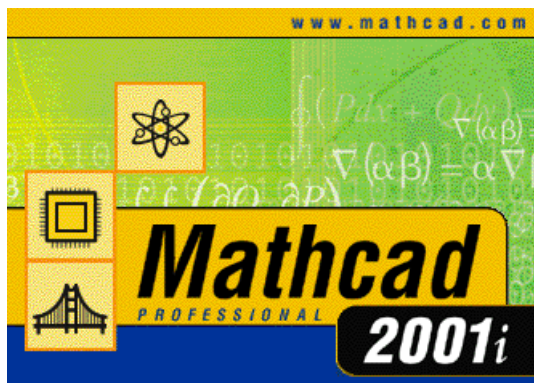




Ю.Е. Воскобойников
В.Ф. Очков

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ ЗАДАЧ В ПАКЕТЕ MATHCAD



НОВОСИБИРСК 2002

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ АРХИТЕКТУРНО-СТРОИТЕЛЬНЫЙ
УНИВЕРСИТЕТ

Ю.Е. Воскобойников, В.Ф. Очков

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ ЗАДАЧ В ПАКЕТЕ MATHCAD

УЧЕБНОЕ ПОСОБИЕ

НОВОСИБИРСК 2002

УДК 004.4
ББК 32.973-018

В 762

Воскобойников Ю.Е.

Программирование и решение задач в пакете MathCAD: Учеб. пособие: Ю.Е. Воскобойников, В.Ф. Очков. – Новосибирск: НГАСУ, 2002. – 136 с.

ISBN-5-7795-0169-6

В учебном пособии рассмотрены основные конструкции встроенного языка математического пакета MathCAD2001i (русифицированная версия) для программирования основных типов вычислительных алгоритмов (линейных, разветвляющихся и циклов) и обмена данными с другими прикладными программами. Подробно рассматривается модульное программирование и его реализация в пакете MathCAD. Для часто встречающихся в строительстве задач приводится их решение в данном пакете. Изложение сопровождается большим числом примеров и задач, что способствует лучшему усвоению материала.

Учебное пособие предназначено для студентов специальности «Информационные системы и технологии» и для студентов дневной формы обучения, изучающих курс «Компьютерные технологии в строительстве», а также оно будет полезно аспирантам и инженерам, использующим в своих расчетах этот математический пакет.

Печатается по решению издательско-библиотечного совета
НГАСУ

Рецензенты:

- В.Н. Зиновьев, к.ф.-м.н., доцент, старший научный сотрудник Института теоретической и прикладной механики СО РАН;
- Н.П. Кисленко, к.т.н., доцент кафедры прикладной математики (НГАСУ)

ISBN-5-7795-0169-6 © НГАСУ, 2002

© Воскобойников Ю.Е.,
Очков В.Ф., 2002

Учебное издание
Воскобойников Юрий Евгеньевич
Очков Валерий Федорович

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ ЗАДАЧ В ПАКЕТЕ MATHCAD

Учебное пособие

Редактор И.Э. Спирина

Санитарно-эпидемиологическое заключение

№ 54.НЦ. 02.953.П.127.10.01 от 01.10.2001 г.

Подписано к печати 10.12.2002. Формат 60х84 1/16 д.л.

Гарнитура Таймс. Бумага газетная. Ризография.

Объем 8,0 уч.-изд.л.; 8,75 п.л. Тираж 400 экз. Заказ №

Новосибирский государственный архитектурно-
строительный университет

630008, Новосибирск, ул. Ленинградская, 113

Отпечатано мастерской оперативной полиграфии НГАСУ

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
РАЗДЕЛ 1. ЭКСПОРТ И ИМПОРТ ДАННЫХ В ПАКЕТЕ MathCAD	7
ТЕМА 1. ЗАПИСЬ И ЧТЕНИЕ ФАЙЛОВЫХ ДАННЫХ	7
1.1. Файловый тип данных MathCAD	7
1.2. Запись данных в файл	9
1.3. Чтение данных из файла	13
ТЕМА 2. ОБМЕН ИНФОРМАЦИЕЙ С ДРУГИМИ ПРОГРАММАМИ-ПРИЛОЖЕНИЯМИ	15
2.1. Обмен информацией с текстовым процессором Word... ..	15
2.2. Обмен информацией с табличным процессором Excel.. ..	19
РАЗДЕЛ 2. ПРОГРАММИРОВАНИЕ В ПАКЕТЕ MathCAD	27
ТЕМА 3. БЕЗМОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ В ПАКЕТЕ MATHCAD	28
3.1. Программирование линейных алгоритмов	28
3.2. Программирование разветвляющихся алгоритмов	29
3.3. Программирование циклических алгоритмов	35
ТЕМА 4. ПОДПРОГРАММА-ФУНКЦИЯ: ОПИСАНИЕ И ВЫЗОВ	39
4.1. Описание подпрограммы-функции и локальный оператор присваивания	39
4.2. Обращение к подпрограмме-функции MathCAD	42
ТЕМА 5. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ В ПОДПРОГРАММЕ-ФУНКЦИИ MathCAD	44
5.1. Программирование линейных алгоритмов в подпрограмме-функции	44
5.2. Программирование разветвляющихся алгоритмов в подпрограмме-функции	45
5.3. Программирование циклических алгоритмов в подпрограмме-функции	51
ТЕМА 6. ПРОГРАММИРОВАНИЕ ТИПОВЫХ ЗАДАЧ В ПОДПРОГРАММАХ-ФУНКЦИЯХ MathCAD	62
6.1. Программирование разветвляющихся алгоритмов	62
6.2. Программирование циклов типа арифметической прогрессии	66

6.3. Программирование итерационных циклов	74
ТЕМА 7. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ В MathCAD	80
7.1. Преимущества модульного программирования	80
7.2. Модульное программирование в пределах одного документа MathCAD	81
7.3. Модульное программирование в нескольких документах MathCAD	84
7.4. Программы MathCAD в Internet	85
РАЗДЕЛ 3. РЕШЕНИЕ НАУЧНО-ИНЖЕНЕРНЫХ ЗАДАЧ В ПАКЕТЕ MathCAD	94
ТЕМА 8. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ В ПАКЕТЕ MathCAD	94
8.1. Решение нелинейных уравнений	94
8.2. Решение систем уравнений	105
ТЕМА 9. РЕШЕНИЕ ОПТИМИЗАЦИОННЫХ ЗАДАЧ В ПАКЕТЕ MathCAD	112
9.1. Решение оптимизационных задач без ограничений	112
9.2. Решение оптимизационных задач с ограничениями	114
ТЕМА 10. ОБРАБОТКА ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ В ПАКЕТЕ MathCAD	118
10.1. Моделирование и обработка статистических данных	118
10.2. Построение эмпирических зависимостей	125
ЗАКЛЮЧЕНИЕ	135
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	135

ВВЕДЕНИЕ

Во все времена инженерам, исследователям (т.е. специалистам в своих областях) был необходим удобный и достаточно эффективный (для своего времени) инструмент для решения своих задач. В этот «инструментальный» ряд можно включить логарифмическую линейку, арифмометр, калькулятор, универсальную ЭВМ, персональный компьютер. При использовании вычислительной техники встала проблема реализации алгоритмов решения в виде так называемых программ. Для решения этой проблемы в различные годы использовались следующие средства:

- программирование в машинных кодах (включая языки типа Ассемблер);
- программирование на языках высокого уровня (включая объектно-ориентированное программирование);
- системы компьютерной математики.

Разработка программы (даже с использованием языков высокого уровня с приставками Visual) требует и соответствующей подготовки (назовем ее «программистской»), и достаточно большего количества времени (и то и другое часто отсутствует у «обычного пользователя»). Поэтому, начиная с 90-х годов прошлого века, широкую известность и заслуженную популярность приобрели так называемые *системы компьютерной математики* [1] или, проще, *математические пакеты*. К ним можно отнести MathCAD [2, 3], MatLab [4, 5], Mathematica [6], Maple [7].

На наш взгляд, наиболее подходящим для выполнения научно-инженерных расчетов является математический пакет MathCAD, особенно его последние версии MathCAD2000 Professional, MathCAD2001i Professional. Эти версии содержат тщательно сбалансированные средства численных и символьных вычислений с графической визуализацией результатов в сочетании с современным интерфейсом пользователя, мощной справочной системой, обширными пакетами расширений (ориентированных на решение определенного класса задач) и средствами для работы в Internet.

Основам работы с последними версиями пакета MathCAD посвящены несколько книг и учебников [2, 8, 9]. К сожалению, в них не уделено должного внимания вопросам программной реализации различных алгоритмов, особенно с использованием программных

модулей – подпрограмм-функций MathCAD. Отчасти это объясняется большим объемом «общеобразовательной» информации, которая необходима для широкого круга пользователей, а также смещением акцента в сторону использования «готовых» функций, входящих как в сам MathCAD, так и в пакеты расширений. Их использование порождает достаточно простые алгоритмические конструкции, реализуемые непосредственно в документе MathCAD.

Однако в ряде случаев возникает необходимость программирования того или иного «нестандартного» вычислительного алгоритма. Здесь необходимы навыки программирования с учетом особенностей конструкций пакета MathCAD.

Поэтому в разделе 2 данного учебного пособия достаточно подробно излагаются конструкции MathCAD (русифицированная версия MathCAD2001i), необходимые для реализации различных типов алгоритмов: линейных, разветвляющихся и циклических. Основное внимание уделяется разработке программных модулей – подпрограмм-функций MathCAD. Обсуждается реализация метода модульного программирования. В третьем разделе рассматривается решение «типовых» задач, встречающихся при расчете и проектировании строительных конструкций, а в первом разделе обсуждаются вопросы «импорта и экспорта» данных в пакете MathCAD.

Предполагается, что читатель уже знаком с основами работы в пакете MathCAD (запуск пакета, работа с окнами, ввод, редактирование конструкций, выполнение в пакете элементарных вычислений). Изложение материала сопровождается большим числом примеров разной сложности, а предлагаемые для самостоятельного выполнения задания позволяют получить практические навыки программирования различных алгоритмов.

Учебное пособие предназначено для студентов специальности «Информационные системы и технологии», для студентов дневной формы обучения, изучающих курс «Компьютерные технологии в строительстве», а также для аспирантов и инженеров, использующих в своих расчетах этот математический пакет. Пособие, безусловно, будет полезно всем, использующим MathCAD при решении «своих» задач и желающим познать радость от эффективной работы «своей» программы.

РАЗДЕЛ 1. ЭКСПОРТ И ИМПОРТ ДАННЫХ В ПАКЕТЕ MathCAD

В этом разделе рассматриваются конструкции MathCAD, позволяющие осуществить обмен информацией между пакетом MathCAD и другими программами-приложениями, включая такие офисные программы как текстовый процессор Word и табличный процессор Excel.

ТЕМА 1. ЗАПИСЬ И ЧТЕНИЕ ФАЙЛОВЫХ ДАННЫХ

В этой теме будут рассмотрены функции MathCAD, позволяющие записывать и считывать числовую и символьную информацию, представляемую ASCII-кодами, когда каждому символу соответствует определённый код со значением от 0 до 255. Обмен данными через файл позволяет «сстыковать» MathCAD с программами, написанными на других языках высокого уровня (например, Си, Pascal и т.д.). Эти программы могут либо «генерировать» данные для дальнейшей обработки MathCAD, либо использовать данные, «генерируемые» в пакете MathCAD.

1.1. Файловый тип данных MathCAD

Данные, читаемые из файла или записываемые в файл, принадлежат к новому типу данных – файловых.

Для удобства изложения операторов для работы с файловыми данными, разделим эти данные на две группы:

- структурированные файлы;
- неструктурированные файлы.

В структурированном файле данные располагаются в виде матрицы, т. е. каждая строка (так называемая запись) имеет одинаковое число элементов. В неструктурированном файле данные располагаются либо последовательно (только одна запись), либо в нескольких записях, но с разным числом элементов в них.

При работе со структурированными файлами необходимо помнить:

- пробелы, запятые, знаки табуляции используются как разделители данных;
 - перевод строки (клавиша [Enter]) осуществляет переход к новой строке (новой записи файла);
 - в качестве разделителя между целой и дробной частью вещественного числа используется только десятичная точка (*внимание при работе с данными из таблиц Excel*);
 - данные в файле должны быть упорядочены в виде матрицы, т.е. каждая строка должна содержать одинаковое количество числовых значений;
 - пустые строки и строки, содержащие ASCII-текст, при считывании игнорируются;
 - если файл не соответствует перечисленным требованиям, то имя файла в функциях ввода-вывода выделится красным цветом.
- При работе с неструктурированными файлами необходимо помнить:
- пробелы, запятые, знаки табуляции, переводы строк используются как разделители данных;
 - в качестве разделителя целой и дробной части вещественного числа используется только десятичная точка.

В функциях работы с файловыми данными аргументом является **Имя файла**, в качестве которого может выступать:

- строковая константа, содержащая полное имя файла или только имя файла (если он находится в текущем каталоге);
- строковая переменная, получившая значение строковой константы, определяющей имя файла.

Пример 1.1.1. Проиллюстрируем два способа задания **Имя файла** на примере функции READ :

а) A := READ("E:\COPY\data1.dat")
б) file_1 := "E:\COPY\data1.dat"
A := READ(file_1) ◆

1.2. Запись данных в файл

Для создания неструктурированных файлов используется две функции:

$WRITE$ («Имя файла») := «ДАННЫЕ»

$APPEND$ («Имя файла») := «ДАННЫЕ»

При использовании этих функций следует помнить:

- если в функции $WRITE$ указано имя существующего файла, то он заменяется новым файлом без предупреждения. Следовательно, этой функцией нельзя пользоваться для добавления данных в конец существующего файла;
- функция $APPEND$ используется для добавления данных в конец существующего файла. Если файл не существует, то эта функция создает его, записывая туда данные;
- каждое обращение к функции $APPEND$ добавляет в файл данные, начиная с новой строки (новую запись);
- ДАННЫЕ – это только один элемент из следующего списка:
 - имя простой переменной;
 - имя массива с указанием индексного выражения;
 - значение числовой константы;
- для записи нескольких элементов необходимо вызов функции поставить в тело цикла, параметр которого индексируется выражением (см. пример 1.2.1).

Пример 1.2.1. Записать в файл следующую информацию:

- значение переменной $n_{samp} = 10$;
- n_{samp} значений случайных чисел из интервала $[0, 99]$;
- n_{samp} значений целых чисел от 0 до $n_{samp} - 1$.

Фрагмент документа, осуществляющий запись таких файловых данных, приведен на рис. 1.2.1. Здесь функция $rnd(a)$ генерирует случайные числа, равномерно распределенные в интервале $[0, a]$, а функция $floor(x)$ – вычисляет наибольшее целое число, меньшее или равное вещественному числу x . В окне программы Блокнот хорошо видна структура сформированного файла. ♦

$n_{samp} := 10 \quad i := 0..n_{samp} - 1$

$x_i := floor(rnd(100))$

$WRITE(file_1) := n_{samp}$

$APPEND(file_1) := x_i$

$APPEND(file_1) := i$

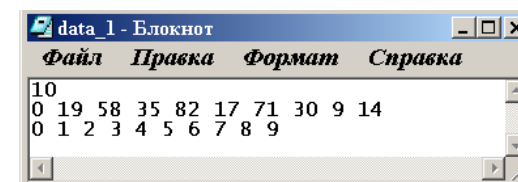


Рис. 1.2.1. Запись неструктурированного файла

Замечание 1.2.1. Рассмотренные функции создания неструктурированного файла входили в состав более ранних версий пакета MathCAD (до версии MathCAD2000 включительно). В русифицированной версии MathCAD2001i обращение к этим функциям вызывает ошибку (см. рис. 1.2.2), поэтому рекомендуется работать только с функциями создания структурированных файлов.

$fl := "a:\data1.txt"$

$x := 2 \quad WRITE(fl) := x$

This function is obsolete.
Press [F1] for Help.

Рис. 1.2.2. Ошибка при обращении к функции $WRITE$

Для создания структурированных файлов используются две функции:

$WRITEPRN$ («Имя файла») := <ДАННЫЕ>

$APPENDPRN$ («Имя файла») := <ДАННЫЕ>

При использовании этих функций необходимо помнить:

- если в функции $WRITEPRN$ указано имя существующего файла, то этот файл заменяется новым файлом без предупреждения;
- функция $APPENDPRN$ используется для добавления данных в конец существующего файла. При этом **число столбцов добавляемого массива должно совпадать с числом столбцов массива, уже**

записанного в файл. Это правило позволяет добавлять в конец файла векторы с любым числом элементов;

- в качестве разделителя между числами записывается пробел, в конце каждой строки осуществляется переход на новую строку;
- **ДАННЫЕ** – это только один элемент из следующего списка:
 - имя простой переменной;
 - имя массива;
 - значение числовой константы.

