МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
Томский государственный университет систем управления и радиоэлектроники

Кафедра автоматизированных систем управления

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Методические указания к практическим занятиям, лабораторным и организации самостоятельной работы

Томск-2020

Лукьянов А.К.
Информационные технологии: Методические указания к практическим занятиям лабораторным и организации самостоятельной работы / – Томск: ТУСУР, 2020. – 82 с.

Методические указания разработаны в соответствии с решением кафедры автоматизированных систем управления

Составитель: доцент каф. АСУ, А.К. Лукьянов

Методические указания утверждены на заседании кафедры автоматизированных систем управления, протокол № 10, 31 октября 2020 года.

Оглавление

ЛАБОРАТОРНАЯ РАБОТА № 1

на тему: «Основы работы в MathCAD»

Цель работы: ознакомиться с основными возможностями, изучить главное меню и панели инструментов пакета MathCAD, а также приобрести практические навыки по выполнению простейших вычислений в нем.

Основные команды MathCAD

Система MathCAD является интегрированной системой, которая ориентирована в основном на проведение математических и инженерно-технических расчётов. В нём объединились понятность, ясность при вычислении с одной стороны и простота в обращении, свойственной электронным таблицам, с другой стороны.

Документ MathCAD, на котором могут быть совмещены текст, графика и формулы, выглядит как страница учебника или журнала, при этом формулы являются «живыми», так как при изменении одной из них, MathCAD скорректирует другие, связанные с первой, атакже пересчитает результаты и перерисует графики.

После запуска приложения MathCAD открывается окно, вид которого представлен на рис. 1.1.

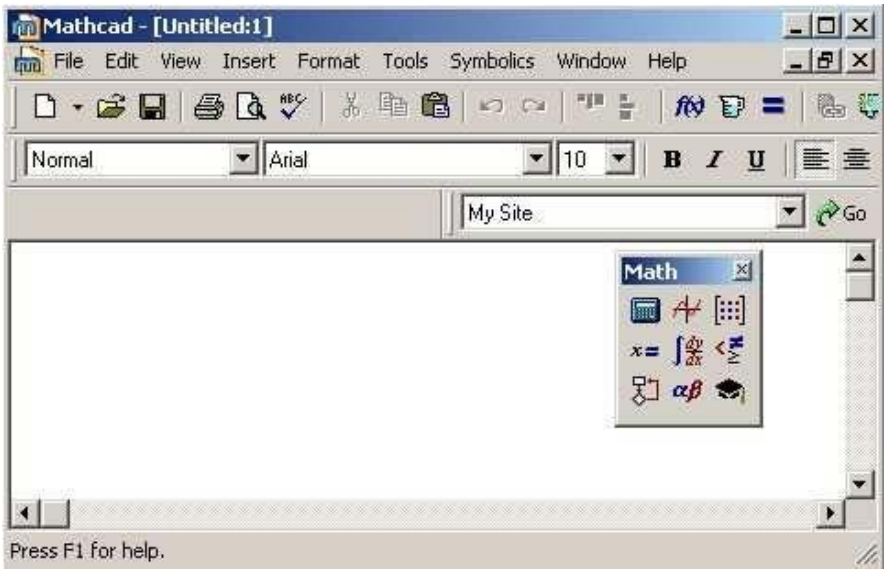


Рис. 1.1 – Рабочее окно системы MathCAD

Главное меню системы MathCAD представлено набором команд, общим для большинства приложений операционной системы MS Windows, а также командами, представляющими специфические возможности [1].

Меню **File** – работа с файлами.

Меню **Edit** – редактирование документов. Меню **View** – настройка элементов окна.

Меню **Insert** – позволяет помещать в MathCAD документ графики, функции, матрицы, гиперссылки, компоненты инастраивать объекты.

Меню **Format** – содержит команды, предназначенные для задания различных параметров, определяющих внешнее представление чисел, формул, текста, абзацев, колонтитулов и т.д.

Меню **Math** – позволяет установить режимы и параметры вычислений.

Меню **Symbolic** – реализует символьные вычисления.

Меню **Window** – содержит команды для упорядочения взаимного расположения нескольких окон и позволяетактивизировать одно их них.

Меню **Help** – информационный центр и справочники.

Кнопки панели Math

Одна из сильных сторон MathCAD – это представление и ввод математических символов и выражений в привычной для человека форме. Открыть соответствующую панель инструментов можно с помощью команды главного меню **View →Toolbars**. Для удобства работы ссылки на них объединены на панели Math.

На панели Math расположены девять кнопок. Каждая из кнопок, в свою очередь, открывает панели инструментов специального назначения, к которым относятся следующие [9].

Calculator. На этой панели находятся кнопки для задания математических операций, а также некоторых часто используемых функций. Эту кнопку можно использовать как калькулятор.

Boolean – для ввода операторов сравнения и логическихопераций.

Evaluation – содержит кнопки для ввода операторов присвоения значений переменных и функций.

Graph – инструменты для построения графика.

Vector and Matrix – инструменты для работы с векторами и матрицами.

Calculus – представляет математические выражения с элементами интегрирования, дифференцирования в привычном виде.Кнопки это панели позволяют вычислять значения пределов, сумм, произведений.

Programming – инструменты для написания программ.

Greek Symbol – графический алфавит.

Symbol – Для символьных вычислений.

В развёрнутом виде перечисленные панели инструментов панели Math представлены на рис. 1.2.

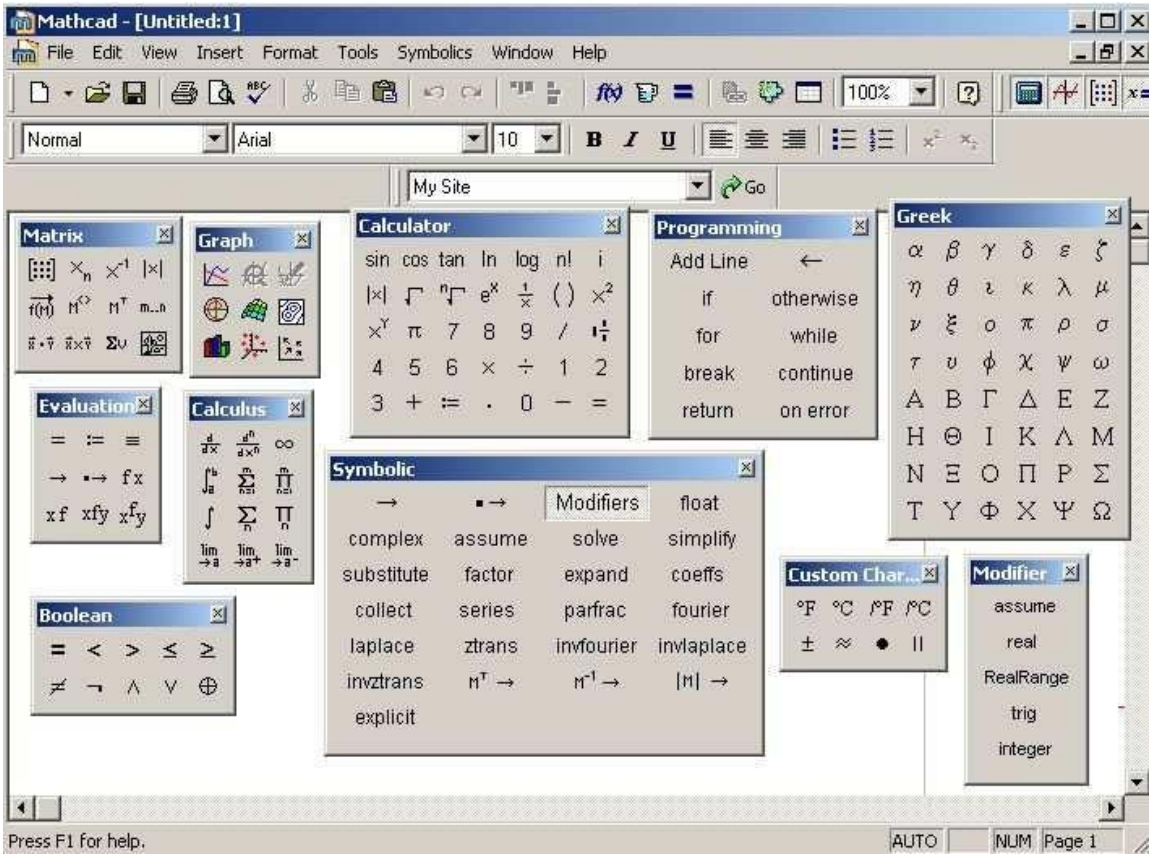


Рис. 1.2 – Рабочее окно системы MathCAD

Математические выражения в MathCAD

К основным элементам математических выражений MathCAD относятся типы данных, операторы, функции и управляющие структуры.

Операторы – элементы MathCAD, с помощью которых можно создавать математические выражения. К ним, например, относятся символы арифметических операций, знаки вычисления сумм, произведений, производной и интеграла и т. д.

Оператор определяет:

1) действие, которое должно выполняться при наличии тех или иных значений операндов;
2) сколько, где и какие операнды должны быть введены в оператор.

Операнд – число или выражение, на которое действует оператор. Например, в выражении «5! + 3» число 3 и выражение «5!» – операнды оператора «+» (плюс), а число 5 – операнд оператора факториал «!». После указания операндов операторы становятся исполняемыми по документу блоками. В Приложении Б данногопособия приведены встроенные операторы пакета MathCAD.

Запись команд в рабочем документе системы MathCAD

В системе MathCAD запись команд очень близка к стандартному языку математических расчётов, выполнимых набумаге, что значительно упрощает постановку и решение задач. В результате главные аспекты решения математических задач смещаются с их программирования на алгоритмическое и математическое описание.

Следует помнить, что MathCAD реализует вычисления в строго определённом порядке, как это делает человек, читая страницу книги, т. е. слева направо и сверху вниз. Правильный порядок выполнения блоков – основа правильного функционирования системы при обработке документа.

Сигнал ошибки в системе имеет вид всплывающей надписи, заключённой в прямоугольник.

Типы данных MathCAD

К типам данных относятся числовые константы, обычные и системные переменные, массивы (векторы и матрицы) и данные файлового типа.

Константами называют поименованные объекты, хранящие некоторые значения, которые не могут быть изменены. Переменные являются поименованными объектами, имеющими некоторое значение, которое может изменяться по ходу выполнения программы. Тип переменной определяется ее значением; переменные могут быть числовыми, строковыми, символьными и т. д. Имена констант, переменных и иных объектов называют идентификаторами. Идентификаторы в MathCAD представляют собой набор латинских или греческих букв и цифр.

В MathCAD содержится небольшая группа особых объектов,которые нельзя отнести ни к классу констант, ни к классу переменных, значения которых определены сразу после запуска программы. Их правильнее считать системными переменными, имеющими предопределенные системой начальные значения (см. Приложение В). Изменение значений системных переменных производят во вкладке Встроенные переменные диалогового окна Math Options команды **Математика → Опции**.

Обычные переменные отличаются от системных тем, что они должны быть предварительно определены пользователем, т. е. им необходимо хотя бы однажды присвоить значение. В качестве оператора присваивания используется знак «:=», тогда как знак «=» отведен для вывода значения константы или переменной.

Различают локальное и глобальное присваивание.

Локальное – присваивание, при котором переменной присваивается начальное значение с помощью оператора «:=», вызываемое нажатием клавиши «:» (двоеточие) на клавиатуре. До этого присваивания переменная не

определена и ее нельзя использовать.

Глобальное – присваивание, при котором переменной присваи- вается начальное значение с помощью знака «□», вызываемое нажатием клавиши «~» на клавиатуре.

Отличие локального присваивания от глобального состоит в следующем. Система MathCAD прочитывает весь документ дважды слева направо и сверху вниз. При первом проходе выполняются все действия, предписанные глобальным оператором присваивания (оператор «≡»), а при втором – производятся действия, предписанные локальным оператором присваивания (оператор «:=»),и отображаются все необходимые результаты вычислений (оператор «=»).

Существуют также жирный знак равенства «=» (комбинация клавиш клавиатуры «Ctrl» + «=»), который используется, например, как оператор приближенного равенства при решении систем уравнений, и символьный знак равенства «→» (комбинация клавиш клавиатуры «Ctrl» + «.»).

Используемые типы данных в системе MathCAD

В системе MathCAD предусмотрены следующие типы данных

1. Целые.
2. Вещественные.
3. Комплексные. Следует иметь в виду, что при записи мнимой единицы следуетиспользовать специальную кнопку панели **Calculus.**
4. Строковые. Обычно это комментарии вида: «Вычисление суммы».
5. Системные. Системная константа – это предварительно определённая переменная, значение которой задаётся в начале загрузки системы. Примерами таких констант являются числа «*e*» или «π».

Простые вычисления в MathCAD

Результат арифметического выражения отображается, если после него стоит знак «=» или знак «→». В первом случае результат представляется в

численном виде, а во втором – в символьном.

Пример 1.1. Символьное вычисление в MathCAD.

$$\frac{2.45}{6.342} + \frac{3}{56} - 45 - \frac{6}{86} \rightarrow -44.629882547505372085$$

При выполнении вычислительной системой арифметического выражения используются знаки арифметических операций с приоритетами, принятыми в обычной математике. Выражение можетсодержать также другие операции:

- извлечение корня;
- возведение в степень;
- интегрирование и дифференцирование;
- знаков факториала и суммирования и т.д.

Часть этих операций можно «взять» на панели **Calculator.**

Пример 1.2. Записи выражения может быть:

$$4.6 \cdot \left(\sqrt[5]{75.7} + \sqrt{24.65}\right) + 7.4^{2.5} - 87.45 + \frac{768}{435} = 97.046$$

Количество значащих цифр, отображаемых при вычислении, можно регулировать с помощью главного меню **Format→Result**. В этом случае команда предоставит диалоговое окно, как это показано на рис. 1.3., в котором следует переустановить параметры для вывода результата.

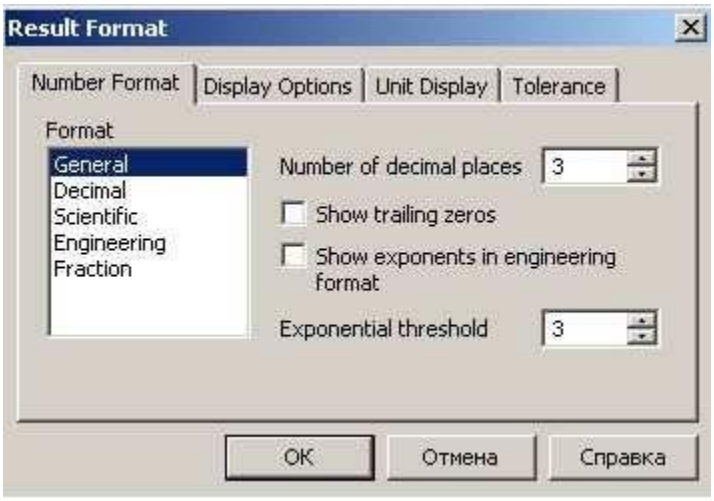


Рис. 1.3 – Рабочее окно команды главного меню **Format** (формат **Result**)

Пример 1.3. Символьное вычисление арифметического выражения:

$$\frac{25}{49} - 3^{-2} + 5\frac{1}{3} + \pi \rightarrow \frac{2528}{441} + \pi \text{ float},4 \rightarrow 8.874$$

После знака «→» отображён результат символьного вычисления. Для замены результата символьного вычисления численным значением применена команда float, расположенная на панели Symbolic. Эта команда представляет шаблон, в котором пользователю предлагается задать количество знаков (цифр) для отображения результата.

Использование встроенных функций в MathCAD

В системе MathCAD имеется множество встроенных функций. Для избегания возможных ошибок не рекомендуется имя функции вводить с клавиатуры. Наиболее часто используемы функции, например, тригонометрические функции, функций натурального логарифма, десятичного логарифма, можно задать, используя их обозначение на панели инструментов **Calculator**. К другим функциям можно обратиться с помощью команды главного меню **Insert**, либо с помощью команды (кнопки)  . В окне, которое появиться после нажатия на кнопку вызова функций (рис. 1.4), пользователь может установить категорию функции, познакомиться с примером её записи и спецификацией (описанием), а затем произвести нужный выбор. После этого система представляет пользователю шаблон, в который требуется вписать необходимые параметры.

Особенностью функции является возврат значения, т.е. функцияв ответ на обращение к ней по имени с указанием её аргументов должна возвратить своё значение.

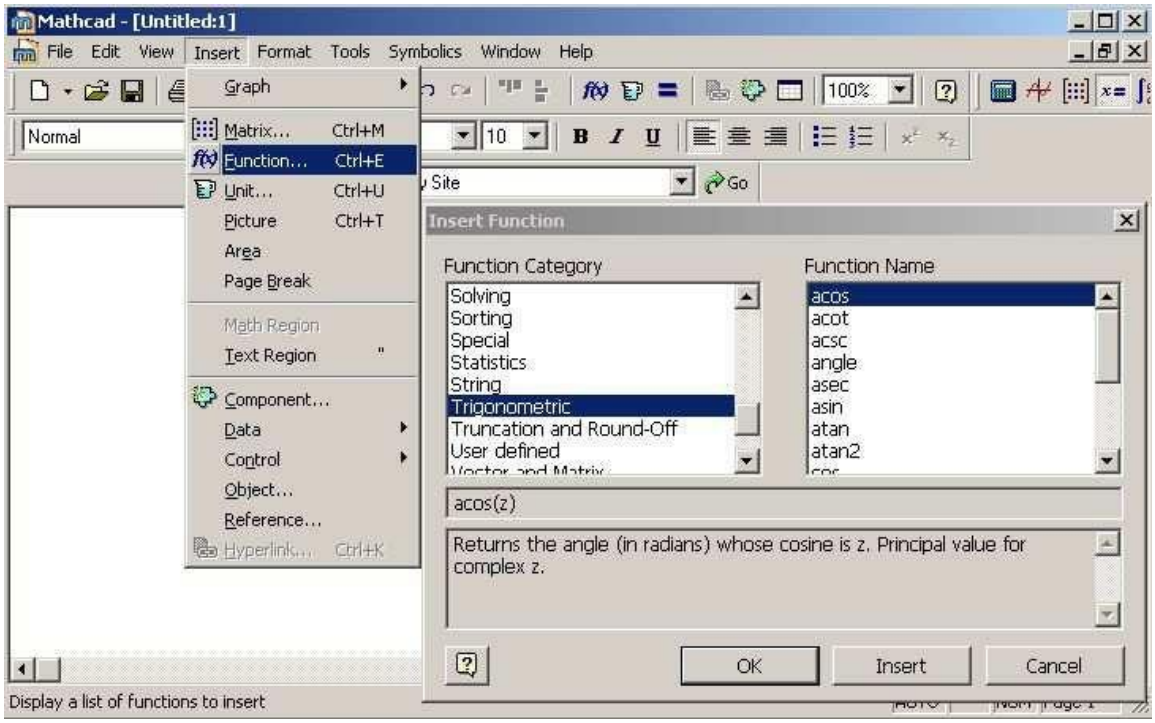


Рис. 1.4 – Рабочее окно команды вставка функции: **Insert→ Function**

Определение переменных и пользовательских функций MathCAD

В системе MathCAD, как и в любых других языкахпрограммирования, каждой ячейке памяти соответствует имя- идентификатор, которое выбирается в соответствии с установленным синтаксисом системы. Идентификаторы в MathCAD могут состоять из букв латинского или греческого алфавита и цифр, но в начальной позиции может стоять только буква. Идентификатор не должен совпадать со служебными словами, предусмотренными в системе. Следует иметь в виду, что MathCAD различает малые и заглавные буквы.

Как и в других языках программирования в MathCAD различают локальные и глобальные переменные. Присваивание локальным переменным своё значение в системе MathCAD реализуют с помощью знака «:=». Для этого достаточно ввести знак двоеточие.

