##### ПРАКТИКУМ

**Ю.А. Медведев**

# Обложка

# Министерство образования и науки Российской Федерации

Государственное образовательное учреждение

высшего профессионального образования

Владимирский государственный педагогический университет

# Ю.А.МЕДВЕДЕВ

ИНФОРМАЦИОННЫЕ

ТЕХНОЛОГИИ

В МАТЕМАТИКЕ

# Практикум

# Владимир 2005

УДК 681.3.06:51(076)

ББК[32.81 + 22.18]я7

М34

**Медведев Ю.А.**

Информационные технологии в математике: Практикум / Владим. гос. пед.ун-т. Владимир, 2005. 96 с.

ISBN 5-87-846-505-1.

В виде практических работ представлен систематизированный материал по основам работы с системами автоматизированного проектирования в математике Mathcad и MatLab. Приводятся примеры с подробной инструкцией для выполнения и задания для самоконтроля и самостоятельной подготовки. Предназначен для студентов педагогических вузов, обучающихся по специальностям «Математика» и «Информатика». Может быть полезен преподавателям математики и информатики, а также широкому кругу читателей, самостоятельно осваивающих вопросы автоматизации решения математических задач с применением информационных технологий. Табл.: 15. Ил.: 30. Библиогр.: 7 назв.

Рецензенты:

Доктор технических наук, профессор, зав. кафедрой «Вычислительные системы и сети» Московского государственного института электроники и математики В.С. Жданов.

Доктор технических наук, профессор, зав. кафедрой информационных технологий и методики обучения Шуйского государственного педагогического университета В.Н. Федосеев.

Печатается по решению редакционно-издательского совета ВлГУ
 УДК 681.3.06:51(076)

## ББК[32.81 + 22.18]я7

ISBN 5-87-846-505-1. © Владимирский государственный

 педагогический университет, 2005

 © Ю.А. Медведев, 2005

ПРЕДИСЛОВИЕ

Настоящее учебное пособие содержит систематизированный материал по современным компьютерным системам. Учебное пособие состоит из двух разделов.

В первом разделе последовательно рассматриваются проблемы, связанные с работой в системе автоматизированного проектирования в математике MATHCAD и содержатся 11 практических работ: введение в MATHCAD – простые и сложные операции; матричные операции; табулирование функции и построение графиков; численные методы решения уравнений; решение уравнений в символьном виде; вычисление сумм и произведений, символьные вычисления; дифференцирование и вычисление интегралов; численные методы решения дифференциальных уравнений; встроенные функции и др. Даны примеры с подробной инструкцией для выполнения, а так же задания для самоконтроля и самостоятельной подготовки.

Во втором разделе рассмотрены вопросы работы в системе автоматизации математических расчетов, построенных на расширенном представлении матричных операций MATLAB. Приведены 5 практических работ: введение в MATLAB – простые и сложные операции; работа с массивами; взаимодействие между системой MATLAB и программами из пакета MICROSOFT OFFICE (WORD и EXCEL); табулирование функции и построение графиков: интегрирование функций, программирование и другие вычисления.

Главы практикума представляют собой законченные учебные модули, каждый из которых включает краткую теорию по теме, типовые практические работы с указанием цели работы, порядка ее выполнения и контрольных заданий.

Практикум подготовлен в соответствии с программой дисциплины ДПП.Ф.13 «Информационные технологии в математике» федерального компонента ДПП.Ф.00 - «Дисциплины предметной подготовки» Государственного образовательного стандарта высшего профессионального образования по специальностям «Математика»

Учебное пособие предназначено для студентов высших учебных заведений и может быть использовано преподавателями информатики и математики, а также лицами, самостоятельно осваивающими применение информационных технологий в математике.

## *Глава 1*

## *РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ СРЕДСТВАМИ MATHCAD[[1]](#footnote-1)\**

MathCad – система компьютерной математики, предназначенная для автоматизации решения массовых математических задач в самых различных областях науки, техники и образования. Название системы происходит от двух слов – MATHematica (математика) и CAD (Computer Aided Design – системы автоматизированного проектирования, или САПР).

Система Mathcad является представителем нового поколения программных средств и чрезвычайно проста в использовании и лёгка в освоении. Ее интерфейс удобно сделан, что позволяет пользователю работать с рабочим листом программы, как с листом бумаги, где он пишет формулы и математические выражения в их привычной нотации.



Рис. 1. Основные элементы интерфейса системы Mathcad

Mathcad может выполнять вычисления любой степени сложности и ограничен лишь техническими возможностями техники. Помимо численных расчетов, система может выполнять и символьные вычисления. Имеются обширные графические возможности, помимо традиционных типов графиков можно строить поверхности, линии уровня и векторные поля.

С помощью системы Mathcad можно готовить статьи, научные отчеты, дипломные и курсовые проекты не только с качественными текстами разного стиля, но и с легко осуществляемым набором самых сложных математических формул, изысканным графическим представлением результатов вычислений и многочисленными "живыми" примерами.

Как интегрированная система Mathcad содержит следующие основные компоненты:

* редактор документов – редактор с возможностью вставки математических выражений, шаблонов графиков и текстовых комментариев;
* центр ресурсов – интегратор ресурсов системы;
* электронные книги – электронные книги с описанием типовых расчетов в различных областях науки и техники;
* справочная система – система для получения справочных данных по тематическому и индексному каталогу, а также для поиска нужных данных по ключевому слову или фразе;
* "быстрая шпаргалка" – короткие примеры с минимальными комментариями, описывающие применение всех встроенных операторов и функций системы;
* браузер Интернета – собственное средство выхода в Интернет.

## Практическая работа №1

## ВВЕДЕНИЕ В MATHCAD – ПРОСТЫЕ И СЛОЖНЫЕ ОПЕРАЦИИ

1. **Цель работы**

Научиться вводить, форматировать и редактировать математические выражения. Производить простейшие вычисления. Использовать палитры математических знаков.

1. **Порядок выполнения**

Система MathCad интегрирует в себе три редактора: формульный, текстовый и графический. Для запуска формульного редактора достаточно установить указатель мыши в любом свободном месте окна редактирования и щелкнуть левой кнопкой. Курсор ввода в виде маленького красного крестика окажется перенесенным в это место, с которого можно начинать набор формул – вычислительных блоков. Для вывода шаблонов математических операторов (цифр, знаков арифметических операций, матриц и т.д), функций системы и отдельных символов, например греческих букв, служат палитры математических знаков (калькулятор, графика, логики и т.д).

*Пример 1.* Вычислим значение выражения, для набора которого используем обычную математическую нотацию: 1/ =. Знак квадратного корня найдём, раскрыв арифметическую палитру, в конце выражения поставим знак равенства "=". В результате на листе документа получим следующую запись: (знак **=** – это команда "Вычислить").

*Пример 2.* Присвоим значения переменным:   .

Ввод заканчивается клавишей Enter или щелчком мыши вне определения. Для обозначения переменных можно использовать произвольный набор символов, при этом имена переменных чувствительны к регистру. Вначале вводится имя переменной, затем символ ":" (или знак =), затем число или выражение (знак **:=** – это оператор присваивания). Синий уголок показывает текущий операнд выражения, он может быть расширен клавишей "Пробел". Теперь этими переменными можно пользоваться при арифметических вычислениях. Например:

  

  

1. **Контрольные задания**

*Задание 1.* Вычислить для каждого из значений *х*=1,5,7 значения следующих функций:

1.1  1.2  1.3 

*Задание 2.* Вычислить значения функций при х=1,23:

2.1  2.3 

2.2 2.4 

 *Задание 3.* Вычислить значения выражения при различных параметрах:

 3.1 *a=2 b=3,2 c=-3,2*

 3.2 *a=-2,4 b=0,2 c= -1,234*

 3.3 *a=0,22 b=1,2 c=12,2*

## Практическая работа №2

## МАТРИЧНЫЕ ОПЕРАЦИИ

1. **Цель работы**

Научиться вводить, форматировать и редактировать матрицы, производить матричные операции.

1. **Порядок выполнения**

Переменной может быть присвоено значение матрицы (вектор-столбец - это матрица с одним столбцом). Для этого используем палитру векторов и матриц .

*Пример 1*. Введем переменную А - матрицу размером 3\*3 и переменную В - вектор-столбец размером 3\*1.

 

С матрицами можно проделать все допустимые операции: вычислить обратную матрицу, перемножить матрицы, сложить и вычесть. Можно также транспонировать матрицу, произвести выборку элементов.

Пример 2. Обратную матрицу получаем, указав -1 степень, а операцию транспонирования выбираем из палитры векторов и матриц.

  

*Пример 3.* Решим систему уравнений матричным способом, в нашем случае:

 

Комментарий: Знак равенства здесь вводится при помощи (Ctrl =) или палитры логических операций. 

Доступ к элементу матрицы производится по индексу, отсчитываемому от 0. Вектор-столбец имеет один индекс, который вводится при помощи символа левой квадратной скобки - [.

*Пример 4.* Решение рассмотренной задачи можно вывести так:

    Вводим X[0= X[1= X[2=.

Двумерный массив имеет уже два индекса, также отсчитываемые от 0, первый из них нумерует строки, второй – столбцы (индексы разделяются запятыми). Так, для матрицы *A*, это будет выглядеть:

    Вводим A[0,0= A[0,2= A[2,2= A[2,0=.

*Пример 5.* Вычислим определитель матрицы (Ctrl |):



*Пример 6*. Вычислим скалярное (Shift 8) и векторное (Ctrl 8) произведение:

 

*Пример 7.* Найдем корни квадратного уравнения для трёх наборов исходных данных.

  

Получим следующее решение:

  

1. **Контрольные задания**

*Задание 1.* Решите матричным способом системы линейных уравнений: 1.1  1.2 

*Задание 2.*Вычислить сумму, разность и произведение двух матриц:

2.1 , 

2.2 , 

*Задание 3.* Вычислить сумму, разность и произведение элементов матрицы, стоящих на главной диагонали:

3.1 , 

3.2 , 

## Практическая работа №3

## ТАБУЛИРОВАНИЕ ФУНКЦИЙ И ПОСТРОЕНИЕ ГРАФИКОВ

1. **Цель работы**

Научиться работать с функциями одной и нескольких переменных, строить графики и производить табулирование. Задавать функции различными способами.

1. **Порядок выполнения**

Построить таблицу значений функции можно двумя способами:

1. Задать интервал изменения аргумента в формате:

*x:=начальное значение[,начальное значение+шаг]..конечное значение*

в скобках указан необязательный параметр, если его нет, шаг, по умолчанию, равен 1. После чего можно определить функцию от этого аргумента, например:

 

Двоеточие ".." вводится символом точка с запятой ";" кнопкой арифметической палитры. Для того, чтобы вывести таблицу значений функции, введите f(x) и знак "=", вы получите первые 50 значений функции. Теперь можно построить график. Воспользуемся графической палитрой, раскрыв которую выберем x-y график. 



Можно явно указать пределы изменения переменной и функции. Двойной щелчок мышкой по графику вызывает меню настройки, где можно изменить его характеристики. Одновременно можно построить до 16 кривых с общим аргументом, указывая функции через запятую. Можно и аргументы указывать через запятую.

2. Определить изменение целого индекса и построить таблицу значений функции в виде вектор - столбца. В частности, для предыдущей задачи:

  

Далее аналогично первому способу можно построить график. Доступ к элементам массива происходит по индексу, например:

  

Выбор способа построения функции, не столь важен, однако при вычислении значения функции как элемента массива упрощается процедура обращения к его отдельным значениям.

Для двумерного массива обращение строится так: M[i,j , получается  . Двумерный массив соответствует значению функции двух переменных, например:











Определим двумерную матрицу:  и построим поверхность, указав в качестве единственного аргумента графика имя матрицы М.