Если необходимо записать в файл несколько переменных, то из них нужно сформировать вектор, который затем записывается в файл.

Структура создаваемого файла определяется следующими системными переменными:

PRNCOLWIDTH – количество позиций, отводимых под один столбец (по умолчанию равно 8);

PRNPRECISION – число задаваемых цифр после десятичной точки в записи числа (по умолчанию равно 4).

Значения этих параметров можно изменить, обратясь к пункту меню **Математика**, команда **Опции**, вкладка **Встроенные переменные**.

Пример 1.2.2. Записать в файл матрицу B , k -й столбец которой есть выборка из нормального распределения с математическим ожиданием $\mu_0 = 10$ и среднеквадратическим отклонением $\sigma_0 = 2$ (дисперсия соответственно $\sigma_0^2 = 4$).

Фрагмент документа, осуществляющий запись структурированного файла, приведен на рис. 1.2.3. В окне программы Блокнот хорошо видна структура сформированного файла (число строк $n = 5$, число столбцов $m = 4$). ♦

$$\mu_0 := 10 \quad \sigma_0 := 2$$

$$n := 5 \quad m := 4$$

$$k := 0..m - 1$$

$$B^{(k)} := \text{rnorm}(n, \mu_0, \sigma_0)$$

$$\text{WRITEPRN}(\text{file_2}) := B$$

Файл	Правка	Формат	Справка
12.04	10.56	16.1	7.593
10.36	10.32	9.427	7.926
11.03	10.86	7.44	9.961
11.48	9.05	11.52	10.06
9.967	8.609	9.215	10.53

Рис. 1.2.3. Запись структурированного файла

Задание 1.2.1. Составить фрагмент документа MathCAD, формирующий файл, содержащий два столбца:

- первый столбец содержит значения x_i , $i = \overline{0, n}$ определяемые по формуле:

$$x_i = a + \frac{b - a}{n};$$

- второй столбец содержит функции $\psi(x_i)$, $i = \overline{0, n}$, где

$$\psi(x) = e^{-x^2} \cdot \cos(5x). \quad \bullet$$

Задание 1.2.2. Составить фрагмент документа MathCAD, который формировал файл, содержащий кроме матрицы B , определяемой условиями примера 1.2.2, добавленную в конец файла матрицу C размерности $8 \times m$, k -й столбец которой есть выборка из нормального распределения с математическим ожиданием $\mu_0 = 10$ и дисперсией $\sigma_k^2 = 0.01 + k$, $k = \overline{0, m - 1}$. •

Замечание 1.2.2. Создаваемый в MathCAD файл данных достаточно просто просмотреть и отредактировать в текстовом редакторе, отображающем ASCII-данные. Примерами таких редакторов может служить программа Блокнот, входящая в состав Windows (Программы $\Rightarrow$ Стандартные), а также редакторы языков программирования – PASCAL, СИ и др.

Задание 1.2.3. Используя программу Блокнот, проверьте правильность формирования файла в задании 1.2.2. •

1.3. Чтение данных из файла

Для чтения данных из неструктурированных файлов используется функция:

$\langle \text{ИМЯ} \rangle := \text{READ}(\langle \text{Имя файла} \rangle),$

где $\langle \text{ИМЯ} \rangle$ – это либо имя простой переменной, либо имя массива с указанным индексным выражением (т.е. элемент массива).

Необходимо помнить, что при каждом обращении к функции *READ* чтение данных начинается с начала файла. Поэтому для чтения большого числа данных целесообразно считать их в массив, а затем через элементы этого массива определять нужные переменные (см. пример 1.3.1).

Пример 1.3.1. Сформировать вектор z из последних m элементов файла *data_1.dat*, сформированного в примере 1.2.1. Значение переменной m определяется первым числом в файле *data_1.dat*. Фрагмент программы показан на рис. 1.3.1. ♦

$\text{file_1} := "F:\COPY\data_1.dat"$

$m := \text{READ}(\text{file_1}) \quad m = 10$

$i := 0..2 \cdot m \quad \text{rab}_i := \text{READ}(\text{file_1}) \quad n := \text{last}(\text{rab})$

$j := 0..m - 1 \quad z_j := \text{rab}_{n-m+j+1}$

$$z^T =$$

	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9

Рис. 1.3.1. Чтение данных из неструктурированного файла

Замечание 1.3.1. Рассмотренная функция чтения неструктурированного файла входила в состав более ранних версий пакета MathCAD (до версии MathCAD2000 включительно). В русифицированной версии MathCAD2001i обращение к этой функции вызывает ошибку и рекомендуется работать только с функцией чтения структурированного файла.

Для чтения данных из структурированных файлов используется функция:

$\langle \text{ИМЯ} \rangle := \text{READPRN}(\langle \text{Имя файла} \rangle),$

где $\langle \text{ИМЯ} \rangle$ – это либо имя простой переменной, либо имя массива.

При использовании этой функции необходимо помнить:

- из файла читается весь записанный массив данных;
- пустые строки и строки, содержащие ASCII-текст, при считывании игнорируются.

Пример 1.3.2. Определить выборочное математическое ожидание и дисперсию по всем элементам матрицы B , сформированной в примере 1.2.2.

Фрагмент программы представлен на рис. 1.3.2. Здесь функция *mean* вычисляет выборочное среднее, а функция *var* – выборочную дисперсию. Обратите внимание на формирование вектора через элементы матрицы. Такой переход часто используется при обработке изображений. ♦

$B := \text{READPRN}(\text{file_2})$

$n := \text{rows}(B) \quad m := \text{cols}(B) \quad n = 5 \quad m = 4$

$j := 0..n - 1 \quad k := 0..m - 1$

$x_{j+k \cdot n} := B_{j,k} \quad \text{mean}(x) = 10.174 \quad \text{var}(x) = 3.215$

Рис. 1.3.2. Чтение данных из структурированного файла

Задание 1.3.1. Составить фрагмент документа, формирующий вектор V , k -я проекция которого равна выборочному среднему k -го столбца матрицы, сформированной в примере 1.2.2 и записанной в файл *file_2.dat*. •

В заключение этой темы заметим, что рассмотренные функции позволяют организовать «гибкую» связь пакета MathCAD с другими вычислительными программами на уровне обмена файловыми данными.

ТЕМА 2. ОБМЕН ИНФОРМАЦИЕЙ С ДРУГИМИ ПРОГРАММАМИ-ПРИЛОЖЕНИЯМИ

В этой теме будут рассмотрены некоторые способы и конструкции MathCAD, позволяющие осуществить обмен информацией между документом MathCAD и документами текстового процессора (редактора) Word и табличного процессора (электронной таблицы) Excel.

2.1. Обмен информацией с текстовым процессором Word

При оформлении дипломного проекта, диссертации, научной публикации возникает необходимость в объединении текстовой информации и результатов вычислений, графиков, полученных в документе MathCAD. Здесь возможны два способа объединения.

Способ 1. Вставка фрагмента документа MathCAD в документ Word.

Способ 2. Вставка фрагмента документа Word в документ MathCAD.

На наш взгляд, первый способ является более предпочтительным из-за «первичной роли» текстового процессора Word для большинства пользователей и мощных функциональных возможностей этого процессора для создания хорошо оформленных научных публикаций. Поэтому начнем изложение с первого способа.

Возможны следующие случаи:

Случай 1. В документе MathCAD выполнены необходимые расчеты, и фрагмент документа необходимо вставить в документ Word.

Для этого нужно:

а) перейти в документ MathCAD, выделить в нем необходимый фрагмент и занести его в буфер обмена, щелкнув правой кнопкой на выделенном фрагменте и выполнив команду контекстного меню **Сору (Копировать)**;

б) перейти в документ Word и щелкнуть левой кнопкой мыши в том месте, куда будет вставляться фрагмент из буфера обмена;

в) вставить из буфера обмена фрагмент MathCAD, выполнив команду **Paste (Вставить)** контекстного меню.

К сожалению, вставляемый таким образом фрагмент плохо позиционируется, т.е. перемещается по документу Word. В последних версиях Word отсутствует команда **Кадр** пункта меню **Вставка**, которая позволяет легко позиционировать вставляемый фрагмент. «Заменить» эту команду можно следующим образом: перед шагом в) включить панель инструментов **Рисование**, выбрать инструмент **Надпись** и установить необходимый размер окна (изменяя мышью его границы), в которое будет вставлен фрагмент из буфера обмена, а затем выполнить шаг в).

Вставляемый таким образом фрагмент легко позиционируется, и его размеры можно изменить стандартным приемом – «перетаскивая» границы окна. Кроме того, это окно легко отформатировать – заливка цветом, удаление линии границы окна и т. д.

На рис. 2.1.1 показан фрагмент документа Word, в который вставлен фрагмент MathCAD – описание программы-функции вычисления определенного интеграла. Для иллюстрации включен режим обтекания фрагмента MathCAD текстом Word.

Приводимая программа-функция выполняет вычисление определенного интеграла по формуле Симпсона (формула парабол).

Описание формальных параметров:

f – имя интегрируемой функции;
 a, b – пределы интегрирования;
 N – количество интервалов интегрирования

$$\begin{aligned} \text{Simpson}(f, a, b, N) := & \quad h \leftarrow \frac{b - a}{N} \\ & S \leftarrow (f(a) + f(b)) \\ & \text{for } i \in 0..N - 1 \\ & \quad S \leftarrow S + 4 \cdot f\left(a + i \cdot h + \frac{h}{2}\right) \\ & \text{for } i \in 1..N - 1 \\ & \quad S \leftarrow S + 2 \cdot f(a + i \cdot h) \\ & \frac{h}{6} \cdot S \end{aligned}$$

Рис. 2.1.1. Вставка фрагмента документа MathCAD

Случай 2. Непосредственно в документе Word необходимо создать фрагмент документа MathCAD, в котором будут выполнены

необходимые вычисления. Это проще всего осуществить установлением объектной связи текстового редактора с пакетом MathCAD, выполнив следующие шаги:

- а) обратиться к пункту меню **Вставка**, команда **Объект**;
- б) в появившемся диалоговом окне (рис. 2.1.2) выделить в списке Тип объекта строку MathCAD document и щелкнуть кнопкой ОК;

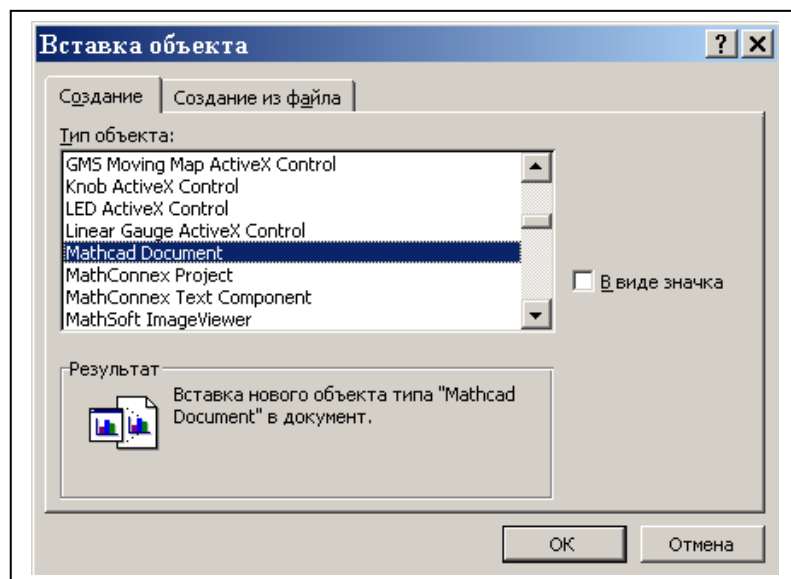


Рис. 2.1.2. Диалоговое окно Вставка объекта

в) в появившемся окне (часть документа выделена рамкой) набрать необходимые для вычислений конструкции и операторы (например, оператор вычисления определенного интеграла, приведенный на рис. 2.1.3);

г) для выхода из режима создания фрагмента MathCAD щелкнуть мышью вне окна MathCAD. После этого в документе Word отображаются только введенные конструкции MathCAD (см. рис. 2.1.4);

д) для редактирования и форматирования фрагмента MathCAD сделать на нем двойной щелчок левой кнопкой мыши и в появившемся окне пакета MathCAD выполнить необходимые операции.

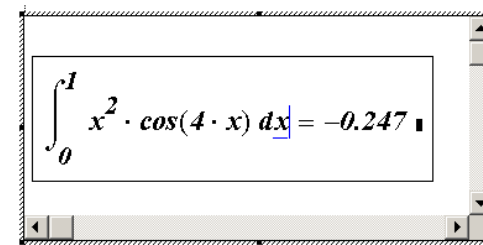


Рис. 2.1.3. Вставка фрагмента MathCAD в документ Word

$$\int_0^1 x^2 \cdot \cos(4 \cdot x) dx = -0.247$$

Рис. 2.1.4. Вставленный фрагмент MathCAD

В некоторых случаях неудобно «отображать» в документе Word фрагмент MathCAD в «полном объеме». Для того чтобы представить этот фрагмент значком, необходимо на шаге б) включить в диалоговом окне (см. рис. 2.1.2) флажок **В виде значка**. Тогда, после щелчка на кнопке ОК, на экране появится полнофункциональное программное окно MathCAD, а в документ Word вставится значок, показанный на рис. 2.1.5.

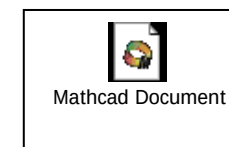


Рис. 2.1.5. Значок вставленного документа MathCAD

После набора необходимых конструкций и вычислений для возвращения в документ Word обратиться к пункту меню MathCAD **Файл**, команда **Выход и возврат к...** (последняя команда в выпадающем меню) или щелкнуть на кнопке завершения работы MathCAD.

Для редактирования и форматирования необходимо сделать двойной щелчок на значке документа MathCAD и в появившемся окне MathCAD выполнить необходимые операции, а затем вернуться в Word.

Весьма привлекательно выглядит документ Word со вставленными в него графиками, созданными программой MathCAD. Для вставки графиков из MathCAD необходимо выделить нужные графики, а затем выполнить шаги а) – в) случая 1.

Обратимся к способу 2, когда в документ MathCAD необходимо вставить фрагмент документа Word. Это можно осуществить:

- вставкой фрагмента Word из буфера обмена (см. рис. 2.1.6, где приведен фрагмент, в котором подчеркнутые слова были созданы редактором Word);
- установлением связи с редактором Word (как в описанном выше случае 2).

Замечание 2.1.1. Точно так же могут импортироваться в MathCAD фрагменты других программ-приложений (графических редакторов, других математических и статистических пакетов). Необходимо лишь одно условие – возможность установления объектной связи с другой программой (т. е. это приложение должно поддерживать технологию OLE).

2.2. Обмен информацией с табличным процессором Excel

Прежде всего заметим, что обмен информацией между MathCAD и Excel может быть осуществлен либо через буфер обмена, либо через установление объектной связи (это было показано в п. 2.1 на примере текстового процессора Word).

Более «быстрый» обмен можно осуществить, используя расширение MathCAD (начиная с версии MathCAD2000), называемое Add-In for Excel, которое устанавливает полноценную объектную связь между MathCAD и Excel. Однако это расширение устанавливается отдельно после инсталляции пакета MathCAD. Предположим, что это расширение установлено, и рассмотрим несколько случаев, часто встречающихся при работе в пакете MathCAD.

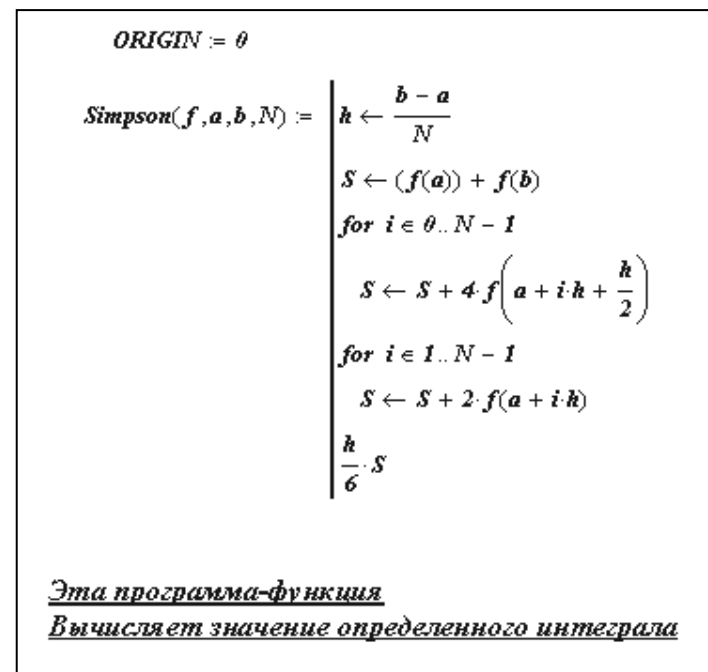


Рис. 2.1.5. Вставка в документ MathCAD фрагмента документа из редактора Word

Случай 1. В программе Excel создан набор табличных данных, которые необходимо обработать с использованием функций операторов MathCAD.