Глобальная переменная вводится с применением следующегошаблона: «переменная~выражение». Вид, который принимает в документе введённое таким образом присваивание: «переменная≡выражение». Отличие

глобальных переменных от локальных переменных в том, что глобальные переменные могут использоваться в любом месте документа (в том числе, слева от их определения и над ним).

Пользовательские функции – функции, определённые и запрограммированные пользователем для выполнения периодических вычислений.

Чтобы воспользоваться собственной функцией, нужно выполнить следующее.

1. Описать функцию.
2. Вызвать описанную функцию для выполнения.

Для определения функции используются идентификаторы: имя функции и имена формальных параметров функции.

Формальный параметр – это идентификатор, конкретное значение которого определяется путём замены его на соответствующее ему значение фактического параметра при обращении к функции. При этом необходимо однозначно определять в создаваемой функции значения аргументов (формальным параметрам) в соответствии с значениями фактических параметров функции при её дальнейшем использовании.

Формат определения функции имеет вид:

ИмяФункции (список формальных параметров):=выражение.

Вызов пользовательской функции производится подобно вызову любой стандартной функции. Можно поместить результат в отдельную переменную, тогдаформат для определения функции примет вид:

Имя_переменной_результата := ИмяФункции (список формальных параметров).

Или напечатать:

ИмяФункции (список формальных параметров) =

Пример 1.4. Пусть требуется определить функцию **Dist**, котораябудет возвращать расстояние заданной точки от начала координат. Использовать эту функцию для вычисления расстояния от точки A(1.96; -3.8) и B(6; 42.5) до

начала координат.

Решение. Из курса линейной алгебры известно, что расстояние от начала координат до некоторой точки $A(x,\ y)$ определятся по формуле $y = \sqrt{x^2 + y^2}$. Здесь $(x,\ y)$ – координаты заданной точки. Эта формула и будет составлять основу функции **Dist.** При описании функции следует предусмотреть два формальных параметра –координаты точки. На это место этих параметров должны будут вписаны фактические координаты заданных точек.

В соответствии с формулой определения расстояния от точки на плоскости доя начала координат функция **Dist** может быть записана в виде:

$Dist(x, y) = \sqrt{x^2 + y^2}$. Обращение к функции для вычислений расстояний от заданных точек может быть представлено либо напрямую, либо с использованием вспомогательной переменной. Ниже представлено решение поставленной задачи в программе MathCAD.

$$Distr(x,y) := \sqrt{x^2 + y^2}$$

$$Distr(1.96, -3.8) = 4.276 \qquad P := Distr(6, 42.6)$$

$$P = 43.02$$

В системе MathCAD предоставлена также возможность определения переменных, принимающих значения из заданного промежутка, причём соседние значения удалены на равные расстояния друг от друга. При этом задаётся только начальное значение, следующее значение и конечное значение.

В качестве переменных, принимающих значение из промежутка,можно использовать только идентификаторы без индексов.

Формат определения переменной имеет следующий вид:

ИмяПеременной := начальное_значение, начальное_значение + шаг .. конечное значение.

Если конечное значение при данном значении шага не достигается точно, последним значением переменной будет наибольшее значение из заданного промежутка, не превышающее конечное значение.

Кроме того MathCAD представляет возможность не задаватьследующее значение, если шаг по величине совпадает со значением1 или -1.

В этом случае формат определения переменной можно представить в виде:

ИмяПеременной := начальное_значение, .. конечное значение.

Пример 1.5. Требуется получить таблицу значений функции $f(x) = \frac{x}{1+x^2}$, на интервале $[a, b]$ с шагом h.

Решение. Решение задачи можно свести к выполнению следующих шагов.

1. Определить функцию $f(x) = \frac{x}{1+x^2}$,
2. Задать *a, b, h.*
3. Задать переменную (например, *t*), принимающую значение изпромежутка интервале [*a, b*] с шагом *h.*
4. Получить таблицу значений функции для переменной *t.*

На рис. 1.5 представлен фрагмент документа с решением задачи.

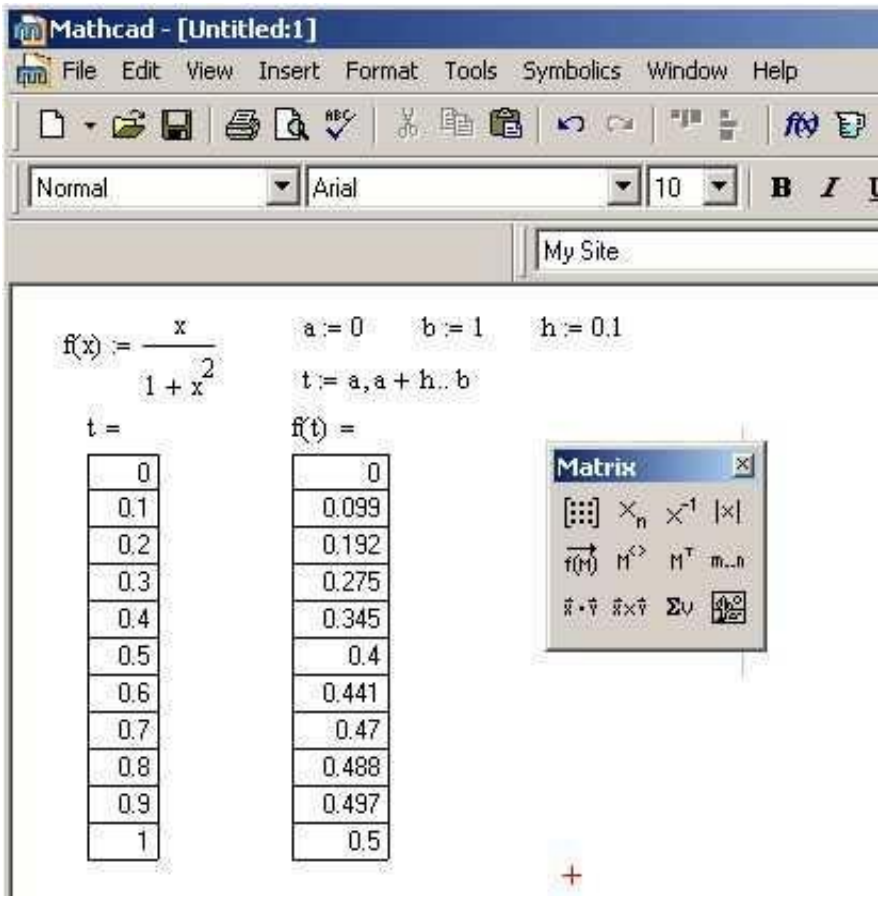


Рис. 1.5 – Получение значений функции в MathCAD

Варианты заданий к лабораторной работе №2

Задание 1

1) Вычислить, добавить комментарии.

$\sqrt{100}$ |-10|= 10

2) Определить переменные *a*, *b* и *c*, причем переменную *c* –глобально.

$a := 3.4$, $b := 6.22$, $c \equiv 0.149$.

Вычислить следующие выражения.

$$Z := \frac{2ab + \sqrt[3]{c}}{\sqrt{\left(a^2 + b^{a+c}\right) \cdot c}} \qquad N := e^{\sin c} \cos\frac{a}{b}.$$

Изменить точность отображения результатов вычисленияглобально.

Примечание. Комментарии добавляются с использованием команды Вставка → Текстовая область. Точность отображения результатов изменяется с помощью команды Формат → Результат → Формат чисел → Число знаков.

3) Вывести на экран значение системной константы □ и установить максимальный формат ее отображения локально.

4) Выполнить следующие операции с комплексными числами

Z1 = -3 + 2i, Z2 = 1 + 2i, Z2 = 3 + 4i.

|Z1| = Re(Z1) = Im(Z1) = arg(Z1) =

Z2 + Z3 = Z2

5) Выполнить следующие операции.

$i := 1 .. 10$ $\sum_i i =$ $\prod_i (i+1) =$

$$\int_0^{0.4} x^2 \cdot \lg(x+2)\,dx = \qquad \int_{0.8}^{1.2} \frac{\operatorname{ctg} 2x}{(\sin 2x)^2}\,dx =$$

$x := 2$ $\frac{d}{dx} x^5 =$ $\frac{d}{dx} \sin(x) =$

Задание 2

Рассчитать выражения в соответствии с вариантом, используя встроенные функции, вывести на экран вспомогательные слова. Ответ должен содержать m знаков после запятой, переменную *x* определить в соответствии с областью определения. Получите таблицу значений функции на интервале

[a, b] с шагом h.

Вариант 1. $y = \dfrac{1+\sin^2\left(b^3+x^3\right)}{\sqrt[3]{b^3+x^3}}$, $m = 4$, $a = -3$, $b = 3$, h=0,1

Вариант 2. $y = \dfrac{\sqrt[3]{ax+b}}{\lg^2 x}$, $m = 3$, $a = 1$, $b = 4$, $h = 0{,}2$

Вариант 3. $y = \dfrac{1+\lg^2 \dfrac{x}{a}}{b - e^{\frac{x}{a}}}$, $m = 2$, $a = 1$, $b = 3$, $h = 0{,}3$

Вариант 4. $y = \sqrt[4]{\left|x^2 - 2{,}5\right|} + \sqrt[3]{\lg x^2}$, $m = 4$, $a = 1$, $b = 9$, $h = 0{,}4$

Вариант 5. $y = \dfrac{a^x - b^x}{\lg\left|\dfrac{a}{b}\right|}\sqrt[3]{ab}$, $m = 3$, $a = 20$, $b = 25$, h = 0,5

Вариант 6. $y = \dfrac{b^3 + \sin^2 ax}{\arccos(xbx) + e^{-x/2}}$, $m = 2$, $a = -5$, $b = 5$, $h = 0{,}6$

Вариант 7. $y = \dfrac{\lg\left(x^2-1\right)}{\log_5\left(ax^2-b\right)}$, $m = 4$, $a = -5$, $b = 5$, $h = 0{,}7$

Вариант 8. $y = \dfrac{\arccos\left(x^2-b^2\right)}{\arcsin\left(x^2-a^2\right)}$, $m = 3$, $a = -5$, $b = 5$, $h = 0{,}8$

Вариант 9. $y = \arcsin\left(x^a\right) + \arccos\left(x^b\right)$, $m = 2$, $a = -7$, $b = 7$, $h = 0{,}9$

Вариант 10. $y = a^{x^2-1} - \lg\left(x^2-1\right) + \sqrt[3]{x^2-1}$, $m = 4$, $a = -4$, $b = 4$, $h = 0{,}5$

Задание 3

Точки, лежащие в плоскости α, $A(x_1, y_1)$, $B(x_2, y_2)$ и $C(x_3, y_3)$, образуют треугольник Δ ABC, а точки пространства $K(k_1, k_2, k_3)$, $N(n_1, n_2, n_3)$, $M(m_1, m_2, m_3)$ являются вершинами треугольника Δ KNM. Необходимо создать собственную функцию для вычисления параметра, представленного в варианте.

Вариaht 1. Периметр Δ ABC при $x_1 = -5$, $y_1 = 0$, $x_2 = 0$, $y_2 = 2$, $x_3 = 6$, $y_3 = 5$.

Вариaht 2. Площадь Δ ABC при $x_1 = -10$, $y_1 = -1$, $x_2 = 3$, $y_2 = 7$, $x_3 = 4$, $y_3 = 2$.

Вариaht 3. Расстояние от точки A до прямой BC при $x_1 = -4$,

$y_1 = 8$, $x_2 = -3$, $y_2 = 4$, $x_3 = 9$, $y_3 = 0$.

Вариант 4. Высоту Δ ABC, опущенную на сторону AC при $x_1 = 4$, $y_1 = 6$, $x_2 = -1$, $y_2 = 2$, $x_3 = 4$, $y_3 = 8$.

Вариант 5. Медиану Δ ABC, опущенную на сторону AB при $x_1 = 4$, $y_1 = 7$, $x_2 = -3$, $y_2 = 5$, $x_3 = 0$, $y_3 = 9$.

Вариант 6. Периметр Δ KNM при $k_1 = -5$, $k_2 = 0$, $k_3 = 3$, $n_1 = 0$, $n_2 = 2$, $n_3 = 9$, $m_1 = 6$, $m_2 = 5$, $m_3 = 0$.

Вариант 7. Площадь Δ KNM при $k_1 = -10$, $k_2 = -1$, $k_3 = 1$, $n_1 = 3$, $n_2 = 7$, $n_3 = 2$, $m_1 = 4$, $m_2 = 2$, $m_3 = 3$.

Вариант 8. Расстояние от точки K до прямой NM при $k_1 = -3$, $k_2 = 2$, $k_3 = 5$, $n_1 = -2$, $n_2 = -4$, $n_3 = 0$, $m_1 = 4$, $m_2 = -3$, $m_3 = 7$.

Вариант 9. Высоту Δ KNM, опущенную на сторону KN при $k_1 = 2$, $k_2 = 7$, $k_3 = 8$, $n_1 = -6$, $n_2 = 2$, $n_3 = 3$, $m_1 = 4$, $m_2 = 4$, $m_3 = 2$.

Вариант 10. Высоту Δ KNM, опущенную на сторону NM при $k_1 = 3$, $k_2 = 2$, $k_3 = 6$, $n_1 = -1$, $n_2 = 2$, $n_3 = -8$, $m_1 = 4$, $m_2 = 8$, $m_3 = -3$.

ЛАБОРАТОРНАЯ РАБОТА №2

на тему: «MatLab в задачах вычислительной математики»

Цель работы: ознакомиться с основными характеристиками, возможностями и особенностями математического макета MatLab, научиться выполнять математические вычисления в MatLab.

Визуализация **расчетов** в системе MatLab

MatLab является одной из самых крупных и мощных систем компьютерной математики. Она, вместе с пакетом ситуационного блочного моделирования Simulink и множеством других пакетов расширения, создана корпорацией MathWork Inc.

При запуске программы появляется окно, представленное на рис. 2.1.

В MatLab, как и в других системах, используются все буквы латинского алфавита от A до Z и арабские цифры от 0 до 9. Как и в C++, большие и малые буквы – это разные переменные и константы. Кроме букв латинского алфавита используются все специальные символы клавиатуры компьютера.

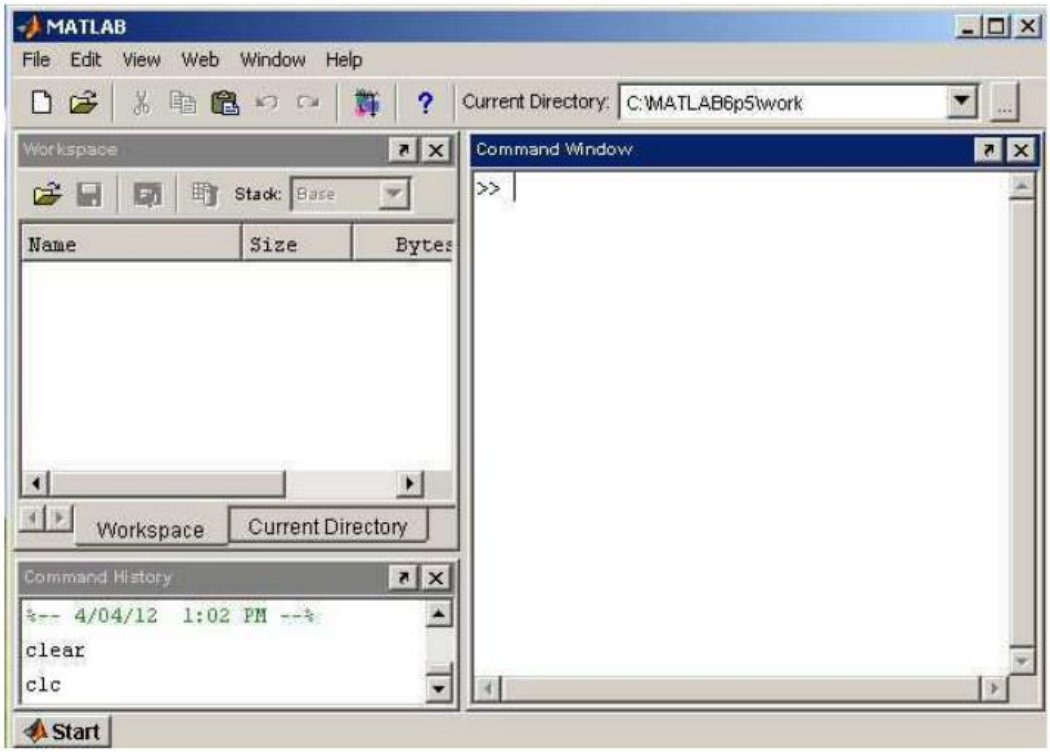


Рис. 2.1 – Окно системы MatLab после запуска

Арифметические и логические операторы

Число арифметических операторов в MatLab значительно расширено и включает в себя матричные и арифметические операции. В таблице 2.1 приводится список арифметических операторов.

Таблица 2.1 – Список арифметических операторов в MatLab

Функция	Обозначение (синтаксис)
Сложение	+ (M1+M2)
Вычитание	– (M1–M2)
Матричное умножение	* (M1*M2)
Поэлементное умножение массивов	.* (M1.*M2)
Возведение матрицы в степень	^ (M1^ x)
Поэлементное возведение массива в степень	.^ (M1.^ x)
Деление матриц слева направо	/ (M1 / M2)
Поэлементное деление массивов слева направо	./ (M1 ./ M2)
Деление матриц справа налево	\ (M1 \ M2)
Поэлементное деление массивов справа налево	.\ (M1 .\ M2)

В математических выражениях операторы имеют определенный приоритет исполнения. В MatLab приоритет логических операторов выше, чем арифметических, приоритет возведения в степень выше приоритетов умножения и деления, приоритет умножения и деления выше сложения и вычитания. Для повышения приоритета операций нужно использовать круглые скобки. Степень вложения скобок не ограничивается. Операторы отношения служат для сравнения двух величин, векторов или матриц, все операторы отношения имеют две сравниваемые величины и записываются, как показано в таблице 2.2.

Таблица 2.2 – Список операторов отношения в MatLab

Функция	Обозначение (синтаксис)
Равно	= = (x = = y)
Не равно	~ = (x ~ = y)
Меньше	< (x < y)
Больше	> (x > y)
Меньше или равно	< = (x < = y)
Больше или равно	> = (x > = y)

Данные операторы выполняют поэлементное сравнение векторов или матриц одинакового размера и логическое выражение принимает значение 1

(True), если элементы идентичны, и значение 0 (False) в противном случае Логические операторы служат для реализации поэлементных логических операций над элементами одинаковых по размеру массивов согласно таблице 2.3.

Таблица 2.3 – Список логических операций над элементами в MatLab

Функция	Обозначение (синтаксис)
Логическое И	&; *and* (*and* (*a, b*))
Логическое ИЛИ	| ; *or* (*or* (*a, b*))
Логическое НЕ	~ ; *not* (*not* (*a, b*))
Исключающее ИЛИ	*xor* (*xor* (*a, b*))
Верно, если все элементы вектора равны нулю	*any* (*any* (a))
Верно, если все элементы вектора не равны нулю	*all* (*all* (*a*))

Элементарные функции

Описание элементарных функций представлено в таблице 4.4.

Следует помнить, что в MatLab в тригонометрических функциях углы измеряются в радианах.

