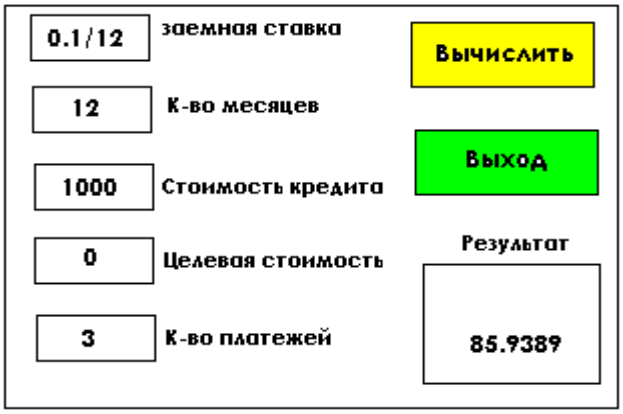
б) изменение кода кнопки нет (клавиша f – net_Callback):

```
function net_Callback(hObject,evendata,handles)
close;                                  %закрытие диалогового окна  выход
```

7 шаг – изменение кода кнопки выход: (клавиша f – Close_Callback):

```
function Close_Callback(hObject,evendata,handles)
danet;                                  %вызов функции danet
```

10.2. Разработать диалоговое окно для работы с функцией payadv( ) (пример 8.1).



Разработка: 1 шаг – создание бланка (File-GUI-Blank)
для ввода цифр–объекты типа Edit (свойство Tag – rate, nper, pv, fv, adv);
для надписей – объекты типа Text ;
для кнопок вычислить и выход–объекты типа Push Button (Tag-pusk и close)
для вывода результата – объект типа Text (Tag – pmt).

2 шаг – сохранение бланка в файле под именем primer2.fig.
3 шаг – изменение программы (файл primer2.m):
изменение кода кнопки вычислить (клавиша f – pusk_Callback):

```
function pusk_Callback(hObject,evendata,handles)
rate=get(handles.rate,'String');        %считывание первого аргумента
rate=eval(rate);                        %вычисление выражения
nper=get(handles.nper,'String');        %считывание второго аргумента
nper=str2double(nper);          %перевод из символьного вида в числовой
pv=get(handles.pv,'String');            %считывание третьего аргумента
pv=str2double(pv);              %перевод из символьного вида в числовой
fv=get(handles.fv,'String';             %считывание четвертого аргумента
fv=str2double(fv);              %перевод из символьного вида в числовой
adv=get(handles.adv,'String');          %считывание пятого аргумента
adv=str2double(adv);            %перевод из символьного вида в числовой
pmt=payadv(rate,nper,pv,fv,adv)         %вызов функции
set(handles.pmt,'String',pmt);          %запись результата в объект pmt
```

4 шаг – работа с диалоговым окном выход (см. пример 10.1).

Задание: Разработать GUI для просмотра нескольких графических окон.
Указание: создать объект PopupMenu, перечислить несколько характерных примеров (график функции одной переменной, эффект comet, график поверхности и пр.). Для вывода графического окна создать объект типа axes. В программе воспользоваться оператором выбора switch.

Приложение. Основные команды ядра системы MATLAB.

Команды общего назначения:

1) Общая информация: help, helpwin, helpdesk, demo, ver, whatsnew, readme.
2) Управление рабочим пространством: who, whos, clear, pack, load, save, quit, exit.
3) Управление командами и функциями: what, type, edit, lookfor, which, pcode, inmem, mex.
4) Управление путями поиска: path, addpath, rmpath, editpath.
5) Управление командным окном: echo, more, diary, format, clc, home.
6) Команды операционной системы: cd, copyfile, pwd, dir, delete, getenv, mkdir, !, dos, web, computer.
7) Отладка М-файлов: debug, dbstop, dbclear, dbcont, dbdown, dbstack, dbstatus, dbstep, dbtype, dbup, dbquit.
8) Профилирование М-файлов: profile.

Конструкции языка программирования:

1) Арифметические операторы: +, -, *, .*, ^, .^, \, /, .\, ./.
2) Операторы отношения: = =, ~ =, <, >, <=, >=.
3) Логические операторы: &, and, or, ~, not, xor, any, all.
4) Специальные символы: : ( ) [ ] { } . .. ... , ; ! = ‘ .’ [,] [;]
5) Форматы чисел: format name.
6) Поразрядные (битовые) операции: bitand, bitcmp, bitor, bitmax, bitxor, bitset, bitget, bitshift.
7) Специальные константы и переменные: ans, eps, realmax, realmin, pi, i, j, inf, NaN, isnan, isinf, isfinite, flops, why.
8) Операторы управления программой: if, else, elseif, end, for, while, break, switch, case, otherwise, try, catch, return.
9) Операторы вычисления и выполнения: eval, feval, evalin, builtin, assignin, run.
10) Операторы работы с аргументами функций: nargchk, nargin, nargout, varargin, varargout, inputname.
11) Вывод сообщений: error, warning, lasterr, lastwarn, errortrap, disp, fprintf, sprintf.
12) Интерактивный ввод: input, keyboard, pause, uimenu, uicontrol, disp.
13) Скрипты, функции и переменные: script, function, global, persistent, mfilename, exist, isglobal, mlock, munlock, mislocked, clear.
14) Функции пользователя: inline, argnames, formula, char.
15) Комментарии: %.