Для конкретности предположим, что в Excel была сформирована прямоугольная матрица, и нам необходимо вычислить ее сингулярные числа, используя функцию MathCAD *svds* (фрагмент таблицы приведен на рис. 2.2.1).

Microsoft Excel - Таблицы к теме 2

Файл Правка Вид Вставка Формат

Arial Cyr 10 Ж К Ч

	A	B	C	D	E
1					
2					
3		12,10	14,00	15,20	
4		3,23	6,00	71,50	
5		3,40	8,00	4,00	
6		4,12	9,00	7,00	
7		5,78	4,00	8,78	
8					
9					

Рис. 2.2.1. Фрагмент таблицы, созданной в Excel

Для этого необходимо выполнить следующие шаги:

а) находясь в документе MathCAD, обратиться к пункту меню **Вставка**, команда **Компонент**;

б) в появившемся диалоговом окне (см. рис. 2.2.2) выбрать строку **Excel** и щелкнуть кнопкой **След**;

в) в появившемся новом диалоговом окне (см. рис. 2.2.3) включить:

- кнопку **Create an empty Excel worksheet** – если будет заполняться пустая страница Excel;
- кнопку **Create from file** – если таблица записана в файле. Во втором случае, используя кнопку **Browse**, необходимо указать полное имя файла, в котором записана вставляемая таблица, и щелкнуть кнопкой **Далее** (в нашем случае, включаем нижнюю кнопку и указываем полное имя файла);

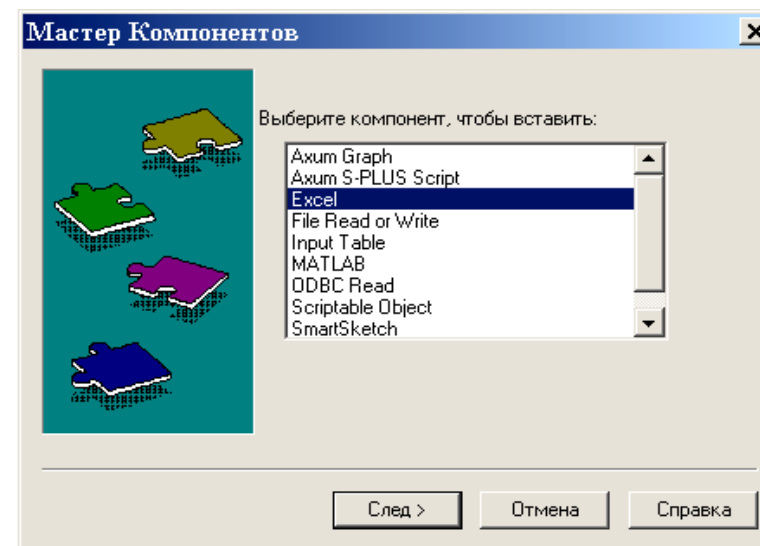


Рис. 2.2.2. Выбор компоненты Excel

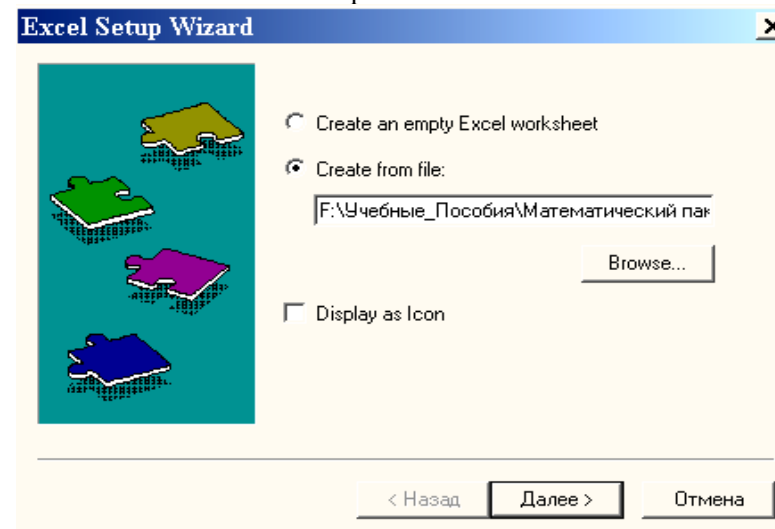


Рис. 2.2.3. Задание полного имени файла с электронной таблицей

г) в появившемся диалоговом окне (рис. 2.2.4) установить $Inputs = 0$, $Outputs = 1$, а в поле **Range** задать диапазон ячеек, занимаемых элементами матрицы (в нашем примере B3:D7), и щелкнуть кнопкой **Готово**;

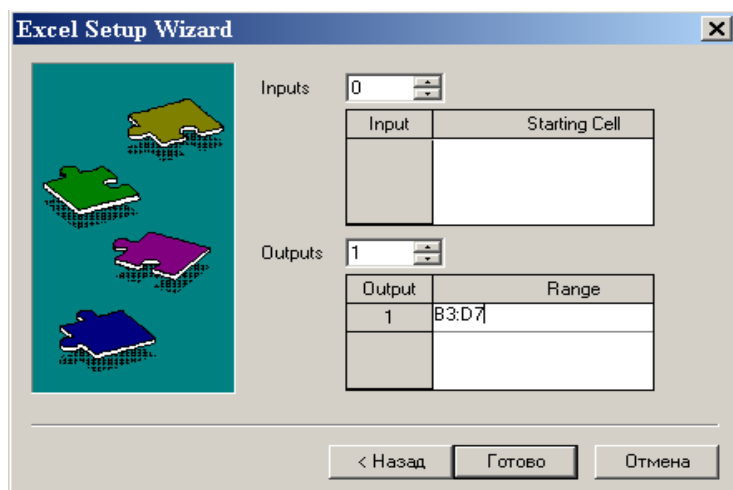


Рис. 2.2.4. Задание фрагмента таблицы

д) в появившемся окне документа MathCAD выводится вставляемый массив, в левом верхнем углу находится шаблон $\blacksquare :=$, в который вводится имя массива (в нашем примере имя массива A). После чего с определенным таким образом массивом можно выполнять нужные вычисления. На рис. 2.2.5 приведен фрагмент документа MathCAD вычисления сингулярных чисел вставленной матрицы A.

Заметим, что существует возможность из одной таблицы Excel сформировать два или более массивов. Возвращаясь к таблице, приведенной на рис. 2.2.1, сформируем две матрицы:

- матрицу A из диапазона ячеек B3:D7;
- матрицу B из диапазона ячеек B3:C7.

Для этого на шаге г) в диалоговом окне устанавливаем $Outputs = 2$, в поле **Range** заполняем две строки (первая строка B3:D7, вторая B3:C7) и щелкаем кнопкой **Готово**. Затем в появившемся окне (см. рис. 2.2.6) в первом поле шаблона вводим имя массива A, во

втором поле – имя B и щелкаем кнопкой **Готово**. На рис 2.2.7 показан фрагмент документа MathCAD, в котором введены сформированные матрицы A, B.

$$A := \begin{pmatrix} 12,10 & 14,00 & 15,20 \\ 3,23 & 6,00 & 71,50 \\ 3,40 & 8,00 & 4,00 \\ 4,12 & 9,00 & 7,00 \\ 5,78 & 4,00 & 8,78 \end{pmatrix} \quad A = \begin{pmatrix} 12.1 & 14 & 15.2 \\ 3.23 & 6 & 71.5 \\ 3.4 & 8 & 4 \\ 4.12 & 9 & 7 \\ 5.78 & 4 & 8.78 \end{pmatrix}$$

$$svals(A)^T = (75.194 \quad 20.67 \quad 3.969)$$

Рис. 2.2.5. Вычисления в документе MathCAD

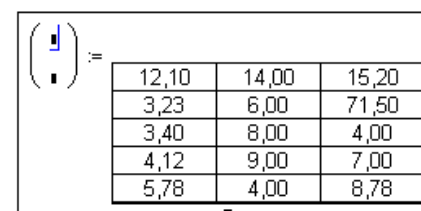


Рис. 2.2.6. Шаблоны для формирования массивов

Очевидно, что «выделение» матрицы B из матрицы A можно осуществить и средствами MathCAD (функция *submatrix*), но в общем случае формирование нескольких массивов MathCAD из различных ячеек таблицы Excel является очень эффективным средством связи MathCAD и Excel.

Остановимся еще на одном важном моменте. Сделав двойной щелчок на ячейках вставленной таблицы, мы вновь входим в электронную таблицу и можем сделать необходимое редактирование и вычисления. Для возврата в документ MathCAD достаточно щелкнуть мышью вне рамки электронной таблицы.

$$\begin{pmatrix} A \\ B \end{pmatrix} := \begin{array}{|c|c|c|} \hline 12,10 & 14,00 & 15,20 \\ \hline 3,23 & 6,00 & 71,50 \\ \hline 3,40 & 8,00 & 4,00 \\ \hline 4,12 & 9,00 & 7,00 \\ \hline 5,78 & 4,00 & 8,78 \\ \hline \end{array} \quad A = \begin{pmatrix} 12.1 & 14 & 15.2 \\ 3.23 & 6 & 71.5 \\ 3.4 & 8 & 4 \\ 4.12 & 9 & 7 \\ 5.78 & 4 & 8.78 \end{pmatrix}$$

$$svds(B)^T = (24.403 \quad 4.038) \quad B = \begin{pmatrix} 12.1 & 14 \\ 3.23 & 6 \\ 3.4 & 8 \\ 4.12 & 9 \\ 5.78 & 4 \end{pmatrix}$$

Рис. 2.2.7. Результаты вычислений в документе MathCAD

Случай 2. В MathCAD получены числовые данные, которые необходимо «передать» в таблицы процессора Excel (например, для построения «красивой» круговой диаграммы).

Для конкретности предположим, что в MathCAD сформированы два вектора: $x = |2, 4, 6, 8|^T$ и $y = |8, 6, 4, 2|^T$. Первый вектор нужно расположить в диапазоне ячеек B2:B5, а второй – в C3:D6.

Выполним следующие шаги:

а) шаг а) из случая 1;

б) включаем кнопку **Create an empty Excel** и щелкаем кнопкой **Далее** (см. рис. 2.2.3);

в) в появившемся окне устанавливаем $Outputs = 0$, $Inputs = 2$ и в поле **Starting Cell** заполняем две строки: в первой строке вводим адрес B2, во второй – C3 (см. рис. 2.2.8) и щелкаем кнопкой **ГОТОВО**;

г) в окне документа MathCAD выводится фрагмент пустой электронной таблицы, в левом нижнем углу находится два пустых шаблона (см. рис. 2.2.9), в которые вводим имена векторов x , y и щелкаем мышью вне рамки электронной таблицы;

д) в документе MathCAD появляется фрагмент электронной таблицы. Сделав на ней двойной щелчок, переходим в табличный процессор Excel для выполнения необходимых операций (форматирование, построение диаграмм и т.д.) На рис. 2.2.10 показан фраг-

мент таблицы с построенной диаграммой, и этот фрагмент расположен в документе MathCAD.



Рис. 2.2.8. Задание фрагмента таблицы

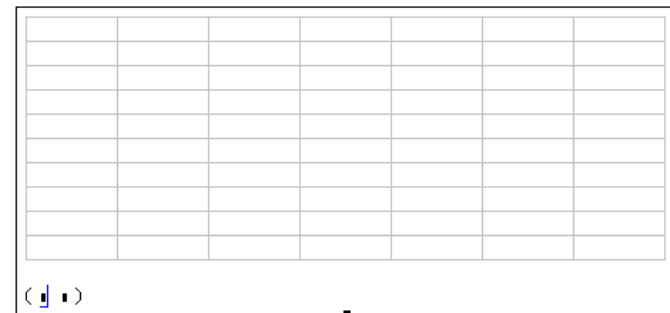


Рис. 2.2.9. Шаблоны имен массивов

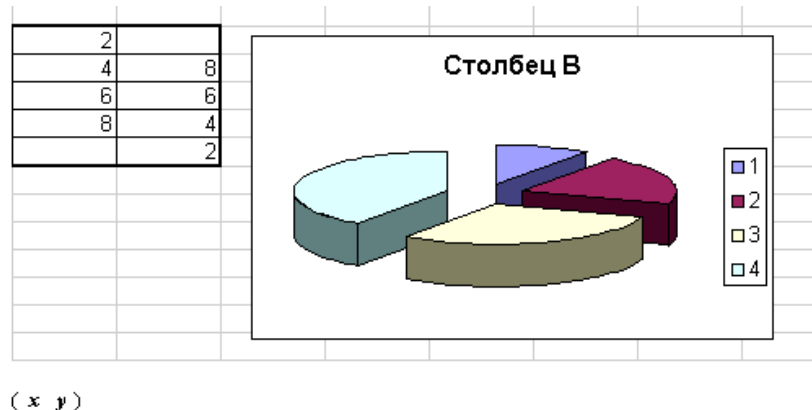


Рис. 2.2.10. Диаграмма, построенная в Excel по данным из документа MathCAD

В заключение параграфа заметим, что рассмотренные способы «связи» MathCAD с табличным процессором Excel существенно увеличивают возможности MathCAD при обработке табличных данных и их графической интерпретации.

РАЗДЕЛ 2. ПРОГРАММИРОВАНИЕ В ПАКЕТЕ MathCAD

В этом разделе рассматриваются конструкции пакета MathCAD, позволяющие реализовать следующие типы алгоритмов: линейный, разветвляющийся и циклический (проще – цикл). При этом будут изучены два способа программирования:

- программирование в пакете MathCAD без использования программных модулей;
- программирование с использованием программных модулей.

Первый способ (в дальнейшем для простоты названный безмодульным программированием) реализуется записью соответствующих конструкций непосредственно в математических областях документа MathCAD, и он приемлем для сравнительно простых алгоритмов.

Второй способ (называемый для простоты модульным программированием) предполагает реализацию отдельных независимых алгоритмов вычисления (например, решение нелинейного уравнения методом «деления отрезка пополам») в виде отдельных программных модулей, которые будем называть подпрограммами-функциями (сокращенно П-Ф). Первое слово «подпрограмма» указывает на свойство «изолированности» этого модуля от других вычислений в документах MathCAD, а второе слово «функция» – на способ вызова модуля и механизмы передачи вычисленных в модуле значений. Заметим, что принцип модульного программирования в свое время (70–80 годы XX века) существенно повысил производительность труда программистов, разрабатывающих программы с использованием алгоритмических языков высокого уровня. Применение его в пакете MathCAD позволяет:

- «распараллелить» разработку программы между несколькими исполнителями;
- создать проблемно-ориентированные библиотеки П-Ф для решения научно-технических задач с размещением библиотек на сайтах Интернета;
- уменьшить затраты на разработку и сопровождение программ для пакета MathCAD.

ТЕМА 3. БЕЗМОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ В ПАКЕТЕ MathCAD

Рассматриваются конструкции пакета, позволяющие реализовать линейный, разветвляющийся и циклический алгоритмы непосредственно в математических областях документа MathCAD.

3.1. Программирование линейных алгоритмов

Характерной особенностью линейных алгоритмов является строго последовательное выполнение всех операций алгоритма без пропусков и повторений вычислений. Поэтому конструкции, реализующие такой алгоритм, записываются в документе MathCAD в нужном порядке их выполнения, т.е. «слева направо – сверху вниз».

Пример 3.1.1. Составить программу для вычисления корней квадратного уравнения: $ax^2 + bx + c = 0$ по известной формуле:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \quad (3.1.1)$$

Алгоритм (3.1.1) является линейным (убедитесь в этом), и фрагмент документа MathCAD содержит конструкции, приведенные на рис. 3.1.1.

$$a := 2 \quad b := 5 \quad c := 8 \quad d := \sqrt{b^2 - 4 \cdot a \cdot c}$$

$$x_1 := \frac{-b + d}{2 \cdot a} \quad x_2 := \frac{-b - d}{2 \cdot a}$$

$$x_1 = -1.25 + 1.561i \quad x_2 = -1.25 - 1.561i$$

Проверка найденных корней

$$a \cdot x_1^2 + b \cdot x_1 + c = 0.00000 \quad a \cdot x_2^2 + b \cdot x_2 + c = 0.00000$$

Рис. 3.1.1. Пример программирования линейного алгоритма

В этом фрагменте x_1 , x_2 являются простыми переменными, цифры 1, 2 являются нижними индексами в именах (а не индексами выражениями у элементов массива) и поэтому вводятся после нажатия клавиши [.] – «десятичная точка». ♦

Задание 3.1.1. Подобрать коэффициенты a , b , c , получить две пары вещественных корней уравнения и две пары комплексных корней. Выполнить их проверку. •

Очевидно, что в реализации линейного алгоритма могут использоваться обращения к встроенным функциям MathCAD и функциям пользователя.