Таблица 2.4 – Элементарные функции в MatLab

Функция	Обозначение (синтаксис)
$\|x\|$ – модуль	*abs*(*x*)
e^x – экспонента	*exp*(*x*)
$\ln x$ – натуральный логарифм	*log*(*x*)
$\log_2 x$ – логарифм по основанию 2	*log*2(*x*)
$\lg x$ – десятичный логарифм	*log*10(*x*)
2^x – 2 в степени *x*	*pow*(*x*)
$\sqrt{x}$ – квадратный корень	*sqrt*(*x*)
arccos x – арккосинус	*acos*(*x*)
arcctg x – арккотангенс	*acot*(*x*)
arccosec x – арккосеканс	*acsc*(*x*)
arcces x – арксеканс	*asec*(*x*)
arcsin x – арксинус	*asin*(*x*)
arctg x – арктангенс	*atan*(*x*)
cos x – косинус	*cos*(*x*)
ctg x – котангенс	*cot*(*x*)
sec x – секанс	*sec*(*x*)
cosec x – косеканс	*csc*(*x*)
sin x – синус	*sin*(*x*)
tg x – тангенс	*tan*(*x*)

arcch x – арккосинус гиперболический	*acosh*(*x*)
arccth x – арккотангенс гиперболический	*acoth*(*x*)
arccosech x – арккосеканс гиперболический	*acsch*(*x*)
arcsech x – арксеканс гиперболический	*asech*(*x*)
arcsh x – арккосинус гиперболический	*asinh*(*x*)
arctgh x – арктангенс гиперболический	*atanh*(*x*)
ch x – косинус гиперболический	*cosh*(*x*)
ctgh x – котангенс гиперболический	*coth*(*x*)
cosech x – косеканс гиперболический	*csch*(*x*)
sech x – секанс гиперболический	*sech*(*x*)
sh x – синус гиперболический	*sinh*(*x*)
tgh x – тангенс гиперболический	*tanh*(*x*)

Все элементарные функции должны записываться в программах малыми буквами. Существуют также специальные математические функции, на которых мы не будем останавливаться.

Оператор присваивания и перенос строки

Для задания переменным определенных значений используется оператор присваивания, вводимый знаком равенства «=»:

Имя _ переменной = Выражение;

Типы переменных заранее не декларируются. Они определяются выражением, значение которого присваивается переменной. Имя переменной может содержать сколько угодно символов, но идентифицируется только 31 начальный символ. Имя любой переменной должно быть уникальным. Имя должно начинаться с буквы, может содержать буквы, цифры и символ подчеркивания «_».

Недопустимо включать в имена пробелы и специальные знаки.

Если математическое выражение выходит за размер экрана монитора, то целесообразно перенести его часть на следующую строку. Для этого используется символ многоточие - три и более точки. В командном режиме число возможных символов в одной строке равно 4096.

Ввод и вывод данных

Для ввода с клавиатуры служит функция **input**. Основными форматами её использования являются следующие два, из которых первый ввод числа, второй ввод текстовой информации (обычно строка):

```
X = input('Введите X: ')
str = input('Введите str: ', 's')
```

С использованием управляющих символов (“\n”), строка приглашения может быть размещена на нескольких строках. Для вывода обратного слэша \ нужно использовать комбинацию символов \\.

Пример:

```
R = input('Введите радиус окружности: \n');
```

Функция **disp** осуществляет вывод значений указанной переменной или указанного текста в командное окно. Обращение к ней имеет вид:

disp (<переменная или текст в апострофах>)

Например:

disp(x1) – в командное окно выводится значение переменной x1.

disp(‘value’) – в командное окно выводится текст value.

Чтобы вывести значения нескольких переменных в одну строку (например, при создании таблиц данных), нужно создать единый объект, который содержал бы все эти значения. Это можно сделать, объединив соответствующие переменные в вектор, пользуясь операцией создания вектора-строки:

x = [x1 x2 ...x].

Тогда вывод значений нескольких переменных в одну строку будет иметь вид:

disp ([x1 x2 ...x]).

Например:

```
» x1=-3.14; x2=-2.5; x3=5.6; x4=-9.33;
» disp([x1 x2 x3 x4])
```

–3.1400 –2.5000 5.6000 –9.3300

Если после математического выражения не ставить символ «;» – точку с запятой, значение результата будет выведено на экран автоматически.

Кроме того, для форматированного ввода и вывода могут использоваться функции fscanf и fprintf.

Системные константы представлены в таблице 2.5.

Таблица 2.5 – Системные константы в MatLab

Системные константы	Функция
pi	число $\pi = 3{,}1415$;
i или ***j***	мнимые единицы;
NaN	неопределенность в виде $\frac{0}{0}$;
Inf	бесконечность типа $\frac{a}{0}$;
ans	результат последней операции и др.

Форматы чисел

При вычислениях в MatLab используется режим двойной точности. Однако, при выводе результатов, по умолчанию выдаются числа в действительной форме, в которых после десятичной точки присутствует четыре цифры. Чтобы изменить данную форму вывода, необходимо в программе перед выводимой величиной использовать команду **format name**, где **name** – имя формата. Для числовых данных наше может быть следующим сообщением.

short – короткое представление в фиксированном формате (5 знаков);

short e – короткое представление в экспоненциальной форме (5 знаков мантиссы и 3 знака порядка);

long – длинное представление в фиксированном формате (15 знаков);

long e - длинное представление в экспоненциальной форме (15 знаков мантиссы и 3 знака порядка).

В качестве примера рассмотрим вывод вектора, содержащего 2 числа, в формате long, представленный на рис. 2.2.

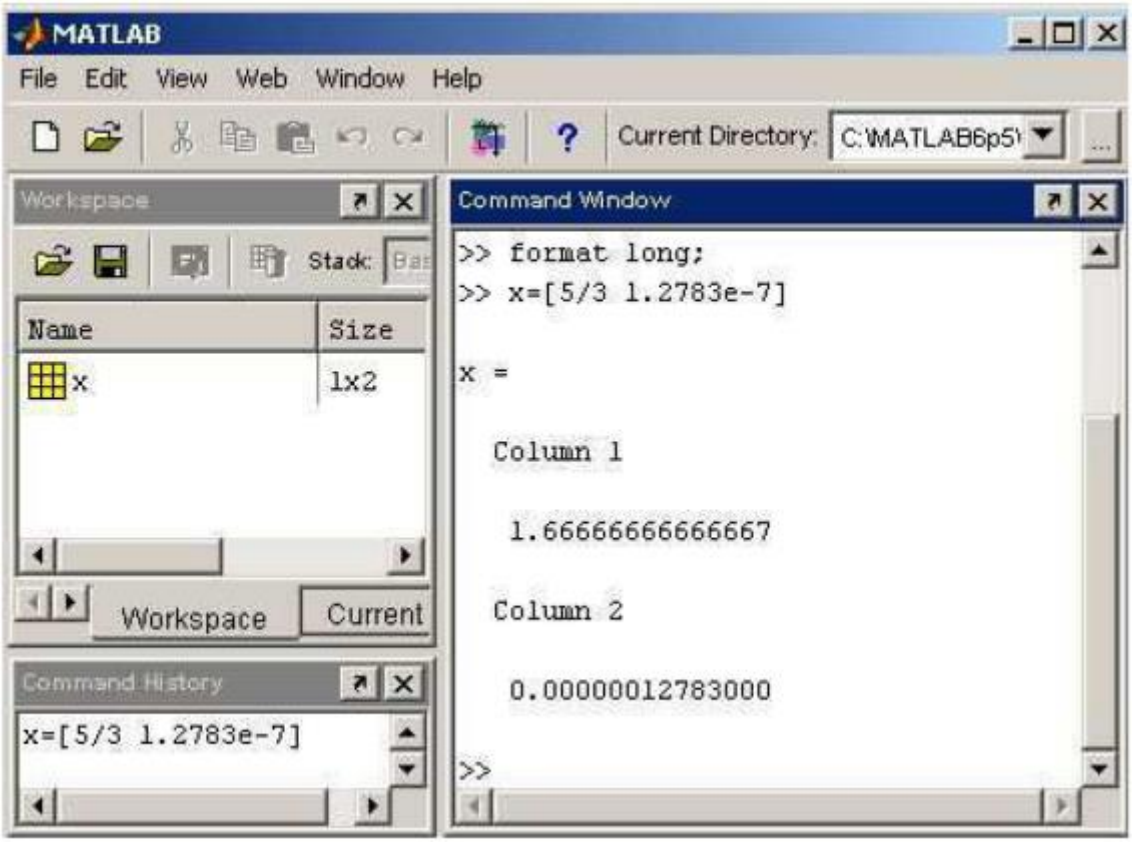


Рис. 2.2 – Представление числа в формате long в системе MatLab

В различных форматах вывод вектора *x* будет иметь различный вид.

Задание формата сказывается только на форме вывода чисел. Вычисления же происходят в режиме двойной точности, а ввод чисел осуществляется в любом удобном виде.

Вектора и матрицы в MatLab

Формирование векторов и матриц

Описанные правила вычислений распространяются и на более сложные вычисления, которые при использовании обычных языков программирования (типа Pasca1, Fortran, C++ и др.) требуют составления специальных программ. MatLab специально предназначен для проведения сложных вычислений с векторами и матрицами. При этом по умолчанию предполагается, что каждая переменная – это вектор или матрица. Например, если задано x = 1, то это значит, что x – это вектор с одним элементом, равным 1. Если надо задать вектор из трех элементов, то их значения надо перечислить в квадратных

скобках, разделяя пробелами, тогда задан вектор-строка. Если разделить элементы точкой с запятой, то получим вектор-столбец (рис. 2.3).

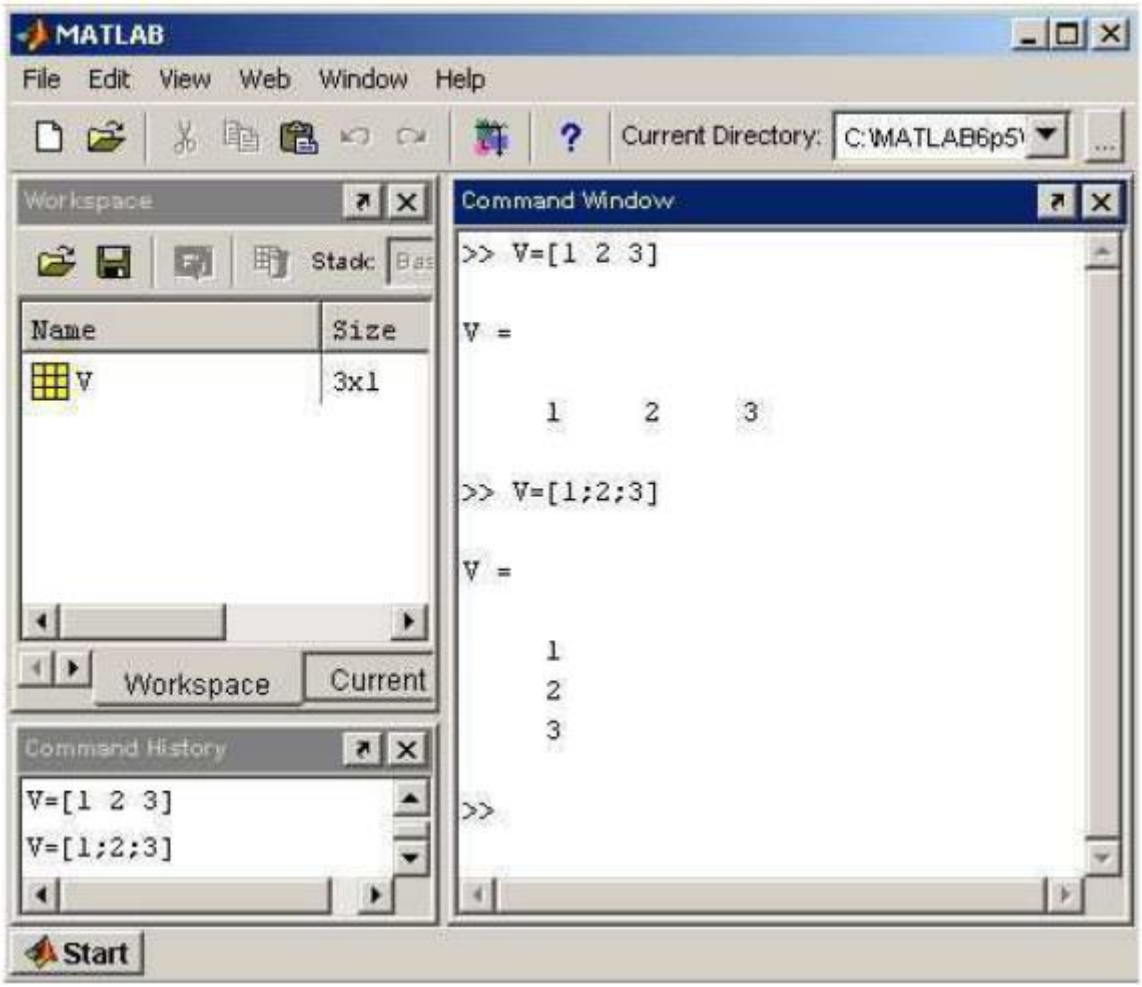


Рис. 2.3. Вектор-строка и вектор-столбец в системе MatLab

Задание матрицы требует указания несколько строк. Для разграничения строк используется символ «;» (точка с запятой). Для указания отдельного элемента вектора или матрицы используются выражения вида $V(i)$ или $T(i, j)$. Пример представлен на рис. 2.4.

Если элементу T(i, j) нужно присвоить новое значение x, то используют оператор присваивания $T(3,2) = x$; Выражение $T(i)$ с одним индексом дает доступ к элементам матрицы, развернутым в один столбец. Такая матрица образуется из исходной, если подряд выписать ее столбцы.

Наряду с операциями над отдельными элементами матриц и векторов MatLab позволяет производить арифметические операции сразу над всеми элементами. Для этого перед знаком операции ставится точка.

Имеются также ряд особых функций для задания векторов и матриц. Отметим функции **ones** и **zeros**. Эти функции служат для создания одномерных и многомерных массивов. Функция *ones* создает массив с единичными элементами. Функция *zeros* создает массив с нулевыми элементами.

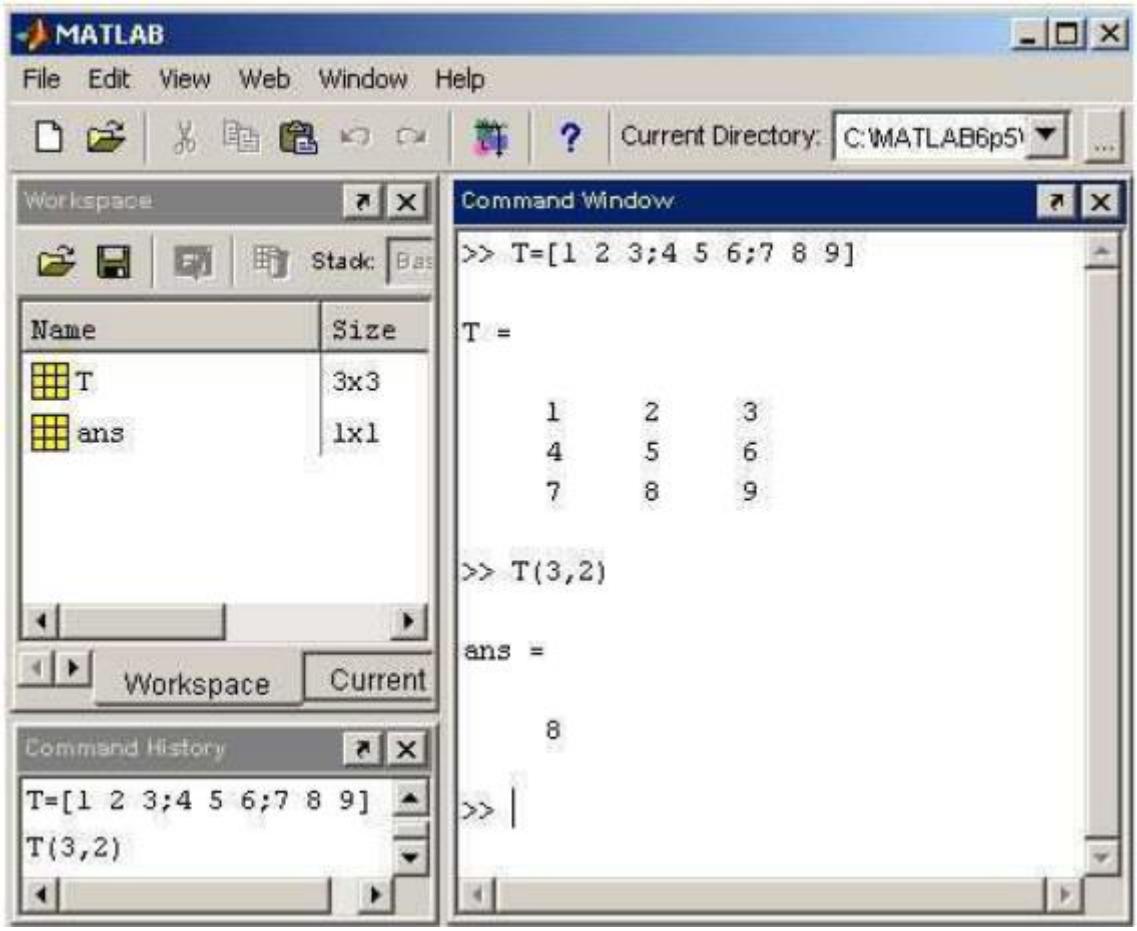


Рис. 2.4. Задание матрицы в системе MatLab

Вычисление определителя квадратной матрицы

Для вычисления определителя квадратной матрицы используется функция **det(a)**. Если матрица *a* содержит только целые числа, то результат – тоже целое число. Определитель вычисляется на основе треугольного разложения методом исключения Гаусса. Пример представлен на рис. 2.5.

Обращение матриц

Обращение матриц – одна из наиболее распространенных операций задач различных технических наук. *Обратной* называют матрицу, получаемую в результате деления единичной матрицы E на исходную матрицу x, т. е. $x^{-1} = E/x$. Эту процедуру выполняет функция $inv(x)$, которая вычисляет элементы обратной матрицы для исходной квадратной матрицы x. Если матрица x плохо масштабирована или близка к вырожденной, выдаётся предупреждающее сообщение.

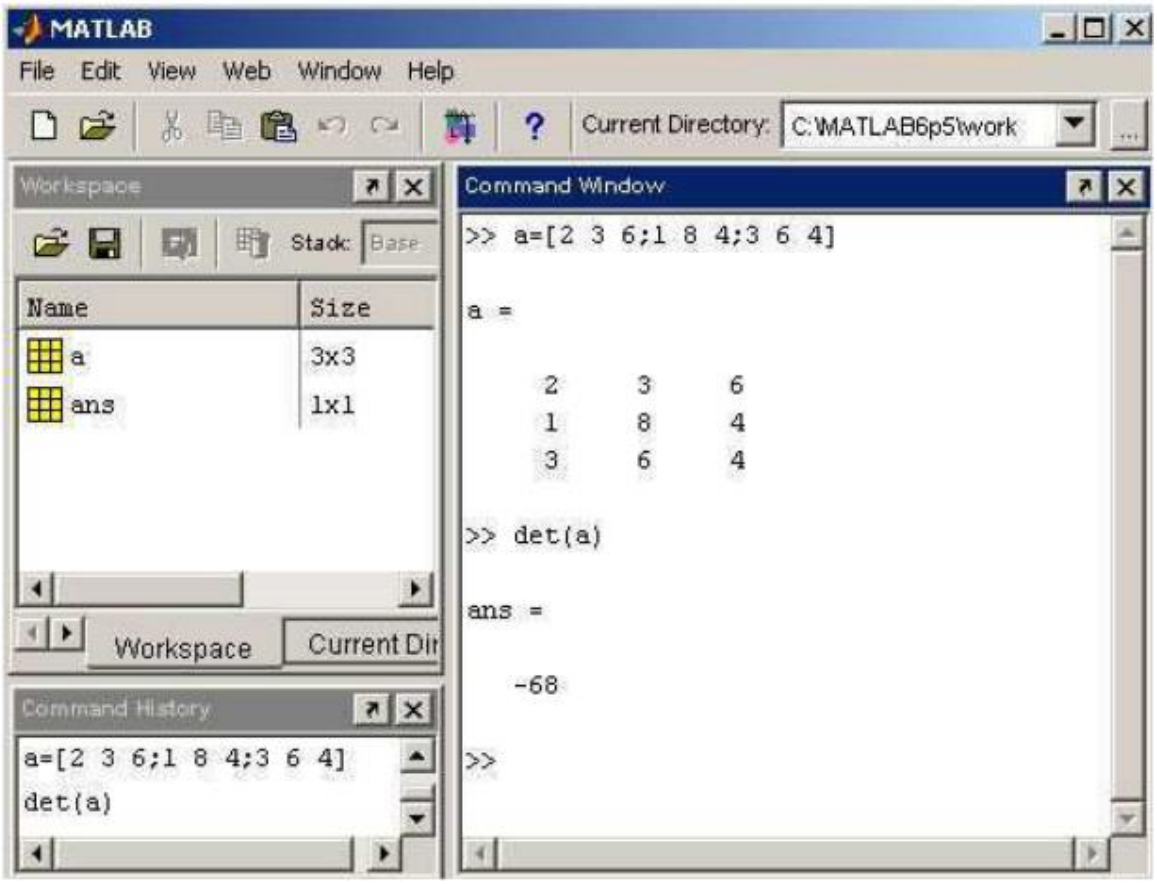


Рис. 2.5. Расчёт определителя матрицы в системе MatLab

Чаще обращение применяют для решения систем линейных алгебраических уравнений вида $ax = b$. Один из путей решения этой системы – x = $inv(x)$ * b, хотя лучше использовать метод исключения Гаусса без формирования обратной матрицы, например $x = a/b$ или $x = b/a$.