*Пример 1*. Построим полярный график, выбрав в качестве кривой, кардиоиду.

|  |  |
| --- | --- |
|  |  Для ввода *, * используем палитру греческих символов.  |

*Пример 2.* Построим столбиковую диаграмму.

|  |  |
| --- | --- |
|  | Для этого зададим матрицу значений: . Как видно из графика, каждая колонка матрицы создаёт ряд значений.   |

1. **Контрольные задания**

*Задание 1.* Построить график функции, произвести форматирование шаблона (растянуть график, изменить цвет линий, увеличить их толщину и т.д.).

* 1.  *x∈[-10,10]* 1.2

1.2  *x∈[-15,15] ; h=0,05* 1.4

 *Задание 2.* Построить графики функций в одной плоскости, произвести форматирование.

* 1. *y=sinx*, *y1=0,25x*, *y2=3sin4x*
	2. *y= sin3x*, *y1= sin2x*, *y2= cosx*

 *Задание 3.* Построить поверхность, произвести ее заливку, включить освещение, увеличить изображение.

3.1 *z=x2+y2* 3.3 *z=sin(xy)* 3.5 *z=-(x2 +y2*)

3.2 *z=cos(xy)* 3.4 *z=cosx+siny* 3.6 *z=x2  - cosy*

*Задание 4.* Построить поверхности в одном пространстве.

4.1 *z=x2+y 2* , z1= –( x2+y 2) 4.2 z=cos(xy), z1=sin(xy)

*Задание 5.* Построить сферу, произвести форматирование, написать название поверхности снизу изображения.

*N=20 i=0,N αi=iπ/N j=0,N βJ=j2π/N xi,J =sinαi cosβJ*

*yi,J =sinαi sinβJ zi,J =cosα i*

Построить сферу с вырезом, произвести форматирование.

*Задание 6.* Построить пересекающиеся поверхности, произвести форматирование: залить обе поверхности, включить освещение, произвести непрерывное вращение, убрать оси, увеличить изображение, растянуть по ширине листа, одну из поверхностей показать как диаграммный график.

 *N=60 i=0,N φi=i2π/N j=0,N θJ=j2π/N*

*xi,J =(5+2cosφi )cosθJ  yi,J =(5+2cosφi )sinθJ  zi,J =2sinφi*

В шаблон вписываются формулы:(x,y+6,z), (z,y+1,x).

*Задание 7.* Построить график поверхности при задании ее в векторной параметрической форме, произвести форматирование:



## Практическая работа №4

## ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ УРАВНЕНИЙ

1. **Цель работы**

Научиться решать уравнения, системы уравнений аналитически и графически средствами Mathcad.

1. **Порядок выполнения**

Найти решение уравнения можно несколькими способами:

1.Простейший способ найти корень уравнения с одним неизвестным обеспечит функция root.

*Пример 1.* Найдем корень трансцендентного уравнения . Для этого зададим начальное значение *x:=1*, решение дается функцией:

*root(x-cos(x), x)=0.74*

Точность вычислений задаётся системной переменной TOL, равной по умолчанию 10-3  и определённой в меню **Математика** \ **Опции…**

Проиллюстрируем полученное решение: 



2. Поиск корней при помощи блока Given .........Find(...)

*Пример 1*. Решим систему уравнений с несколькими неизвестными, однако, как и в предыдущем случае, необходимо задание начальной точки от которой будет происходить поиск решения.

*x:=1 y:=1*

*Given*

 *x3+sin(y)* ***=*** *25*

 *y2-cos(x)* ***=*** *27*



 *x=2.96*

 *y=5.101*

Решение ищется методом итераций и при наличии нескольких корней, очевидно, будет найдено лишь ближайшее решение, если оно существует.

Таким же образом можно решать и системы линейных уравнений, однако приходится задавать начальную итерацию, потому системы линейных уравнений лучше решать матричным методом.

1. Поиск решения при помощи блока Given .........Minerr(...).

Практически то же, что и в предыдущем случае, однако здесь решение будет найдено в любом случае, даже при его отсутствии. Дело в том, что здесь ищется не решение системы, а минимальная невязка уравнений.

*Пример 1*. Рассмотрим функцию, заведомо не имеющую действительных корней и найдем точку, в которой эта функция наиболее приближена к оси *х*.

*y(x):= x2 – 2x+1.5*

 *x:=0,0.05..2*

При построении графика необходимо указать начальное значение *y = 0*.



Для этого простого случая очевидно, что наименьшая невязка функции будет при *x=1*.

*x:=0*

*Given*

 *y(x)* ***=*** *0*

*Minerr(x) = 1*

Первая строка даёт нам решение *х=1*, а системная переменная ERR показывает невязку уравнения.

*ERR=0.25*

Аналогично решаются и более сложные уравнения или их системы.

**3. Контрольные задания**

*Задание 1.* Для данной функции произвести графическое отделение корней и аналитически вычислить корни c помощью функции root().

* 1. *y=2e-x/20x2+20x2-50* 1.3 *y=-sinπx+2x2*
	2. *y=x 2+5x-10* 1.4 *y=cosπx-x2*

*Задание 2.* Вычислить корни уравнения с помощью блока Given, используя функции Find() и Minner(). Ответить на вопрос: чем отличаются данные функции.

2.1 *x2 =0*2.4 *3x-cosx-1=0*

2.2 *8+3x=0* 2.5 *0,1ln(x+2)+3x=0*

2.3 *x2 –2x+1=0* 2.6 *1,4x-arcsinx=0*

##

## Практическая работа №5

## РЕШЕНИЕ УРАВНЕНИЙ В СИМВОЛЬНОМ ВИДЕ

1. **Цель работы**

Научиться решать уравнения и системы уравнений символьном виде средствами Mathcad.

1. **Порядок выполнения**

Некоторые уравнения Mathcad может разрешить в символьном виде. Для этого существуют три возможности:

* 1. Использование меню Символика;
	2. Использование оператора *solve, x* ;
	3. Использование блока Given .... Find(...) .

*Пример 1.* Запишем квадратный трёхчлен, выделим переменную *x* и выберем в меню пункт Символика \ Переменная \ Разрешить. Получим решение в символьном виде:

 

*Пример 2.* При использовании оператора необходимо иметь в виду, что переменные не должны быть определены заранее, так попытка раскрытия квадратного уравнения приведёт к ошибке. Однако, квадратный трёхчлен



раскрывается вполне удовлетворительно.

*Пример 3.* Попробуем решить систему линейных уравнений.









1. **Контрольные задания**

*Задание 1.* Найдите решение полинома третьей степени.

## Практическая работа №6

## ВЫЧИСЛЕНИЕ СУММ И ПРОИЗВЕДНИЙ.

## СИМВОЛЬНЫЕ ВЫЧИСЛЕНИЯ

1. **Цель работы**

Получить практические навыки вычисления сумм и произведений в числовом и символьном виде.

1. **Порядок выполнения**

**Вычисление сумм и произведений**

|  |  |
| --- | --- |
| Для вычисления сумм и произведений воспользуемся палитрой вычислений.  |  |

*Пример 1*.  

 

Замечание: нельзя считать суммы с бесконечными пределами.

*Пример 2*. Значок суммирования с указанием индекса используем для работы с матрицей и функцией.

  

  

*Пример 3*. По определению: .

    

**Символьные вычисления**

Символьными называются такие вычисления, результаты которых представляются в аналитическом виде.

|  |  |
| --- | --- |
| b |  Команды, относящиеся к работе символьного процессора, содержаться в меню *Символика*. Чтобы операции выполнились, необходимо выделить выражение, над которым это будет производиться. Для некоторых операций следует указать переменную, относительно которой выполняется символьная операция.  |

*Пример 1.* Вычисление сумм. Для вычисления можно воспользоваться палитрой оценки  



 или по конечному пределу:



 или получаем ряд из 8 слагаемых, это значит, что система не смогла упростить выражение:



*Пример 2.* Аналогично вычислим произведение:





1. **Контрольные задания**

*Задание 1.* Вычислить сумму и произведение элементов матрицы.

1.1 , 

1.2 , 

## Практическая работа №7

## ДИФФЕРЕНЦИРОВАНИЕ И ВЫЧИСЛЕНИЕ ИНТЕГРАЛОВ

1. **Цель работы**

Изучить процедуры символьного и численного дифференциального и интегрального исчисления в системе Mathcad.

1. **Порядок выполнения**

**Дифференцирование**

Дифференцирование осуществляется через меню **Символика** \ **Переменная** \ **Дифференцировать**. Предварительно необходимо выделить переменную дифференцирования в выражении.

*Пример 1*. Вычислим производную функции от одной переменной.

 

*Пример 2*. Вычислим производную, используя панель Исчислений, поставим функцию под знак .

 

 

*Пример 3.* Определим перед вычислением производной значение переменной, получим численное значение:







Проверим: 

**Вычисление интегралов**

*Пример 1.* Если  

|  |  |
| --- | --- |
|  | Определённый интеграл - есть площадь криволинейной трапеции.Интеграл достаточно хорошо вычисляется, если подынтегральная функция не имеет особенностей. |

Интегрирование можно осуществить через меню **Символика** \ **Переменная** \ **Интегрировать** для получение ответа в символьном виде. Предварительно необходимо выделить переменную дифференцирования в выражении.

*Пример 2*. Вычислим интеграл в символьном виде.





1. **Контрольные задания**

*Задание 1.*  Построить график функции и вычислить определенный интеграл.

1.1  1.3 

1.2  1.4 

*Задание 2.*  Найти неопределенный интеграл и результаты проверить дифференцированием.

2.1 ∫ esinxsin2x dx 2.3 ∫ exln(1+3ex) dx

2.2  2.4 

*Задание 3.*  Вычислить площадь фигуры, ограниченной параболой y=3x2+1 и прямой y=3x+7. Построить график криволинейной трапеции.

*Задание 4.* Найти производные  данных функций.

4.1 *y=(ecosx+3)2* 4.3 

4.2 *y=ln sin(2x+5)*  4.4 *y=0.3tg3x – tgx + x*

*Задание 5.*  Найти производные первого  и второго порядка  для данных функций.

5.1  5.3 *y=arctgx*

5.2 *y=ln ctg2x* 5.4 *y=ex cosx*

## Практическая работа №8

## ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ

## ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

1. **Цель работы**

Научится применять численные методы решения дифференциальных уравнений, встроенных в систему Mathcad.

1. **Порядок выполнения**

**Дифференциальные уравнения 1-го порядка.**

**Решение задачи Коши**

*Пример 1.* Пусть задано дифференциальное уравнение 

При начальном условии .

Численное решение осуществляется при помощи встроенной функции *rkfixed(y,x1,x2,n,D),* которая использует метод Рунге-Кутта 4-го порядка, где

*y -* вектор начальных условий, в данном случае вектор из одного элемента;

*x1,x2 -* границы интервала для поиска решения;

*n* - количество точек на интервале;

*D(x,y) -* вектор-функция первых производных, в данном случае вектор из одного элемента.

Решим заданное дифференциальное уравнение на интервале (1,5).  – начальное условие;

 – правая часть уравнения.

 

|  |  |
| --- | --- |
|  | Матрица Z имеет 2 столбца и 40 строк. Первый столбец содержит х переменную, второй - y. |
|  | Крестиками показано точное решение уравнения  |

**Системы линейных уравнений первого порядка**

Системы линейных уравнений первого порядка решаются с помощью той же встроенной функции *rkfixed(y,x1,x2,n,D),* которая использует метод Рунге-Кутта 4-го порядка.

*Пример 1.* Решим систему 2-х дифференциальных уравнений 1-го порядка.









 – начальные условия, теперь уже в виде вектора.

 – правые части уравнений, также вектор, где в качестве аргументов используются компоненты вектора y.