3.2. Программирование разветвляющихся алгоритмов

Характерной чертой разветвляющихся алгоритмов является наличие в них нескольких возможных ветвей вычислений. Выбор конкретной ветви зависит от выполнения (или не выполнения) заданных условий на значения переменных алгоритма.

Пример 3.2.1. Значение переменной y зависит от значений переменной x и определяется выражением:

$$y = \begin{cases} x^2, & \text{если } x \leq 0; \\ \sqrt{x}, & \text{в противном случае.} \end{cases} \quad (3.2.1)$$

Выбор одной из двух ветвей вычислений определяется текущим x . На рис. 3.2.1 представлена блок-схема этого алгоритма, хорошо подтверждающая название алгоритма – «разветвляющийся». ♦

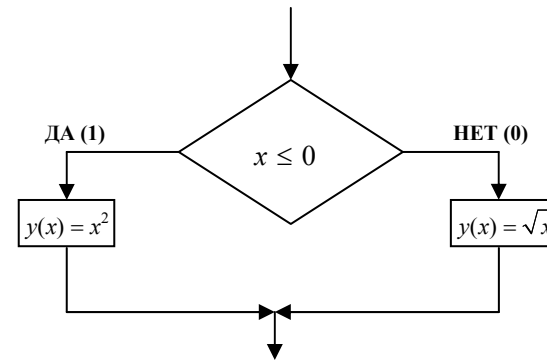


Рис. 3.2.1. Блок-схема разветвляющегося алгоритма (3.2.1)

Возникает вопрос: какие конструкции необходимы для реализации разветвляющегося алгоритма? Анализ алгоритма (3.2.1) и «программистская интуиция» подсказывают необходимость использования:

- конструкций, проверяющих выполнение заданных условий (чаще гораздо более сложных, чем условие алгоритма (3.2.1));
- конструкций, выбирающих нужную ветвь вычислений в зависимости от результатов проверки заданных условий.

Для проверки заданных условий в MathCAD используется: *выражение отношений, логические операции и логические выражения.*

Выражением отношений (или просто отношением) называется конструкция вида:

$$\langle \text{выр.1} \rangle \langle \text{операция отношения} \rangle \langle \text{выр.2} \rangle,$$

где $\langle \text{выр.1} \rangle$, $\langle \text{выр.2} \rangle$ – произвольные арифметические выражения, $\langle \text{операция отношения} \rangle$ – любая из следующих операций:

$| < | \leq | > | \geq | \neq | = |$ (здесь вертикальные чёрточки являются разделительным символом при перечислении).

Смысл этих операций понятен и не нуждается в пояснении. Для ввода знаков операций отношений можно использовать палитру

Логический (приведённую на рис. 3.2.2) или использовать клавиши, обозначения которых приведены в табл. 3.2.1.

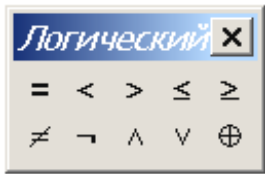


Рис. 3.2.2. Палитра инструментов **Логический**

Внимание! Не следует путать знак операции сравнения = с похожим знаком вывода значений переменных. Знак операции = имеет больший размер и более жирное начертание.

Таблица 3.2.1

Знаки операции	Клавиши
<	[<]
≤	[Ctrl] + [9]
>	[>]
≥	[Ctrl] + [0]
=	[Ctrl] + [=]
≠	[Ctrl] + [3]

Выражение отношений принимает одно из двух значений: 1 – если заданное отношение выполняется, 0 – в противном случае. Значение 1 можно интерпретировать как значение ИСТИНА, а 0 – как ЛОЖЬ.

Задание 3.2.1. Пусть значение целой переменной $x = 3$. Определить значение следующих выражений отношений:
 а) $x \geq 4$; б) $x + 1 \geq 4$; в) $x - 4 > 1$. •

Для проверки более сложных условий используются *четыре логические операции*, обозначения которых приведены в табл. 3.2.2.

Таблица 3.2.2

Название операции	Знак
Логическое отрицание (NOT)	¬
Логическое ИЛИ (OR)	∨
Логическое И (AND)	∧
Исключающее ИЛИ (XOR)	⊕

Знаки этих операций вводятся с палитры **ЛОГИЧЕСКИЙ**. Результат выполнения этих операций приведен в табл. 3.2.3.

Таблица 3.2.3

NOT ¬	AND ∧	OR ∨	XOR ⊕
$0 \neg 1 = 1$	$0 \wedge 0 = 0$	$0 \vee 0 = 0$	$0 \oplus 0 = 0$
$1 \neg 0 = 0$	$0 \wedge 1 = 0$	$0 \vee 1 = 1$	$1 \oplus 0 = 1$
	$1 \wedge 0 = 0$	$1 \vee 0 = 1$	$0 \oplus 1 = 1$
	$1 \wedge 1 = 1$	$1 \vee 1 = 1$	$0 \oplus 0 = 0$

Логическим выражением называется конструкция, состоящая из выражений отношений, логических операций и круглых скобок. Логическое выражение принимает только одно из двух значений: 1 или 0; вычисляется слева направо с учетом приоритета входящих в выражение операций. Наивысший приоритет – круглые скобки, а затем по убыванию: AND, OR и XOR – одинаковый приоритет и самый низкий приоритет выражения отношений.

Задание 3.2.2. Определите порядок вычисления значений логических выражений в документе MathCAD, приведенных на рис. 3.2.3. •

$$\begin{aligned}
 x &:= 0.2 \quad y := -4 \\
 (x > 3) \wedge (y > 0) &= \blacksquare \\
 \neg(x > 3) \wedge (y < 0) &= \blacksquare
 \end{aligned}$$

Рис. 3.2.3. Примеры логических выражений

Для выбора нужной ветви разветвляющегося алгоритма используется конструкция, названная *условной функцией if*, записываемая в виде:

$if(<\text{логическое выражение}>, <\text{выр. 1}>, <\text{выр. 2}>)$,

где имя функции *if* вводится с клавиатуры. Если логическое выражение равно 1, то значение функции определяется *выр. 1*, в противном случае – *выр. 2*. Блок-схема этой функции приведена на рис. 3.2.4.

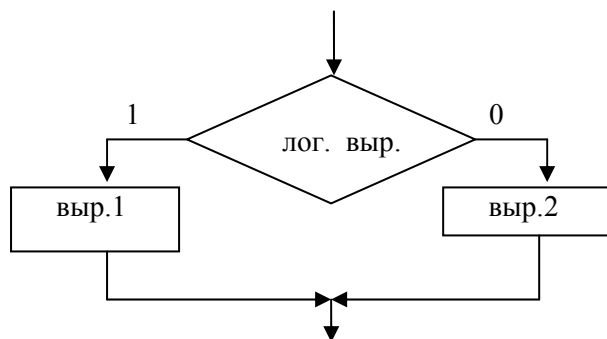


Рис. 3.2.4. Блок-схема функции *if*

При программировании разветвляющихся алгоритмов с тремя и более вычислительными ветвями в качестве *выр. 1* и *выр. 2* вновь может использоваться функция *if* (см. пример 3.2.1).

Пример 3.2.1. Используя условную функцию *if*, запрограммировать два разветвляющихся алгоритма.

$$\text{А. } y(x) = \begin{cases} x^2, & \text{если } x \leq 0; \\ \sqrt{x}, & \text{если } x > 0. \end{cases}$$

$$\text{Б. } z(x) = \begin{cases} 30, & \text{если } x \leq -1; \\ |x|, & \text{если } -1 < x \leq 1; \\ x^2 - 30, & \text{если } x > 1. \end{cases}$$

$$x := 3$$

$$y := if(x \leq 0, x^2, \sqrt{x}) \quad y = 1.732$$

$$z(x) := if(x \leq -1, 30, if(-1 \leq x \leq 1, |x|, x^2 - 30))$$

$$x := -10, -9.95 \dots 10$$

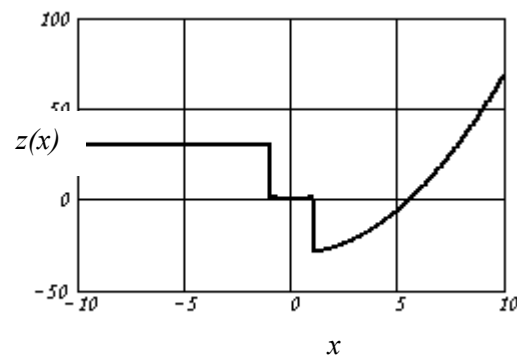


Рис. 3.2.5. Реализация разветвляющихся алгоритмов

Конструкции, реализующие эти алгоритмы, показаны на рис. 3.2.5. Для алгоритма Б была определена функция пользователя $z(x)$, а затем был построен ее график.

В MathCAD имеется ряд встроенных функций, которые возвращают результат, зависящий от знака или величины аргумента, и могут использоваться при программировании разветвляющихся алгоритмов. Приведем некоторые из них:

- $ceil(x)$ – наименьшее целое, большее или равное x ;
- $trunc(x)$ – целая часть вещественного числа x ;
- $floor(x)$ – наибольшее целое, меньшее или равное x ;
- $round(x, n)$ – округленное значение вещественного x с точностью до n знаков после десятичной точки;
- $\Phi(x)$ – функция Хевисайда – равна 0 при $x < 0$ и 1 в противном случае;
- $sign(x)$ – функция знака (равна 0 если $x = 0$; -1, если $x < 0$ и 1, если $x > 0$);

- $\text{sgnum}(x)$ – возвращает 1, если $x = 0$ и $\frac{x}{|x|}$ в остальных случаях.

3.3. Программирование циклических алгоритмов

Циклическим алгоритмом (или просто *циклом*) называется алгоритм, содержащий вычисления, повторяющиеся при различных значениях некоторой переменной, названной параметром цикла, а сами повторяющиеся вычисления составляют тело цикла.

Типы циклов. По способам организации цикла можно выделить:

- а) *цикл типа арифметической прогрессии*;
- б) *итерационный цикл*.

Особенностью цикла типа арифметической прогрессии является изменение параметра цикла по закону арифметической прогрессии, и поэтому можно, не выполняя цикла, определить количество повторений цикла.

Пример 3.3.1. Сформировать вектор z из n элементов, определяемых соотношением:

$$z_i = \frac{1}{(i+4)}, \quad i = 1, \dots, n; \quad n = 10.$$

В этом примере i – параметр цикла, а тело цикла (вычисление элементов z_i) будет повторено $n = 10$ раз, т.е. алгоритм является циклом типа арифметической прогрессии. ♦

В итерационном цикле невозможно «априори», т.е. не выполняя вычислений, определить число повторений тела цикла из-за разнообразных условий завершения цикла.

Пример 3.3.2. Вычислить приближенное значение $\sqrt{a}$, используя итерационную процедуру:

$$x_{n+1} = 0.5 \cdot \left(x_n + \frac{a}{x_n} \right), \quad n = 0, 1, \dots; \quad x_0 = a. \quad (3.3.1)$$

В качестве приближенного значения $\sqrt{a}$ принимается x_{n+1} , удовлетворяющее условию:

$$|x_{n+1}^2 - a| \leq \varepsilon, \quad (3.3.2)$$

где ε – заданная точность вычисления корня квадратного. Даже задав конкретные значения a, ε (например, $a = 9, \varepsilon = 0.01$), невозможно априори определить количество повторений цикла. ♦

Программирование цикла типа арифметической прогрессии. Параметр такого цикла задается дискретной переменной (называемой также ранжированной переменной), и тогда конструкции, входящие в тело цикла, располагаются начиная от этого описания и до конца документа MathCAD или до конструкции, переопределяющей дискретную переменную – параметр цикла.

Фрагмент документа, в котором реализован алгоритм решения задачи примера 3.3.1 приведен на рис. 3.3.1.

$n := 10$
 $z_i := \frac{1}{i+4}$

$i := 1..10$

$z^T =$		1	2	3	4	5	6	7	8	9	10
	1	0.2	0.167	0.143	0.125	0.111	0.1	0.091	0.083	0.077	0.071

Рис. 3.3.1. Формирование вектора z примера 3.3.1

На рис. 3.3.2 приведен фрагмент документа, формирующий матрицу (двумерный массив) по следующему правилу:

$$B_{i,j} = \frac{1}{i+j+1}; \quad i = 1, \dots, n; \quad j = 1, \dots, m. \quad (3.3.3)$$

Очевидно, что этот цикл имеет уже два параметра: i – параметр, определяющий номер строки; j – параметр, определяющий номер столбца матрицы. Напомним, что такой цикл называется *двойным циклом*.

$$\begin{aligned}
 n &:= 3 & m &:= 6 \\
 i &:= 1..n & j &:= 1..m \\
 B_{i,j} &:= \frac{1}{i+j+1} & B &= \begin{pmatrix} 0.333 & 0.25 & 0.2 & 0.167 & 0.143 & 0.125 \\ 0.25 & 0.2 & 0.167 & 0.143 & 0.125 & 0.111 \\ 0.2 & 0.167 & 0.143 & 0.125 & 0.111 & 0.1 \end{pmatrix}
 \end{aligned}$$

Рис. 3.3.2. Формирование матрицы B по алгоритму (3.3.3)

Заметим, что в приведенных на рис. 3.3.1 и 3.3.2 фрагментах системная переменная $ORIGIN=1$.

Программирование итерационных циклов. Для этого используется функция *until*, записываемая в виде:

$$until (<выр. 1>, <выр. 2>).$$

Эта функция принимает значение равное *выр. 2*, если *выр. 1* больше или равно 0. Как только *выр. 1* принимает отрицательное значение, функция *until* принимает значение 0, и дальнейшее выполнение функции прекращается. Это свойство позволяет применять функцию *until* для программирования итерационных циклов.

На рис. 3.3.3 приведен фрагмент документа, реализующий алгоритм примера 3.3.2. ♦

Замечание 3.3.1. В русифицированной версии MathCAD2001i обращение к этой функции **вызывает ошибку** и рекомендуется для программирования итерационных циклов использовать подпрограмму-функцию с оператором *while*.

$$\begin{aligned}
 ORIGIN &:= 0 \\
 a &:= 9 & x_0 &:= a & \varepsilon &:= 10^{-6} \\
 i &:= 0..10 \\
 x_{i+1} &:= until \left[\left| (x_i)^2 - a \right| - \varepsilon, \frac{x_i + \frac{a}{x_i}}{2} \right]
 \end{aligned}$$

$$x^T = (9.0000 \quad 5.0000 \quad 3.4000 \quad 3.0235 \quad 3.0001 \quad 3.0000 \quad 0.0000)$$

Рис. 3.3.3. Приближенное вычисление корня квадратного с помощью функции *until*

Последний, 7-й элемент массива x (значение индекса равно 6) равен нулю из-за того, что выражение $|(x_i)^2 - a| - \varepsilon$ стало меньше нуля, а значение функции *until* – 0. Поэтому в качестве приближенного значения корня необходимо принять предыдущий элемент, номер и значение которого определяются в следующем фрагменте:

$$n_{solve} := last(x) - 1 \quad n_{solve} = 5 \quad x_{n_{solve}} = 3.0000000$$

Следует отметить, что такой способ программирования *не эффективен при большом числе итераций* из-за необходимости хранить в памяти все предыдущие значения x_n . На самом деле необходимо хранить в памяти только два приближения: x_n («старое») и x_{n+1} («новое»).

В заключение этого параграфа заметим, что рассмотренные способы безмодульного программирования можно рекомендовать для реализации достаточно простых алгоритмов. Более сложные алгоритмы целесообразно реализовать в виде подпрограммы-функции, которая содержит конструкции, подобные конструкциям таких алгоритмических языков, как FORTRAN или Pascal: оператор присваивания, условные операторы, операторы цикла и т.д. В следующих темах будут рассмотрены конструкции пакета MathCAD, позволяющие реализовывать различные алгоритмы в подпрограмме-функции.

ТЕМА 4. ПОДПРОГРАММА-ФУНКЦИЯ: ОПИСАНИЕ И ВЫЗОВ

В этой теме будут рассмотрены описание подпрограммы-функции и ее вызов.

4.1. Описание подпрограммы-функции и локальный оператор присваивания

Перед тем как использовать подпрограмму-функцию (П-Ф), нужно ее задать, т.е. выполнить описание. Описание П-Ф размещается в рабочем документе перед ее вызовом и включает в себя имя подпрограммы-функции, список формальных параметров (который может отсутствовать) и тело подпрограммы-функции. Для ввода конструкций в тело П-Ф используется палитра инструментов **ПРОГРАММИРОВАНИЕ**, приведенная на рис. 4.1.1.