Табулирование функций

Данная задача широко используется в различных областях науки и техники. Обычно функции, описывающие какой-либо процесс, весьма громоздки и создание таблиц их значений требует большого объема вычислений.

Рассмотрим два случая табулирования функции:

1) с постоянным шагом изменения аргументов,

2) с произвольным набором значений аргумента.

Алгоритм реализуется путем организации какого-либо цикла.

Пример 4.1. Вычислить

$$y_i = R\sqrt[3]{\left|\ln\left(1-\lambda^{x_i}\right)^2 - x_i^3\right|}$$

при $R = 4.28 * 10^{-2}$; $\lambda = 2.87$;

x_i изменяется с шагом $\Delta x = 2$; $x_n = 2$; $x_k = 10$.

Введем обозначение λ -› $la = 2.87$.

Листинг программы представлен на рис. 2.6. Для вывода значения y в конце строки символ « ; » *не ставится*!

В окне команд появляются после нажатия кнопки *выполнить* значения функции y, которые затем можно скопировать в какой-либо файл.

```
>> R=0.0428;
>> la=2.87;
>> x= 2.0:2.0:10.0;
>> y=R*(abs(log((1-la.^x).^2)-x.^3)).^(1/3);
>> [x;y]

ans =

    2.0000    4.0000    6.0000    8.0000   10.0000
    0.0682    0.1634    0.2517    0.3386    0.4250

>>
```

Рис. 2.6 – Расчёт функции в MatLab

Численное решение обыкновенных дифференциальных уравнений

Многие задачи физики, химии, экологии, механики и других разделов науки и техники при их математическом моделировании сводятся к дифференциальным уравнениям. Поэтому решение дифференциальных уравнений является одной из важнейших математических задач. В вычислительной математике изучаются численные методы решения дифференциальных уравнений, которые особенно эффективны в сочетании с использованием персональных компьютеров.

Среди множества численных методов решения дифференциальных уравнений наиболее простые – это явные одношаговые методы. К ним относятся различные модификации метода Рунге-Кутты.

Постановка задачи: требуется найти функцию $y = y(x)$, удовлетворяющую уравнению:

$$\frac{dy}{dx} = F(x, y) \qquad (4.1)$$

и принимающую при $x = x_0$ заданное значение y_0:

$$y(x_0) = y_0 \qquad (4.2)$$

При этом решение необходимо получить в интервале $x_0 \le x \le x_k$. Из теории дифференциальных уравнений известно, что решение $y(x)$ задачи Коши (4.1), (4.2) существует, единственно и является гладкой функцией, если правая часть $F(x, y)$ удовлетворяет некоторым условиям гладкости. Численное решение задачи Коши методом Рунге-Кутты 4-го порядка заключается в следующем. На заданном интервале $[x_0, x_k]$ выбираются узловые точки. Значение решения в нулевой точке известно $y(x_0) = y_0$. В следующей точке $y(x_1)$ определяется по формуле:

$$y_1 = y_0 + (k_0 + 2k_1 + 2k_2 + k_3)/6, \qquad (4.3)$$

здесь

$$k_0 = h \cdot F(x_0, y_0);\ k_1 = h \cdot F(x_0 + h/2,\ y_0 + k_0/2);$$
$$k_2 = h \cdot F(x_0 + h/2,\ y_0 + k_1/2);\ k_3 = h \cdot F(x_0 + h,\ y_0 + k_2); \qquad (4.4)$$

h – шаг сетки.

Т. е. данный вариант метода Рунге-Кутты требует на каждом шаге четырехкратного вычисления правой части уравнения (4.2). Этот алгоритм реализован в программе **ode45**. Кроме этой программы MatLab располагает обширным набором аналогичных программ, позволяющих успешно решать обыкновенные дифференциальные уравнения.

Пример 4.2. Решить задачу Коши

$$\frac{dy}{dx} = 2\left(x^2 + y^2\right)$$ на [0, 1] при $y(0) = 1$ (4.5)

Точное решение имеет вид:

$y(x) = 1.5e^{2x} – x^2 – x – 0.5$

Выполним решение данной задачи с помощью программы **ode45**. Вначале в М-файл записываем правую часть уравнения (4.5), сам М-файл оформляется как файл – функция, даем ему имя P. Создаём с помощью главного меню программы MatLab: File-›New-›M-file. (В Octave: Файл-› Создать-›Создать функцию) В открывшемся окне записываем:

```
function dydx = F(x, y)
dydx = zeros(1,1);
dydx(1) = 2*(x^2+y(1));
endfunction
```

Для численного решения задачи Коши в окне команд (лучше в М-файле) набираются следующие операторы.

Протокол программы

```
[X Y] = ode45 (@F, [0 1], [1]);
% Дескриптор @ обеспечивает связь с файлом – функцией правой части
% [0 1] - интервал на котором необходимо получить решение
% [1] - начальное значение решения
plot (X,Y);
% Построение графика численного решения задачи Коши
hold on; gtext ('y(x)')
```

% Команда позволяет с помощью мышки нанести на график надпись y(x)

[X Y]

% Последняя команда выводит таблицу численного решения задачи.

Результаты решения. График решения задачи Коши показан на рис. 2.7.

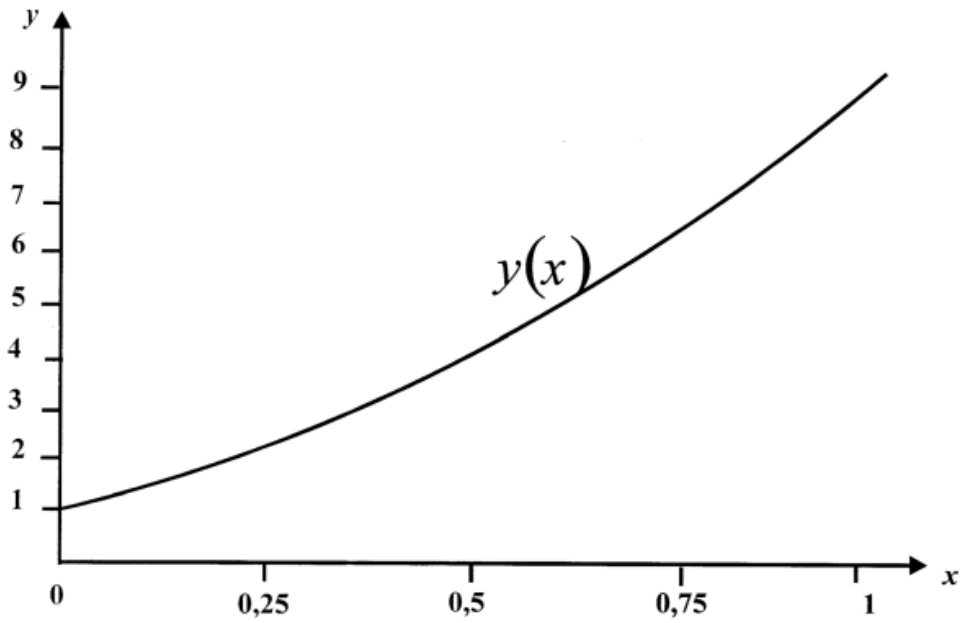


Рис. 2.7 – График решения задачи Коши

Численное решение представлено в таблице 4.2, где приведены только отдельные узловые точки. В программе **ode45** по умолчанию интервал разбивается на 40 точек с шагом $h = 1/40 = 0.025$.

Таблица 2.6 – Численное решение задачи Коши в MatLab

x_i	Метод Рунге-Кутта	Точное решение
0.0	1.0	1.0
0.1	1.2221	1.2221
0.2	1.4977	1.4977
0.3	1.8432	1.8432
0.4	2.2783	2.2783
0.5	2.8274	2.8274
0.6	3.5202	3.5202
0.7	4.3928	4.3928
0.8	5.4895	5.4895
0.9	6.8645	6.8645
1.0	8.5836	8.5836

Как следует из таблицы 2.6, численное решение программой **ode45** является точным.

Приближённое вычисление интегралов

К вычислениям определенных интегралов сводятся многие практические задачи физики, химии, экологии, механики и других естественных наук. На практике формулой Ньютона-Лейбница не всегда удается воспользоваться. В этом случае используются методы численного интегрирования. Они основаны на следующих соображениях: с геометрической точки зрения определенный интеграл

$$\int_a^b f(x)dx$$

представляет собой площадь криволинейной трапеции. Идея методов численного интегрирования сводится к разбиению интервала [a; b] на множество меньших интервалов и нахождению искомой площади как совокупности элементарных площадей, полученных на каждом частичном промежутке разбиения. В зависимости от использованной аппроксимации получаются различные формулы численного интегрирования, имеющие различную точность. Рассмотрим методы трапеций и Симпсона (парабол).

Метод трапеций.

Здесь используется линейная аппроксимация, т. е. график функции $y = f(x)$ представляется в виде ломаной, соединяющей точки y_i. Формула трапеции при постоянном шаге $h = (b - a) / n$, где n – число участков, имеет вид

$$\int_a^b f(x)dx = h\left(\frac{y_0 + y_n}{2} + \sum_{i=1}^{n-1} y_i\right) \qquad (4.6)$$

В MatLab данную формулу реализует программа **trapz(x,y).**

Метод Симпсона

Если подынтегральную функцию заменить параболой, то формула Симпсона с постоянным шагом интегрирования примет вид:

$$\int_a^b f(x)dx = \frac{h}{3}\left[y_0 + 4(y_1 + y_3 + ... + y_{n-1}) + 2(y_2 + y_4 + ... + y_{n-2}) + y_n\right] \quad (4.5)$$

В MatLab формула Симпсона реализуется программой **quad**. Подынтегральная функция может задаваться с помощью дескриптора @, тогда она программируется в файле – функции, или с помощью апострофов, тогда она записывается в самой программе **quad**. Точность вычисления интегралов по умолчанию принята равной 10^{-6}.

Пример 4.3. Вычислить и вывести на печать по методам трапеций и Симпсона значения интеграла:

$$\int_0^1 \frac{dx}{1+x^2}$$

Протокол программы метода трапеций:

```
x=0:0.0001:1.0;
y=1./(1+x.^2);
z=trapz(x, y)
```

Результат вычислений

```
z =
0.7854
```

Протокол программы метода Симпсона:

```
quad ('(1./(1+x.^2))', 0, 1)
```

Или так:

```
quad (@Q, 0, 1)
```

если подынтегральная функция записана в файле – функции Q.m

Результат вычислений:

```
ans  =
0.7854
```

Точное значение интеграла равно 0.785398163.

Как видно из примера 4.3 полученные результаты являются практически точными, а сами протоколы весьма просты.

Численное решение нелинейных уравнений

Задача нахождения корней нелинейных уравнений встречается в различных областях научно-технических исследований. Проблема формулируется следующим образом. Пусть задана непрерывная функция $f(x)$ и требуется найти корень уравнения $f(x) = 0$.

Будем предполагать, что имеется интервал изменения x $[a; b]$, на котором необходимо исследовать функцию $f(x)$ и найти значение x_0, при котором $f(x_0)$ равно или весьма мало отличается от нуля.

Данная задача в системе MatLab может быть решена следующим образом. Вначале необходимо построить график функции $f(x)$ на заданном интервале и убедиться в существовании корня или нескольких корней. Затем применить программы поиска корней. Если существует один корень и график $f(x)$ пересекает ось ox, то можно применить программу **fzero**. Если $f(x)$ имеет больше одного корня и может касаться и пересекать ось ox, то следует применить более мощную программу **fsolve** из пакета **Optimization Toolbox** (может быть не установлена по умолчанию), которая решает задачу методом наименьших квадратов. Программа **fsolve** использует известные численные методы: деление отрезка пополам, секущей и обратной квадратичной интерполяции.

Пример 7.4. Найти корень нелинейного равнения:

$10^x + 2x – 100 = 0$ на интервале [1.0; 2.0].

Протокол программы

```
% Строим график заданной функции
x = 1.0:0.001:2.0;
y = 10.0.^x+2.0*x–100.0;
plot (x, y); grid on
```

Появляется окно с графиком функции $10^x + 2x – 100$ (рис. 2.8), из которого следует, что корень функции на заданном интервале существует.

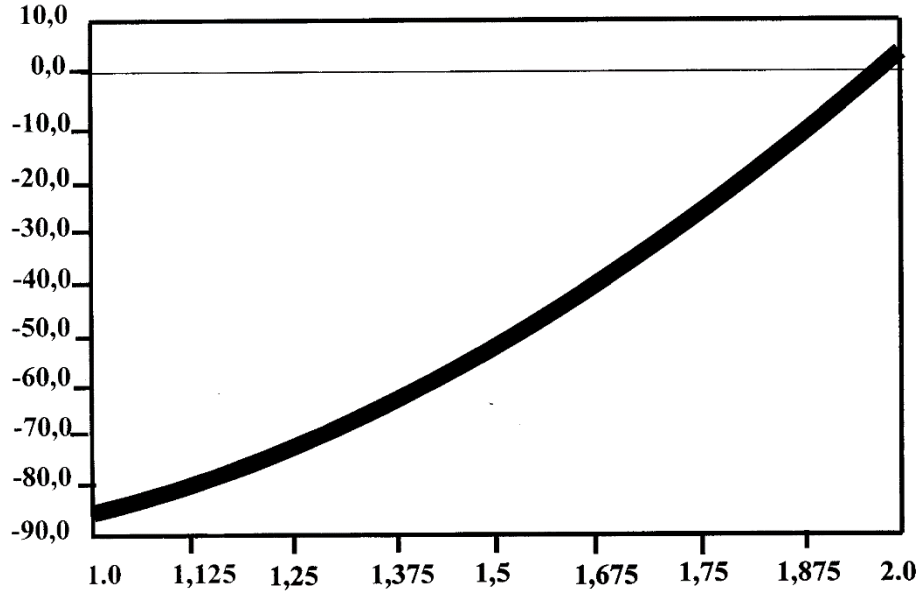


Рис. 2.8 – График функции

Для точного определения корня применяем **fzero** и **fsolve**.

X1=fzero(‘(10.0.^x + 2.0*x – 100.0)’, [1.0 2.0])

Результат решения

X1 =

1.9824

X1=fsolve(‘(10.0.^x + 2.0*x – 100.0)’, [1.0 2.0])

Результат решения

X2 =

1.9824

В Octave функция fzero в таком виде работать не будет. Уравнение необходимо записать в отдельную функцию, например F4.m. И вызывать fzero так:

X1=fzero(@F4, [1.0 2.0])

Поиск минимума функций нескольких переменных

Данная задача значительно сложнее первой. Рассмотрим ее решение на примере функции двух переменных. Алгоритм может быть распространен на функции большего числа переменных. Для минимизации функций нескольких переменных MatLab использует симплекс – метод Нелдера-Мида. Данный

метод является одним из лучших методов поиска минимума функций многих переменных, где не вычисляются производные или градиент функции. Он сводится к построению симплекса в n-мерном пространстве, заданного $n+1$ – вершиной. В двухмерном пространстве симплекс является треугольником, а в трехмерном - пирамидой. На каждом шаге итераций выбирается новая точка решения внутри или вблизи симплекса. Она сравнивается с одной из вершин симплекса. Ближайшая к этой точке вершина симплекса заменяется этой точкой. Таким образом, симплекс перестраивается и позволяет найти новое, более точное положение точки решения. Алгоритм поиска повторяется, пока размеры симплекса по всем переменным не станет меньше заданной погрешности решения. Программу, реализующую симплекс-методы Нелдера-Мида, удобно использовать в следующей записи

[x, minf] = fminsearch (…),

где x - вектор координат локального минимума; *minf* – значение целевой функции в точке минимума.

Саму целевую функцию удобно представить с помощью дескриптора @ в М-файле.

Пример 4.5. Найти и вывести на печать координаты и значение минимума функции двух переменных:

$f(x, y) = (x^2 + y^2 – 3)^2 + (x^2 + y^2 – 2x – 3)^2 + 1,$

если начальная точка поиска имеет координаты M_0 (1; 1).

Решение. Анализ функции показывает, что *minf* = 1 x = 0,

$y = \sqrt{3} = 1.73205.$

Строим трехмерный график этой функции, чтобы убедиться в наличии минимума. Возьмем интервалы $x \in [-1; 1]$; $y \in [1; 3]$.

Протокол программы

```
[X, Y] = meshgrid([-1 : 1], [1 : 3] ) ;
Z = (X.^2 + Y.^2  – 3).^2 + (X.^2 + Y.^2 – 2*X – 3).^2 + 1;
surf(X, Y, Z)
```

После построения трехмерного графика выполняем поиск минимума. В

М-файле программируем целевую функцию (не забываем, что М-файл должен называться, как сама функция – Fxy.m):

```
function f = Fxy(x)
f=(x(1)^2+x(2)^2–3)^2–3)^2+(x(1)^2+x(2)^2–2*x(1)–3)^2 +1;
```

Здесь вместо X и Y функции передаётся массив *x* из двух элементов, где *x*(1) соответствует X, а *x*(2) соответствует Y. Решаем поставленную задачу в окне команд:

```
Результаты поиска
[xmin, minf] = fminsearch (@Fxy, [1; 1])
xmin =
        –0.0000        1.7320
minf =
1.0000
```

Таким образом, результаты решения задачи точные.

Варианты заданий к лабораторной работе №2

Задание 1

Задать произвольную матрицу (6×6). Найти её определитель и обратную матрицу.

Задание 2

Составить программу вычисления значений функции y_i для значений аргумента x_i. Данные представлены в таблице 2.7.

Таблица 2.7 – Исходные данные задания 2

Вариант	Функция	Значения переменных				
		a	в	$x_н$	$x_к$	Δx
1	$y=\frac{1+\sin^2\left(b^3+x^3\right)}{\sqrt[3]{b^3+x^3}}$	–	2.5	1.28	3.28	0.4
2	$y=\frac{1+\lg^2\frac{x}{a}}{b-e^{\frac{x}{a}}}$	2.0	0.95	1.25	2.75	0.3
3	$y=\frac{a^x-b^x}{\lg\frac{a}{b}}\sqrt[3]{ab}$	0.4	0.8	3.2	6.2	0.6
4	$y=\frac{b^3+\sin^2 ax}{\arccos(xbx)+e^{-x/2}}$	1.2	0.48	0.7	2.2	0.3
5	$y=\frac{\lg\left(x^2-1\right)}{\log_5\left(ax^2-b\right)}$	1.1	0.09	1.2	2.2	0.2
6	$\begin{cases} y=\frac{\lg^2\left(a^2+x\right)}{(a+x)^2}, & x>5 \\ \frac{(a+bx)^{2,5}}{1{,}8+\cos^3(ax)}, & \geq 5 \end{cases}$	–2.5	3.4	3.5	6.5	0.6
7	$y=\sqrt[4]{\lvert x^2-2{,}5\rvert}+\sqrt[3]{\lg x^2}$	–	–	1.25	3.25	0.4
8	$y=1{,}2^x-x^{1{,}2},\ \lvert x\rvert\geq 1$ $\arccos x,\ \lvert x\rvert<1$	–	–	0.2	2.2	0.4
9	$y=\frac{\arccos\left(x^2-b^2\right)}{\arcsin\left(x^2-a^2\right)}$	0.05	0.06	0.2	0.95	0.15
10	$y=\frac{\lg^2(a+x)}{(a+x)^2}$	2.0	–	1.2	4.2	0.6

Задание 3

Построить график и вывести в виде таблицы решение задачи Коши на интервале [0; 1] методом Рунге-Кутты 4-го порядка. Данные взять из таблицы 2.8.