 

|  |  |
| --- | --- |
|  *t x(t) y(t)* | Обратите внимание, что здесь мы, для отображение *x* и *y*, воспользовались немного другой формой, учитывая, что  |

**Дифференциальное уравнение 2-го порядка**

Дифференциальное уравнение 2-го порядка решается аналогично, путём сведения уравнения 2-го порядка к системе 2-х уравнений 1-го порядка.

Пример 1. Решим уравнение:







Эквивалентная форма – система двух уравнений:



 

|  |  |
| --- | --- |
|  *x y(x) y'(x)* |  |

**Уравнения или системы более высокого порядка**

*Пример 1*. Решим систему двух уравнений второго порядка.

|  |  |
| --- | --- |
|  | С начальными условиями:  *v(0)=1*   |

Сводим к системе 4-х уравнений 1-го порядка и решаем стандартным

способом.

 – запись начальных условий как вектор-столбца:

 

 – вектор - столбец правых частей системы уравнений.

 



**Медленно изменяющиеся функции**

Если известно, что искомое решение достаточно гладкое, можно использовать функцию *Rkadapt(y, x1, x2, npoints, D)*, которая ищет решение с переменным шагом, то есть там, где решение меняется медленнее, шаг увеличивается, а в области быстрого изменения функции шаг уменьшается, это ускоряет поиск решения. Возвращается же решение с равным шагом.

Функция имеет те же параметры, что и *rkfixed(y, x1, x2, npoints, D)*.

*Пример 1.* Решим предыдущую задачу двумя способами.

 







|  |  |  |
| --- | --- | --- |
|  |  | Как видно, в простом случае мы получим полное совпадение решений. |

**Гладкие системы**

Когда известно, что решение является гладкой функцией, более точное решение даёт функция *Bulstoer(y, x1, x2, npoints, D),* которая использует метод Bulirsch-Stoer, но имеет те же аргументы, что и

 *rkfixed(y, x1, x2, npoints, D)*.

*Пример 1.* Решим предыдущую задачу.



|  |  |
| --- | --- |
|  | Нетрудно убедиться, что мы получили практически то же самое решение. Вы можете сравнить и матрицу значений *М*. |

**Жёсткие системы**

Если матрица правых частей системы дифференциальных уравнений почти вырождена, такие системы являются жёсткими. В этом случае, решение, возвращаемое функцией *rkfixed(y, x1, x2, npoints, D)* будет неустойчивым и для решения таких систем необходимо применять функцию *Stiffb(y, x1, x2, npoints, D, J),* использующую метод Bulirsch-Stoer, или *Stiffr(Y, x1, x2, npoints, D, J),* использующую Rosenbrock метод. Здесь аргументы функции те же самые, что и прежде, но добавляется матрица J размером n\*(n+1), первый столбец которой содержит частные производные *d****D****/dx*, остальные столбцы и строки представляют собой матрицу Якоби *d****D****/dyk .*



*Пример 1.* Рассмотрим решение для жёсткой системы:

|  |  |
| --- | --- |
|  | При начальных условиях: |

 

  

|  |  |
| --- | --- |
|  | Как видно, обе функции дают одинаковое решение. Выбор того, или иного метода определяется особенностями задачи. |

1. **Контрольные задания**

Прорешайте, используя рассмотренный набор функций, дифференциальные уравнения и их системы из стандартного задачника по дифференциальным уравнениям.

Сравните решение c аналитическим, если удастся его найти.

## Практическая работа №9

## ВСТРОЕННЫЕ ФУНКЦИИ

1. **Цель работы**

Научиться использовать в решениях математических задач встроенных функции.

1. **Порядок выполнения**

Mathcad имеет богатый набор встроенных функций. Большинство из них просто возвращает значение, однако, имеется две функции, которые служат для управления вычислениями.

1. Функция *if (условие, оператор 1,оператор 2)* – если условие истинно, выполняется оператор 1, иначе оператор 2.

*Условие Соответствующая комбинация клавиш*

 *x=y* Ctrl=

 *x>y* >

 *x<y* <

 Ctrl 0

 Ctrl 9

 Ctrl 3

Результат логической операции равен 0, если условие не выполнено, и 1, если условие истинно. Этим свойством можно пользоваться для создания более сложных логических конструкций.

*Пример 1.* Логическое умножение:  равно 1, если  и 0 в противном случае.  действует подобно логическому сложению.

*Пример 2*. Используем функцию *if*  для корректного определения корней квадратного уравнения:













Однако:





1. Функция *until(x,z) –* возвращает *z* пока *х* не становится отрицательным. Функция позволяет останавливать вычисления при выполнении определённого условия и может быть полезна при организации итерационной процедуры.

*Пример 3.* Рассмотрим процедуру нахождения корней трансцендентного уравнения методом Ньютона. Для уравнения *f(x)=0* итерационная процедура реализуется формулой:



Рассмотрим решение уравнения *x=cos(x),* считая, что решение найдено, если разность левой и правой частей уравнения не превосходит системной переменной TOL=0.001.

   

Начальное значение.

 

Как видно, для достижения заданной точности потребовалось не 100 шагов, а всего лишь 3.

 

Обе эти функции позволяют достичь большей гибкости вычислений, не прибегая к программированию.

3. Импульсные функции.

1). Функция Хэвисайда *(x)=if(x<0,0,1)* используется для электротехнических расчётов. Служит для создания ступенчатого импульса шириной *w.*

*Пример 4.* Проведем расчет, при наборе формулы символ  выбираем на палитре греческих символов

  



2). Дельта символ Кронекера *(m,n):=if(m=n,1,0)*.

 

*Пример 5.* При помощи этой функции несложно создать единичную матрицу, или на обратной диагонали:

  

 



 

4. Прочие функции.

Mathcad содержит большую библиотеку встроенных функций, с перечнем которых можно познакомиться во встроенной системе помощи.

1). Трансцендентные функции - тригонометрические, показательные, гиперболические, функции Бесселя.

2). Усечение и функции округления – функции, которые извлекают какую-либо часть числа, включая реальную, мнимую, дробную или целую части.

3). Дискретные преобразования - преобразование Фурье.

4). Функции сортировки - функции упорядочения элементов векторов и матриц.

5). Векторные и матричные функции - функции преобразования матриц и операций над ними.

6). Статистические функции - функции распределения различных статистик, вычисления различных статистических характеристик.

7). Функции обмена данными - набор функций позволяющий сохранять матрицы данных в файле и читать данные из файла.

**3. Контрольные задания**

*Задание 1.* Вычислить Z – сумму значений функций

*Z=f(a,b)+f(a2,b2)+f(a2-1,b)+f(a-b,b)+f(a2+b2,b2-1),*

 *a2+b2, если a>0, b>0;*

*где f(a,b)= a2+b2, если a<=0, b<=0;*

 *a – b, если a>0, b<=0;*

 *a + b, если a<=0, b>0.*

*a) a=2.5, b=-7.3;*

*b) a=-0.5, b=4.2;*

*c) a=-0.2, b=-0.42;*

*d) a=23.7, b=41.2.*

*Задание 2*. Вычислить Z – сумму значений функций

*Z=f(sin(x)+cos(y),x+y)+f(sin(x),cos(y))+f(x-y,x)+f(sin2(x)-2,a)+f(a+3,b+1),*

 *a+b, если a>1;*

*где f(a,b)= a-b, если 0<=a<=1;*

 *a-b, если a<0.*

*a) x=π/4, y=0.41, a=0.1, b=-2.1;*

*b) x=0.32, y= π/10, a=-0.21, b=4.1;*

*c) x=19.2, y=0.48, a=-4.3, b=-6.1;*

*d) x=0.2 π, y=-2/3, a=17.1, b=0.2.*

## Практическая работа №10

## ПРОГРАММИРОВАНИЕ

1. **Цель работы**

Получить практические навыки в составлении и оформлении алгоритмов решения математических задач в системе Mathcad.

1. **Порядок выполнения**

Для повышения гибкости Mathcad в системе предусмотрена возможность написания небольших программ для решения тех проблем, которые не могут быть реализованы стандартными средствами.

Обычно прибегать к программированию приходится в тех случаях, когда стандартные средства либо не могут решить задачу, либо неэффективны.

Для написания программ используется программная палитра, которая вызывается кнопкой панели управления.

 

Как видно, всего имеется 10 операторов, из которых и строится программа.

*Пример 1.* Приведём простую программу возвращающую 1, если число чётное и 0, если нечётное.

Начинаем создание программы с кнопки **Add Line**. Вертикальная линия играет роль операторных скобок.



После того, как функция определена, она может использоваться наравне со встроенными функциями.





*Пример 2.* Более сложный пример определения максимальной координаты вектора и её позиции.

|  |  |
| --- | --- |
|  | Присваивание начальных значений переменным.Цикл по элементам вектора (элементы вектора отсчитываются от 0).Присваивание больше- го значения и сохра- нение его координаты.Возвращаемые значения. |

Определим теперь вектор:



Действительно, максимальное значение 8, имеет номер 2.



*Пример 3.* Рассмотрим, как программным способом построить скалярное произведение.

Вычислим квадрат модуля вектора .



 

Программные строки создаются кнопкой Add Line, операторы вводятся соответствующей кнопкой. Обратите внимание, что программах мы не пользуемся оператором присваивания :=, а вместо него пишем оператор локального присваивания .

*Все переменные, определённые в программе, теряют своё значение при выходе из неё.*

*Пример 4.* Рассмотрим проблему вычисления несобственного интеграла. Несложно убедиться, что попытка подставить бесконечный предел интегрирования не увенчается успехом.

Например: 

Этот интеграл система вычислить не смогла, но в символьном виде он считается.



Составим программу вычисления таких интегралов.



 

Сравним с точным решением: 

Алгоритм работает, но выбран не самый удачный, поскольку приходится многократно пересчитывать интегральные суммы на одном и том же диапазоне.

Очень интересная особенность программирования в Mathcad заключается в том, что в программе мы можем использовать операторы типа вычисления интеграла, суммы, производной и т.п.

Более удачный алгоритм получится, если мы будем продолжать интегрирование от той точки, на которой закончили предыдущее интегрирование.

Составим программу по данному алгоритму.





Получили тот же результат, что и не удивительно, но алгоритм работает быстрее.

По наглядности текста, вряд ли какой-нибудь язык программирования сравнится с Mathcad-программой.

Здесь, для ускорения работы программы, мы на каждом шаге удваиваем интервал интегрирования.

Можно использовать программные возможности Mathcad просто для задания функций более сложного вида.

*Пример 5.* Определим функцию, которая равна 1, если аргумент размещён между чётным и нечётным числом, и 0, в противном случае.



Функция  возвращает ближайшее целое большее .

Например: 



*Пример 6.* Приведём ещё программу перевода десятичного числа в двоичное представление.

Функция - ближайшее целое меньшее .

Функция  - остаток от деления по модулю.



Например:





Примечание:  - транспонированная матрица.

**3. Контрольные задания**

1. Написать программу, вычисляющую модуль числа.

2. Написать программу для вычисления суммы чисел из диапазона от 1 до n.

3. Написать программу для вычисления произведения чисел из диапозона от 1 до n.

4. Написать программу для вычисления факториала n! при n>0, используя цикл WHILE.

5. Написать программу для вычисления суммы делителей числа.

6. Написать программу, проверяющую является ли число совершенным.

## Практическая работа №11

## РАЗМЕРНОСТИ

1. **Цель работы**

Научиться использовать при решении задач переменные с размерностями.

1. **Порядок выполнения**

В Mathcad реализована возможность использования переменных с размерностями, даже зарезервированы некоторые константы с их размерностями, например, ускорение свободного падения .

Причём возможен выбор системы единиц в меню **Математика \ Опции** из перечня: SI, MKS, CGS, US или отказ от выбора размерностей. При работе с размерными величинами мы можем вводить размерности вручную после знака умножения, или же выбрать из списка по команде **Вставка \ Объект** (Ctrl U) или кнопкой.