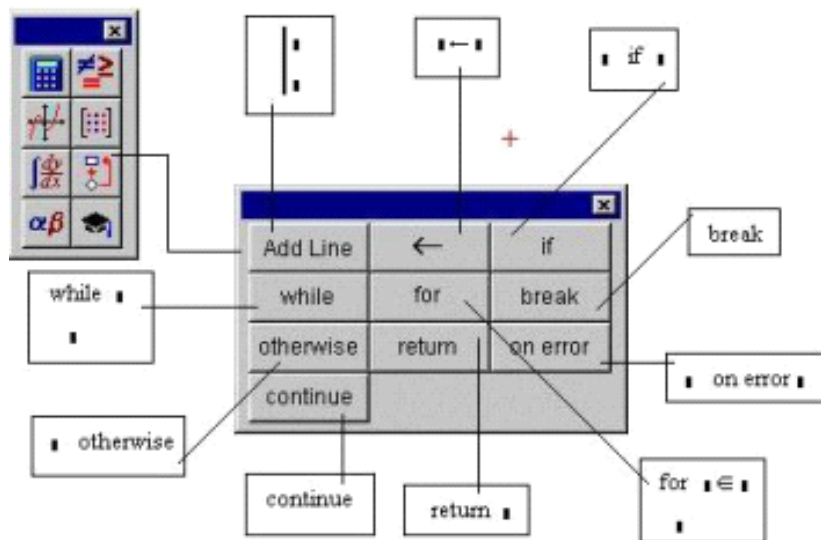


Рис. 4.1.1. Палитра **ПРОГРАММИРОВАНИЕ**

Каждая П-Ф MathCAD имеет *оригинальное имя*, посредством которого осуществляется обращение к ней. Через это же имя (*и только через это имя*) «возвращается» результат выполнения П-Ф.

После имени П-Ф идет *список формальных параметров*, заключенный в круглые скобки. Через формальные параметры «внутри» П-Ф «передаются» данные, необходимые для выполнения вычислений внутри программы, т.е. *все формальные параметры являются входными*. В качестве формальных параметров могут использоваться имена простых переменных, массивов и функций. Формальные параметры отделяются друг от друга запятой.

Замечание 4.1.1. П-Ф может не иметь формальных параметров, и тогда данные передаются через имена переменных, определенных выше описания П-Ф.

Тело подпрограммы-функции включает любое число операторов: локальных операторов присваивания, условных операторов и операторов цикла, а также вызов других П-Ф и функций пользователя.

Порядок описания подпрограммы-функции MathCAD. Для ввода в рабочий документ описания П-Ф необходимо выполнить следующие действия:

- ввести имя П-Ф и список формальных параметров, заключенный в круглые скобки (см. замечание 4.1.1);
- ввести символ “:=” — на экране отображается как “:=”;
- открыть палитру **ПРОГРАММИРОВАНИЯ** и щелкнуть кнопкой Add line (см. рис. 4.1.1). На экране появится вертикальная черта и вертикальный столбец с двумя полями для ввода операторов, образующих тело П-Ф (см. рис. 4.1.2);

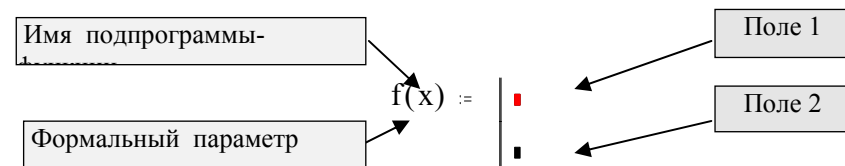


Рис. 4.1.2. Структура подпрограммы-функции

- перейти в поле 1 (щелкнув на нем мышью или нажав клавишу [Tab]) и ввести первый оператор тела П-Ф. Так как самое нижнее поле всегда предназначено для возвращаемого П-Ф значения, то поля ввода для дополнительных операторов открываются с помощью

щелчка на кнопке Add line палитры **ПРОГРАММИРОВАНИЕ** (см. рис. 4.1.3). При этом поле ввода добавляется внизу выделенного к этому моменту оператора. Для удаления того или иного оператора или поля ввода из тела П-Ф нужно заключить его в выделяющую рамку и нажать клавишу [Delete];

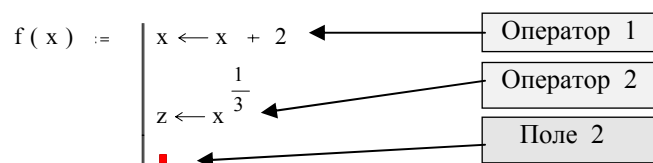


Рис. 4.1.3. Добавление операторов в тело подпрограммы-функции

- заполнить самое нижнее поле ввода (поле 2), введя туда выражение, определяющее возвращаемое через имя П-Ф (см. рис. 4.1.4).

$$f(x) := \begin{array}{|l} x \leftarrow x + 2 \\ \\ z \leftarrow x^{\frac{1}{3}} \\ z \end{array}$$

Рис. 4.1.4. Окончательная структура подпрограммы-функции

В приведенном примере формальным параметром является простая переменная x , тело П-Ф включает два локальных оператора присваивания (см. следующий пункт) и значение переменной z , которая определяет возвращаемый через имя функции результат выполнения П-Ф.

Замечание 4.1.2. Если результатом работы П-Ф являются несколько величин, то из них в теле П-Ф необходимо сформировать массив и его имя поместить в последней строке тела П-Ф.

Локальный оператор присваивания. Для задания внутри программы значения какой-либо переменной используется так называемый *локальный оператор присваивания*, имеющий вид:

$\langle \text{имя переменной} \rangle \leftarrow \langle \text{выражение} \rangle .$

Внимание! Использование «обычного» оператора присваивания в теле П-Ф (на экране отображается $:=$) приводит к синтаксической ошибке.

4.2. Обращение к подпрограмме-функции MathCAD

Для выполнения П-Ф необходимо обратиться к ее имени с указанием *списка фактических параметров* (если в описании программы присутствует список формальных параметров), т.е.:

$\langle \text{имя П-Ф} \rangle (\langle \text{список фактических параметров} \rangle) .$

Фактические параметры указывают, при каких конкретных значениях осуществляются вычисления в теле программы. Фактические параметры отделяются друг от друга запятой.

Очевидно, что между фактическими и формальными параметрами должно быть *соответствие по количеству, порядку следования и типу*. Последнее соответствие означает:

- если формальным параметром является простая переменная, то в качестве фактического может использоваться константа, переменная, арифметическое выражение;
- если формальным параметром является вектор или матрица, то фактическим должен быть вектор или матрица;
- если формальным параметром является имя встроенной функции или другой программы, то и фактическим параметром должен являться тот же объект.

Замечание 4.2.1. Обращение к П-Ф должно находиться после ее описания, и *к моменту обращения фактические параметры должны быть определены*.

Пример 4.2.1. Обращение к программе $f(x)$, приведенной на рис. 4.1.4, может иметь следующий вид:

$$\begin{aligned}
 x &:= 2 & f(x) &= 1.587 & f(-3.23) &= 0.536 + 0.928i \\
 z &:= f(x + 2.5) & z &= 1.866
 \end{aligned}$$

Заметим, что переменная z никак не связана с «локальной» переменной z , используемой внутри тела П-Ф. ♦

Замечание 4.2.2. Передать данные внутрь П-Ф можно используя внутри подпрограммы переменные, определенные до описания П-Ф (см. пример на рис. 4.2.1).

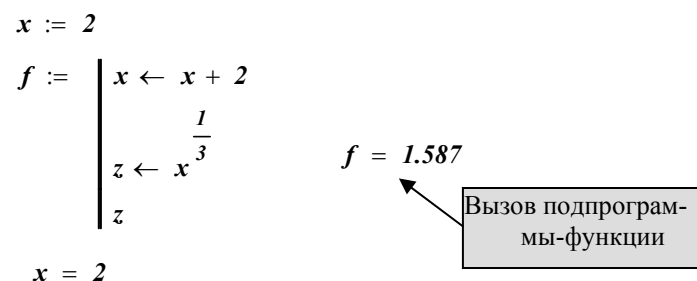


Рис. 4.2.1. Подпрограмма-функция без формальных параметров

Хотя значение переменной x изменилось внутри П-Ф, вне описания П-Ф эта переменная сохранила свое прежнее значение.

Замечание 4.2.3. Имена фактических параметров при вызове П-Ф могут либо совпадать, либо не совпадать с именами ее формальных параметров.

ТЕМА 5. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ В ПОДПРОГРАММЕ-ФУНКЦИИ MathCAD

В этой теме будут рассмотрены конструкции для программирования линейных, разветвляющихся и циклических алгоритмов в подпрограмме-функции MathCAD.

5.1. Программирование линейных алгоритмов в подпрограмме-функции

Напомним, что под линейным алгоритмом понимается вычислительный процесс, в котором необходимые операции выполняются строго последовательно (см. п. 3.1). Операторы, реализующие этот алгоритм, в теле П-Ф также размещаются последовательно и выполняются все, начиная с первого и заканчивая последним.

Пример 5.1.1. Оформи́м в виде П-Ф вычисление корней квадратного уравнения $ax^2 + bx + c = 0$ по формуле

$$x_{1,2} = \frac{-b \mp (b^2 - 4ac)^{1/2}}{2a}.$$

Описание П-Ф *root_poly2* и обращение к ней приведено на рис. 5.5.1. П-Ф имеет три входных формальных параметра – коэффициенты квадратного уравнения. Выходом является вектор с двумя компонентами. Заметим, что величины x_1 , x_2 являются простыми переменными, а не элементами одномерного массива. Поэтому нижние индексы в их именах вводятся после нажатия клавиши $[.]$ – «десятичная точка». Подпрограмма-функция реализует **линейный алгоритм** – все операторы выполняются всегда строго последовательно. ♦

$$\begin{aligned}
 \text{root_poly2}(a, b, c) &:= \begin{cases} d \leftarrow \sqrt{b^2 - 4 \cdot a \cdot c} \\ x_1 \leftarrow \frac{-b + d}{2 \cdot a} \\ x_2 \leftarrow \frac{-b - d}{2 \cdot a} \\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \end{cases} \\
 z &:= \text{root_poly2}(12, 7, 10) \quad z = \begin{pmatrix} -0.292 + 0.865i \\ -0.292 - 0.865i \end{pmatrix}
 \end{aligned}$$

Рис. 5.1.1. Пример программирования линейного алгоритма

5.2. Программирование разветвляющихся алгоритмов в подпрограмме-функции

Напомним, что в разветвляющихся алгоритмах присутствует несколько ветвей вычислительного процесса. Выбор конкретной ветви зависит от выполнения (или не выполнения) заданных условий на значения переменных алгоритма.

Для программирования разветвляющихся алгоритмов в подпрограмме-функции MathCAD можно использовать *условную функцию и условный оператор if*. Используя эти конструкции, можно «изменить» последовательное выполнение операторов. Условная функция *if* была рассмотрена в п. 3.2, поэтому перейдем к изучению условного оператора *if*.

Условный оператор. Этот оператор используется *только в теле П-Ф* и для его ввода необходимо щелкнуть на кнопке **if** палитры **ПРОГРАММИРОВАНИЕ**. На экране появляется конструкция с двумя полями ввода, изображенная на рис. 5.2.1.

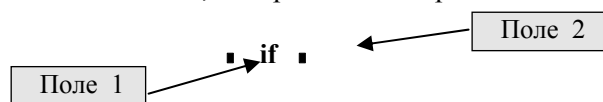


Рис. 5.2.1. Структура условного оператора *if*

В поле 2 вводится логическое выражение УСЛ (в простейшем случае это выражение отношений). В поле 1 вводится конструкция ВЫР1 (как правило, арифметическое выражение), которая выполняется, если проверяемое логическое выражение принимает значение 1. Если УСЛ = 0, то ВЫР1 не выполняется. Это соответствует условной структуре, называемой **ЕСЛИ – ТО**.

Для получения условной структуры **ЕСЛИ – ТО – ИНАЧЕ** используется оператор *otherwise*, вводимый с палитры **ПРОГРАММИРОВАНИЕ**, в поле которого размещается конструкция ВЫР2, которая выполняется, если проверяемое логическое выражение принимает значение 0 (см. рис. 5.2.2). Оператор *otherwise* непосредственно следует после условного оператора *if*.

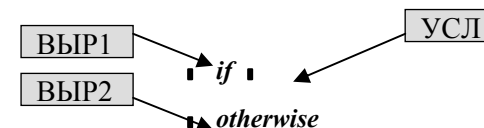


Рис. 5.2.2. Реализация структуры **ЕСЛИ – ТО – ИНАЧЕ**

Для ввода ВЫР2 в поле оператора *otherwise* необходимо:

- выделить поле, стоящее после оператора *if*;
- щелкнуть на кнопке *otherwise* палитры

ПРОГРАММИРОВАНИЕ;

- в появившееся поле оператора *otherwise* ввести необходимую конструкцию ВЫР2.

Пример 5.2.1. Составим описание П-Ф, вычисляющей функцию $y(x)$, заданную в примере 3.2.1. Описание и вызов П-Ф приведены на рис. 5.2.3. ♦

$$y(x) := \begin{cases} x^2 & \text{if } x \leq 0 \\ \sqrt{x} & \text{otherwise} \end{cases}$$

$$y(2) = 1.414 \quad y(-4) = 16$$

Рис. 5.2.3. Реализация разветвляющегося алгоритма примера 3.2.1

Таким образом, конструкция, стоящая перед словом *otherwise* выполняется только в том случае, если не выполнено заданное перед этим условие. В программе можно использовать несколько следующих друг за другом условных операторов с одним оператором *otherwise*.

Пример 5.2.2. Составим описание П-Ф для вычисления переменной $z(t)$ по формуле

$$z(t) = \begin{cases} t^3, & t < -3; \\ \ln(t), & t > 4; \\ t^2, & -3 \leq t \leq 4. \end{cases}$$

Описание П-Ф и ее вызов приведены на рис. 5.2.4.

$$z(t) := \begin{cases} t^3 & \text{if } t \leq -3 \\ t^2 & \text{if } -3 \leq t \leq 4 \\ \ln(t) & \text{otherwise} \end{cases}$$

$$z(-8) = -512 \quad z(2) = 4 \quad z(7) = 1.946$$

Рис. 5.2.4. Реализация разветвляющегося алгоритма примера 5.2.2

Из описания видно, что функция $z(t)$ получит значение $\ln(t)$ только тогда, когда не выполняются условия, записанные в двух вышестоящих строках тела П-Ф.

Внимание! Если в строке 3 ввести просто $\ln(t)$, то это выражение *будет вычисляться всегда* вне зависимости от выполнения заданных выше условных операторов.

Задание 5.2.1. Составьте описания П-Ф, реализующих следующие разветвляющиеся алгоритмы:

$$\text{А. } y = \begin{cases} \frac{1}{2\sqrt{x}}, & \text{если } x \geq 1; \\ \frac{1}{3}\sqrt[3]{x}, & \text{если } 0 \leq x \leq 1; \\ \frac{1}{4}\sqrt[4]{|x|}, & \text{если } x < 0. \end{cases} \quad \text{Б. } \mu(x, \varepsilon) = \begin{cases} \frac{1}{2\sqrt{x+1}}, & \text{если } |x-y| < \varepsilon; \\ \frac{1}{3}\sqrt[3]{x+y}, & \text{если } |x-y| \geq \varepsilon, \end{cases}$$

где $y = |x|$.

Рассмотрим использование условного оператора *if* при программировании более сложных разветвляющихся алгоритмов. Выделим следующие варианты.

Вариант 1. При выполнении заданного условия УСЛ необходимо выполнить несколько конструкций MathCAD.

В этом случае необходимо выделить *поле* 1 условного оператора (см. рис. 5.2.2), щелкнуть на кнопке Add line палитры **ПРОГРАММИРОВАНИЕ** нужное число раз и заполнить появившиеся поля (рис. 5.2.5).

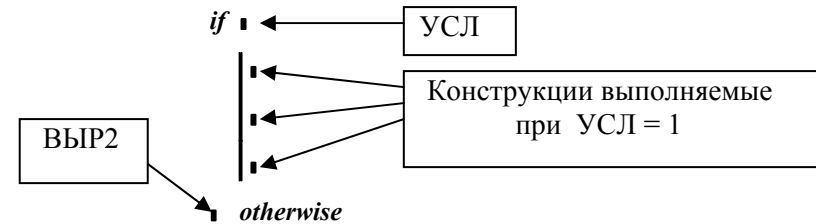


Рис. 5.2.5. Вариант 1 реализации условного оператора

Вариант 2. При невыполнении заданного условия УСЛ необходимо выполнить несколько конструкций MathCAD.