Таблица 2.8 – Исходные данные задания 3

Вариант	f(x,y)	y_0
1	$x^3\sin y+1$	0.0
2	$x^2\sin y-1$	0.1

Вариант	f(x,y)	y_0
3	$e^{+x}+3y$	2.0
4	$\sqrt{y^2+x^3}$	0.3
5	$\sqrt{y^3+x^2}$	0.4
6	$1/(1+y^2)+x^2$	0.0
7	$1/(1+y^2)+xy$	0.1
8	$\cos y+xy$	0.2
9	$x^2\cos y+0.1$	0.3
10	$x^3\cos y+0.1$	0.4

Задание 4

Вычислить и вывести на печать значения определенного интеграла методами трапеций и Симпсона. Данные представлены в таблице 2.9.

Таблица 2.9 – Исходные данные задания 4

Вариант	Подынтегральная функция f(x)	Интервал интегрирования [a; b]	Точность вычислений интеграла
1	$\ln x/x\sqrt{1+\ln x}$	[1; 3.5]	0.001
2	tg^2x+ctg^2x	[π/6; [π/3]	0.002
3	$1/x\ln x$	[1.5; 3.]	0.0001
4	$\ln^2 x/x$	[1.0; 4,0]	0.003
5	$\sqrt{e^x-1}$	[0; ln2]	0.0015
6	$xe^x\sin x$	[1.0; 4.0]	0.002
7	$x\frac{e^x-e^{-x}}{2}$	[0.0; 2.0]	0.001
8	$1/\sqrt{9+x^3}$	[2.0; 5.0]	0.001
9	$\sin(1/x)x^4$	[1.0; 2.5]	0.0005
10	$x^3 arctgx$	[0.0; $\sqrt{3}$]	0.003

Задание 5

Построить график и найти корень нелинейного уравнения. Данные представлены в таблице 2.10.

Таблица 2.10 – Исходные данные задания 5

Вариант	Уравнение f(x) = 0	Отрезок [a; b]
1	$xarctg-1=0$	[1.0; $\sqrt{3}$]
2	$e^{x-2}-\ln(x+2)=0$	[2.0; 3.0]
3	$x^3-9x^2+5x-6=0$	[8.0; 9.0]
4	$e^x-\frac{1}{x}-1=0$	[0.5; 1.0]
5	$arctg2x-\frac{1}{1+x}=0$	[0.0; 1.0]
6	$e^x-\ln x-20=0$	[3.0; 3.2]
7	$\sqrt{x}-tgx(1-x)=0$	[0.0; 1.0]
8	$\sin\sqrt{x}-\cos\sqrt{x}+2\sqrt{x}=0$	[0.0; 0.2]
9	$x^4+2x^3-x-1=0$	[0.8; 1.0]
10	$x^3-e^{4x}-5.5=0$	[2.6; 3.0]

Задание 6

Найти и вывести на печать координаты и минимальное значение функции двух переменных. Поиск начать с точки MO (xo, yo). Данные взять из таблицы 2.11.

Таблица 2.11 – Исходные данные задания 6

Вариант	Функция *f*(*x*, *y*)	Координаты начальной точки M_0 (x_0, y_0).
1	$(2x^2-y-3)^2+x^2+2x+2$	(1; 1)
2	$(xy+2)^2+y^2+2y+4$	(2; 2)
3	$(x^2y^2-y+2)^2+x^2+1$	(2; 2)
4	$(3x^2+2y^2-1)^2+(xy-3)^2$	(2; 2)
5	$(2x^2-7y^2-2)^2+(x^2+y^2-20)^2+3$	(2; 2)
6	$(x^2+y^2-2x-3)^2+(x^2+y^2-2y-3)^2$	(2; 2)
7	$(x^2-6x+y^2+8)^2+x^2y^2+1$	(2; 2)
8	$(x^2-y=2)^2+(x-y+3)^2$	(2; 2)
9	$\ln(1+x^2+y^2)^2+(x-y-1)^2$	(2; 2)
10	$(x^2+y^2-1)^2+(x^2-6x+y^2+8)^2$	(2; 2)

ЛАБОРАТОРНАЯ РАБОТА № 3

на тему «Управляющие структуры, массивы и операции с многочленами в MatLab»

Цель работы: получить практические навыки разработки и программирования вычислительного процесса линейных, разветвляющихся и циклических структур в системе Matlab; получить практические навыки работы с массивами в системе Matlab; освоение приемов обработки данных.

Операции, выражения и переменные

Пакет MatLab является интерпретирующим языком непосредственных вычислений. Это означает, что выражения, которые вы вводите, интерпретируются и вычисляются. Операторы пакета MatLab обычно имеют форму:

>> имя_переменной = выражение

или просто

>> выражение

Выражение, как правило, формируется из операторов, функций и имен переменных. После выполнения выражения генерируется матрица, которая выводится на экран и присваивается соответствующей переменной для последующего использования. Если имя переменной в левой части и знак = отсутствуют, автоматически генерируется переменная ans (answer − ответ), которой присваивается результат вычислений.

Обычно оператор завершается клавишей <Enter>. Однако при необходимости оператор может быть продолжен на следующей строке. Для этого его необходимо завершить тремя или более точками, после которых следует <Enter>. С другой стороны, в одной строке может быть несколько операторов, разделенных запятой или точкой с запятой.

Если последним символом в строке является точка с запятой, то вывод значений результата не производится, но присвоение значения выражения

переменной выполняется. Это помогает подавить вывод ненужных промежуточных результатов. Важно помнить, что MatLab различает строчные и прописные буквы в именах команд, функций и переменных.

Для получения списка всех переменных, расположенных в рабочем пространстве, используется команда who. Переменная может быть удалена из рабочего пространства командой clear <имя_переменной>. Команда clear без аргументов очищает все непостоянные переменные, расположенные в рабочем пространстве. Примером постоянной переменной является переменная eps (epsilon), значение которой по умолчанию равно 10^{-6}. Данная переменная используется для оценки точности вычислений итеративных процессов.

Вывод на дисплей или вычисления могут быть прерваны на большинстве компьютеров, не покидая MATLAB, с помощью комбинации клавиш <Ctrl>+<C> (<Ctrl>+<Break> на PC).

Программирование алгоритмов разветвляющейся и циклической структуры в MatLab

Для решения сложных задач режима прямых вычислений недостаточно. Возникает необходимость сохранять используемые последовательности вычислений и модифицировать их. Иными словами, нужно программировать решение задачи.

Программа в MatLab представляет собой запись в m-файле (файл с расширением ".m") всей той последовательности команд, которые потребовались бы для решения задачи в режиме прямых вычислений. Запуск на выполнение m-файла осуществляется введением в строке его имени без расширения и нажатием клавиши "Enter".

Ввод данных с клавиатуры в программе организуется при помощи оператора input, а вывод на дисплей – disp.

Например:

```
r = input ('Введите r');
disp (r)
```

Кроме того, управление выводом информации на экран может осуществляться посредством символа ";".

В системе MatLab **оператор ветвления** имеет следующий сокращенный вид:

if Условие

Операторы

end

а в общем случае записывается следующим образом:

if Условие

Операторы1

else

Операторы2

end

В первом варианте "Операторы" выполняются только в том случае, если условие истинно, и не выполняются если ложно. Во втором варианте если условие истинно, то выполняется группа операторов "Операторы1", в противном случае, выполняется группа операторов "Операторы2".

Для организации **циклов** в программах используются операторы **for ... end** (со счетчиком) и **while ... end** (с предварительной проверкой условия). Первый из них записывается следующим образом:

for Счетчик = Нач_знач : Шаг : Кон_знач

Тело цикла (операторы)

end

где *Счетчик* – имя переменной параметра цикла (не обязательно целочисленный тип);

Нач_знач и *Кон_знач* – начальное и конечное значение переменной цикла;

Шаг − задает величину приращения переменной Счетчик при ее изменении от *Нач_знач* до *Кон_знач*.

Например, задать случайным образом вектор X из 10 элементов и

подсчитать в нем число положительных элементов.

```
X = rand(1, 10)–0.5
n = 0;
for i=1:1:10
if X(i)>0
n=n+1;
end
end
disp (n)
```

Конструкция цикла **while ... end** имеет вид:

```
while Условие
Тело цикла (операторы)
end
```

В данном случае операторы цикла будут выполняться, пока будет истинно условие "Условие".

Задание новых функций

Центральным моментом программирования в среде MatLab является задание новых функций. Для этого используется конструкция

```
function Имя_переменной = Имя_функции (Список_аргументов)
% Комментарий, выводимый при использовании команды help
Операторы % Комментарии
Имя_переменной=Выражение
```

Все переменные, используемые внутри блока функции, являются локальными, т.е. область их действия распространяется только на блок функции.

Для того, чтобы создать свою функцию необходимо войти в любой текстовый редактор (например, в FAR – клавиши "Shift"+"F4"), ввести весь текст с описанием функции и сохранить его в файле с именем "Имя_функции.m". ***Имя функции обязательно должно совпадать с именем***

файла, в который она записана! Например, создав запись:

```
function y =func(x);
y = sqrt(2*x+1);
```

и сохранив ее в файле "func.m", мы получаем доступ к этой функции ($y = \sqrt{2 * x + 1}$) по имени, как, скажем, к стандартной функции sin(x), т.е. следующая команда даст результат:

```
>> z = func(4)
z =
     3
```

Пример. Вычисление факториала числа. В файле с именем "fact.m" создаем запись:

```
function f = fact(x)
% fact(x) - функция вычисления факториала x!
f = 1;
if x > 0
for i = 1:x
f = f*i;
end
end
```

Первая строка с комментарием (%) будет выводиться на экран в результате выполнения команды

```
>> help Имя_функции
```

то есть

```
>> help fact
fact(x) - функция вычисления факториала x!
```

Пример определения функции трех переменных $y = \frac{x_1^2}{2} + \frac{x_2^3}{3} + \frac{x_3^3}{4}$

```
function y = func3(x)
% Файл func3.m
y = x(1)^2/2+x(2)^3/3+x(3)^4/4;
```

Тогда вызов функции может быть осуществлен следующими способами:

```
>> x=[1 2 3]; func3(x)
        ans =
23.4167
```

Или

```
>> func3([1 2 3])
        ans =
                23.4167
```

Вообще, здесь используется упаковка трёх переменных в вектор, но ничего не мешает объявить функцию и таким образом:

```
function y = func3(x1, x2, x3)
```

Вызывать её можно будет так:

```
func3(1, 2, 3)
```

Изменения формата

Для изменения формата представления действительных чисел используется команда

```
>> format Тип_формата
```

В MatLab доступны следующие форматы:

short – короткое представление в фиксированном формате (4 знака после десятичной запятой)

short e – короткое представление в экспоненциальном формате (4 знака после запятой и 3 знака порядка)

l**ong** – длинное представление в фиксированном формате (14 знаков после запятой)

long e – длинное представление в экспоненциальном формате (14 знаков после запятой и 3 знака порядка)

Например:

```
>> format short; x = 4/3
x =
```

1.3333

>> format short e; x = 4/3

x =

1.3333e+000

Одномерные и многомерные массивы в системе MatLab

MatLab работает практически с одним видом объектов − с числовыми прямоугольными матрицами, элементами которых могут быть в общем случае комплексные числа. Все переменные представляют собой матрицы. В некоторых случаях матрицы 1 × 1 интерпретируются как скаляры, а матрицы с одной строкой или одним столбцом интерпретируются как векторы. В MatLab матрицы могут быть созданы разными способами:

1. Введены явно с помощью списка элементов.
2. Сгенерированы встроенными операторами или функциями.
3. Созданы в m-файлах.
4. Загружены из внешнего файла данных.

Явное задание матриц

Например, любой из приведенных далее операторов

>> A=[1 2 3; 4 5 6; 7 8 9] <Enter>

>> A=[1 2 3 <Shift>+<Enter>

4 5 6 <Shift>+<Enter>

7 8 9] <Enter>

создает матрицу 3 × 3 и присваивает ее значение переменной

A =

1 2 3

4 5 6

7 8 9.

Элементы внутри строки матрицы могут отделяться друг от друга не только пробелами, но и запятыми. При вводе чисел в экспоненциальной форме

(например, 2,34e–94) нельзя использовать пробелы. Ввод больших матриц лучше выполнять с помощью m-файлов, в которых легко находить и исправлять ошибки. В MatLab встроен ряд функций, позволяющих создавать функции специального вида, например, rand, magic, zeros и ones позволяют легко сгенерировать матрицы.

Функция **rand(n)** создает матрицу размером n × n, каждый элемент которой − случайное число с равномерным законом распределения в диапазоне [0, 1].

Функция **rand(m,n)** создает матрицу размера m × n, каждый элемент которой − случайное число с равномерным законом распределения в диапазоне [0, 1].

Функция **magic(n)** создает матрицу размером n × n, которая является магическим квадратом (суммы элементов по строкам и столбцам равны).

Функция **zeros(m,n)** создает нулевую матрицу размера m × n.

Функция **ones(m,n)** создает матрицу размера m × n, каждый элемент которой равен единице.

Матрицы могут быть сгенерированы также с помощью цикла for.

Ссылки на отдельные элементы матриц и векторов осуществляются с помощью индексов в круглых скобках обычным образом. Например, A(1,3) означает элемент матрицы, стоящий на 1-й строке и 3-м столбце матрицы A, а x(3) означает 3-й элемент вектора x. В качестве индексов векторов и матриц могут использоваться только положительные числа. Ссылаться на элементы матрицы A можно, используя единственный индекс A(k). В этом случае данная матрица рассматривается как один длинный вектор-столбец, сформированный из столбцов исходной матрицы. Например, обратиться ко второму элементу второй строки матрицы A можно, указав A(2,2) или A(5).

Подматрицы и использование двоеточия

Для записи алгоритмов сложной обработки данных в компактной форме в системе MATLAB используются векторы и подматрицы. Использование

нотации с двоеточием (которая используется и для генерации векторов и подматриц) и векторов вместо индексов является ключом к эффективной манипуляции этими объектами. Эффективное использование этих возможностей позволяет минимизировать число явных циклов, использование которых существенно замедляет работу MATLAB, и делает написанную программу простой и легко читаемой. (Правда, овладение данной технологией требует от пользователя пакета определенных усилий.) Например, выражение 1 : 5 фактически является вектором-строкой [1 2 3 4 5]. Отметим, элементы вектора могут быть не только целыми, но и действительными.

Например, команда

```
>> x=0.2:0.2:1.2
```

создает вектор

x = [0.2, 0.4, 0.6, 0.8, 1.0, 1.2]

а команда

```
>> 5:-1:1
```

создает вектор

[5 4 3 2 1]

Для создания и вывода на экран таблицы синусов необходимо выполнить следующую последовательность команд:

```
>> x = [0.0:0.1:2.0]'; % транспонирование вектора
>> y = sin(x); % вычисление вектора, содержащего значения sin
>> [x y] % вывод таблицы значений.
```

Отметим, что так как sin является скалярной функцией (то есть функцией, действующей поэлементно), поэтому результатом ее применения к вектору x будет вектор y.

Для доступа к подматрицам может быть использовано двоеточие.

Например, A(1:2,3) является вектором-столбцом, состоящим из последних двух элементов третьего столбца матрицы A:

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

6

9

Двоеточие само по себе означает всю строку или весь столбец, например:

```
>> A(:,3)
ans =
3
6
9
>> A(3,:)
ans =
7 8 9
```

В качестве индекса подматрицы может использоваться произвольный целый вектор:

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

Описанные способы индексирования могут использоваться с обеих сторон знака присваивания. Например, после выполнения команды

```
>> A(:,[2 4 5]) = B(:,1:3)
```

2, 4 и 5-й столбцы матрицы A будут заменены на первые три столбца матрицы B.

Функции построения матриц

В MATLAB имеются следующие стандартные функции для построения матриц:

eye(m,n) − создание единичной матрицы размером m × n;

zeros(m,n) − создание нулевой матрицы размером m × n;

ones(m,n) − создание матрицы размером m × n, каждый элемент которой равен единице;

diag(x) − создание матрицы, у которой на главной диагонали стоят элементы вектора x;

diag(A) − создание матрицы, у которой на главной диагонали стоят диагональные элементы матрицы A;

triu(A) − создание верхней треугольной матрицы, элементы которой, расположенные на главной диагонали и выше главной диагонали, равны соответствующим элементам матрицы A;

tril(A) − создание нижней треугольной матрицы, элементы которой, расположенные на главной диагонали и ниже главной диагонали, равны соответствующим элементам матрицы A.

Матричные операции

В MATLAB доступны следующие матричные операции:

< + > − сложение;

< - > − вычитание;

< * > − умножение;

< ^ > − возведение в степень;

< ′ > − транспонирование;

< \ > − левое деление;

< / > − правое деление.

Данные матричные операции применимы, конечно, и к скалярам − матрицам 1 × 1. Если размерность матриц не соответствует используемой операции, то система генерирует сообщение об ошибке, за исключением случаев, когда одним из операндов является скаляр, потому что в этом случае операция выполняется между скаляром и каждым элементом матрицы второго операнда.

Отметим, что операция возведения в степень применима только для квадратных матриц.

Если A является невырожденной квадратной матрицей, а b − вектор-столбец или вектор-строка соответственно, тогда вектор x = A \ b является

решением уравнения A x = b, а x = b / A является решением уравнения x A = b. Если A − квадратная матрица, то при левом делении для факторизации используется метод исключения Гаусса. Если матрица не квадратная, то для ее факторизации используется метод ортогонализации Хаусхольдера с ведущим столбцом, а приведенная матрица используется для решения переопределенной системы уравнений в смысле наименьших квадратов. Правое деление определяется в терминах левого деления по формуле b / A = (A' \ b')'.

Матричные операции сложения и вычитания действуют поэлементно, а остальные приведенные выше операции − нет, они являются матричными операциями. Следует отметить, что приведенные выше операции *, ., \, / могут стать поэлементными, если перед ними поставить точку. Например,

команды

```
>> [1,2,3,4].*[1,2,3,4]
```

или

```
>> [1,2,3,4].^2
```

дадут один и тот же результат

```
ans=
[1,4,9,16].
```

Функции MatLab

В MATLAB существует большое количество функций, созданных разработчиками системы, большинство из которых предоставлено в виде m-файлов, содержащих исходные тексты. Можно подойти к классификации данных функций различными способами, например по областям их использования (тригонометрические, спецфункции, функции линейной алгебры и т. д.). Далее мы используем подход, основанный на виде аргумента функции: скаляр, вектор, матрица.

Скалярные функции

Скалярные функции MATLAB действуют только на скаляры. Если

аргументом данных функций является матрица, то они действуют поэлементно. К таким функциям,

например, относятся

sin asin exp abs round

cos acos log (натуральный логарифм) sqrt floor

tan atan rem (остаток от деления двух чисел) sign ceil.

Векторные функции

Аргументами векторных функций являются векторы (строки или столбцы). Если в качестве аргумента функции указана матрица размером m × n (m >= 2), то данная функция действуют постолбцово, то есть результатом действия является векторстолбец, каждый элемент которой является результатом действия этой функции на соответствующий столбец. Построчное действие такой функции (если необходимо) может быть достигнуто использованием операции транспонирования.