Базовыми единицами системы SI являются: m - метр (1L), kg - килограмм (1M), s - секунда (1T), K - Кельвин (единица температуры 1K), A - ампер (единица силы тока 1A), cd - Кандела (единица силы света 1C), и mole - моль (количество вещества 1S).

Если щелкнуть мышкой по любому выражению Mathcad, справа появляется маркер для ввода размерностей. Таким образом, можно вводить размерности, или преобразовывать значение из одних единиц в другие.

*Пример 1.* Преобразуем футы в метры, а мили в метры и футы.











Нужно иметь в виду, что при вычислениях переменных с размерностями происходит контроль размерности операндов и, при несовпадении размерности, выдаётся сообщение об ошибке.

*Пример 2.* Рассмотрим, в качестве примера, задачу о теле, брошенном под углом  к горизонту с начальной скоростью V0=(Vx,Vy).



Пусть начальная скорость 

Угол бросания 

Уравнения движения:

 

 

Примечание: *для обозначения переменных используйте текстовый индекс*.

Для построения графика необходимо найти время движения. Мы можем воспользоваться симметрией графика. Очевидно, что время движения будет вдвое больше времени достижения максимальной точки траектории. А эту точку мы найдём из равенства:



Решаем это уравнение, считая производную функции заданной параметрически.





  



Можно найти максимальное расстояние по оси *х* и максимальную высоту подъёма.









**ЗАДАНИЯ ДЛЯ САМОПРОВЕРКИ**

1. Дано уравнение плоскости Ax+Bx+Cz=D. Построить плоскость, параллельную заданной и отстоящую от нее на расстоянии R (изобразить графически).
2. Исследовать функцию y=f(x) (непрерывность, четность, выпуклость, точки перегиба и т.д.).

2.1  2.2  2.3 

1. Решить систему линейных уравнений методом Крамера.



1. Найти площадь фигур, ограниченных линиями:

4.1 Параболами y = x2 , y2 = x.

4.2 Гиперболой xy = a2 и прямыми y = 0, x = b, x = 2b (b>0).

1. Составить программу, определяющую, является ли число N простым.
2. Составить программу канонического разложения числа N<1000 на простые множители.

## *Глава 2*

## *РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ СРЕДСТВАМИ MATLAB[[2]](#footnote-2)\**

**MatLab –** одна из старейших, тщательно проработанных и апробированных систем автоматизации математических расчетов, построенная на расширенном представлении матричных операций. Этот факт отражает название системы **MAT**rix **LAB**oratory – матричная лаборатория.

В настоящее время MatLab далеко вышла за пределы специализированной матричной системы и стала одной из наиболее мощных универсальных математических систем. В новую версию вошли такие мощные типы данных, как многомерные массивы, массивы ячеек и разреженные матрицы, что открывает возможности применения системы при создании и отладке новых алгоритмов матричных и основанных на них параллельных вычислений. В целом MatLab – это уникальная коллекция современных численных методов.

Достоинством системы является возможность MatLab интегрироваться с другими математическими системами (MathCad, Maple и Mathematica) и прикладными программами пакета MS Office (Word и Excel).

## Практическая работа №1

## ВВЕДЕНИЕ В MATLAB – ПРОСТЫЕ И СЛОЖНЫЕ ОПЕРАЦИИ

1. **Цель работы**

Овладение практическими навыками вычисления и составления расчетных формул в системе MatLab.

1. **Порядок выполнения**

Прежде всего, надо знать, что рабочая среда (место в которой находится мигающий вертикальный курсор) MatLab отличается в некоторых версиях. Все команды следует набирать в командной строке, после символа **>>**. Этот символ обозначает приглашение командной строки. Чем больше функции (команд) знаем, чем легче работать или решить какую либо задачу.

Важно знать и запомнить, что набор любой команды или выражения должен заканчиваться нажатием клавиши <**Enter>**.

**Простейшие вычисления**

*Пример 1.* Наберите в командной строке 23+17 и нажмите клавишу <**Enter>**. В результате в командном окне отображается следующее:

>> 23+17

*ans* =

 40

>> |

Сначала программа MatLab вычисляет сумму 23+17, затем записывает результат в специальную переменную *ans* . После того как курсор снова мигает после символа >>, это означает, что MatLab готова к дальнейшим вычислениям.

Если требуется продолжить работу с предыдущим выражением, например, вычислить , то проще всего воспользоваться уже имеющимся результатом, который хранится в переменной *ans* до тех пор, пока не будем завершать работу (закрытие программы MatLab) или освобождать память.

*Пример 2.* Наберите в командной строке *ans*/11 и нажмите клавишу ‘**Enter**’, получается:

>> *ans*/11

*ans* =

 3.6364

>> |

*Задание 1.* Вычислите выражение и выведите результат на экран.



Требуемый формат вывода результата определяется пользователем из меню MatLab. Выберите в меню **File** пункт **Preferences**. На экране появится диалоговое окно **Preferences**, в котором следует перейти на вкладку **General**. На панели **Numeric Format** расположены переключатели, при помощи которых устанавливается формат вывода результатов вычислений.

*Задание 2.* Вычислите выражения из предыдущих примеров в каждом формате. Например: 3.3333e-004 обозначает 3.3333**.**10-4 или 0.0003333.

**Использование элементарных функций**

*Пример 1.* Предположим, что требуется вычислить значение следующего выражения:



Для этого введите в командной строке это выражение в соответствии с правилами MatLab и нажмите клавишу ‘**Enter**’.

>>exp(-2.5)\*log(11.3)^0.3-sqrt((sin(2.45\*pi)+cos(3.78\*pi))/tan(3.3))

ans =

 -3.2105

Если теперь требуется вычислить значение выражения, похожего на предыдущее, например



то необязательно снова набирать его в командной строке. Можно использовать клавиши <↑> и/или <↓> для повторного занесения их в командную строку.

*Задание 1.* Сделайте все изменения и вычислите значение предыдущего выражения. Результат должен быть равен 121.2446. Сделайте выводы и заполните следующую таблицу (не менее 12 строк):

|  |  |  |
| --- | --- | --- |
| **№** | **Команда или функция** | **Представление в MatLab** |
| 1 | степень | ^ |
|  |  |  |

*Задание 2*. Зная функции кнопки <↑> и <↓>, можно быстро изменить формат командой, а затем посмотреть результат. Попробуйте этого делать.

При делении на ноль в MatLab получает ‘*Inf*’– значит бесконечность, а при делении отрицательного числа на ноль получается *‘*–*Inf*’ *(*минус бесконечность). При делении нули на ноль получается ‘*NaN*’(не число). При вычислении, например , MatLab автоматически переходит в область комплексных чисел.

Чтобы вызывать встроенные элементарные функции, которые можно использовать, надо в команде строке набрать команду *help elfun*, при этом в командное окно выводится список всех встроенных элементарных функций с их кратким описанием.

*Задание 3*.Наберите все часто используемые функции и напишите их обозначение на русском языке. Разделить на 4 группы:

- тригонометрические, гиперболические и обратные к ним функции;

- экспоненциальная функция, логарифмы, степенные функции;

- функции для работы с комплексными числами;

- округление и остаток от деление.

**Понятие переменных**

Переменные - это именованный объект, который в процессе выполнения программы может принимать различные значения.

*Пример 1.* Для того чтобы присвоитьпеременной t значение 6.74, надо написать в командной строке t=6.74, при этом MatLab сразу же выведет значение t:

>> t=6.74

t =

 6.7400

*Задание 1.* Наберите команды из примера 1 и добавите в конце ‘**;**’. Какие произошли изменения?

Примечание: В MatLab, когда ставят знак **%** перед фразой, система MatLab понимает его как комментарий.

*Пример 2.* Найдем значение следующего выражения:

Для этого наберите последовательность команд, приведенную ниже:

>> x=sin(1.3\*pi)/log(3.4); %нажмите клавишу ‘**Enter**’

>> y=sqrt(tan(2.75)/tanh(2.75)); %нажмите клавишу ‘**Enter**’

>> z = (x+y) / (x-y) %нажмите клавишу ‘**Enter**’

z =

 0.0243 – 0.9997i

Наберите сразу всю формулу и нажмите клавишу ‘**Enter**’, какой результат Вы получили?

Переменные, определенные выше, можно использовать и в других формулах.

*Задание 2.*  Какую команду достаточно ввести для вычисления следующего выражения

**

 и какой будет результат вычисления?

**Процесс сохранения значения всех переменных**

Предположим, что мы хотим еще использовать эти переменные на следующий день. Тогда надо будет сохранить содержимое рабочей области с помощью командой save <*имя файла*>. (см. лекцию). Чтобы потом использовать эти переменные, надо их загрузить с помощью команды load <*имя файла*>.

*Задание 1.* Используйте команды diary, who, whos и clear. Чем они отличаются друг от друга?

## Практическая работа №2

## РАБОТА С МАССИВАМИ

1. **Цель работы**

Овладение практическими навыками работы с массивами в системе MatLab.

1. **Порядок выполнения**

**Сложение, вычитание и деление векторов**

*Пример 1.* Студенты одной группы прошли два раза тест по предметам. После первого и второго теста, студенты получили следующие результаты:

|  |  |  |
| --- | --- | --- |
| *ФИО Студентов* | *Результат теста №1* | *Результат теста №2* |
| Студенты 1 | 6.2 | 5.2 |
| Студенты 2 | 7 | 6 |
| Студенты 3 | 4.5 | 7.32 |
| Студенты 4 | 2.8 | 9 |

Для итога рейтинга преподаватель определяет среднее значение. Для этого существует в системе MatLab путь решения данной задачи. Попробуем для начала разместить эти значения в массиве. Назовем первый массив ***a***и второй ***b****.*

1) Введите массив ***a*** в командной строке MatLab:

>> a = [6.2; 7; 4.5; 2.8]

a =

 6.2000

 7.0000

 4.5000

 2.8000

a= b=

Далее, введите массив **b**; завершить выражение точкой с запятой.

Для нахождения суммы векторов используется знак +. Запишите результат в массив ***c***; для этого, введите следующее команд:

>> c=a+b

c =

 11.4000

 13.0000

 11.8200

 11.8000

Теперь определите среднее значение d с помощью формулы. Для этого наберете в командой строке это: **d=c/2** и напишите ответ. Так же можно делать другие операции, например определить квадратный корень из элементов d со знаком минус “>> sqrt(-d)”.

*Задание 1.* Определите разность массивов **a** и **b**.

**Определение размерности и размера массивов**

Определите размерность и размер массива ***с*** при помощи функций *ndims* и *size*

>> ndims(c)

>> size(c)

ans =

 4 1

ans =

 2

Итак, векторы **а**,**b**,**c** и **d** хранятся каждый в двумерном массиве **a**,**b**,**с** и **d** размерностью четыре на один.

*Задание 2.* Ответьте на вопрос: если размеры векторов, не совпадают можно ли их сложить или вычитать?

*Задание 3.* Выполняете следующие шаги и сделайте выводы.

>> v1=[1.3 4.2 7 5.25]

v1 =

1.3000 4.2000 7.0000 5.2500

>> v2=[2.7 0.2 3.35 1]

v2 =

 2.7000 0.2000 3.3500 1.0000

ans =

 4 1

a) Вычислите значение v = v1+v2;

b) Сцепляете вектор v1 и v2, результат записать в массиве u. (>> u = [v1 v2]).

с) Повторите тоже сами для вектора v11=[1.3; 4.2; 7; 5.25] и v22=[2.7; 0.2; 3.35; 1]

d) Какой вывод можно делать?

e) Можно ли сложить вектор v1 и вектор v22? Аргументируйте ваш ответ.