В этом случае необходимо выделить *поле* оператора *otherwise*, щелкнуть на кнопке Add line палитры **ПРОГРАММИРОВАНИЕ** нужное число раз и заполнить появившиеся поля (рис. 5.2.6).

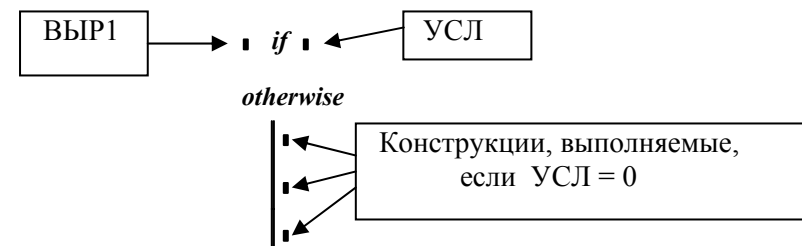


Рис. 5.2.6. Вариант 2 реализации условного оператора

Пример 5.2.3. Составьте описание П-Ф, вычисляющей значения двух полиномов $x(t)$, $y(t)$ нулевой или первой степени. Порядок полиномов задается переменной n . Если $n < 0$ или $n > 1$, то значения полиномов равны 0. Описание П-Ф приведено на рис. 5.2.7. Обратите внимание, что коэффициенты полиномов передаются через массивы a , b , а в полях условных операторов находятся не арифметические выражения, а локальные операторы присваивания. ♦

$$\begin{array}{l}
 \text{poly}(n, a, b, t) := \text{if } n = 0 \\
 \quad \left| \begin{array}{l} x \leftarrow a_0 \\ y \leftarrow b_0 \end{array} \right. \\
 \text{if } n = 1 \\
 \quad \left| \begin{array}{l} x \leftarrow a_0 + a_1 \cdot t \\ y \leftarrow b_0 + b_1 \cdot t \end{array} \right. \\
 \text{otherwise} \\
 \quad \left| \begin{array}{l} x \leftarrow 0 \\ y \leftarrow 0 \end{array} \right. \\
 \quad \left(\begin{array}{c} x \\ y \end{array} \right)
 \end{array}$$

Рис. 5.2.7. Реализация алгоритма примера 5.2.3

Пример 5.2.4. Даны два числа x, y . Составить описание П-Ф, которая переменной x присваивает максимальное значение из этих двух чисел, а y – минимальное. Описание приведено на рис. 5.2.8. Выходным параметром является массив v , нулевой элемент которого содержит новое значение x , а первый элемент – новое значение y . Вызов этой П-Ф показан на рис. 5.2.9. ♦

$$\begin{array}{l}
 \text{arrangement}(x, y) := \text{if } x > y \\
 \quad \left| \begin{array}{l} v_0 \leftarrow x \\ v_1 \leftarrow y \end{array} \right. \\
 \text{otherwise} \\
 \quad \left| \begin{array}{l} v_0 \leftarrow y \\ v_1 \leftarrow x \end{array} \right. \\
 \quad v
 \end{array}$$

Рис. 5.2.8. Реализация алгоритма примера 5.2.4

$$x := 2 \quad y := 8 \quad \left(\begin{array}{c} x \\ y \end{array} \right) := \text{arrangement}(x, y) \quad x = 8 \quad y = 2$$

Рис. 5.2.9. Вызов подпрограммы-функции *arrangement*

Задание 5.2.2. Даны три числа a, b, c . Составить П-Ф, реализующую следующий алгоритм. Если $a \leq b \leq c$, то все числа заменить их квадратами, если $a > b > c$, то каждое число заменить максимальным значением из этих трех чисел, в противном случае — сменить знаки у чисел.

Задание 5.2.3. Координаты точки на плоскости задаются двумя числами x, y . Составить П-Ф, вычисляющую номер четверти на плоскости, в которую попала точка.

Задание 5.2.4. Длина сторон треугольника задается числами a, b, c . Составить П-Ф, вычисляющую значение целой переменной n по следующему правилу: $n = 3$, если три стороны равны; $n = 2$, если любые две стороны равны; $n = 1$, если все три стороны имеют разную длину.

5.3. Программирование циклических алгоритмов в подпрограмме-функции

Напомним, что циклические алгоритмы (циклы) содержат повторяющиеся вычисления, зависящие от некоторой переменной. Такая переменная называется *параметром цикла*, а сами повторяющиеся вычисления составляют *тело цикла*. Циклы можно условно разделить на две группы:

- циклы типа арифметической прогрессии;
- итерационные циклы.

Особенности этих циклов были рассмотрены в п. 3.3.

Программирование цикла типа арифметической прогрессии. Для программирования таких циклов используется оператор цикла *for* (часто называемый оператором цикла с параметром). Для ввода такого оператора необходимо выполнить следующие действия:

- щелкнуть на кнопке *for* палитры

ПРОГРАММИРОВАНИЕ. На экране появятся поля ввода, изображенные на рис. 5.3.1;

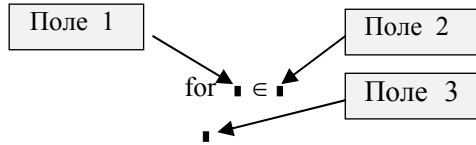


Рис. 5.3.1. Поля оператора цикла *for*

- в поле ввода 1 ввести имя переменной, являющейся параметром цикла;
- в поле 2 — закон изменения параметра цикла, используя для этого описание дискретной переменной или *описание массива*;
- в поле 3 — операторы, составляющие тело цикла. Если одной строки недостаточно, то дополнительные поля ввода (дополнительные строки) создаются щелчком на кнопке Add line палитры **ПРОГРАММИРОВАНИЕ**, и тогда слева от тела цикла появляется вертикальная черта, охватывающая тело цикла.

Пример 5.3.1. Составить описание П-Ф, реализующей алгоритм примера 3.3.1.

Описание П-Ф и ее вызов приведены на рис. 5.3.2. Заметим, что значение системной переменной *ORIGIN* (начальное значение индексного выражения) задается равным 1. ♦

$$form_vec1(n) := \left| \begin{array}{l} \text{for } i \in 1..n \\ z_i \leftarrow \frac{1}{i^2 + 1} \\ z \end{array} \right. \quad form_vec1(5) = \begin{pmatrix} 0.5 \\ 0.2 \\ 0.1 \\ 0.059 \\ 0.038 \end{pmatrix}$$

Рис. 5.3.2. Подпрограмма-функция формирования вектора

Пример 5.3.2. Для x меняющегося от -2 до 2 с шагом 0.5 вычислить значение $f(x) = e^{-x} \cdot \cos(2x)$ и сформировать из этих значений вектор y , т.е. $y_1 = f(-2)$, $y_2 = f(-1.5)$ и т.д.

В этом примере количество повторений тела цикла определяется по формуле

$\left\lceil \frac{x_k - x_0}{d} \right\rceil + 1$, где x_k , x_0 — конечное и начальное значения параметра цикла, d — шаг его изменения. Подставив значения, получаем $(2 - (-2))/0.5 + 1 = 9$. Следовательно, сформированный вектор y будет содержать 9 элементов. Необходимо отметить, что алгоритм формирования содержит два параметра цикла: первый параметр — переменная x , определяющая текущее значение аргумента функции $f(x)$; второй — это переменная i , определяющая текущее значение индекса y элемента формируемого вектора. Но между этими двумя параметрами имеется однозначное соответствие, поэтому цикл нужно организовывать только по одному параметру, например, по переменной x , а значение второго параметра будет вычисляться в теле цикла.

Описание П-Ф и ее вызов приведены на рис. 5.3.3. Видно, что в теле цикла выполняется два оператора. Первый оператор формирует элемент массива y , а второй изменяет на 1 значение индекса. ♦

$$form_vec2(x_0, x_k, d) := \left| \begin{array}{l} i \leftarrow 1 \\ \text{for } x \in x_0, x_0 + d..x_k \\ y_i \leftarrow e^{-x} \cdot \cos(2 \cdot x) \\ i \leftarrow i + 1 \\ y \end{array} \right. \quad z := form_vec2(0, 0.8, 0.1)$$

$$z^T = (1 \quad 0.887 \quad 0.754 \quad 0.611 \quad 0.467 \quad 0.328 \quad 0.199 \quad 0.084 \quad -0.013)$$

Рис. 5.3.3. Формирование вектора примера 5.3.2

Пример 5.3.3. Составить описание П-Ф, где значения параметра цикла задаются вектором.

На рис. 5.3.4 приведено описание такой П-Ф.

Обращение
к П-Ф

$sum_vec_3 :=$

$S \leftarrow 0$ $for\ i \in \begin{pmatrix} 2 \\ 5 \\ 7 \end{pmatrix}$ $S \leftarrow S + i$ S

$sum_vec_3 =$ ■

Рис. 5.3.4. Описание П-Ф примера 5.3.3

Задание 5.3.1. Составьте описание П-Ф формирования вектора у примера 5.3.2, приняв в качестве параметра цикла переменную i .

Программирование итерационных циклов. Для программирования таких циклов используется оператор цикла *while*. Для ввода этого оператора необходимо выполнить следующие действия:

- щелкнуть на кнопке **while** палитры **ПРОГРАММИРОВАНИЕ**. На экране появляются элементы, показанные на рис. 5.3.5;

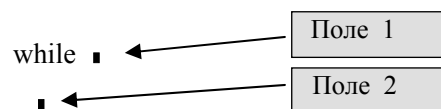


Рис. 5.3.5. Структура оператора цикла *while*

- в поле 1 ввести условие выполнения цикла;
- в поле 2 ввести операторы тела цикла. В теле цикла должны присутствовать операторы, которые *могут изменить значение условия цикла*, иначе цикл будет продолжаться бесконечно.

Оператор цикла *while* выполняется следующим образом: обнаружив оператор *while*, MathCAD проверяет указанное в операторе условие. Если оно равно 1 (т.е. выполняется), то выполняется тело

цикла, и снова проверяется условие. Если условие принимает значение 0, то цикл заканчивается.

Пример 5.3.4. Составим П-Ф, реализующую итерационную процедуру приближенного вычисления корня квадратного, описанную в примере 3.3.2. Описание П-Ф приведено на рис. 5.3.6.

Как видно из текста, П-Ф нет необходимости хранить в памяти все приближенные решения $x_0, x_1, x_2, \dots$ (как это делалось при использовании функции *until* — см. рис. 3.3.3). Достаточно хранить предыдущее («старое») значение (обозначим его как xc) и последующее («новое») значение xn .

К сожалению, организация итерационного цикла с помощью оператора *while* без дополнительных средств контроля может привести к заикливанию, т.е. повторению тела цикла «бесконечное» число раз. Например, задав при обращении к П-Ф $\varepsilon < 0$, получаем заикливание.

$sroot(a, \varepsilon) :=$

$xn \leftarrow a$ $while\ xn^2 - a > \varepsilon$ <table border="0"> <tr> <td style="border-left: 1px solid black; padding-left: 10px;"> $xc \leftarrow xn$ $xc \leftarrow \frac{xn + \frac{a}{xn}}{2}$ $xn \leftarrow xc$ xn </td> </tr> </table>	$xc \leftarrow xn$ $xc \leftarrow \frac{xn + \frac{a}{xn}}{2}$ $xn \leftarrow xc$ xn
$xc \leftarrow xn$ $xc \leftarrow \frac{xn + \frac{a}{xn}}{2}$ $xn \leftarrow xc$ xn	

Вызов подпрограммы-функции

$sroot(9, 0.1) = 3.00009$

$sroot(9, 0.0001) = 3.00000$

Рис. 5.3.6. Реализация итерационного цикла примера 5.3.4

Поэтому в MathCAD имеется специальный оператор *break*, который позволяет выйти из цикла или приостановить исполнение программы при выполнении заданного в операторе *break* условия. Для ввода оператора *break* необходимо щелкнуть на кнопке **break** панели **ПРОГРАММИРОВАНИЕ** (нельзя вводить этот оператор с клавиатуры по символам). Оператор *break* используется в левом поле ввода условного оператор *if*, а в правом размещается условие,

при выполнении которого происходит прекращение работы цикла или программы. Поэтому *первоначально вводится оператор if*, а затем заполняются поля этого оператора. Следующий пример показывает написание подпрограммы без «зацикливания» с использованием оператора *break*.

Пример 5.3.5. Составим П-Ф, реализующую итерационную процедуру вычисления корня квадратного без «зацикливания». Описание П-Ф приведено на рис. 5.3.7.

В этой подпрограмме число повторений тела ограничено 10000. Если за это число итераций приближенное значение корня с заданной точностью не найдено, то параметр *ierr* – индикатор ошибки получает значение 1, что говорит об ошибке в вычислительном процессе. Для передачи двух вычисленных значений *xn*, *ierr* используется вектор. Если *ierr* = 1, то переменной *xn* присваивается значение строковой константы “not solve” для привлечения внимания пользователя к ошибочному завершению работы П-Ф. Заметим, что после обращения к П-Ф необходимо обязательно проверить значение индикатора ошибки *ierr* и, в зависимости от его значения, либо продолжить работу с вычисленным значением корня квадратного, либо прекратить дальнейшие вычисления и определить причину появления ошибки. ♦

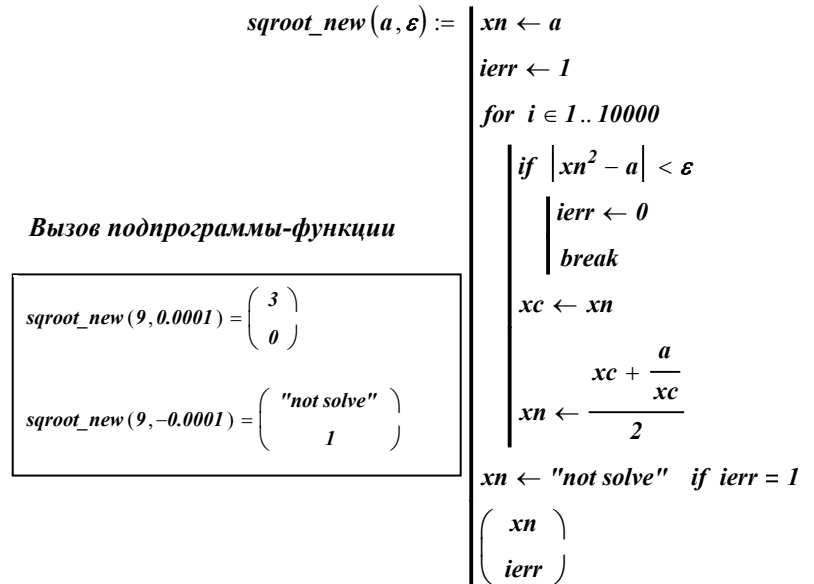


Рис. 5.3.7. Реализация итерационного цикла без «зацикливания»

Пример 5.3.6. Составить П-Ф, осуществляющую суммирование ряда с бесконечным числом слагаемых. Накопление суммы прекращается, как только очередное слагаемое по абсолютной величине становится меньше заданной погрешности ε .