Названия некоторых из этих функций приведены ниже:

max sum median any

min prod mean all

sort std.

Например, максимальный элемент прямоугольной матрицы находится с помощью команды max(max(A)), а не с помощью max(A), так как результат, возвращенный функцией max(A), − вектор, каждая компонента которого есть максимальный элемент соответствующего столбца.

Матричные функции

Наибольшую мощь системе MatLab дают матричные функции, наиболее употребительные из которых приведены ниже:

Eig − собственные значения и собственные векторы;

Chol − факторизация Холецкого;

Svd −сингулярная декомпозиция;

Inv − обратная матрица;

Lu − LU-факторизация;

Qr − QR-факторизация;

Hess − вычисление формы Хессенберга;

Schur − декомпозицияШура;

Rref − приведение к треугольной форме методом Гаусса;

Expm − матричная экспонента;

Sqrtm − матричный корень квадратный;

Poly −характеристический полином;

Det −определитель;

Size − размерность;

Norm − норма вектора или матрицы;

cond− число обусловленности;

rank − ранг матрицы;

Функции MATLAB могут возвращать одновременно несколько значений. Например, функция

y = eig(A)

или просто

eig(A)

генерирует вектор-столбец, содержащий собственные значения матрицы A, в то время как оператор

[U,D] = eig(A)

генерирует матрицу U, чьи столбцы являются собственными векторами A, а диагональная матрица D содержит на главной диагонали собственные значения этой матрицы.

Операции с многочленами

Многочлен, определяемый следующим выражением:

$$P(x)=a_n x^n+\ldots+a_2 x^2+a_1 x+a_0,$$

задается в виде вектора, хранящего коэффициенты от a_n до a_0. Для нахождения корней полинома в MatLab используется функция **roots(P)**, возвращающая вектор, элементы которого являются корнями многочлена,

заданного вектором P. Например, пусть требуется найти корни уравнения

$x^5 + 8x^4 + 31x^3 + 80x^2 + 94x + 20 = 0$

```
>> P=[1 8 31 80 94 20];
>> roots (P)
 ans =
        -1.0000 + 3.0000i
        -1.0000 - 3.0000i
        -3.7321
        -2.0000
        -0.2679
```

Обратная функция poly (R), наоборот, восстанавливает по корням коэффициенты полинома:

```
>> R = [1 -1];
>> P = P=poly(R)
 P =
1 0 -1
```

т.е. для корней x = ±1, мы получили полином $P = x^2 – 1$.

Для проведения полиномиальной аппроксимации используется функция polyfit(x,y,n), которая реализует вычисление коэффициентов полинома n-ой степени, аппроксимирующего зависимость y(x). Для вычисления значений аппроксимирующей полиномиальной функции в точках сетки используется функция polyval(P,x), где P – вектор коэффициентов аппроксимирующего полинома, x – точки сетки. Например (рис. 5.1),

```
>> x = [1 2 3 4 5]; y = [3 4 7 10 11];
>> P = polyfit(x,y,2)
P =
 0.0000 2.2000 0.4000
>> yp = polyval(P,x)
 yp =
 2.6000 4.8000 7.0000 9.2000 11.4000
```

>> plot (x, y, 'r-', x, yp, 'm:')

В результате мы получили уравнение аппроксимирующей полиномиальной зависимости $y_p(x) = a_2x^2 + a_1x + a_0$, где $a_2 = 0$; $a_1 = 2.2$; $a_0 = 0{,}4$.

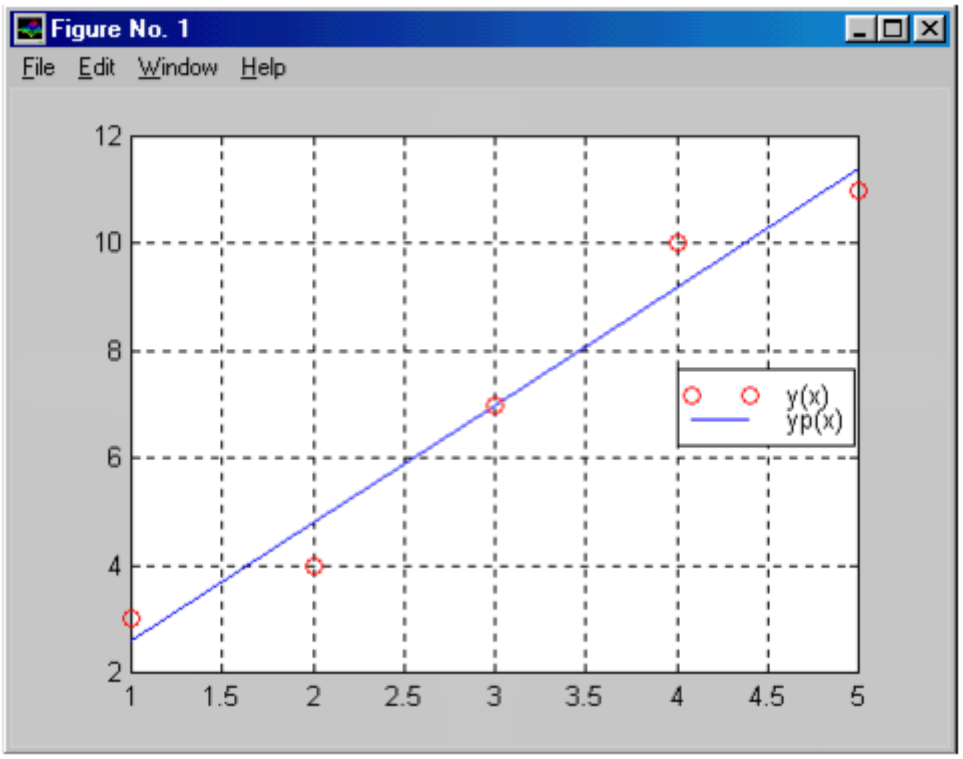


Рис. 3.1 – Графики таблично заданной функции y(x) и аппроксимирующей полиномиальной зависимости $y_p(x)$

Варианты заданий к лабораторной работе №3

Задание 1

Вычислить значение функции, заданной в таблице 3.1 (в соответствии с вариантом). Осуществить вывод значений вводимых исходных данных и результат вычисления значения функции, сопровождая вывод наименованиями переменных.

Таблица 3.1 Исходные данные задания 1

Вариант	Функция	Условие	Исходные данные	Диапазон и шаг изменения аргумента
1	2	3	4	5
1	$Y=\begin{cases} at^2\cdot\ln t \\ 1 \\ e^{at}\cdot\cos bt \end{cases}$	$1\le t\le 2$ $t<1$ $t>2$	$a=-0.5$ $b=2$	$t\in[0;3]$ $\Delta t=0.15$
2	$Y=\begin{cases} \pi x^2-\dfrac{7}{x^2} \\ ax^3+7\sqrt{x} \\ \lg\left(x+7\sqrt{x}\right) \end{cases}$	$x<1.3$ $x=1.3$ $x>1.3$	$a=1.5$	$x\in[0.8;2]$ $\Delta x=0.1$
3	$W=\begin{cases} ax^2+bx+c \\ \dfrac{a}{x}+\sqrt{x^2+1} \\ \dfrac{(a+bx)}{\sqrt{x^2+1}} \end{cases}$	$x<1.2$ $x=1.2$ $x>1.2$	$a=2.8$ $b=-0.3$ $c=4$	$x\in[1;2]$ $\Delta x=0.05$
4	$Q=\begin{cases} \pi x^2-\dfrac{7}{x^2} \\ ax^3+7\sqrt{x} \\ \ln\left(x+7\sqrt{\lvert x+a\rvert}\right) \end{cases}$	$x<1.4$ $x=1.4$ $x>1.4$	$a=1.65$	$x\in[0.7;2]$ $\Delta x=0.1$
5	$Y=\begin{cases} 1{,}5\cdot\cos^2 x \\ 1{,}8\cdot ax \\ (x-2)^2+6 \\ 3\cdot tg\,x \end{cases}$	$x<1$ $x=1$ $1<x\le 2$ $x>2$	$a=2.3$	$x\in[0.2;2.8]$ $\Delta x=0.2$

Таблица 3.1 (Продолжение)

1	2	3	4	5
6	$W=\begin{cases} e^{-ax}\cdot\cos ax \\ x\cdot\sin ax \\ x\cdot\sqrt[3]{x-a} \end{cases}$	$x<a$ $x=a$ $x>a$	$a=2.5$	$x\in[1;5]$ $\Delta x=0.5$
7	$Y=\begin{cases} \sin x\cdot\lg x \\ \cos^2 x \end{cases}$	$x>3.5$ $x\le 3.5$		$x\in[2;5]$ $\Delta x=0.25$
8	$F=\begin{cases} \lg(x+1) \\ \sin^2\sqrt{ax} \end{cases}$	$x>1$ $x\le 1$	$a=20.3$	$x\in[0.5;2]$ $\Delta x=0.2$
9	$Z=\begin{cases} \cos x+t\cdot\sin^2 x \\ \frac{1}{x}+\sqrt{x+t} \\ \frac{(\ln^3 x+x^2)}{\sqrt{x+t}} \end{cases}$	$x>0.5$ $x=0.5$ $x<0.5$	$t=2.2$	$x\in[0.2;2]$ $\Delta x=0.2$
10	$S=\begin{cases} \frac{a+b}{e^x+\cos x} \\ \frac{a+b}{x+1} \\ e^x+\sin x \end{cases}$	$x<2.8$ $2.8\le x<6$ $x\ge 6$	$a=2.6$ $b=-0.39$	$x\in[0;7]$ $\Delta x=0.5$
11	$Y=\begin{cases} a\cdot\lg x+\sqrt[3]{\lvert x\rvert} \\ 2a\cdot\cos x+3x^2 \end{cases}$	$x>1$ $x\le 1$	$a=0.9$	$x\in[0.8;2]$ $\Delta x=0.1$
12	$W=\begin{cases} \frac{a}{i}+bi^2+c \\ ai+bi^3 \end{cases}$	$i<4$ $4\le i\le 6$ $i>6$	$a=2.1$ $b=1.8$ $c=-20.5$	$i\in[0;12]$ $\Delta i=1$
13	$Z=\begin{cases} a\cdot\sin\left(\frac{i^2+1}{n}\right) \\ \cos\left(i+\frac{1}{n}\right) \end{cases}$	$\sin\left(\frac{i^2+1}{n}\right)>0$ $\sin\left(\frac{i^2+1}{n}\right)<0$	$a=0.3$ $n=10$	$i\in[1;10]$ $\Delta i=1$
14	$Q=\begin{cases} \sqrt{at^2+b\cdot\sin t+1} \\ at+b \\ \sqrt{at^2+b\cdot\cos t+1} \end{cases}$	$t<0.1$ $t=0.1$ $t>0.1$	$a=2.5$ $b=0.4$	$t\in[-1;1]$ $\Delta t=0.2$

Задание 2

Написать программу для вычисления значения суммы членов бесконечного ряда (см. табл. 3.2) с заданной точностью. На печать вывести значение суммы и число членов ряда, вошедших в сумму.

Таблица 3.2 Исходные данные задания 2

Вариант	Сумма членов ряда	Точность
1	2	3
1	$S = -\frac{(2x)^2}{2} + \frac{(2x)^4}{24} + \ldots + (-1)^n \cdot \frac{(2x)^{2n}}{2n!} + \ldots$	10^{-4}
2	$S = x - \frac{x^3}{3} + \ldots + (-1)^{n+1} \cdot \frac{x^{2n-1}}{2n-1} + \ldots$	10^{-5}
3	$S = \frac{x^3}{5} - \frac{x^5}{17} + \ldots + (-1)^{n+1} \cdot \frac{x^{2n+1}}{4n^2+1} + \ldots$	10^{-4}
4	$S = 1 + \cos\left(\frac{\pi}{4}\right) \cdot \left(\frac{x}{1!}\right) + \ldots + \cos\left(\frac{n\pi}{4}\right) \cdot \left(\frac{x^n}{n!}\right) + \ldots$	10^{-6}
5	$ch\,x = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + \frac{x^{2n}}{2n!} + \ldots$	10^{-5}
6	$\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots + \frac{(-1)^{n+1}}{2n-1} + \ldots\right)$	10^{-8}
7	$arctg\,x = \frac{1}{x} - \frac{1}{3x^3} + \frac{1}{5x^5} - \ldots + \frac{(-1)^n}{((2n+1) \cdot x^{2n+1})} + \ldots$	10^{-6}
8	$\cos\left(\frac{\pi}{6}\right) = -\frac{\left(\frac{\pi}{6}\right)^2}{2!} + \frac{\left(\frac{\pi}{6}\right)^4}{4!} - \ldots + (-1)^n \cdot \frac{\left(\frac{\pi}{6}\right)^{2n}}{(2n)!} + \ldots$	10^{-4}
9	$sh\,x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + \frac{x^{2n+1}}{(2n+1)!} + \ldots$	10^{-5}
10	$\sin\left(\frac{\pi}{3}\right) = \frac{\pi}{3} - \frac{\left(\frac{\pi}{3}\right)^3}{3!} + \frac{\left(\frac{\pi}{3}\right)^5}{5!} + \ldots + (-1)^n \cdot \frac{\left(\frac{\pi}{3}\right)^{2n+1}}{(2n+1)!} + \ldots$	10^{-4}
11	$S = 1 + \frac{x^2}{2!} - \frac{3x^4}{4!} + \ldots + (-1)^n \cdot \frac{(2n-1) \cdot x^{2n}}{(2n)!} + \ldots$	10^{-5}
12	$S = \frac{2}{3} \cdot \sin 2x - \frac{3}{8} \cdot \sin 3x + \ldots + (-1)^n \cdot n \cdot \sin\frac{nx}{n^2-1} + \ldots$	10^{-6}
13	$S = 1 + \frac{\cos x}{1!} + \frac{\cos 2x}{2!} + \ldots + \frac{\cos nx}{n!} + \ldots$	10^{-5}

Таблица 3.2 (Продолжение)

1	2	3
14	$S = \frac{x^3}{3} - \frac{x^5}{15} + ... + (-1)^{n+1} \cdot \frac{x^{2n+1}}{4n^{2-1}} + ...$	10^{-4}
15	$S = x \cdot \cos\left(\frac{\pi}{3}\right) + x^2 \cdot \cos\frac{\left(\frac{2\pi}{3}\right)}{2} + ... + x^n \cdot \cos\frac{\left(\frac{n\pi}{3}\right)}{n} + ..$	10^{-5}

Задание 3

Обработать массив в соответствии с вариантом задания (см. табл. 3.3).

Таблица 3.3 Исходные данные задания 3

Вар. зад.	Массив	Действия	Условия и ограничения
1	$x(100)$	Вычислить сумму элементов массива X, значения которых больше 0,5.	$0 < x_i < 1$
2	$A(80)$	Вычислить среднее арифметическое значение элементов массива A.	$0 < a_i < 100$
3	$X(15)$	Переписать положительные элементы массива X в массив Y и подсчитать их количество. Вывести массивы на экран.	$-1 < x_i < 1$
4	$C(10)$	Определить минимальный элемент массива C и его номер.	$x_i < 0$
5	$X(N)$	Вычислить среднее гармоническое значение элементов массива X, кратных двум.	$x_i > 0, N < 30$
6	$Y(20)$	Вычислить среднее геометрическое элементов массива	$y_i > 0$
7	$Z(30)$	Расположить в массиве R сначала положительные, а затем отрицательные элементы.	-
8	$N(50)$	Определить сумму элементов массива N кратных трем.	$Int(n_i / 3) + 3 = n_i$
9	$D(80)$	Найти максимальный элемент массива D.	-
10	$A(N)$	Найти среднее геометрическое элементов массива A.	$a_i > 0, N < 50$
11	$X(N)$	Переписать в массив Y положительные элементы массива X. Массивы вывести на экран.	$x_i > 0, N < 40$
12	$X(N)$	Переписать подряд в массив Y отрицательные элементы массива X. Массивы вывести на экран.	$N < 40$
13	$B(K)$	Определить максимальный элемент массива B и его порядковый номер.	$x_i < 0, K < 40$
14	$C(K)$	Определить среднее квадратичное элементов массива C.	$-1 < x_i, K < 20$

Справочный материал

Даны n чисел: $x_1, x_2, \ldots, x_n$.

Среднее арифметическое этих чисел:

$$A = \frac{x_1 + x_2 + ... + x_n}{n}.$$

Среднее геометрическое:

$$G = \sqrt[n]{x_1 x_2 \ldots x_n}, (x_i > 0).$$

Среднее гармоническое:

$$H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \ldots + \frac{1}{x_n}}, (x_i > 0).$$

Среднее квадратичное:

$$K = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}}.$$

Задание 4

Аппроксимировать полиномом таблично заданную функцию y = f(x). Степень полинома выбрать самостоятельно по виду графика аппроксимирующего полинома, но не более 4. (Поэкспериментировать можно и с полиномами большей степени)

Найти корни полученного полинома.

Вывести график заданной функции и аппроксимирующей полиномиальной функции.

Варианты заданий даны в таблице 3.4.

Таблица 3.4 Исходные данные задания 4

№	Критерий	Данные y=f(x)								
1	а	x	1	2	3	4	5	6	7		
		y	2,45	2,56	2,35	2,48	2,51	2,66	2,98		
2	б	x	1	2	3	4	5	6	7	8	9
		y	10,56	10,98	10,54	10,78	10,32	10,58	10,91	10,25	10,56
3	а	x	11	22	33	44	55	66	77	88	
		y	121,2	132,6	129,6	130,5	127,6	126,4	135,1	128,2	
4	б	x	1,2	1,3	1,4	1,5	1,6	1,7			
		y	18,0	17,5	16,8	17,4	18,6	18,1			
5	а	x	1	2	3	4	5	6	7		
		y	121	131	141	125	135	138	129		
6	б	x	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
		y	4.52	4.35	4.28	4.68	4.55	4.70	4.29	4.56	4.85
7	а	x	5	5.5	6	6.5	7	7.5	8	8.5	
		y	21.3	22.1	21.9	22	22.6	23.1	22.4	21.8	

Таблица 3.4 (Продолжение)

№	Критерий	Данные y=f(x)								
8	б	x	17	18	19	20	21	22	23		
		y	0.56	0.63	0.58	0.49	0.63	0.55	0.59		
9	а	x	14	15	16	17	18	19			
		y	121	124	126	128	130	124			
10	б	x	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
		y	14.23	14.56	14.28	14.98	14.11	14.78	15.11	14.32	
11	а	x	120	130	140	150	160	170	180		
		y	6.33	6.52	6.42	6.81	6.12	6.35	6.45		
12	б	x	1	2	3	4	5	6	7	8	9
		y	55.23	54.96	53.89	56.36	51.69	54.32	55.31	52.89	54.56
13	а	x	0.5	1	1.5	2	2.5	3	3.5	4	
		y	5	7.3	6.4	8.1	4.9	6.3	5.6	7.2	
14	б	x	71	75	79	83	87	91			
		y	2.36	2.45	2.8	2.12	2.65	2.74			
15	а	x	12	13	14	15	16	17	18	19	
		y	281	254	266	239	255	263	278	245	
16	б	x	65	70	75	80	85	90	95	100	
		y	16.36	15.98	16.21	16.54	16.83	17.23	15.99	16.87	
17	а	x	7	8	9	10	11	12	13	14	15
		y	7.41	7.23	7.56	7.12	7.9	8.1	6.98	7.56	7.49
18	б	x	11	11.5	12	12.5	13	13.5			
		y	33.3	12.4	0.5	44.5	12	27.6			

ЛАБОРАТОРНАЯ РАБОТА № 4

на тему «Работа с изображениями в Octave»

Примечание! Работу желательно выполнять в аналоге MatLab GNU Octave. В MatLab тоже можно, но работа с функцией blockproc() там выполняется иначе и в аудиториях на машинах невозможно будет подключить необходимые пакеты image и signal.

Создание графического окна

Для выполнения работы нужно будет создать графическое окно. Для создания пустого графического окна служит функция figure.