Операции над матрицами и множествами:

1) Множества: union, unique, intersect, setdiff, setxor, ismember.
2) Элементарные матрицы: zeros, ones, eye, repmat, rand, randn, linspace, logspace, meshgrid.
3) Информация о матрицах и массивах: size, length, ndims, disp, isempty, isequal, isnumeric, islogical, logical.
4) Формирование матриц: reshape, diag, tril, triu, fliplr, flipud, flipdim, rot90, **:**, find, end, sub2ind, ind2sub.
5) Специальные матрицы: compan, gallery, hadamard, hankel, hilb, invhilb, magic, pascal, rosser, toeplitz, vander, wilkinson.
6) Матричные функции: norm, normest, rank, det, trace, null, orth, rref, subspace, inv, chol, cholinc, lu, luinc, qr, nnls, pinv, lscov, expm, logm, sqrtm,funm.
7) Собственные значения и сингулярные числа: eig, svd, gsvd, eigs, svds, polyeig, condeig, hess, qz, schur.
8) Переупорядочение: colmmd, symmmd, symrcm, colperm, randperm, dmperm.
9) Разреженные матрицы: speye, sprand, sprandn, sprandsym, spdiags, sparse, full, find, spconvert, nnz, nonzeros, nzmax, spones, spalloc, issparse, spfun, spy.

Математические функции:

1) Тригонометрические и гиперболические: sin, sinh, asin, asinh, cos, cosh, acos, acosh, tan, tanh, atan, atan2, atanh, sec, sech, asec, asech, csc, csch, acsc, acsch, cot, coth, acot, acoth.
2) Экспоненциальные, степенные и логарифмические: exp, log, log10, log2, pow2, sqrt.
3) Комплексные: abs, angle, conj, imag, real, unwrap, isreal, cplxpair.
4) Округления, знаки, модули и остатки: fix, floor, ceil, round, mod, rem, sign.
5) Специальные функции: airy, besselj, bessely, besselh, besseli, besselk, beta, betainc, betaln, ellipj, ellipke, erf, erfc, erfcx, erfinv, expint, gamma, gammainc, gammaln, legendre, cross.
6) Теоретико-числовые: factor, isprime, primes, gsd, lcm, rat, rats, perms, nchoosek, kron.

Анализ данных и их математическая обработка:

1) Основные операции: max, min, mean, median, std, sort, sortrows, sum, prod, hist, trapz, cumsum, cumprod, cumtrpz.
2) Конечные разности: diff, gradient, del2.
3) Корреляция: corrcef, cov, subspace.
4) Фильтрация и свертка: filter, filter2, conv, conv2, convn, deconv.
5) Интерполяция и аппроксимация: interpl, interplq, interpft, interp2, interp3, interpn, griddata, spline, ppval.
6) Операции с полиномами: poly, roots, polyval, polyvalm, residue, polyfit, polyder, conv, deconv, polyeiq.
7) Преобразования координат: cart2sph, cart2pol, pol2cart, sph2cart.
8) Преобразование Фурье: fft, fft2, fftn, ifft, ifft2, ifftn, fftshift, ifftshift.

Численные методы:

1) Минимизация и нахождение нулей функции: fmin, fmins, fzero.
2) Численное интегрирование: quad, quad8, dblquad.
3) Решение обыкновенных дифференциальных уравнений: ode45, ode23, ode113, ode23t, ode15s, ode23s, ode23tb, odeset, odeget, odeplot, odeprint, odephas2, odephas3.
4) Решение линейных алгебраических уравнений: pcg, bicg, bicgstab, cgs, gmres, qmr.

Графика:

1) Двумерные графики: plot, loglog, semilogx, semilogy, polar, plotyy.
2) Трехмерные графики: plot3, mesh, meshc, meshz, surf, surfc, fill, fill3, surface.
3) Управление осями координат: axis, zoom, grid, box, hold, axes, subplot, daspect, pbaspect, xlim, ylim, zlim, gca, cla,caxis, ishold.
4) Надписи на графиках: legend, title, xlabel, ylabel, zlabel, text, gtext, colorbar, plotedit (on-off), edtext.
5) Вывод графической страницы на печать: print, printopt, orient, vrml.
6) Управление цветом: colormap, shading, hidden, brighten, contrast, spinmap, rgbplot, colstyle.
7) Освещение: surfl, lighting, material, specular, diffuse, surfnorm, camlight, lightangle.
8) Палитры: hsv, hot, gray, bone, copper, pink, white, flag, lines, colorcube, jet, prism, cool, autumn, spring, winter, summer.
9) Выбор точки наблюдения: view, viewmtx, rotate3d.
10) Управление фотокамерой: campos, camtarget, camva, camup, camproj, camorbit, campan, camdolly, camzoom, camroll, camlookat.
11) Специальная графика: area, bar, barh, bar3, bar3h, comet, errorbar, feather, fill, hist, pareto, pie, pie3, plotmatrix, ribbon, scatter, stem, staris, ezplot, fplot, comet3, quiver3, scatter3, slice, trisurf, trimesh, waterfall, cylinder, sphere, path, line .
12) Линии уровня и поле направлений: contour, contourc, contourf, contour3, clabel, quiver, voronoi, pcolor.
13) Отображение образов: image, imagesc, imread, imwrite, iminfo.
14) Движение и анимация: capture, moviein, getframe, movie, rotate, frame2im, im2frame.

15) Управление графическими окнами и объектами: figure, clf, shg, close, refresh, closerec, newplot, ishandle, set, get, reset, delete, gco, gcbo, gcbf, drawnow, findobj, copyobj, allchild, findall, hidegui.
16) Графический интерфейс пользователя (GUI): uicontrol, uimenu, uiconextmenu, ginput, dragrect, rbbox, selectmoveresize, waitforbuttonpress, waitfor, uiwait, uiresume, uisuspend, uirestore, guide, align, cbedit, menuedit, propedit, dialog, axlimdlg, errordlg, helpdlg, warndlg, inputdlg, listdlg, menu, msgbox, questdlg, uigetfile, uiputfile, uisetcolor, uisetfont, pagedlg, printdlg, waitbar, makemenu, menubar, imtoggle, winmenu, btngroup, btnstate, btnpress, btndown, btnup, clruprop, getuprop, setuprop.

Звук и аудио:

1) Общие команды: sound, soundsc, speak, recordsound, soundcap, mu2lin, lin2mun.
2) Прием и передача: auwrite, auread, wavwrite, wavread.

Символьные строки:

1) Общие команды: char, double, cellstr, blanks, deblank, eval.
2) Проверка строк: ischar, iscellstr, isletter, isspace.
3) Операции со строками: strcat, strvcat, strcmp, strncmp, strcmpi, strncmpi, findstr, strjust, strmath, strrep, strtok, upper, lower.
4) Преобразование строк в числа и чисел в строки: num2str, int2str, mat2str, str2num, sprintf, sscanf, hex2num, hex2dec, dec2hex, bin2dec, dec2bin, base2dec, dec2base.

Работа с файлами:

1) Открытие и закрытие: fopen, fclose.
2) Бинарный ввод-вывод: fread, fwrite.
3) Форматированный ввод-вывод: fscanf, fprintf, fgetl, fgets, input.
4) Позиционирование: ferror, feof, fseek, ftell, frewind.
5) Обработка имен: fullfile, fileparts, partialpath, tempdir, tempname.
6) Импорт – экспорт: load, save, dlmread, dlmwrite, wklread, wklwrite, hdf.

Время и даты:

1) Текущие дата и время: now, date, clock.
2) Операции с датами: calendar, weekday, eomday, datetick, datenum, datestr, datevec.
3) Операции со временем: cputime, tic, toc, etime, pause.

Типы данных и структуры:

1) Типы данных (классы): double, sparse, char, cell, unit8, inline.
2) Операции с многомерными массивами: cat, ndims, ngrid, permute, ipermute, shiftdim, squeeze, cell, celldisp, cellplot, num2cell, deal, cell2struct, struct2cell.
3) Операции со структурами: struct, fieldnames, getfield, setfield, rmfield, isfield, isstruct.

Все функции приведены без аргументов. Напомним, что более подробную информацию о любой из перечисленных выше функций и примеры их применения можно получить из командного окна:

```
>> help  имя функции
или
>> demo имя функции
```

ЛИТЕРАТУРА

1. Дьяконов В.П. MATLAB 6.5 SP1/7 + Simulink 5/6. Основы применения. Серия «Библиотека профессионала». – М.: СОЛОН-Пресс, 2005, 800 стр.

2. Кондрашов В. Е., Королев С.Б. MATLAB как система программирования научно-технических расчетов. - М.: Мир, Институт стратегической стабильности Минатома РФ, 2002, 350 стр.

3. Фазылов В.Р., Шульгина О.Н., Щербакова Н.К. Применение MATLAB для обработки экспериментальных данных. Методическое пособие. – Казань: Казанский государственный университет, 2005, 56 стр.

4. Кривилев А. В. Основы компьютерной математики с использованием системы MATLAB. - М.: Лекс-Книга, 2005, 496 стр.