Описание П-Ф и ее вызов показаны на рис. 5.3.8. Заметим, что вторым формальным параметром является имя функции пользователя, определяющей зависимость величины члена ряда от его номера. При вызове этот формальный параметр заменяется фактическим – именем функции пользователя, описанной до обращения к П-Ф. ♦

$sum_conv(\varepsilon, a) :=$

```

sum ← 0
ierr ← 1
for i ∈ 1..10000
    if |a(i)| < ε
        ierr ← 0
        break
    sum ← sum + a(i)
sum ← "not convergence" if ierr = 1
( sum
  ierr )

```

Описание слагаемых
суммируемых рядов

$$b(i) := \frac{1}{i^2} \quad d(i) := \frac{1}{\sqrt{i}}$$

Вызов подпрограммы-функции

$sum_conv(0.00001, b) = \begin{pmatrix} 1.64177 \\ 0 \end{pmatrix}$

$sum_conv(0.00001, d) = \begin{pmatrix} "not convergence" \\ 1 \end{pmatrix}$

Рис. 5.3.8. Реализация итерационного цикла примера 5.3.6

Программирование двойных циклов. Напомним, что элементы матриц (двумерных массивов) имеют два индекса: первый индекс определяет номер строки, а второй – номер столбца, на пересечении которых находится указанный элемент. Например, запись $A_{2,4}$ – означает обращение к элементу, стоящему на пересечении второй строки и четвертого столбца. Поэтому в общем случае для работы с матрицами необходимо использовать так называемый двойной цикл, у которого тело одного цикла, называемого внутренним, «вкладывается» в тело другого – «внешнего» цикла. Каждый из этих циклов имеет свой параметр: параметр внешнего цикла и параметр внутреннего цикла. Следовательно, для организации такого двойного цикла необходимо использовать два оператора цикла, один из которых

размещен в теле другого (различные варианты вложений операторов цикла приведены на рис. 5.3.9).

```

for i ∈ 1..n      for i ∈ 1..n      while i ≤ n
  for j ∈ 1..m    while j ≤ m        for j ∈ 1..m
    B[i,j] ← 1    B[i,j] ← 1        B[i,j] ← 1

```

Рис. 5.3.9. Варианты вложений операторов цикла

Пример 5.3.7. Составить описание П-Ф формирующей матрицу по следующему правилу:

$$B_{i,j} = \frac{1}{i+j+1}, \quad i=1,\dots,n; \quad j=1,\dots,m.$$

Описание и вызов П-Ф приведены на рис. 5.3.9. В этой П-Ф параметром внешнего цикла является переменная i , а параметром внутреннего — переменная j . ♦

$ORIGIN := 1$ $form_mat1(n, m) :=$

```

for i ∈ 1..n
  for j ∈ 1..m
    B[i,j] ← 1/(i+j+1)
B

```

$form_mat1(4, 2) = \begin{pmatrix} 0.333 & 0.25 \\ 0.25 & 0.2 \\ 0.2 & 0.167 \\ 0.167 & 0.143 \end{pmatrix}$

Рис. 5.3.10. Реализация двойного цикла примера 5.3.7

Пример 5.3.8. Дана матрица A размерности $n \times m$. Составить описание П-Ф, вычисляющей индексы и значение элемента матрицы, который по модулю максимально близок к заданному числу c . Описание П-Ф и ее вызов приведены на рис 5.3.10. Напомним, что системная переменная $ORIGIN = 1$. ♦

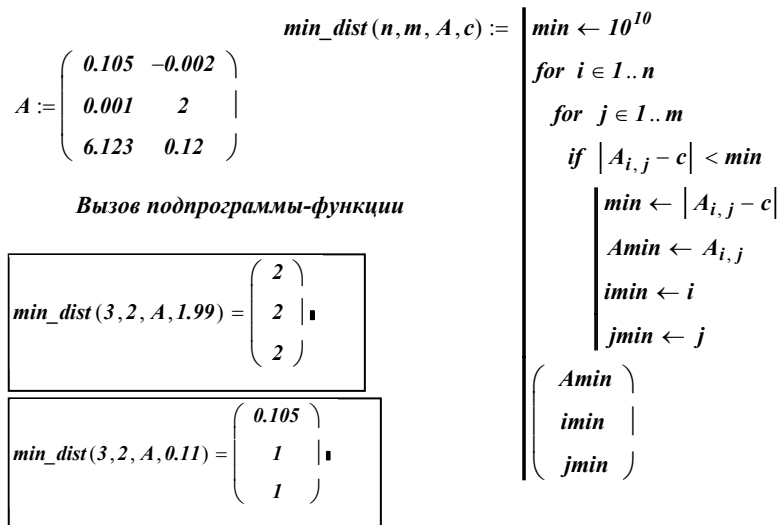


Рис. 5.3.11. Реализация двойного цикла примера 5.3.8

Дополнительные операторы, используемые при программировании циклов. Рассмотрим несколько операторов, которые могут быть полезны при программировании циклов.

Оператор *continue*. Обычно используется для продолжения выполнения цикла путем возврата в начало тела цикла. Следующий пример поясняет работу этого оператора.

Пример 5.3.9. Составить описание П-Ф, формирующей новый вектор из положительных проекций исходного вектора. Описание приведено на рис. 5.3.12. В теле П-Ф используется функция *last(v)*, определяющая значение индекса последнего элемента массива *v*. Если очередной элемент *v_i* не больше нуля, то пропускаются все нижележащие операторы тела цикла (в нашем случае — два оператора, формирующие очередную проекцию вектора *w*), и тело цикла повторяется при новом значении параметра цикла *i*.

Сравнение исходного вектора *x* с результатом работы П-Ф — вектором *z* показывает правильность работы запрограммированного алгоритма. ♦

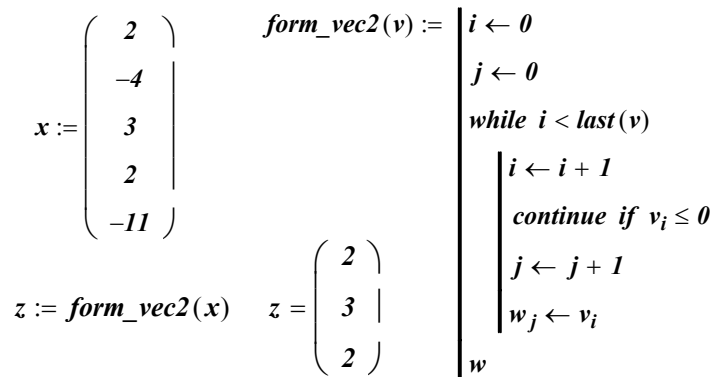


Рис. 5.3.12. Реализация алгоритма примера 5.3.9

Оператор *return*. Прерывает выполнение П-Ф и возвращает значение операнда, стоящего в поле 1 (см. рис. 5.3.13).

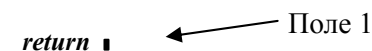


Рис. 5.3.13. Структура оператора *return*

Следующий пример поясняет работу этого оператора.

Пример 5.3.10. Составить описание П-Ф, находящей первую положительную проекцию исходного вектора. Возможны два варианта программной реализации алгоритма решения этой задачи (приведены на рис. 5.3.14). Вариант В представляется более простым и «элегантным». ♦

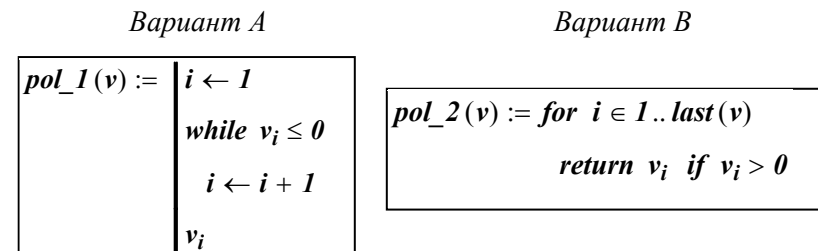


Рис. 5.3.14. Два варианта реализации алгоритма примера 5.3.10

Оператор *on error*. Этот оператор является обработчиком возникающих при выполнении тех или иных вычислений ошибок и записывается в виде:

$$< \text{конструкция 1} > \text{ on error } < \text{конструкция 2} >.$$

Оператор выполняется следующим образом. Если при выполнении <конструкция 2> возникает ошибка, то выполняется <конструкция 1>. Если ошибка не возникает, то выполняется <конструкция 2>.

Пример 5.3.11. Используем оператор *on error* для предотвращения появления ошибки «деление на ноль» при вычислении функции *angl(x,y)*. Фрагмент представлен на рис. 5.3.15. |

$$\begin{aligned} \text{angl}(x,y) &:= \frac{x}{y} & \text{angl}(2,0) &= \text{Ошибка} \\ & & & \text{«деление на ноль»} \\ \text{angl}(x,y) &:= 10^{10} \text{ on error } \frac{x}{y} \\ \text{angl}(2,0) &= 1 \times 10^{10} & \text{angl}(2,2) &= 1 \end{aligned}$$

Рис. 5.3.15. Использование функции *on error* в примере 5.3.11

Функция *error*. Используется для вывода диагностических сообщений при возникновении в вычислениях ошибки и записывается в виде:

$$\text{error} ("< \text{диагностическое сообщение пользователя} >").$$

Имя функции вводится с клавиатуры. Функция используется в левом поле условного оператора *if*, как показано в следующем примере.

Пример 5.3.12. Запрограммируем вывод диагностического сообщения при попытке спроецировать вектор *v* на нулевой вектор *w*. Описание П-Ф и ее вызовы приведены на рис. 5.3.16.

$$\text{proj}(v,w) := \begin{cases} \text{error}("You \text{ cannot onto the } 0 \text{ vector}") & \text{if } |w| < 10^{-10} \\ \frac{w}{|w|} \cdot (v \cdot w) & \text{otherwise} \end{cases}$$

$$x := \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix} \quad w := \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix} \quad \text{proj}(x,w) = \begin{pmatrix} 16.036 \\ 10.69 \\ 5.345 \end{pmatrix}$$

Рис. 5.3.16. Использование в П-Ф оператора *error*

Следует заметить, что взятое в кавычки диагностическое сообщение появится на экране только после щелчка мышью на выделенном красным цветом обращении к П-Ф, при выполнении которой произошла ошибка (см. рис. 5.3.17).

$$x := \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix} \quad w := \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \boxed{\text{proj}(x,w) = \dots}$$

You cannot onto the 0 vector

Рис. 5.3.17. Применение функции *error* в примере 5.3.12

ТЕМА 6. ПРОГРАММИРОВАНИЕ ТИПОВЫХ ЗАДАЧ В ПОДПРОГРАММАХ-ФУНКЦИЯХ MathCAD

В этой теме будет рассмотрено программирование некоторых типовых задач, которые могут входить в состав более сложных алгоритмов.

6.1. Программирование разветвляющихся алгоритмов

Будет рассмотрено несколько примеров программирования разветвляющихся алгоритмов, в некоторых из которых используются встроенные функции MathCAD.

Пример 6.1.1. Составить описание П-Ф, формирующей новый вектор из первых n элементов исходного вектора. Описание П-Ф и ее вызов приведены на рис. 6.1.1. В этой П-Ф использованы две функции:

- $length(v)$ – вычисляет общее число элементов массива v ;
- $submatrix(A, i1, i2, j1, j2)$ – формирует новую матрицу, состоящую из элементов исходной матрицы A , находящихся на пересечении строк с индексами с $i1$ по $i2$ и столбцов с индексами с $j1$ по $j2$.

Значение системной переменной $ORIGIN = 0$. ♦

$$form_vec3(n, v) := \begin{cases} N \leftarrow length(v) \\ v & \text{if } n \geq N \\ submatrix(v, 0, n-1, 0, 0) & \text{if } n \geq 1 \text{ otherwise} \end{cases}$$

**Обращение
к подпрограмме-функции**

$$x := \begin{pmatrix} 2 \\ 5 \\ 7 \\ 9 \\ 2 \\ 1 \end{pmatrix} \quad form_vec3(3, x) = \begin{pmatrix} 2 \\ 5 \\ 7 \end{pmatrix} \quad form_vec3(10, x) = \begin{pmatrix} 2 \\ 5 \\ 7 \\ 9 \\ 2 \\ 1 \end{pmatrix}$$

Рис. 6.1.1. Реализация алгоритма примера 6.1.1

Пример 6.1.2. Составить описание П-Ф, вычисляющей значения прямоугольных сигналов с периодом 2 и амплитудой +1 и -1 (график приведен на рис. 6.1.2). Описание П-Ф приведено на рис. 6.1.2. Здесь использовались функции:

- $mod(y, modul)$ – вычисляет остаток от деления значения вещественной переменной x на $modul$;
- $ceil(y)$ – вычисляет наименьшее целое, большее или равное y . ♦

Пример 6.1.3. Составить описание П-Ф, вычисляющей прямоугольный сигнал амплитудой A на интервале $[a, b]$. Вне этого интервала значение равно B .

Описание П-Ф имеет вид:

$$f(x, a, b, A, B) := B + A \cdot (\Phi(x - a) - \Phi(x - b)),$$

а график сигнала при $a = 2, b = 4, A = 6, B = -2$ приведен на рис. 6.1.3. Здесь использовалась функция $\Phi(x)$, равная нулю при $x < 0$ и 1 для всех других значений аргумента. ♦

$$f(x) := \begin{cases} 1 & \text{if } mod(ceil(x), 2) = 1 \\ -1 & \text{if } mod(ceil(x), 2) = 0 \end{cases}$$

$$x := 0, 0.05..6$$

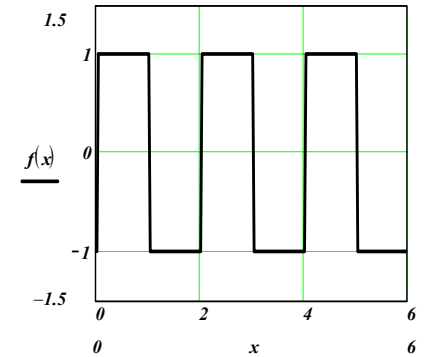


Рис. 6.1.2. Реализация алгоритма примера 6.1.2

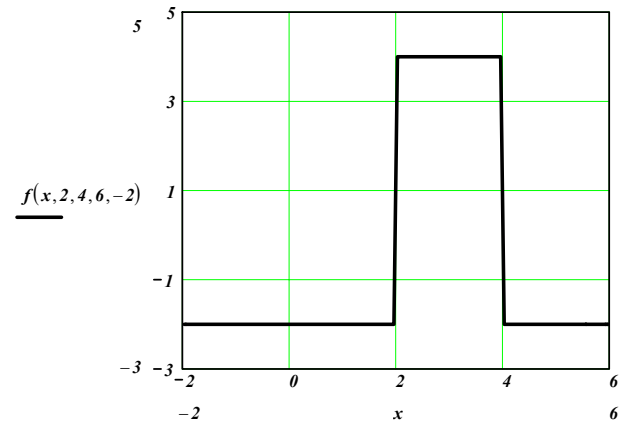


Рис. 6.1.3. График сигнала примера 6.1.3

Пример 6.1.4. Составить фрагмент документа MathCAD для вычисления значения $z = x_{\max} + y_{\text{pol}}$, где $x_{\max}$ — максимальный по модулю корень уравнения $x^2 + 5.45x - 8.12 = 0$; y_{pol} — любой положительный корень уравнения $y^2 + 25.3y - 9.5 = 0$. Фрагмент приведен на рис. 6.1.4. В этом фрагменте вызывается П-Ф вычисления корней квадратного уравнения, описание которой приведено на рис. 5.1.1 (см. пример 5.1.1). ♦

$$\begin{aligned}
 X &:= \text{root_poly2}(1, 5.45, -8.12) & X &= \begin{pmatrix} 1.218 \\ -6.668 \end{pmatrix} \\
 Y &:= \text{root_poly2}(1, 25.3, -9.5) & Y &= \begin{pmatrix} 0.37 \\ -25.67 \end{pmatrix} \\
 x_{\max} &:= \text{if}(|X_0| > |X_1|, X_0, X_1) & x_{\max} &= -6.668 \\
 y_{\text{pol}} &:= \text{if}(Y_0 > 0, Y_0, Y_1) & y_{\text{pol}} &= 0.37 \\
 z &:= x_{\max} + y_{\text{pol}} & z &= -6.298
 \end{aligned}$$

Корни уравнений

Рис. 6.1.4. Реализация алгоритма примера 6.1.4

Задание 6.1.1. Составить описание П-Ф, вычисляющей значения прямоугольных сигналов такой же формы, как на рис. 6.1.2, но с периодами: а) равным 1; б) равным 8. Построить графики этих сигналов (см. пример 6.1.2). ●

Задание 6.1.2. Составить описание П-Ф, реализующей алгоритм, приведенный на рис. 6.1.4, а. Составить фрагмент для проверки правильности работы П-Ф. ●

Задание 6.1.3. Составить описание П-Ф, вычисляющей площадь S геометрической фигуры по ее номеру $n=1, 2, 3$. Расчетные соотношения приведены на рис 6.1.4, б, h, a, b, r, n — исходные данные. ●

$$\begin{aligned}
 &y = \max(a, b), \\
 \text{где} \\
 &a = \sqrt[3]{2\sin^2 x + 3\cos(x^2 + 2)} \\
 &b = \sqrt{\frac{e^{\sin 2x} - e^{\cos 3x}}{\ln x^2 + d^2}}
 \end{aligned}$$

$$S = \begin{cases} ab, & \text{если } n = 1; \\ \pi r^2, & \text{если } n = 2; \\ h \frac{a+b}{2}, & \text{если } n = 3. \end{cases}$$

а б
Рис. 6.1.4. Разветвляющиеся алгоритмы

6.2. Программирование циклов типа арифметической прогрессии

В этом параграфе будут рассмотрены четыре типа циклических алгоритмов, входящих в более сложные алгоритмы. Во всех ниже приведенных фрагментах MathCAD системная переменная $ORIGIN=1$.