F = figure();

В результате выполнения этой команды будет создано данном графическое окно с именем objfigure1. По умолчанию первое окно получает имя objfigure1, второе – objfigure2 и т.д. Указатель на графическое окно записывается в переменную F. Размер и положение окна на экране монитора можно задавать с помощью параметра 'position', [x y dx dy], где

- x, y - положение верхнего левого угла окна (по горизонтали и вертикали соответственно) относительно верхнего левого угла экрана;
- dx - размер окна по горизонтали (ширина окна) в пикселях;
- dy - размер окна по вертикали (высота окна) в пикселях.

Параметры окна можно задавать одним из двух способов.

1. Непосредственно при создании графического окна задаются его параметры. В этом случае обращение к функции figure имеет вид:

F = figure('Свойство_1', 'Значение_1', 'Свойство_2', 'Значение_2', … 'Свойство_N', 'Значение_N')

здесь 'Свойство_1' – название первого параметра, Значение_1 – его значение, 'Свойство_2' – название второго параметра, Значение_2 – значение второго параметра и т.д.

Например,

f = figure('position', [10 100 300 200]);

2. После создания графического окна с помощью функции set(f, 'Свойство', 'Значение') устанавливается значение параметров, здесь f - указатель на графическое окно, 'Свойство' – имя параметра, 'Значение' – его значение.

```
f = figure();
set(f, 'position', [20,40,600,450])
```

Для изменения заголовка окна используется параметр 'figure_name', определяющий заголовок окна:

```
f = figure('position',[20,40,600,450], 'figure_name', 'FIRST WINDOW');
```

либо:

```
f = figure();
set(f, 'position', [20,40,600,450]);
set(f, 'figure_name', 'FIRST WINDOW');
```

Графическое окно можно закрыть с помощью функция close(f) (здесь f – указатель на окно). Удаляется окно с помощью функции delete(f), где f – указатель на окно.

Создание объектов управления (функция uicontrol)

В Octave используется динамический способ создания интерфейсных компонентов. Он заключается в том, что на стадии выполнения программы могут создаваться (и удаляться) те или иные графические объекты (кнопки, метки, флажки и т.д.) и их свойствам присваиваются соответствующие значения.

C = uicontrol(F, 'style', 'тип_компонента', 'Свойство_1', Значение_1, 'Свойство_2', Значение_2, ..., 'Свойство_N', Значение_N);

где C – указатель на создаваемый компонент;

F –указательна объект, внутри которого будет создаваться компонент; первый аргумент функции uicontrol не является обязательным, и если он отсутствует, то родителем (владельцем) создаваемого компонента является текущий графический объект – текущее графическое окно;

'style'– служебная строка, указывает, что дальше будет тип создаваемого

компонента;

'тип_компонента' – определяет, к какому классу принадлежит создаваемый компонент, это может быть PushButton, Radiobutton, Edittext, StaticText, Slider, Panel, Button Group, Listbox и другие компоненты;

'Свойство_N', Значение_N – определяют свойства и значения отдельных компонентов, они будут описаны ниже конкретно для каждого компонента.

У существующего интерфейсного объекта можно изменить те или иные свойства с помощью функции set:

set(C, 'Свойство_1', Значение_1,...)

где C – указатель на компонент, параметры которого будут меняться;

Получить значение параметра компонентов можно с помощью функции get следующей структуры:

get(C, 'Свойство')

где C – указатель на динамический интерфейсный компонент, значение параметра которого необходимо узнать;

'Свойство'- имя параметра, значение которого нужно узнать.

Функция возвращает значение параметра.

Типы компонентов:

pushbutton – кнопка;

text – текстовое поле для отображения текстовой информации;

edit – окно редактирования;

radiobutton – кнопка со значением on или off – переключатель;

checkbox – флажок;

listbox – список строк.

Свойства компонентов: string – заголовок;

position – координаты расположения компонента;

callback – обработка события (например, при нажатии на кнопку);

BackgroundColor – цвет фона (значением является ибо строка, либо вектор, состоящий из трех величин типа real. В случае, если это строка, т значения компонент цвета разделяются с помощью знака «|»: «R|G|B». Каждая

величина представляет значение в интервале [0,1]).

HorizontalAlignment – выравнивание текста (используется в компонентах 'text', 'edit' и 'checkbox', значения: left – выравнивание по левому краю; center – выравнивание текста по центру (значение по умолчанию); right – выравнивание по правому краю).

Пример создания текстового поля.

```
w = figure('Position', [50,50,200,200]);
t = uicontrol('Style', 'text', 'Position', [100,100,50,20], 'String', '0.1');
set(t, 'BackgroundColor', [1 1 1]);
set(t, 'HorizontalAlignment', 'left');
```

Пример создания кнопки

```
w=figure();
pbtn=uicontrol(w, 'Style', 'push', 'string', 'OK', 'Callback', 'sinus');
```

Функция, которая будет вызываться при нажатии на кнопку:

```
function y=sinus()
x=-5*%pi:0.2*%pi:5*%pi;
y=sin(x);
plot(x,y);
endfunction
```

Функция sinus() должна быть записана в отдельном файле с именем sinus.m.

Пример создания переключателя

```
hFig=figure('Position',[50,50,200,200]);
//Создание радиокнопок
hRb1 = uicontrol('Style', 'radiobutton',' String', 'sin(x)', 'value', 0, Position', [25,100,60,20], 'callback', 'Radio1');
hRb2 = uicontrol('Style', 'radiobutton', 'String', 'cos(x)', 'value', 0, 'Position', [25,140,60,20], 'callback', 'Radio2');


//обработчик нажатия на кнопки
```

```
function Radio1()
newaxes;
x = -2*%pi:0.1:2*%pi;
set(hRb2, 'value', 0);
y = sin(x);
plot(x,y);
xgrid();
endfunction

function Radio2()
newaxes;
x = -2*%pi:0.1:2*%pi;
set(hRb1, 'value', 0);
y = cos(x);
plot(x,y);
xgrid();
endfunction
```

Пример создания списка строк

```
f=figure();
h=uicontrol(f, 'style', 'listbox', 'position', [10 10 150 160]);
set(h, 'string', "item 1|item 2|item3");
set(h, 'value', [1 3]);
```

Для выполнения текущей лабораторной нам понадобится создать окно, в котором будет два элемента:

- поле, в которое будет заноситься значение качества;
- кнопка, запускающая процесс сжатия картинки.

Кроме того, в окне должно отображаться изображение до сжатия и после сжатия. Для этого необходимо создать два подокна. Подокно создаётся

функцией subplot(). У неё 3 параметра: количество подокон по горизонтали, количество подокон по вертикали, порядковый номер текущего окна. Пример:

subplot(2, 3, 5)

Это означает, что у нас будет шесть подокон: две строки и три столбца, где что-то будет рисоваться в пятом подокне (среднее в нижнем ряду).

Открытие файла с изображением в Octave

Для ознакомления с методами обработки изображений в Octave рассмотрим следующий пример.

Пример 1. *Откроем изображение lily.jpg из архива.*

```
%открытие файла изображения в матрицу A
My_lily= imread('lily.jpg');

%посмотрим картинку
imshow(My_lily);
```

Внимание! Для корректной работы Octave имя файла и путь к нему не должны содержать кириллических символов.

Для функций обработки изображений необходимо подключить пакет **image**. Для этого в командном окне Octave нужно набрать:

pkg load image

Обратите внимание, что изображение содержит три канала, а значит открытие создаст трёхмерный массив. Для изучения переменных и объектов, находящихся в памяти, можно использовать окно «Область переменных» Octave или «Workspace» Matlab. Наряду с этим, свойства переменной можно узнать с помощью команды whos VAR_NAME.

```
%узнаем свойства переменной My_lily
whos My_lily

Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
        My_lily     600x800x3                1440000  uint8
```

Переменная My_lily представляет собой трехмерный массив размерностью 600x800 (600x800x3), занимающий в памяти 1440000 байт, элементы массива представляют собой данные класса uint8 (8-ми битное

целое). В элемент класса uint8 можно записать данные от 0 до 255, т.е. всего 256 значений. Ясно, что этот тип данных наиболее приемлем для хранения в них матриц изображений

Разложим изображение на цветовые каналы и выведем каждый из них отдельно:

```
A=imread('lily.jpg');

%выбор в отдельные матрицы каналы изображения
%красный
A_R=A(:,:,1);
%зеленый
A_G=A(:,:,2);
%голубой
A_B=A(:,:,3);

figure;
%создание новой картинки
subplot(2,2,1); %значит, что создается изображение 2x2, текущий
сегмент 1
imshow(A);
title('RGB');
subplot(2,2,2); %значит, что создается изображение 2x2, текущий
сегмент 2
imshow(A_B);
title('Blue channel');
subplot(2,2,3); %значит, что создается изображение 2x2, текущий
сегмент 3
imshow(A_G);
title('Green channel');
subplot(2,2,4); %значит, что создается изображение 2x2, текущий
сегмент 4
imshow(A_R);
title('Red channel');

saveas(gcf,'MyFigure.png'); %сохраним картинку в файл png

figure; %create new figure
%combine channels into older picture - nothing been changed
comb_lily(:,:,1)=A_R; comb_lily(:,:,2)=A_G; comb_lily(:,:,3)=A_B;
comb_lily=uint8(comb_lily);
imshow(comb_lily);
title('lily combined from separated channels');
```

В рамках лабораторной работы главная программа должна выглядеть похожим образом:

```
%Определяем разрешение экрана
ScreenSize = get(0,'ScreenSize');

%handles - структура для хранения всех элементов графического интерфейса
%Создаём окно
%ScreenSize(3)/2,ScreenSize(4)/2 - левый нижний угол в центре экрана
%800, 400 - ширина: 800, высота: 400
handles.figure = figure('Position',[ScreenSize(3)/2,ScreenSize(4)/2,800,400]);

%Создаём кнопку
%[200 10 75 30] - 200, 10 координаты относительно нижнего левого угла окна, 75, 30 - размер
%PushButtonCallback - имя функции, которая будет вызываться при событии callback (нажатии)
handles.computeButton = uicontrol('style', 'push','Position',[200 10 75 30],'String','PushMe', 'callback', @PushButtonCallback);

%Создаём редактируемое поле
%Значение качества сжатия в нём по умолчанию 90
handles.edit1 = uicontrol('style', 'edit','Position',[100 10 75 30],'String','90');

%Делим наше окно на два подокна, и делаем активным первое подокно
subplot(1,2,1);
%Считываем картинку
I = imread('AJ.png')
%Рисуем её в подокне
imshow(I);

%Раскладываем картинку на цветовые составляющие
R = I(:,:,1);
G = I(:,:,2);
B = I(:,:,3);

%Конвертируем в другое цветовое пространство и округляем
handles.Y = 0.299 * double(R) + 0.587 * double(G) + 0.114 * double(B);
handles.U = -0.14713 * double(R) - 0.28886 * double(G) + 0.436 * double(B) + 128;
handles.V = 0.615 * double(R) - 0.51499 * double(G) - 0.10001 * double(B) + 128;

%Сохраняем handles (данные графического интерфейса для текущего окна (handles.figure))
guidata(handles.figure, handles);
```

Обратите внимание, что figure, computeButton, edit1, Y, U и V упакованы в структуру handles, которая в последней строчке привязывается к нашему окну. Таким образом обратившись к этой структуре внутри функции PushButtonCallback, которая начнёт выполняться по нажатию кнопки, мы сможем получить доступ к значению качества в поле edit1, цветовым составляющим картинки и самому окну, в котором нам нужно будет отобразить сжатое изображение.

Краткое описание алгоритма JPEG

Общая характеристика алгоритма. Алгоритм JPEG разработан как метод сжатия непрерывно-тоновых изображений. На практике он характеризуется сравнительно высоким коэффициентом сжатия, большим числом параметров кодирования, позволяющих найти баланс между степенью и качеством компрессии, возможностью использования "быстрой реализации" алгоритма прямого и обратного преобразования изображения (ДКП), возможностью использования только целочисленной арифметики при реализации. К недостаткам алгоритма могут быть отнесены дефекты, проявляющиеся в виде регулярных блоков одинакового размера на восстановленном изображении при сильной степени компрессии. Кроме того, возможно проявление эффекта

Гиббса – ореолов по границам резких переходов цветов.

Компрессия/декомпрессия изображения выполняется путем последовательного применения нескольких различных алгоритмов обработки данных. Рассматривается стандартный режим работы алгоритма.

Краткое описание алгоритма кодирования. Основные шаги сжатия по алгоритму JPEG состоят в следующем:

Цветное изображение преобразуется из цветового пространства RGB в цветовое пространство, состоящее из компоненты яркости (Y) и двух компонент цветности (U и V), например:

$$Y = 0,299 \times R + 0,587 \times G + 0,114 \times B;$$
$$U = -0,14713 \times R - 0,28886 \times G + 0,436 \times B + 128;$$
$$V = 0,615 \times R - 0,51499 \times G - 0,10001 \times B + 128$$

Глаз менее чувствителен к ошибкам в передаче цветности, нежели яркости. Поэтому кодировать компоненты цветности можно с меньшей точностью, чем компоненту Y. Для этого выполняется понижение частоты дискретизации (прореживание) компонент цветности в 2 или даже 4 раза по каждой размерности. Далее каждая компонента кодируется отдельно.

2) Пикселы цветовой компоненты разбиваются на блоки NxN пикселов. Если количество пикселов не кратно N, то последняя строка и/или столбец повторяются необходимое число раз.

3) К каждому блоку применяется дискретно-косинусное преобразование

$$D(i,j) = C(i)C(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} S(x,y)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

где S(x, y) – исходное значение амплитуды пиксела с координатами x и y внутри блока, D(i, j) – значение элемента (i, j) матрицы коэффициентов преобразования (0 <= i, j <= N - 1), а значения C(i) и C(j) рассчитывается по формуле

$$C(t) = \begin{cases} 1/\sqrt{N}, t = 0; \\ \sqrt{2/N}, 1 \le t \le N-1 \end{cases}.$$

4) Каждая из 64 компонент делится на специальное число –коэффициент квантования. Обычно таблица коэффициентов квантования является статически заданной и передается вместе с закодированным изображением. Один из способов формирования таблицы квантования Q является использование формулы:

Q(i, j) = 1 + (i + j) * R

где i, j – номер строки и столбца в блоке (0 <= i, j <= N – 1), R – некоторый показатель качества (R = 100 – Quality + 1). Чем больше значение R, тем сильнее квантование и, соответственно, ниже качество восстановленного изображения.

5) Квантованные коэффициенты ДКП после округления преобразуются из матричного представления в линейное таким образом, чтобы в элементы были расположены в векторе в порядке убывания значимости (с точки зрения "важности" кодируемых частот). Для этого используется зигзагообразный обход матрицы коэффициентов, проиллюстрированный на рисунке:

1	2	6	7
3	5	8	13
4	9	12	14
10	11	15	16

Обход обычно реализуется на основе рассчитываемых заранее индексов табличной перестановки. Например, для блока 4x4 такой таблицей является:

(1,1) (1,2) (2,1) (3,1) (2,2) (1,3) (4,1) (2,3) (3,2) (4,1) (4,2) (3,3) (2,4) (3,4) (4,3) (4,4)

6) К полученному вектору коэффициентов применяются дополнительные методы кодирования. При этом обычно первый коэффициент (DC)кодируется в виде кода Хаффмана для разности с DC-коэффициентом предыдущего блока, а остальные (AC) коэффициенты кодируются с использованием комбинации RLE и метода Хаффмана (или арифметического

кодирования).

При кодировании DC коэффициента сначала унарным кодом кодируется длина бинарного представления DC-коэффициента, а затем записывается собственно значение коэффициента.

В последовательности AC-коэффициентов, как правило, значительная часть нулевых значений. Для кодирования каждого ненулевого значения x кодер а) определяет число предшествующих ему нулей (Z), б) определяет длину унарного кода (R) числа x и номер числа x среди всех чисел с данной длиной двоичного представления (C), в) заменяет пару <R, Z> на код переменной длины Хаффмана (H), выбираемый из заранее заданной статической таблицы с монотонно возрастающими длинами кодов, г) приписывает H к C (C представлено в виде R битов) и выдает полученное кодовое слово в битовый поток. После кодирования последнего ненулевого коэффициента в выходной поток записывается специальный четырехбитовый код окончания блока (EOB), отсутствующий в таблице кодов Хаффмана.

7) Полученный битовый поток дополняется заголовком, содержащим параметры изображения, таблицы квантования, таблицу Хаффмана и записывается в файл или передается в канал связи.

Краткое описание алгоритма декодирования. Основные шаги декодирования по алгоритму JPEG состоят в следующем:

1) Из файла (канала связи) читается битовый поток, из которого выбирается заголовок, таблицы квантования и таблица Хаффмана, а также параметры (размеры) изображения. Рассчитывается количество блоков по вертикали и горизонтали.

2) Последовательно выполняется декодирование DC и AC коэффициентов каждого блока и формирование вектора коэффициентов ДКП.

3) Выполняется формирование матрицы коэффициентов ДКП-преобразования каждого блока путем применения такой же таблицы для обхода, как и при кодировании (только в обратном направлении).

4)Выполняется обратное масштабирование (квантование)

коэффициентов ДКП каждого блока путем поэлементного умножения значений блока на соответствующие значения матрицы коэффициентов квантования Q.

5) К каждому блоку применяется обратное ДКП-преобразование:

$$S(x,y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} C(i)C(j)D(i,j)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

где S(x, y) –исходное значение амплитуды пиксела с координатами x и y внутри блока, D(i, j) –значение элемента (i, j) матрицы коэффициентов преобразования (0 <= i, j <= N - 1), а значения C(i) и C(j) рассчитывается по формуле.

$$C(t) = \begin{cases} 1/\sqrt{N}, t = 0; \\ \sqrt{2/N}, 1 \le t \le N-1 \end{cases}$$

6) Выполняется обратное изменение (повышение) частоты дискретизации компонент цветности.

7) Выполняется переход из пространства YUV в пространство RGB

$$R = Y + 1,13983 \times (V - 128);$$
$$G = Y - 0,39465 \times (U - 128) - 0,58060 \times (V - 128);$$
$$B = Y + 2,03211 \times (U - 128);$$

Шаблон функции для кодирования декодирования:

```
function PushButtonCallback()
%Получаем данные графического интерфейса для текущего окна
%Теперь мы можем обращаться ко всем полям структуры handles
handles=guidata(gcf);
%Получаем значение из редактируемого поля в пременную Quantity
Quality = str2num(get(handles.edit1,'String')); % = 90

%Извлекаем компоненты картинки
Y = handles.Y;
U = handles.U;
V = handles.V;

%Проводим ДКП всех трёх компонент

%Рассчитываем матрицу квантования

%Квантуем коэффициенты ДКП всех трёх компонент

%Округляем результаты

%Деквантуем коэффициенты ДКП всех трёх компонент

%Проводим обратное ДКП

%Проводим переходим к цветовому пространству RGB

%Собираем картинку и выводим её в правом подокне

endfunction
```

Описание работы функций blockproc(), dct2 и idct2()

B = blockproc(A, [m, n], fun) делит изображение A на блоки размером m на n и передает их предоставленной пользователем функции fun, результат работы которой объединяется для построения возвращаемой матрицы B. Если ширина и высота изображения не делятся нацело на m и n, то изображение расширяется снизу и справа до необходимого размера. В качестве значения заполнения используется 0.

Пример:

DY = blockproc(Y, [8 8], @dct2);

Здесь матрица Y делится на блоки 8x8 элементов и каждый блок передаётся функции dct2 как параметр. Функция dct2 производит прямое ДКП над блоком и возвращает блок коэффициентов ДКП. Блоки коэффициентов объединяются в матрицу DY.

B = blockproc(A, [m, n], fun, …) ведет себя так, как описано выше, но передает дополнительные параметры функции fun.