**Операции с массивом**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Функция** | **Пример синтаксиса** | **Описание** |
| 1 | length | length(имя вектора)max(size(имя вектор)) | Определяет длину вектора |
| 2 | prod | prod(имя вектора) | Позволяет перемножить элементы вектора |
| 3 | sum | sum(имя вектора) | Суммирует элементы вектора |
| 4 | max | max(имя вектора)[m,i]=max(имя вектора) | Найти максимум из элементов вектораНайти максимум(m) и его индекс(i) из элементов вектора |
| 5 | min | min(имя вектора)[m,i]=min(имя вектора) | Найти минимум из элементов вектораНайти минимум(m) и его индекс(i) из элементов вектора  |
| 6 | sort | sort(имя вектора)-sort(-имя вектора) | Упорядочить вектор по возрастаниюУпорядочить вектор по убыванию |

*Задание 1.* Приведите пример, используя все функции, которые находятся в таблице.

**Деление и умножение**

*Пример 1.* Введите две вектор - строки:

>> m1 = [3, 6, -7, 9];

>> m2 = [2, 8, 4, -3];

Для умножения m1 и m2, надо набрать в командой строке следующие выражения:

>> m1**.**\*m2

для деления: >> m1**.**/m2

для обратного деления: >>m1**.**\m2

для возведения в степень: m1.^m2

*Задание 1.* Напишите ответ на все вычисления.

**Построение таблицы значений функции.**

*Пример 1.* Пусть требуется вывести в командное окно таблицу значений функции



в точках -0.5, 0.1, 0.2, 0.4; 0.6, 1.2, 1.6. Можно решить эту задачу в 2 этапа:

a) Создайте вектор-строку x, содержащую координаты заданных точек.

б) Вычислите функцию y(x) от каждого элемента вектора x и напишите полученные значения в вектор-строку y. Важно только сделать это правильно!

>> x=[-0.5 0.1 0.2 0.4 0.6 1.2 1.6]

x =

 -0.5000 0.1000 0.2000 0.4000 0.6000 1.2000 1.6000

>>y = 2.\*exp(-x)+sin(x).^2./cos(x)-log(x).\*abs(-2.\*x)

y =

4.2525 - 3.1416i 2.2802 2.3215 2.2383 2.0969 2.5622 -35.3182

Часто требуется вывести значение функции в точках отрезка, отстоящих друг от друга на равное расстояние (шаг). Предположим, что необходимо вывести таблицу значений функции y(x) на отрезке [-2, 2] с шагом 0.4. Можно обычно путем ввести вектор-строку значений аргумента *x* из командой строки и вычислить все значения y(x). Чтобы не трудиться при введения значений аргумента *x* , в MatLab предусмотрено простое создание векторов, каждый элемент которых отличается от предшествующего на постоянную величину, т.е. шаг.

Синтаксис описание этого вектора: [начальное значение: шаг: конечное значение]. Шаг, так же может быть отрицательный. Если шаг равен единице, то можно не указывать его.

*Задание 1.* Выведите теперь таблицу значений предыдущей функции y(x) на отрезке [-1,3] с шагом 0.4, произведя следующие действия:

a) Сформируйте вектор-строку x при помощи двоеточия.

>> x=[-1:0.4:3];

b) Вычислите значения y(x) от элементов x и запишите результат в вектор-строку y.

>> y = 2.\*exp(-x)+sin(x).^2./cos(x)-log(x).\*abs(-2.\*x);

c) Выведите x и y

>>x

x =

 Columns 1 through 6

 -1 -0.6 -0.2 0.2 0.6 1

 Columns 7 through 11

 1.4 1.8 2.2 2.6 3

>>y

y =

 Columns 1 through 3

 6.7471 - 6.2832i 4.6435 - 3.7699i 3.1269 - 1.2566i

 Columns 4 through 6

 2.3215 2.0969 2.0463

 Columns 7 through 9

 5.2646 -5.9596 -4.3583

 Columns 10 through 11

 -5.1302 -6.5122

d) Сделать заключения.

Комментарий: Результат, отображенный на экране, не очень напоминает таблицу. Вектор-строки x и ,y состоят из одиннадцати элементов, не помещаются на экране в одну строку и выводятся по частям. Так как x и y хранятся в двумерных массивах размерностью один на одиннадцать, то выводятся по столбцам, каждый из которых состоит из одного элемента.

Columns 1 through 3 - с первого по трети

Columns 4 through 6 - с четвертого по шестой

Columns 10 through 11 - с десяти по одиннадцати

Более наглядным и удобным является графическое представление функции, которая будет рассмотрена в следующей лабораторной работе.

**Типы произведений**

*Скалярное:*

*Пример 1.* Определите скалярное произведение r векторов c и d.

c= d=.

 Для этого, надо использовать следующую формулу: r = sum (c.\*d)

Результат должен быть 107.18.

*Векторное:*

Векторное произведение определено только для векторов из трехмерного пространства, т.е. состоящих из трех элементов. Для этого в MatLab использует функцию cross.

*Пример 2.* Определим векторное произведение z=cross(c,d).

Получим:

z =

 24.24

 -21.984

 0.8

**Способы ввода матриц**

*Первый способ*:

Что бы хранить матрицы, надо использовать двумерный массив. *Пример 1.* Предположим, что надо вывести матрицу A= 

Для этого наберите следующие команды:

>> A = [2 -3 5; 3 4 1]

 A =

 2 -3 5

 3 4 1

*Второй способ*:

*Пример 2.* Пусть требуется написать следующую матрицу:

B=

Начните набирать в командной строке:

>> B=[-1 4 2

 3 5 1

-7 0 8]

Нажмите клавишу <Enter>. Курсор мигает на следующей строке без символа>>. Продолжите ввод матрицы построчно, нажимая в конце каждой строки <Enter>. Последнюю строку завершите закрывающей квадратной скобкой, получается:

B =

 -1 4 2

 3 5 1

 -7 0 8

*Третий способ*:

*Пример 3.* Матрицу A можно ввести при помощи команды:

>> A = [[2; 3] [-3; 4] [5; 1]]

A =

 2 -3 5

 3 4 1

*Задание 1.* Попробуйте также ввести матрицу B. Какой способ более удобно и почему?

**Обращение к элементам матриц**

Для обращения к определенному элементу массива, используется два индекса: первый – это номер строки и второй – номер столбцы. Например:

>> B (1, 3)

ans =

 2

*Задание 1.* Используйте возможность обращения к элементам матрицы для вычисления суммы и произведение элементов массива A.

**Математические операции с массивами**

*Задание 1.* Произведите сложение, вычитание и умножение следующих массивов и выведите результаты на экран:

A и B, B и A (Для сложения используется символ **+**, для вычитания **–**, для умножения \*)

Для возведения квадратной матрицы в целую степень используется оператор **^** . Например:

>> B^2

*ans =*

 *-1 16 18*

 *5 37 19*

 *-49 -28 50*

**Решение систем линейных уравнений**

 Пример 1. Решим систему линейных уравнений матричным способом

 2x - 4y + z = 3

-x + 2y + 3z = 2

-0.5x + 4y - 6z =-1

Чтобы решить эту систему, надо ее разделить на два массива. Введите левую часть системы в матрице **D,** а правую часть системы (вектор) в массиве **E**. Решите систему при помощи оператора "**\**" и в результате получите:

>> D\E

ans =

 4.6667

 1.8333

 1

*Задание 1.* Для проверки, умножьте матрицу **D** на ans. Что вы получили?

**Удаление строк и столбцов**

*Пример 1.* Удалим вторую строку матрицы B, для этого, надо написать следующее:

>> B(2,:)=[] тогда получим:

B =

 -1 4 2

 -7 0 8

Чтобы удалить нескольких идущих подряд столбцов, например, удалить первой и второй, надо написать следующее:

B(:,1:2)=[] тогда получим:

B =

 2

 8

*Задание 1.* Удалите первую строку и третий столбец массива **D**. После этого напишите полученную систему и определите значения **x** и **y**.

## Практическая работа №3

## ВЗАИМОДЕЙСТВИЕ МЕЖДУ СИСТЕМОЙ MATLAB И ПРОГРАММАМИ ИЗ ПАКЕТА MS OFFICE (WORD И EXCEL)

1. **Цель работы**

Получить практические навыки по одновременной работе в системе **MatLab** и в прикладных программах.

1. **Порядок выполнения**

Система MatLab интегрируется с прикладными программами, такими как MS Word и MS Excel. Связь MatLab и Word обеспечивает возможность написания в редакторе Word интерактивных документов, так называемых **M-книг**, основанных на специальном шаблоне. Можно осуществлять запуск блоков команд MatLab непосредственно из документа Word, причем результат выполнения команд будет автоматически отображаться в редакторе Word, т.е. в M-книге. Таким образом, можно быстро создавать отчеты при решении задач в MatLab.

Excel Link, входящая в поставку системы MatLab, снабжает пользователя программы Excel доступом ко всем функциям MatLab, которые значительно расширяют возможности табличного процессора.

**Знаки, показывающие готовность Word работать с М-книгой:**

- появление в меню **Файл** пункта **Создать** (New) – **Общие шаблоны**… **шаблон** M-book (М-книги);

- пополнения, определенные в М-книге *стили*: AutoInit, Calc, Error, Input, NoGraph, Output;

- присутствие меню **Notebook**, предназначенного для управления и редактирования интерактивной М-книгой (после создания файла М-книги);

**Некоторые обозначений:**

*Define Input Cell* – определить ячейки ввода;

*Input cell* – ячейка ввода;

*Evaluate Cell* – вычислить ячейку;

*Evaluate Cells* – вычислить ячейки;

**Простейшие операции**

Имеется несколько способов, позволяющих начать работу над новой М-книгой:

1. Запустить MatLab, набрать в командной строке команду *notebook* и нажмите <Enter>. После этого в Word появляется новый файл, основанный на шаблоне *m-book.doc* . Если Word не был открыт, то он запустится после выполнения данной команды;

2. Запустить Word, создать новый файл при помощи пункта **Создать…** меню **Файл** Microsoft Word. В диалоговом окне при создании документа на вкладке **Общие** (Создание с помощью шаблона) следует выбрать шаблон m-book, установить переключатель **Создать документ** и нажать кнопку **ОК** (см. рис 1).



Рис. 1

*Пример 1.* Наберите в документе, следующую команду:

I=log2(32)+7

Поместите курсор в набранную строку и выберите в меню **Notebook** пункт Define Input Cell. Не трудно заметить, что стиль набранного текста и цвет шрифта изменились, сам текст заключился в квадратные скобки.

Замечание: при печати М-книги скобки не выводятся.

[**I=log2(32)+7**]

Таким образом, можно сказать, что образовалась так называемая ячейка ввода (Input Cell). Для получения значение I или выполнения команды MatLab, следует убедиться, что данная ячейка является текущей, т.е. в ней находится курсор, и выбрать в меню **Notebook** пункт **Evaluate Cell**. Ниже ячейки ввода в документе появляется ячейка вывода с результатом для пользователя MatLab виде:

[I =

 12]

Абзацы ячейки вывода имеют стиль **Output**, начало и конец ячейки ограничены квадратными скобками, а цвет шрифта другой (синий на пример).

Комментарий: Для пользователя имеется возможность переопределить стили шаблона m-book.doc так же, как и любого другого стиля, выбрав в меню **Формат** пункт **Стиль** и произведя нужные установки в появившемся диалоге окне.

**Выполнение нескольких команд одновременно в М-книге**

Для этого, сначала следует набрать операторы в тексте документа; потом каждый оператор или весь текст (путем выделения) заключить в ячейку ввода, выбирая в меню **Notebook** пункт **Define Input Cell** либо используя комбинацию клавиш <Alt>+<D>.

*Пример 1.* Наберите следующий текст:

q=[0 1 2 3 4 5 6 7 7.5 8 9 10 11 12 13 14 15];

p1=[30 40 45 40 30 25 30 40 50 60 70 75 80 90 80 90 100];

p2=[60 68 70 65 70 75 75 80 80 80 75 70 80 80 90 80 70];

plot(q, p1, 'b\*--', q, p2, 'rs-')

grid on

title ('посещение сайта')

xlabel('день')

ylabel('количество человек')

legend('МАЯ','ИЮЛЯ')

Команды образовавшейся в группу выполняются из пункта **Evaluate Cell**(или **Evaluate Cells**) в меню **Notebook**. В результате содержимое М-книги дополняется ячейкой вывода с решением задачи линейного программирования (рис. 2).

0

5

10

15

20

30

40

50

60

70

80

90

100

посещение сайта

день

количество человек

МАЯ

ИЮЛЯ

Рис.2 Результат выполнения группы ввода ячеек

Работа с М-книгой большого объема становится проще, если предусмотреть разбивку ее на разделы (Calc Zone). Изучите остальные команды пункта **Notebook** и решите следующую задачу в М-книге.

**Знак, показывающий готовность Excel работать с MatLab**

- присутствие панели инструментов **Excel Link**, содержащей три кнопки: *putmatrix, getmatrix, evalstring*. Эти кнопки реализуют основные действия, требуемые для осуществления взаимосвязи между Excel и MatLab.

**Роль команды putmatrix и getmatrix**

*Пример 1.* Наберите на рабочем листе Excel таблицу с данными, как показано на рис. 3. Далее выделите диапазон ячеек **A1**:**D3** и нажмите кнопку **putmatrix –** появится диалоговое окно Excel, предназначенное для определения имени переменной матрицы (введите переменную S и нажмите кнопку OK) , в которую следует экспортировать данные из программы Excel в MatLab.



рис. 3

Перейдите к командному окну MatLab, наберите S и нажмите <Enter>; таким образом, получим:

» S %<Enter>

S =

 3.0000 6.0000 -2.0000 5.1000

 11.0200 0.3400 9.0000 21.0000

 8.0000 -7.0000 -10.0000 4.0000

Можно эти данные обрабатывать в MatLab, например:

» R=S(1:3,2:4) %<Enter>

R =

 6.0000 -2.0000 5.1000

 0.3400 9.0000 21.0000

 -7.0000 -10.0000 4.0000

а результат импортировать в Excel либо, например, создать магический квадрат из 25 элементов для их обработки в Excel. В MatLab наберите в командной строке следующую команду и нажмите <Enter>:

» mk=magic(5) %<Enter>

mk =

 17 24 1 8 15

 23 5 7 14 16

 4 6 13 20 22

 10 12 19 21 3

 11 18 25 2 9

Полученные данные можно отправить в Excel для обработки, к примеру, для проверки, является ли данная матрица магическим квадратом. Чтобы импортировать эти данные Excel, надо указать пустую ячейку, которая будет являться верхним левым элементом матрицы (например B6), потом нажать на кнопку **getmatrix**; после этого появляется диалоговое окно со строкой ввода, в которой требуется ввести имя переменной (в нашем примере **mk**) импортируемой в электронной таблице Excel, и нажать OK, чтобы переместить эти данные в ячейке листа согласно размера массива.

**Роль команды evalstring**

Находясь в рабочей области Excel, можно выполнить команды MatLab в строке ввода, которые появляются при нажатии кнопки **evalstring**, расположенные на панели Excel Link.

*Пример 1.* Нажмите кнопку **evalstring** , в появившемся диалоговом окне, наберите команду MatLab: F=factorial(5) и нажмите OK. (F берет значение факториала целого числа 5: это 5x4x3x2x1=120). Чтобы увидеть результат вычисления, перейдите в рабочую область MatLab, и наберите F, и нажмите <Enter>; получим сразу значение 120, которое можно использовать для других вычислений. Возможно, и не переходить в рабочую область MatLab и использовать кнопку **getmatrix** для получения результата. Можно было получить такой же результат, если вы набрали бы команду F=factorial(5) в среде MatLab.

## Практическая работа №4

## ТАБУЛИРОВАНИЕ ФУНКЦИЙ И ПОСТРОЕНИЕ ГРАФИКОВ

1. **Цель работы**

Научиться работать с функциями одной и нескольких переменных, строить графики и производить табулирование. Задавать функции различными способами.

1. **Порядок выполнения**

**Построение диаграмм с помощью функции bar**

Диаграмма позволяет представить числовые данные в графическом виде. Значение элемента вектора пропорционально высоте столбика диаграммы или площади сектора диаграммы. Для отображения данных в графическом виде используется функция ***bar*** с аргументом ***x***.

*Пример 1.* Зафиксируем в переменную **fd** результаты измерения погоды в течение недели. Введите эти данные в массиве “A” в виде вектор-столбец. После этого нарисуете диаграмму значение с помощью функции bar. Для этого, надо писать в командной строке следующее: bar(A).

Если, правильно выполнили алгоритм действий, то появится графическое окно, содержащее столбчатую диаграмму результаты измерений погоды.

В массиве “В” находится дата (день) соответствующие каждому измерению.

*Пример 2.* Напишите массив “В” в виде вектор-строка и нарисуете диаграмму с помощью функции bar. В этот раз, надо писать в командной строке следующее: bar(В, A). Чем эти графики отличаются друг от друга?

*Задание 1.* Выбор ширины столбцов осуществляется заданием третьего дополнительного аргумента. По умолчанию ширина равна 0.8. Что происходит если ширина α меньшее, большее или равно 1?

**Построение диаграмм с помощью функций barh** **и bar3**

*Пример 1.* Отобразите функцию **у** (вариант функции, полученной в практической работе №3) на отрезке [a, b] в виде столбчатой диаграммы с промежутков и без него. Для этого, используйте следующие аналогичные функции bar: **barh** и **bar3**.

*Задание 1.* Напишите, чем отличается каждая из этих функции.

**Построение диаграмм с помощью функций pie и pie3**

Если требуется оценить вклад каждого из элементов вектора в общую сумму его элементов, то удобно построить круговую диаграмму при помощи функции pie или pie3. Синтаксис функции pie и pie3 одинаковый: ***pie(x, y)***, где x – числовые данные для построения круговой диаграммы и у – векторы позволяющие разделить каждый сектор. Они могут быть либо ноль, либо другие значения. Размер вектора **х** должен быть равен размеру вектора **у**.

*Задание 1.* Построить круговую диаграмму с помощью функции pie и pie3. Используйте данные в массиве A.

*Задание 2.* Постройте диаграмму без параметра **у**. Чем они отличаются полученные диаграммы? Когда функция pie(x, y) дает тоже сам результат, что и функция pie(x)?

**Интерпретация команд**

*Задание 1.* Выполните следующие команды и сделайте комментарии для каждой команды.

>> d=[23.5 10.24 34 57.02 40];

>> p=zeros(size(d));

>> ind=3;

>> p(ind) = 1;

>> pie(d, p)

**Распределение данных по интервалам**

Для получения наглядного представления о распределении данных, которые попали в тот или иной интервал, применяется функция **hist**. Синтаксис описание функции **hist**: hist(x, y), где х – вектор, содержащий распределенные данных по интервалам и у – интервалы (или центры интервалов) попадания значения х. Для увеличения числа интервалов следует в качестве второго аргумента указать число интервалов. Функцию hist можно вызывать с одним или двумя выходными аргументами.

>> % Генерируете случайный массив с размером 10*x*1:

>> mas = randn(10, 1);

>> hist(mas)

Функция rose предназначена для построения угловых гистограмм (в полярных координатах). Аргументом функции rose является вектор значений в радианах.

*Пример 1.*  Пусть в течение суток каждый час, измерялось направление ветра в градусах. Результат измерений содержится в файле *data.txt*. Для выяснения преобладающего направления используйте круговую гистограмму, считав значения из файла в вектор в *dat* и преобразовав их значения в радианах. Файл *data.txt* имеет следующие данные: 20, 300, 50, 166, 70, 212.5, 400.34

Для того чтобы нарисовать гистограмму распределения направлений ветра надо выполнить следующие действие:

>> dat=load ('data.txt');

>> datr=dat\*pi/180;

>> rose(datr)

**Графики функций одной переменной**

С помощью системы MatLab, можно построить графики для визуализации функций одной и двух переменных, задав минимальный набор параметров. Построение графиков функций одной переменной в линейном масштабе осуществляется при помощи функции **plot**. В зависимости от входных аргументов функция *plot* позволяет строить один или несколько графиков, изменять цвет и стиль линий и добавлять маркеры на каждый график.

*Пример 1*. Постройте график функций f(x)=ln(10.2+x2)+cos2 (x/9.2) при х=[-30\*pi:0.05:30\*pi]. Для этого надо набирать следующие команды в командной строке:

>> x=[-30\*pi:0.05:30\*pi];

>> y=log(10.2+x.^2)+cos(x./9.2);

>> plot(x, y)

В результате получите график, который находится на рисунке 1.



рис. 1

Рассмотрим, как построить два графика в одной координатной плоскости.

*Пример 2.* Постройте графики функций f(x)=ln(10.2+x2)+cos(x/9.2) и h(y)= ln(5.2+x2)+cos(x/2) на отрезке [-15, 15], с шагом 3/100 . Для этого надо:

- сгенерировать вектор-строку значений аргумента х и вектор-строки f и h, содержащие значения функций;

- с помощью команды **plot(x, f, x, h)** постройте график.

 В результате вы получите график, изображенный на рисунке 2.

 

рис. 2

Иногда требуется сравнить поведение двух функций, значения которых сильно отличаются друг от друга. В этой ситуации помогает функция **plotyy(x1, f1, x2, f2)** – где x1, x2 - вектор-строки значений аргумента и f1, f2- вектор-строки, содержащие значения функций.

*Пример 3.* Построимграфик прямой от точки *k(x1, y1)* до точки *z(x2,y2)*. Для этого, надо разместить эти координаты в массиве D и E таким образом: D=[ x1 x2 …] и E=[ y1 y2 …]. Выберите конкретные координаты точек и проведите построение.

**Оформление графиков одной переменной**

Кроме того, что система MatLab ставит автоматически координаты оси, есть еще возможность более красиво оформить график.

*Пример 1.* Пусть в каждом определенном времени (секунды – массив **q**), web-мастер определяет количество посещений (массив **p**) сайта. Введем результаты наблюдения в массиве, и потом рассмотрим в графическом виде, как часто пользователи посещают этот сайт в два разных периода (p1, p2).

Введите следующие команды:

>> q=[0 1 2 3 4 5 6 7 7.5 8 9 10 11 12 13 14 15];

>> p1=[30 40 45 40 30 25 30 40 50 60 70 75 80 90 80 90 100];

>> p2=[60 68 70 65 70 75 75 80 80 80 75 70 80 80 90 80 70];

>> %

Для того чтобы нарисовать оба графика, зададим формы маркеров, цвет и тип линий:

>> plot(q, p1, ‘b\*--’, q, p2, ‘rs-’)

*Задание 1.* Выполняете последовательность следующих команд и прокомментируйте каждую из них.

>> grid on

>> title (‘посещение на сайт’)

>> xlabel (‘Время (секунда)’)

>> ylabel (‘Кол-во посещение’)

>> legend (‘за мая’, ‘за августа’)

При добавлении легенды следует учесть, что порядок и количество аргументов команды legend должны соответствовать линиям на графике.

*Задание 2.* Выполняете последовательно следующие команды и напишите, что означает каждый последний аргумент или какую роль это новый аргумент играет.

>> legend (‘за мая’, ‘за августа’, -1)

>> legend (‘за мая’, ‘за августа’, 0)

>> legend (‘за мая’, ‘за августа’, 1)

>> legend (‘за мая’, ‘за августа’, 2)

>> legend (‘за мая’, ‘за августа’, 3)

>> legend (‘за мая’, ‘за августа’, 4)

**Графики функций двух переменных**

С помощью MatLab можно построить графики функций двух переменных. Построение графиков функций двух переменных осуществляется при помощи функции **mesh, surf, …**.

*Пример 1.* Разбиваем области определения прямоугольной сеткой с помощью команды **meshgrid**.