Пример:

DY = blockproc(DY, [8 8], @mulMatrix, Q);

Здесь матрица DY делится на блоки 8x8 элементов и каждый блок передаётся функции mulMatrix как первый её параметр. Функции mulMatrix требует два параметра, поэтому второй параметр – матрица Q – указывается через запятую после ссылки на функцию.

Квантование и деквантование можно оформить в виде одной функции. Тогда внутри неё нужно будет произвести последовательно 3 операции:

1. Поэлементно разделить блок на матрицу квантования.
2. Округлить элементы получившегося блока до целого (для этого можно использовать функцию round()).
3. Умножить округлённый результат на матрицу квантования.

Задание к лабораторной работе №4

Написать программу, которая:

1) Читает файл с изображением.

2) Изображение из RGB преобразует в YUV.

3) Кодируется это изображение. При кодировании выполняются только шаги ДКП и квантования.

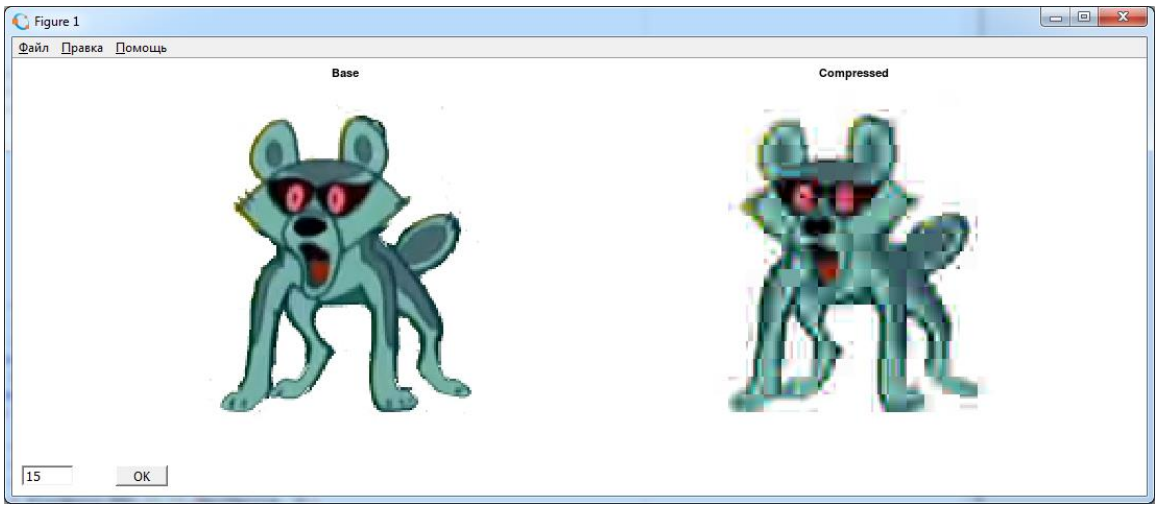
4) Декодирует изображение.

5) Преобразует изображение из YUV в RGB

6) Выводит оба изображения на экран.

7) Заменить встроенные функции ДКП и обратного ДКП на свои собственные. (При запуске уменьшите размер картинки до размера 32x32, так как будет работать намного медленнее встроенных)

Программа должна иметь такой вид.



В поле указывается качество, после нажатия кнопки запускается процедура кодирования и декодирования. Для упрощения задачи используйте картинки с размерами кратными 8.

Примеры программ:

```
figure;
I = imread('f:\jackal.png');
imshow(I);

I2 = rgb2gray(I);
I2 = blockproc(I2,[8 8],@dct2); %Прямое ДКП

mask = [1   1   1   0   0   0   0   0
        1   1   0   0   0   0   0   0
        1   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0];

I2 = blockproc(I2,[8 8],@mulMatrix, mask);  %Приравниваем младшие коэффициенты 0
I2 = blockproc(I2,[8 8],@idct2);  %Обратное ДКП

figure;
imshow(I2)
```

Примеры результатов

Данные для картинки AJ.png для самопроверки.

Цветовая компонента R:

```
ans(:,:,1) =

  186   150   243   253   154   192   245   241   242   240   251   253   251   236   168   148
  163   173   233   232   205   191   183   212   231   247   248   251   249   243   199   118
  130   220   204   239   194   157   224   247   250   240   240   252   253   242   235   203
  106   255   184   237   182   195   222   238   253   255   255   246   250   254   255   234
  106   255   148   228   196   230   251   233   227   247   254   255   252   246   246   250
  102   255   122   226   164   244   249   255   235   195   229   253   251   245   247   252
   79   255   157   240   140   106   244   253   255   228   116   126   199   242   230   245
  100   215   255   205   168    78    85   194   221   251   242   104    27   143   236   216
  185   147   237   146   211    38    49   239   181   179    81    35   142    58   190   232
  223   151   234   199   175    64    30   255   252   250    63    27   184    86   121   231
  228   195   214   202   238   139    44   236   255   255   104   126   208   156   140   191
  226   227   173   255   240   243   181   217   255   250   164   248   229   184   169   166
  227   253   182   204   255   255   255   235   255   251   228   242   216   170   199   165
  225   255   252   171   182   249   247   219   232   219   200   213   174   199   200   202
  226   243   255   253   157    85   186   237   236   233   194   189   204   212   222   231
  220   241   235   252   255   155    86   144   143   103   135   208   201   206   208   216
```

Y после преобразования из RGB:

```
 118.789    90.352   192.242   189.721   103.123   160.021   215.990   211.061   211.605   208.334   221.551   226.452   227.011
  99.277   127.896   175.899   162.948   147.882   154.849   140.865   174.008   196.986   219.392   223.652   224.908   218.850
  76.530   178.469   137.017   172.068   138.398   119.151   191.354   221.364   225.407   210.631   205.433   221.052   228.572
  64.202   209.297   113.934   172.285   133.522   163.021   190.739   206.659   225.979   229.022   229.723   216.534   220.192
  67.952   214.090    85.138   162.225   153.603   204.216   227.205   206.743   195.072   217.437   224.648   229.136   224.620
  67.457   210.454    68.365   154.942   117.460   218.478   223.837   231.239   207.016   159.778   193.715   218.547   220.149
  48.173   197.460   105.536   169.808    90.217    71.702   215.691   225.899   232.971   194.442    77.023   101.111   166.485
  65.571   168.904   188.102   144.981   167.476    73.008    39.207   136.979   183.367   221.619   208.151    84.064    19.318
 146.222   114.924   178.303   127.774   213.411    25.627    23.405   161.764   107.070   107.532    52.521    20.849   137.299
 188.576   117.049   172.955   179.845   174.595    50.834    24.688   210.129   202.222   181.455    42.118    16.291   183.881
 193.234   160.137   153.850   165.861   238.764   142.384    41.732   179.993   208.858   208.613    97.941   132.656   203.806
 189.718   193.687   122.761   212.346   193.465   219.411   166.570   160.503   201.472   192.021   127.970   201.215   164.792
 220.599   221.938   143.974   141.012   206.510   203.136   193.174   170.302   201.472   189.972   163.399   186.825   147.324
 211.361   229.056   222.112   130.364   119.662   181.139   177.754   151.219   168.459   156.143   133.473   135.388   158.413
 206.656   220.966   230.099   223.568   125.363    49.984   130.119   172.188   166.720   160.067   137.692   205.562   239.751
 198.342   216.538   209.022   226.170   233.604   129.199    44.772    89.567    94.648    67.671   140.802   240.947   233.229

olumns 14 through 16:

 183.202   110.700    91.270
 212.394   158.820    69.534
 220.963   201.915   163.914
 228.398   232.869   203.229
 217.628   220.318   226.108
 216.919   214.650   223.936
 207.872   199.343   212.975
 110.286   204.448   189.811
  33.756   151.940   197.399
  58.952    79.544   196.040
  90.545   116.884   162.657
 113.022   173.907   134.329
 137.378   237.914   152.855
 238.256   232.964   233.921
 238.621   241.058   243.686
 234.137   235.322   236.654
```

DY после ДКП (выделен первый блок):

1.2094e+03	-1.8155e+02	1.5415e+01	-1.5642e+02	-5.8741e+01	-5.3114e+01	-1.1055e+02	-9.2497e+01	1.5614e+03	6.7418e+01
5.2186e+01	-2.6540e+01	-6.7038e+00	1.5646e+01	6.2232e+00	8.1769e+01	6.8974e+01	2.6065e+01	6.7709e+01	1.0895e+02
-6.5295e+01	8.2865e+01	-3.8855e+01	-6.5220e+01	4.7376e+01	4.1636e+01	8.6419e+01	7.6265e+01	-1.6528e+02	8.2797e+01
5.8291e+01	-9.4445e+01	4.9377e+01	5.3739e+01	-4.0284e+01	5.4841e+01	-1.0174e+01	-4.8764e+01	2.6454e+01	-2.1231e+01
7.0812e-01	3.3658e+01	-4.0251e+01	1.0299e+01	1.0974e+00	-6.2398e+00	3.2871e+01	1.5153e+01	-9.0662e-01	3.3855e+01
7.0030e+00	-3.7308e+01	5.5378e+01	-5.2067e+01	-4.6978e+00	4.7628e+01	-4.8911e+01	-1.3108e+01	7.8550e+00	-2.5474e+01
1.4757e+01	-8.8444e+00	2.2380e+00	1.3287e+01	1.2586e+00	-7.6727e+00	1.7856e+01	-7.9814e+00	1.1108e+01	-1.1122e+00
7.6350e+00	-1.7822e+01	2.0361e+01	-2.3232e+01	5.7262e+00	2.2715e+01	-3.6781e+01	1.6433e+01	-6.0249e+00	-1.1955e+01
1.3092e+03	1.8267e+02	-2.0824e+01	-5.0225e+01	1.3142e+02	-9.0676e+01	2.1660e+01	2.3397e+01	1.2863e+03	-1.0188e+02
-1.1287e+02	-6.6170e+01	-6.0133e+00	-1.3655e+01	1.1984e+02	-5.5933e+01	4.6300e+01	4.0945e+01	-2.4273e+02	1.4501e+02
-1.1629e+02	1.0244e+02	-5.9055e+01	-1.2602e+02	9.6644e+01	-2.3288e+01	2.4429e+01	6.1581e+01	-7.6668e+01	-1.4445e+02
-1.4428e+01	-1.9612e+01	8.6891e+01	-2.8979e+01	-7.7040e+01	3.1010e+01	-8.1265e+00	4.4803e+01	-2.4621e+01	-5.6067e+01
1.5189e+01	-2.3514e+01	-5.7781e+01	7.9838e+01	-1.0206e+01	-2.1841e+01	-3.2627e+01	-3.1923e+00	-3.1874e+01	-1.8514e+01
-4.5440e+00	-3.7969e+00	-1.7864e+00	-3.5730e+01	-7.0291e+00	2.6815e+01	-2.6660e+01	-9.5631e+00	2.6286e+01	-7.2526e+00
-5.8788e+00	9.1451e+00	-3.0976e+01	5.3318e+01	-3.6875e+01	-4.7431e+01	3.0781e+01	1.8492e+01	-1.5432e+01	-3.8766e+01
-2.9230e+00	8.1934e+00	4.4215e+00	-2.6643e+00	3.3616e+01	-3.2850e+01	-1.2991e+01	3.4369e+01	2.7029e+01	2.0636e+01

Columns 11 through 16:

3.4882e+00	4.2411e+01	-3.4887e+01	7.8688e+00	-2.7953e+00	-2.7197e+00
-1.9920e+02	3.3331e+01	2.1696e+01	-3.0396e+00	2.5181e+00	5.0777e+00
4.5811e+01	-2.5678e+00	-3.9179e+01	-3.5182e+00	2.8395e+00	-5.0936e+00
-7.0826e+01	2.3774e+01	6.8432e+01	-1.7226e+01	1.2544e+01	-2.7820e+00
2.5217e+01	-8.1725e+01	-3.8755e+01	4.4899e+00	6.5876e+00	8.7157e+00
3.5496e+01	6.4546e+01	4.3468e+01	-2.6905e+01	-3.5235e+00	-6.1732e+00
-8.7847e+00	-4.7840e+01	1.3182e+01	2.5663e+01	1.9293e+01	1.2638e+01
1.7170e+01	1.9825e+01	2.0060e+01	-2.5341e+01	-1.0575e+01	-9.9434e+00
9.5876e+01	5.2153e+01	9.2887e+01	-4.1468e+01	-9.2776e+01	3.3550e+01
1.3862e+02	-2.6926e+00	5.1445e+01	-1.0963e+02	-5.5912e+01	6.9994e+01
-7.6139e+00	-5.3495e+00	4.3520e+01	-5.6297e+01	3.1402e+01	3.4535e+01
7.4029e+01	-2.1808e+00	-6.5132e+01	1.3025e+01	1.4065e+01	1.5556e+01
-5.0226e+01	-5.1643e+01	-1.8109e+01	9.0484e+01	-2.3514e+01	7.4206e+00
-2.6249e+01	-7.2136e+00	3.1675e-01	1.3500e+01	-1.7693e+01	-1.1354e+01
-1.1185e+01	-1.1891e+01	-1.7430e+01	1.1308e+01	1.8571e+01	-6.0447e+00
-1.2460e+01	-1.3107e+01	-1.2959e+00	1.2446e+01	-2.5283e+01	1.5507e+01

Матрица квантования при качестве равном 90:

1	12	23	34	45	56	67	78
12	23	34	45	56	67	78	89
23	34	45	56	67	78	89	100
34	45	56	67	78	89	100	111
45	56	67	78	89	100	111	122
56	67	78	89	100	111	122	133
67	78	89	100	111	122	133	144
78	89	100	111	122	133	144	155

DY после квантования при качестве равном 90:

1209	-180	23	-170	-45	-56	-134	-78	1561	72	0	34	-45	0	0	0
48	-23	0	0	0	67	78	0	72	115	-204	45	0	0	0	0
-69	68	-45	-56	67	78	89	100	-161	68	45	0	-67	0	0	0
68	-90	56	67	-78	89	0	0	34	0	-56	0	78	0	0	0
0	56	-67	0	0	0	0	0	0	56	0	-78	0	0	0	0
0	-67	78	-89	0	0	0	0	0	0	0	89	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1309	180	-23	-34	135	-112	0	0	1286	-96	92	68	90	-56	-67	0
-108	-69	0	0	112	-67	78	0	-240	138	136	0	56	-134	-78	89
-115	102	-45	-112	67	0	0	100	-69	-136	0	0	67	-78	0	0
0	0	112	0	-78	0	0	0	-34	-45	56	0	-78	0	0	0
0	0	-67	78	0	0	0	0	-45	0	-67	-78	0	100	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Видно, что младшие коэффициенты (в правом нижнем углу) обратились в ноль. Для менее пёстрой картинки или более низкого качества сжатия нулевых значений было бы больше.

Y после обратного ДКП:

```
Columns 1 through 13:

  114.310   78.734  212.063  183.825  100.035  142.837  223.648  209.933  212.082  224.067  232.679  232.876  224.270
  114.017  127.362  158.216  177.546  146.131  150.692  162.009  173.366  186.815  219.163  231.303  215.884  211.060
   62.461  193.058  122.890  167.714  133.798  120.452  145.003  244.361  212.435  218.868  206.070  195.561  224.531
   38.686  240.019  108.538  176.256  131.204  140.836  175.719  247.711  229.641  225.997  222.008  225.098  234.561
   76.255  241.510   85.169  174.700  159.208  213.793  226.108  162.594  205.270  212.798  241.099  258.925  226.775
   77.449  203.266   83.278  136.416  119.203  187.231  272.424  223.682  215.169  178.215  168.520  198.808  216.707
   54.229  169.385  122.325  133.187  104.936   85.128  205.280  259.506  220.525  180.710  125.718  105.955  147.494
   69.040  162.397  161.197  183.072  185.709   38.299   57.133  115.463  173.908  221.465  188.013   76.044   37.451
  181.430  127.861  151.037  116.826  197.899   67.309   15.180  151.402  106.884  134.944   27.485   22.975  144.805
  150.163  144.538  184.239  173.837  178.492   46.065   22.803  219.632  184.141  194.470   32.747   30.879  200.084
  193.455  149.540  119.382  187.558  227.893  160.270   66.536  181.748  229.257  209.206   85.933  103.509  210.596
  185.627  184.476  128.203  205.201  223.099  217.407  114.820  170.624  205.364  169.113  160.182  192.406  156.232
  206.095  233.093  144.778  170.519  174.923  223.953  165.064  170.626  181.189  162.436  176.302  192.389  127.578
  237.212  258.520  162.885  136.488  138.395  175.215  180.150  167.228  181.937  185.644  136.446  145.468  165.726
  184.900  245.263  241.524  198.353  145.495   60.607  130.464  167.151  158.883  149.581  118.093  182.585  210.669
  196.757  194.649  212.083  229.945  243.177   77.938   90.149   63.848  115.093   73.964  136.288  276.083  222.129

Columns 14 through 16:

  189.387  119.523   55.606
  208.749  166.749  109.425
  252.020  210.279  137.780
  233.422  210.199  183.582
  183.789  199.505  250.045
  208.416  214.485  238.401
  205.203  217.531  195.448
  115.602  192.447  205.866
   31.098  113.714  196.073
   76.150  100.232  202.124
  101.956  113.409  173.070
  111.512  165.422  131.421
  165.816  217.911  147.035
  246.172  235.837  212.519
  269.914  226.210  250.552
  241.248  214.566  244.345
```

Компонента R до округления после преобразования из YUV:

```
Columns 1 through 13:

  188.953  143.339  265.087  231.214  146.561  186.517  259.693  238.730  245.116  255.913  263.419  264.297  259.143
  178.975  188.150  214.508  231.496  197.473  194.972  195.117  197.582  217.940  248.955  259.718  244.627  242.873
  112.302  247.942  184.069  231.010  191.260  164.688  173.834  262.891  241.119  245.948  231.275  220.442  251.776
   74.212  289.442  173.851  246.328  191.228  182.763  201.213  264.246  257.263  251.662  245.146  247.060  257.966
  103.226  287.896  152.081  245.166  215.064  250.347  250.866  183.911  234.457  239.646  264.715  280.443  248.738
  102.928  249.501  149.012  200.834  164.796  216.165  299.157  255.827  248.309  208.664  195.085  222.424  239.845
   82.725  217.229  185.435  188.817  138.515  106.507  235.248  304.100  258.319  215.543  156.167  132.803  173.159
  100.674  211.734  222.263  232.441  211.282   54.966   89.475  168.299  214.808  259.258  221.153  105.231   65.073
  210.630  170.731  196.062  138.439  192.287   66.410   56.211  234.227  168.654  204.093   65.923   35.692  165.608
  180.395  189.340  233.019  200.518  175.703   40.224   47.877  278.323  250.859  250.586   45.165   27.429  228.361
  220.925  193.741  172.557  224.569  236.845  158.589   81.650  219.124  294.982  257.094   87.827   99.923  253.288
  204.485  222.486  182.348  254.461  253.408  237.070  142.470  211.665  258.040  226.341  193.783  221.773  212.827
  216.128  262.080  194.203  226.214  224.780  270.226  217.843  232.505  218.191  237.228  255.589  257.987  182.721
  246.440  281.609  202.952  186.592  191.769  232.505  245.760  240.578  216.487  265.351  221.768  203.324  194.078
  202.407  268.170  271.380  233.558  185.382  107.451  186.532  230.227  207.428  214.800  156.586  182.754  195.884
  223.103  219.783  235.366  252.379  268.047  109.512  130.556  110.598  179.183  121.726  123.090  219.390  174.123

Columns 14 through 16:

  229.957  166.080  105.993
  245.906  209.624  155.982
  283.958  247.435  178.350
  260.668  242.012  218.455
  208.670  228.248  281.466
  233.622  242.900  269.142
  232.284  247.322  227.295
  144.286  223.571  238.900
   47.497  129.550  240.605
  112.583  126.944  242.259
  155.517  144.099  202.272
  157.854  179.627  144.162
  182.060  202.998  144.702
  229.526  200.014  204.322
  234.091  188.107  246.410
  199.788  182.912  246.403
```