>> [a, b] = meshgrid (-5\*pi:0.5:5\*pi, -10\*pi:1.05:6\*pi); %

Замечание:Можно убрать точку с запятой, чтобы проконтролировать генерацию массивов.

*Пример 2.* Вычислим значения функции c=ln(10.2+a2)+cos(b/9.2) в точках пересечения линий сетки и записываем их в матрицу.

>> c=log(10.2+a.^2)+cos(b./9.2);

Для завершения, построим график.

>> mesh(a, b, c)

*Задание 1.* Произведите увеличение и уменьшение шага интервального значения a и b на 50%, и построить графики. Какие изменение вы наблюдаете?

Цвет линий поверхности соответствует значениям функции. Система MatLab рисует только видимую часть поверхности. При помощи команды **hidden off** (команда hidden on убирает невидимую часть поверхности), можно сделать каркасную поверхность “прозрачной”, добавив скрытую часть.

*Задание 2.* Выполняете следующие команды и опишите их роль:

*>> surf(a, b, c)*

*>> colorbar*

*>> shading flat*

*>> shading interp*

*>> shading faceted*

Команды **meshc** и **surfc** позволяют получить более точное преставление о поведении функции. Функция **contour3** позволяет построить поверхность, состоящую из линий уровня. Число линий уровня выбирается автоматически.

*Пример 3.* Построим поверхность, состоящую из линий уровня, соответствующих значениям от -20 до 20 с шагом 0.05:

>> uroven = [-20: 0.05: 20];

>> contour3 (a, b, c, uroven)

>> colorbar

Система MatLab предоставляет возможность получать различные типы контурных графиков при помощи функций **contour** и **contourf**. (contour (a, b, c); contour (a, b, c, uroven); …).

**Оформление графиков двух переменных**

Для оформления графиков, используются те же функций, как и с одной переменной. Кроме того, можно использовать команду **colormap (тип палитры)** для изменения палитры графического окна. Тип палитры: *bone, colorcube, cool, copper, flag, gray, hot, hsv, jet, pink, prism spring, summer, vga, white, winter*.

При построении трехмерных поверхностей оси координат располагаются всегда одинаковым образом. Часть поверхности остается при этом скрытой. Для получения полной информации о поверхности ее желательно “осмотреть” со всех сторон. Для этого MatLab использует функцию **view**.

*Пример 1.* Для того чтобы узнать текущее положение наблюдателя, следует вызвать view с двумя выходными аргументами:

>> [coorX, coorY] = view

Для того, чтобы увидеть конкретную часть поверхности *xy* под углом u  и v , используем команду view(u , v ).

*Задание 1.* Измените, координаты таким образом, чтобы можно было просмотреть график сверху. При этом вы получите вид графика, изображенного рис. 3



рис. 3

**Работа с несколькими графиками**

До сих пор, все графики, которые вы нарисовали, выводились в специальное окно с заголовком **Figure No. 1**. Каждый раз новый график выводился в то же самое окно. Для того чтобы выводить несколько графиков в одном окне, надо набрать в командной строке команду **figure**. Команда создает пустое графическое окно, которое становится текущим. Для того чтобы сделать какое-то графическое окно текущим, следует щелкнуть на него мышкой. Для очистки всего текущего окна, используется команда **clf** , а для того, чтобы убрать только график, но оставить оси, заголовок и названия осей, следует применить команду **cla**.

Ранее, мы уже рассмотрели, как можно отобразить нескольких графиков функций одной переменной в одной координатной плоскости. Для трехмерных графиков используется команда **hold on**, которую нужно задать перед построением графика. Команда **hold on** может так же применяться и для расположения нескольких графиков функций одной переменной.

Команда >> plot(x, f, x, g) эквивалентна последовательности команд:

>> plot (x, f)

>> hold on

>> plot (x, g)

Система MatLab позволяет разбить графическое окно на несколько подграфиков со своими осями. Для этого используется команда **subplot**, которая располагает подграфики в виде матрицы и используется с тремя параметрами: subplot(i, j, n), где i и j – число подграфиков по вертикали и горизонтали, n-номер подграфика, которые надо сделать текущем.

*Задание 1.* Выполните следующие команды:

>> subplot (2, 3, 1)

>> subplot (2, 3, 2)

>> subplot (2, 3, 3)

Можно ли выполнить следующую команду: >> subplot (2, 3, 7) и почему?

## Практическая работа №5

## ИНТЕГРИРОВАНИЕ ФУНКЦИЙ, ПРОГРАММИРОВАНИЕ И ДРУГИЕ ВЫЧИСЛЕНИЯ

1. **Цель работы**

Научиться использовать М-файлы, находить корни уравнений, интегрировать, создавать собственные функции. Освоить основы программирования в системе MatLab.

1. **Порядок выполнения**

Для выполнения данной работы, необходимо иметь навыки работы в системе MatLab или выполнить обязательно все предыдущие практические работы. Кроме этого, работа из командной строки затрудняется, если надо вводить много команд и часто их изменять. Ведение дневника при помощи команды diary и сохранение рабочей среды незначительно облегчает работу. Самым удобным способом выполнения команд в системе MatLab является использование М-файлов, в которых можно набирать все команды сразу. При помощи редактора М-файлов можно создавать собственные функции и вызывать их, в том числе и из командной строки.

**Работа в редакторе М-файлов**

М-файлы в системе MatLab бывают двух типов:

– файл-программы (Script M-Files), содержащие последовательности команд;

– файл-функции (Function M-Files), в которых описываются функции, определяемые пользователем.

Для того чтобы создать М-файл раскройте меню **File** основного окна системы MatLab и в пункте **New** выберите подпункт M-file.

**Файл-программы**

*Задание 1.* Создайте М-файл и наберите в редакторе последовательность команд, позволяющих решить следующую задачу: пусть требуется вывести таблицу значений и построить график функции y1(x) на отрезке [x1, x2] с шагом p1 (выберите конкретные значения, соответствующие формулировке задачи). Определите алгоритм для решения этой задачи в системе MatLab.

Далее сохраните файл с именем *filepro1.m* в вашей папке.

Для запуска на выполнение всех команд, содержащихся в файле, следует выбрать пункт **Run** в меню **Tools** или в меню **Debug** (в зависимости от версии программы MatLab).

Запустить на выполнение команды можно и следующим образом: набрать в командной строке:

>> имя\_файла\_для\_выполнения

В нашем случае: >> filepro1

Комментарий: Если при наборе сделана ошибка, и MatLab не может распознать команду, то происходит выполнение команд до неправильно введенной, после чего выводится сообщение об ошибке. Команды MatLab файл-программы осуществляют вывод в командное окно. Для подавления вывода следует завершать команды точкой с запятой.

Необходимо запомнить, что для выполнения части команд их следует выделить и нажать <**F9**>.

*Задание 2.* Что происходит после того, как будут выполнены следующие команды ?:

>> edit

>> edit filepro1

*Замечание:* Для того чтобы понять действие, надо закрыть появившееся окно и потом выполнить следующую команду. Какое заключение вы можете сделать?

**Файл-функции**

*Задание 1.* Определить средний балл Ci каждого студента в группе по пятьсотбалльной системе с помощью формулы: Сi = (*x*1*+x*2*+x*3)5*+x*4\*35; где *x*1 *,x*2 *,x*3 – соответственно результат теста, оценка посещение занятия и оценка конспекта в 10-ти балльной системе, x4 – результат последнего теста. ЕслиСi  < 200, то студент – очень слабый; если200  Сi  < 300, то студент – слабый; если 300 Сi < 400, то студент получает удовлетворительную оценку; если 400 Сi  < 475 , то студент получает оценку хорошо; если 475 Сi , то студент получает оценку отлично/

Алгоритм решение этой задачи представлен на рис. 1.

Начало

Ввод n, x1, x2, x3, x4

i := 

Сi := (*x*1*+x*2*+x*3)5*+x*4\*35

Конец

да

нет

Сi <200

оц:=очень слабый

Сi >=200 & Сi <300

да

оц:= слабый

нет

нет

да

Сi >=300 & Сi <400

да

нет

Сi >=400 & Сi <475

оц:= удовл.

оц:= отлично

оц:= хорошо

Вывод отметки Сi и оценки оц.

Рис. 1

Можно решить эту задачу по-разному. Самый удобный и быстрый способ следующий: имеет смысл один раз написать отдельно формулу в виде функции, которая часто повторяется при вычислении. И потом вызывать функцию там, где она используется.

Выполните следующие действия:

– откройте в редакторе М-файлов новый файл, и наберите следующий текст:

function C = filefun1(a,b,c,d)

 C = (a+b+c)\*5+35\*d;

 % a, b, c и d – входные аргументы;

 % С – единственный выходной аргумент (вычисленное значение записывается в С);

– сохраните файл в вашем рабочем каталоге с названием filefun1. Название файла должно быть то же, что и название функции.

Далее созданную функцию можно использовать так же, как и встроенные sin, cos, sqrt и другие.

Например, из командной строки можно определить средний балл студента, получившегося следующие оценки (c переводом в пятисотбалльную систему):

*5 из 10 - результат теста;*

*7 из 10 - оценка посещение занятия;*

*8 из 10 - оценка конспекта;*

*9 из 10 – результат последнего теста.*

>> my\_mark = filefun1(5,7,8,9)

my\_mark =

 415

% функция filefun1 должна всегда иметь 4 входных аргумента.

– теперь, чтобы решить поставленную задачу, надо написать программу на языке MatLab. Мы будем продолжать решать эту задачу в разделе программирования.

**Файл-функции с несколькими выходными аргументами**

Здесь, функция возвращает несколько значений при вычислении. Выходные аргументы добавляются также через запятую в список выходных аргументов, а сам список заключается в квадратные скобки.

*Пример 1.* Определим количество байтов, мегабайтов и килобайтов, если знаем размер в гигабайтах.

Для этого, надо писать следующую функцию:

function [Mb, Kb, b] =convertGtobyte (gigabyte)

 Mb = gigabyte\*1024;

 Kb = gigabyte\*(1024^2);

 b = gigabyte\*(1024^3);

Для того чтобы вызвать данную функцию, надо написать следующие инструкции в командной строке:

>> [m,k,b]=convertGtobyte(0.5)

% 0.5Гб равен, сколько Мб, Кб и Байтов?

Тогда получим:

m =

 512

k =

 524288

b =

 536870912

то есть: что 0.5Гб = 512Мб; 0.5Гб = 524288Кб; 0.5Гб = 536870912

*Задание 1.*  Напишите функцию, позволяющую определить количество байтов, гигабайтов и килобайтов, если известен размер в мегабайтах.

Замечание: Если список выходных аргументов пустой, т.е. заголовок выглядит следующим образом:

function filefun1(a,b) или function [ ] = filefun1(a, b),

то файл-функция не будет возвращать никаких значений.

**Решение произвольных уравнений**

Нахождение корней произвольных уравнений осуществляет встроенная функция **fzero**, для определения всех корней полиномов применяется **roots**.

Для решения произвольных уравнений, надо написать следующую команду:

x = fzero(‘myf’, x0) , где myf – имя файл-функции, вычисляющая левую часть уравнения; x0 – начальное приближение к корню; x – найденное приближенное значение корня.

*Пример 1.* Решим на отрезке [-5, 5] уравнение: sin*x*-*x*2cos*x* = 0 . Для этого выполним следующие действия:

– перед нахождением корней полезно построить график функции, входящей в левую часть уравнения. Это позволит визуально определить область пересечения функции и оси *x* . Конечно, построить график можно при помощи plot, но все равно понадобится написать файл-функцию, поэтому имеет смысл воспользоваться **fplot**, которая к тому же позволяет получить более точный график по сравнению с plot.

– написать файл-функцию:

function y = mayafun(x)

 y = sin(x)-x.^2.\*cos(x);

– построить график *mayafun*, используя *fplot*, и нанести сетку.

fplot(‘mayafun’, [-5 5])

grid on

– уточнить значение корня, расположенного вблизи x=-5, при помощи функции fzero:

x1 = fzero(‘mayafun’, -5)

% после того, как вы нашли значение x1 , надо проверить ответ, вычислив значение функции *mayafun* в точке x1 . (>> mayafun(x1)). Результат должен быть равен нулю или приближен к нулю.

– уточнить следующее значение корня, расположенного вблизи x=-2, при помощи fzero:

x2 = fzero(‘mayafun’, -2)

*Задание 1.* Проверьте ответ и найдите остальные корни, расположенные на других отрезках, если они есть.

**Вычисление всех корней полинома**

*Пример 1.* Вычислим все корни полинома:

s = .

Для этого, необходимо пройти следующие этапы:

>> %Нахождение коэффициенты:

>> k = [2 -4.7 0 1 1.3 0 -4];

Нахождение сразу всех корней полиномов осуществляется при помощи функции **roots**, в качестве аргумента которой указывается вектор с коэффициентами полинома.

>> s = roots (k)

s =

 2.2268

 0.89629 + 0.58868i

 0.89629 - 0.58868i

 -0.38839 + 0.85098i

 -0.38839 - 0.85098i

 -0.89264

Число корней полинома (в нашем случае - 6), как известно, совпадает со степенью полинома.

С помощью функции **polyval**, можно вычислить значения полинома от вектора его корней.

Проверим, правильно ли функция *roots* определила все корни полинома.

Для этого, набираем следующую команду:

>> polyval (k, s)

-4.4453e-013

-5.3291e-015 +6.6613e-015i

-5.3291e-015 -6.6613e-015i

 1.5987e-014 +5.5511e-015i

 1.5987e-014 -5.5511e-015i

 8.8818e-016

% Все шесть значений почти равны нулю.

**Минимизация и максимизация функций**

Встроенные функции MatLab позволяют минимизировать функции одной или нескольких переменных. Результатом является локальный минимум, т.е. точка, в окрестности которой исследуемая функция имеет большие значения по сравнению со значением локального минимума.

Для поиска локального минимума функции одной переменной на некотором отрезке используется функции ***fmin***, а для функции нескольких переменных по заданному начальному приближению ***fmins***.

*Пример 1.* Найдем локальные минимумы следующей функции:

y =  на отрезке [0.01, 4.5].

Для этого выполним следующую последовательность действий:

– перед нахождением локальных минимумов постройте график исследуемой функции командой fplot.

Файл-функция:

function y = fprimer(x)

y=cos(2.\*pi.\*x+3)./sqrt(x);

Построим график, используя файл-функцию fprimer, он будет соответствовать рисунку 2.

Для этого наберите следующие команды:

fplot(‘fprimer’, [0.01 4.5])

grid on

% для нанесения сетки.

0.5

1

1.5

2

2.5

3

3.5

4

4.5

-10

-8

-6

-4

-2

0

2

Рис. 2

– из графика, приведенного на рис. 2, видно, что функция имеет четыре локальных минимума. Для нахождения всех локальных минимумов необходимо выполнить четыре этапа.

Находим первый:

>> x1 = fmin(‘fprimer’, 0.7, 1.3) тогда x1 = 1.01;

Находим второй:

>> x2 = fmin(‘fprimer’, 1.8, 2.3) тогда x2 = 2.0163;

Находим третий:

>> x3 = fmin(‘fprimer’, 2.7, 3.3) тогда x3 = 3.0183;

Находим четвертый:

>> x4 = fmin(‘fprimer’, 3.8, 4.3) тогда x4 = 4.0194.

*Замечание:* Для нахождения локального максимума нет специальной функции, очевидно, что следует искать минимум функции с обратным знаком.

*Задание 1.* Найдите локальные максимумы*.*

**Интегрирование функций**

Для вычисления интегралов, пользователь имеет возможность выбрать подходящий метод численного интегрирования в зависимости от свойств подынтегральной функции.

*Пример 1.* Пусть необходимо найти значение I1 =.

На первом этапе, надо создать файл-функции, вычисляющую подинтегральное выражение:

function f=integr1(x)

f = 2.\*x.\*cos(x.^2);

На втором этапе используем функцию **quad** или **quad8** (основанной на более точных квадратурных формулах Ньютона-Котеса) для вычисления интеграл с точностью 10-3.:

>> I1 = quad('integr1',0, 2)

 %тогда получим I1 = -0.7568

Для повышения точности вычислений следует задать дополнительный четвертый аргумент:

>> I1 = quad8('integr1',0, 2, 1.0e-06)

Для вычисления двойных интегралов используется функция **dblquad**.

*Пример 2.* Найдем значение следующего двойного интеграла:

I2 = 

Процедура вычисления аналогична, только надо использовать функцию *dblquad*. Опишем функцию:

function f=integr2(x,y)

f = (2.\*y+1).\*cos(x.^3);

>> I2 =dblquad('integr2',-pi, pi, 0, 2)

I2 = 9.11089663553435

Если седьмым аргументом указать ‘*quad8*’, то вычисления будут основаны на квадратурных формулах Ньютона-Котеса

>> I2 = dblquad('integr2',-pi, pi, 0, 2, 1.0e-012, 'quad8')

I2 = 9.11092317515518

**Программирование**

Продолжим решение задачи, которая была поставлена в разделе *файл-функции*. Чтобы решить данную задачу, в целом, надо создать файл-программу, для этого написать на языке MatLab коды каждого блока и соединить их как указано на рисунке 1.

%Программа на MatLab

n=input(' Введите количество студентов = ');

for i=1:n

 disp('Введите следующие данные в 10-ти балльной системе')

 disp('для студента № ')

 disp(i)

 disp(' ')

 x1=input('Результат теста № 1 = ');

 x2=input('Оценка посещение занятия = ');

 x3=input('Оценка конспекта = ');

 x4=input('Оценка последнего теста = ');

 Ci = filefun1(x1,x2,x3,x4);

 if Ci < 200

 Ots = 'Очень слабый';

 elseif (Ci >=200) & (Ci < 300)

 Ots = 'Слабый';

 elseif (Ci >=300) & (Ci < 400)

 Ots = 'Удовлетворительно';

 elseif (Ci >=400) & (Ci < 475)

 Ots = 'Хорошо';

 else

 Ots = 'Отлично';

 end

 disp('Полученная Отметка:')

 disp(Ci)

 disp('Полученная Оценка:')

 disp(Ots)

 disp(' ')

end

disp('Конец программы !!!')

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Дьяконов В.П* Расширяемые системы для численных расчетов MatLAB. Монитор - Аспект. 1993. – №2.
2. *Дьяконов В.П* Справочник по применению системы PC MatLAB. – М.: Наука, Физматлит, 1993.
3. *Потемкин В.Г.* MATLAB 5 для студентов. – М.: Диалог - МИФИ, 1998.
4. *Потемкин В.Г.* Система инженерных и научных расчетов MATLAB 5.х В 2-х т. – М.: Диалог - МИФИ, 1999.
5. *Дьяконов В.П., Абраменкова И.В.* MATLAB 5.0/5.3 Система символьной математики. – М.: Нолидж, 1999.
6. *Дьяконов В.П*  Matcad 2001: учебный курс. –СПб.: Питер, 2001.
7. *Могилев А.В., Пак Н.И., Хеннер Е.К.* Информатика: Учеб. пособие для студентов пед. вузов. – М.: Академия, 2003 (издание 3-е).
8. *Угринович Н.Д.* Информатика и информационные технологии: Учебник для 10-11 классов. – М.:БИНОМ. Лаборатория знаний, 2003.
9. *Захарова И.Г.* Информационные технологии в образовании. – М.: Академия, 2003.
10. *Шульга С.Б., Медведев Ю.А.* Информационные и коммуникационные технологии в образовании (практикум). – Владимир: ВГПУ, 2008.
11. *Медведев Ю.А.* Информатика (практикум): учебное пособие для студентов вузов с грифом УМО. – Владимир: ВГПУ, 2008.

ОГЛАВЛЕНИЕ

**ПРЕДИСЛОВИЕ**…………………………………………………………………………….

**ГЛАВА 1. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ СРЕДСТВАМИ MATHCAD**. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

Практическая работа № 1. ВВЕДЕНИЕ В MATHCAD – ПРОСТЫЕ И СЛОЖНЫЕ ОПЕРАЦИИ . . . . . . . . . . . . . . . . 5

Практическая работа № 2. МАТРИЧНЫЕ ОПЕРАЦИИ . . . . . . . . . . . . . . 7

Практическая работа № 3. ТАБУЛИРОВАНИЕ ФУНКЦИИ И ПОСТРОЕНИЕ ГРАФИКОВ . . . . . . . . . . . . 10

Практическая работа № 4. ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ УРАВНЕНИЙ . . . . . . . . . . . . . . . . . . . . . . . . 14

Практическая работа № 5. РЕШЕНИЯ УРАВНЕНИЙ В СИМВОЛЬНОМ ВИДЕ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

Практическая работа № 6. ВЫЧИСЛЕНИЕ СУММ И ПРОИЗВЕДЕНИЙ. СИМВОЛЬНЫЕ ВЫЧИСЛЕНИЯ . . . . . . . . 19

Практическая работа № 7. ДИФФЕРЕНЦИРОВАНИЕ И ВЫЧИСЛЕНИЕ ИНТЕГРАЛОВ. . . . . . . . . . . . . . . . . . . . . . . . 22

Практическая работа № 8. ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ . .25

Практическая работа № 9. ВСТРОЕННЫЕ ФУНКЦИИ. . . . . . . . . . . . . 33

Практическая работа № 10. ПРОГРАММИРОВАНИЕ. . . . . . . . . . . . . . 38

Практическая работа № 11. РАЗМЕРНОСТИ . . . . . . . . . . . . . . . . . . . . . 45

**ГЛАВА 2. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ СРЕДСТВАМИ**

 **MATLAB** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .49

Практическая работа № 1. ВВЕДЕНИЕ В MATLAB – ПРОСТЫЕ И СЛОЖНЫЕ ОПЕРАЦИИ . . . . . . . . . . . . . . . 50

Практическая работа № 2. РАБОТА С МАССИВАМИ .. . . . . . . . . . . . . 37

Практическая работа № 3. ВЗАИМОДЕЙСТВИЕ МЕЖДУ СИСТЕМОЙ MATLAB И ПРОГРАММАМИ ИЗ ПАКЕТА MS OFFICE (WORD И EXCEL) . . . . . . . . . 64

Практическая работа № 4. ТАБУЛИРОВАНИЕ ФУНКЦИИ И ПОСТРОЕНИЕ ГРАФИКОВ . . . . . . . . . . . . 71

Практическая работа № 5. ИНТЕГРИРОВАНИЕ ФУНКЦИЙ, ПРОГРАММИРОВАНИЕ И ДРУГИЕ ВЫЧИСЛЕНИЯ . . . . . . . . . . . . . . . . . . . . . . . 81

**БИБЛИОГРАФИЧЕСКИЙ СПИСОК** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .94

**Медведев Юрий Алексеевич**

ИНФОРМАЦИОННЫЕ ТЕХНЛОГИИ В МАТЕМАТИКЕ (практикум)

Учебное пособие для студентов вузов

с грифом УМО

Редактор А. А. Масленникова

Владимирский государственный педагогический университет

План университета 2005 г.

Позиция 67

Подписано в печать 26.12.2005 Формат 84х108 1/32

Усл. п. л. – 6,1 Уч. – изд. л. – 6,4

Заказ 2-06 Тираж 1000 экз.

Отпечатано в отделе оперативной полиграфии ВГПУ

600024, г. Владимир, ул. Университетская, 2, тел. 33-87-40

1. \* Глава написана совместно с Л.М.Груздевой [↑](#footnote-ref-1)
2. [↑](#footnote-ref-2)