Тип 1 (формирование элементов массива, удовлетворяющих заданным условиям)

Пример 6.2.1. Составить П-Ф, формирующую матрицу A по следующему правилу:

$$a_{i,j} = \begin{cases} \sin(i+j), & \text{если } i = j; \\ \sin i + \cos j, & \text{если } i > j; \\ \sin j + \cos i, & \text{если } i < j. \end{cases}$$

Описание П-Ф приведено на рис. 6.2.1. ♦

$$\text{form_mat_1}(n, m) := \begin{cases} \text{for } i \in 1..n \\ \quad \text{for } j \in 1..m \\ \quad \quad a_{i,j} \leftarrow \sin(i+j) \text{ if } i = j \\ \quad \quad a_{i,j} \leftarrow \sin(i) + \cos(j) \text{ if } i > j \\ \quad \quad a_{i,j} \leftarrow \sin(j) + \cos(i) \text{ if } i < j \\ \quad a \end{cases}$$

Рис. 6.2.1. Реализация алгоритма примера 6.2.1

Задание 6.2.1. Даны два вектора x и y , состоящие из n элементов. Составить П-Ф, формирующую матрицу B размерности $n \times n$ по следующему правилу:

$$B_{i,j} = \begin{cases} x_i^2 + 2 \cdot y_i, & \text{если } x_i > |y_i|; \\ \sqrt{|x_i + y_i|}, & \text{если } x_i \leq |y_i|. \end{cases} \bullet$$

Задание 6.2.2. Даны два вектора x и y , состоящие из n элементов. Составить П-Ф, формирующую вектор q размерности n , i -й элемент которого равен 1, если точка с координатами (x_i, y_i) принадлежит кругу радиуса r с центром в начале координат. $\bullet$

Задание 6.2.3. Дана матрица A размерности $n \times n$. Составить П-Ф, заменяющую все элементы матрицы, с четной суммой индексов на 1, а другие элементы – на 0. $\bullet$

Задание 6.2.4. Даны два вектора x , y , состоящие из n элементов. Преобразовать эти векторы по правилу: большее из чисел x_i и y_i принять в качестве нового значения x_i , а меньшее – в качестве нового значения y_i . $\bullet$

Тип 2 (подсчет числа элементов массива или последовательности, удовлетворяющих заданным условиям)

Пример 6.2.2. Дан массив y , состоящий из n элементов. Составить П-Ф, определяющую:

- число элементов, удовлетворяющих условию $a \leq |y_i| \leq b$;
- число положительных элементов.

Описание П-Ф приведено на рис. 6.2.2. $\blacklozenge$

Пример 6.2.3. Дана квадратная матрица C размерности $n \times n$. Составить П-Ф, вычисляющую число элементов, находящихся ниже главной диагонали и удовлетворяющих условию $a \leq |C_{i,j}| \leq b$. Заметим, что для элементов, лежащих ниже главной диагонали справедливо неравенство $i > j$. Поэтому возможны два варианта выбора элементов, лежащих ниже главной диагонали: а) проверять все элементы матрицы на выполнение этого условия; б) организовать двойной цикл так, чтобы в нем было обращение только к элементам, удовлетворяющим этому условию. В П-Ф, приведенной на рис. 6.2.3 реализован второй вариант. $\blacklozenge$

```
calc_num(n, y, a, b) :=
    k1 ← 0
    k2 ← 0
    for i ∈ 1..n
        if a ≤ |y_i| ≤ b
            k1 ← k1 + 1
        if y_i > 0
            k2 ← k2 + 1
    ( k1
      k2 )
```

Рис. 6.2.2. Реализация алгоритма примера 6.2.2

```
calc_num2(n, C, a, b) :=
    k ← 0
    for i ∈ 2..n
        for j ∈ 1..i - 1
            k ← k + 1 if a ≤ |C_{i,j}| ≤ b
    k
```

Рис. 6.2.3. Реализация алгоритма примера 6.2.3

Пример 6.2.4. Дана последовательность

$$a_i = \cos\left(\frac{i}{10} + i\right) \text{ где } i=1, 2, \dots, n; \quad n=10.$$

Составить П-Ф, подсчитывающую число отрицательных членов последовательности и число членов, принадлежащих отрезку $[0.5, 1]$.

При составлении описания учтем два момента: а) для универсальности П-Ф элементы последовательности задаются функцией пользователя, имя которой будет параметром П-Ф; б) для подсчета числа элементов необязательно внутри П-Ф формировать массив из элементов последовательности – достаточно работать со значением текущего элемента последовательности. Описание П-Ф и ее вызов приведены на рис. 6.2.4. $\blacklozenge$

$$\begin{array}{l}
 \text{calc_posl_1}(n, a, a1, a2) := \left| \begin{array}{l} k1 \leftarrow 0 \\ k2 \leftarrow 0 \\ \text{for } i \in 1..n \\ \quad \left| \begin{array}{l} c \leftarrow a(i) \\ k1 \leftarrow k1 + 1 \text{ if } c < 0 \\ k2 \leftarrow k2 + 1 \text{ if } a1 \leq c \leq a2 \end{array} \right. \\ \quad \left(\begin{array}{l} k1 \\ k2 \end{array} \right) \end{array} \right. \\
 \\
 a(l) := \cos\left(\frac{l}{10} + l\right) \\
 \boxed{\text{calc_posl_1}(10, a, 0.5, 1) = \left(\begin{array}{l} 5 \\ 2 \end{array} \right) \blacksquare}
 \end{array}$$

Рис. 6.2.4. Реализация алгоритма примера 6.2.4

Пример 6.2.5. Дана матрица D размерности $n \times m$. Составить П-Ф, формирующую вектор, i -й элемент которого равен количеству положительных элементов в i -й строке матрицы D . Описание П-Ф приведено на рис. 6.2.5. Здесь (в отличие от предыдущего примера) искомая характеристика (количество положительных элементов) вычисляется для каждой строки матрицы. Поэтому нулевое начальное значение задается в теле внешнего цикла до начала внутреннего цикла. ♦

$$\begin{array}{l}
 \text{calk_mat_1}(n, m, D) := \left| \begin{array}{l} \text{for } i \in 1..n \\ \quad \left| \begin{array}{l} k \leftarrow 0 \\ \text{for } j \in 1..m \\ \quad k \leftarrow k + 1 \text{ if } D_{i,j} > 0 \\ D_{i,m} \leftarrow k \end{array} \right. \\ \quad D \end{array} \right. \\
 \\
 A := \left(\begin{array}{cc} 1 & -2 \\ 1 & 3 \\ -1 & -9 \end{array} \right) \\
 \boxed{\text{calk_mat_1}(3, 2, A) = \left(\begin{array}{cc} 1 & 1 \\ 1 & 2 \\ -1 & 0 \end{array} \right) \blacksquare}
 \end{array}$$

Рис. 6.2.5. Реализация алгоритма примера 6.2.5

Задание 6.2.4. Даны две матрицы A, B размерности $n \times m$. Составить П-Ф, подсчитывающую число случаев $A_{i,j} \geq B_{i,j}$ •

Задание 6.2.5. Дана матрица D размерности $n \times m$. Составить П-Ф, которая вместо последнего элемента i -й строки записывает коли-

чество элементов i -й строки матрицы D , удовлетворяющих условию $|D_{i,j}| \geq \delta$, где δ – задаваемая величина. •

Тип 3 (подсчет суммы или произведения элементов массива, удовлетворяющих заданным условиям)

Пример 6.2.6. Дан массив y , состоящий из n элементов. Составить П-Ф, определяющую:

- произведение строго положительных элементов массива;
- сумму элементов, находящихся в интервале $[a, b]$.

Описание П-Ф приведено на рис. 6.2.6. Обратите внимание, что начальные значения для переменных sum , pr задаются до начала цикла, в котором происходит «накопление» значений этих переменных. ♦

$$\begin{array}{l}
 \text{calc_sum1}(n, y, a, b) := \left| \begin{array}{l} sum \leftarrow 0 \\ pr \leftarrow 1 \\ \text{for } i \in 1..n \\ \quad \left| \begin{array}{l} sum \leftarrow sum + y_i \text{ if } a \leq y_i \leq b \\ pr \leftarrow pr \cdot y_i \text{ if } y_i > 0 \end{array} \right. \\ \quad \left(\begin{array}{l} sum \\ pr \end{array} \right) \end{array} \right.
 \end{array}$$

Рис. 6.2.6. Реализация алгоритма примера 6.2.6

Пример 6.2.7. Дана матрица A размерности $n \times m$. Составить П-Ф, которая вместо первого элемента i -й строки записывает сумму всех элементов этой строки матрицы. Описание П-Ф приведено на рис. 6.2.7.

Задание 6.2.6. Дан вектор x , состоящий из n элементов. Составить П-Ф, вычисляющую наибольшую из двух сумм элементов с четными и нечетными индексами. •

Задание 6.2.7. Дан вектор x , состоящий из n элементов. Составить П-Ф, вычисляющую среднее арифметическое элементов вектора, а затем преобразующую исходный вектор по правилу: те эле-

менты, которые меньше среднего арифметического, заменить нулями.

•

$$A := \begin{pmatrix} 1 & -2 \\ 1 & 3 \\ -1 & -9 \end{pmatrix}$$

$$sum_mat_2(n, m, A) := \begin{array}{l} \text{for } i \in 1..n \\ \quad sum \leftarrow 0 \\ \quad \text{for } j \in 1..m \\ \quad \quad sum \leftarrow sum + A_{i,j} \\ \quad A_{i,1} \leftarrow sum \end{array}$$

$$sum_mat_2(3, 2, A) = \begin{pmatrix} -1 & -2 \\ 4 & 3 \\ -10 & -9 \end{pmatrix}$$

Рис. 6.2.7. Реализация алгоритма примера 6.2.7

Задание 6.2.8. Дана матрица A размерности $n \times m$. Составить П-Ф, формирующую вектор, i -й элемент равен среднему арифметическому i -й строки матрицы. •

Задание 6.2.9. Дана матрица A размерности $n \times m$. Составить П-Ф, вычисляющую сумму и количество только тех элементов матрицы, которые удовлетворяют условию $|A_{i,j}| \geq \varepsilon$, где ε — задаваемая величина. •

Задание 6.2.10. Имеется экзаменационная ведомость студенческой группы из $n = 20$ человек по $m = 5$ дисциплинам. Оценки из этой ведомости занесены в матрицу размерности $n \times m$. Составить П-Ф, вычисляющую число студентов, получивших только четверки и пятёрки. •

Задание 6.2.11. Имеется экзаменационная ведомость студенческой группы из $n = 20$ человек по $m = 5$ дисциплинам. Оценки из этой ведомости занесены в матрицу размерности $n \times m$. Составить П-Ф, вычисляющую средний балл по каждой дисциплине. •

Тип 4 (вычисление максимального или минимального элемента массива и его индекса)

Пример 6.2.8. Дан массив y , состоящий из n элементов. Составить П-Ф, меняющую максимальный и минимальный элемент местами. Описание П-Ф приведено на рис. 6.2.8. ♦

$min_max_vec(n, y) :=$

```

min ← y1
max ← y1
imin ← 1
imax ← 1
for i ∈ 1..n
    if yi > max
        max ← yi
        imax ← i
    if yi < min
        min ← yi
        imin ← i
yimin ← max
yimax ← min
y

```

$$x := \begin{pmatrix} 2 \\ 10 \\ 4 \\ -3 \\ 2 \end{pmatrix}$$

$$min_max_vec(5, x) = \begin{pmatrix} 2 \\ -3 \\ 4 \\ 10 \\ 2 \end{pmatrix}$$

Рис. 6.2.8. Реализация алгоритма примера 6.2.8

Пример 6.2.9. Дана функция $f(x)$ и вектор x , состоящий из n элементов. Составить описание П-Ф, определяющей: а) максимальное значение этой функции на элементах этого вектора; б) номер индекса и значение элемента массива, на котором функция достигает максимального значения.

Описание П-Ф и обращение к ней приведены на рис. 6.2.9. Отметим, что в качестве третьего параметра П-Ф используется имя функции пользователя. Формальным параметром является функция $f(x)$, а фактическим — функция $\psi(x)$, описание которой должно *находиться до обращения к П-Ф*. ♦

Пример 6.2.10. Дана матрица A размерности $n \times m$. Составить П-Ф, формирующую вектор, i -й элемент равен номеру столбца в котором на i -й строке матрицы стоит минимальный элемент. Описание П-Ф приведено на рис. 6.2.10. ♦

$$\psi(x) := e^{\frac{-(x-3.5)^2}{3}}$$

$$\max_fun(n, x, f) := \begin{array}{l} x_{max} \leftarrow x_1 \\ i_{max} \leftarrow 1 \\ f_{max} \leftarrow f(x_{max}) \\ \text{for } i \in 1..n \\ \quad \text{if } f(x_i) > f_{max} \\ \quad \quad \left| \begin{array}{l} f_{max} \leftarrow f(x_i) \\ i_{max} \leftarrow i \\ x_{max} \leftarrow x_i \end{array} \right. \end{array}$$

$$x := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{pmatrix}$$

$$\max_fun(6, x, \psi) = \begin{pmatrix} 0.92 \\ 3 \\ 3 \end{pmatrix}$$

Рис. 6.2.9. Реализация алгоритма примера 6.2.9

$$A := \begin{pmatrix} 1 & -2 \\ 1 & 3 \\ -1 & -9 \end{pmatrix}$$

$$\min_mat_1(n, m, A) := \begin{array}{l} \text{for } i \in 1..n \\ \quad j_{min} \leftarrow 1 \\ \quad \min \leftarrow A_{i,1} \\ \quad \text{for } j \in 1..m \\ \quad \quad \text{if } A_{i,j} < \min \\ \quad \quad \quad \left| \begin{array}{l} j_{min} \leftarrow j \\ \min \leftarrow A_{i,j} \end{array} \right. \\ \quad x_i \leftarrow j_{min} \end{array}$$

$$\min_mat_1(3, 2, A) = \begin{pmatrix} 2 \\ 1 \\ 2 \end{pmatrix}$$

Рис. 6.2.10. Реализация алгоритма примера 6.2.10

Задание 6.2.12. Даны две матрицы A, B размерности $n \times m$. Вычислить величину $\max = \max A + \max B$, где $\max A, \max B$ – максимальные значения матриц A, B соответственно. Для вычисления значения максимального элемента составить П-Ф. •

Задание 6.2.13. Дана матрица A размерности $n \times m$. Составить П-Ф, меняющую местами максимальный и минимальный элементы матрицы. •

Задание 6.2.14. Дана матрица A размерности $n \times m$. Составить П-Ф, вычисляющую номер столбца, который содержит наименьший по модулю элемент этой матрицы. •

Задание 6.2.15. Имеется экзаменационная ведомость студенческой группы из $n = 20$ человек по $m = 5$ дисциплинам. Оценки из этой ведомости занесены в матрицу размерности $n \times m$. Составить П-Ф, вычисляющую номер дисциплины (т.е. номер столбца), имеющей максимальный средний балл. •

Задание 6.2.16. Имеется экзаменационная ведомость студенческой группы из $n = 15$ человек по $m = 4$ дисциплинам. Оценки из этой ведомости занесены в матрицу размерности $n \times m$. Составить П-Ф, вычисляющую номер студента (т.е. номер строки), имеющего максимальный средний балл по учебным дисциплинам. •

6.3. Программирование итерационных циклов

Пример 6.3.1. Составить описание П-Ф, вычисляющей факториал числа $n \geq 0$ по формуле

$$n! = 1 \cdot 2 \cdot 3 \cdots n.$$

Описание П-Ф с использованием оператора произведения приведено на рис. 6.3.1.

$$FI(n) := \begin{cases} 0 & \text{if } n \leq 0 \\ \prod_{i=1}^n i & \text{otherwise} \end{cases}$$

$$FI(5) = 120$$

$$FI(0) = 0 \quad FI(-3) = 0$$

Рис. 6.3.1. Вычисление факториала числа n

Описание П-Ф с использованием итерационного цикла и оператора *break* приведено на рис. 6.3.2. Обратите внимание на запись условия в операторе *while*, а именно *while 1*, что обеспечивает повторение тела цикла до тех пор, пока не выполнится условие $n = 1$, что приведет к выполнению оператора *break* и выходу из цикла. ♦

```

F2(n) :=  $\left\{ \begin{array}{l} f \leftarrow 0 \text{ if } n \leq 0 \\ \text{otherwise} \\ \quad f \leftarrow n \\ \quad \text{while } 1 \\ \quad \quad f \leftarrow f \cdot (n - 1) \\ \quad \quad n \leftarrow n - 1 \\ \quad \quad \text{break if } n = 1 \end{array} \right.$ 

```

$F2(5) = 120$ ■

Рис. 6.3.2. Вычисление факториала числа n

Пример 3.6.2. Дан вектор v , состоящий из n элементов. Составить П-Ф, определяющую первый элемент этого массива, который удовлетворяет условию $|v_j - x| \leq \varepsilon$, где x, ε — заданные величины.

Описание П-Ф приведено на рис. 6.3.3. Эту же задачу можно реализовать с помощью оператора цикла *for*, что предотвратит «зацикливание», которое может иметь место в П-Ф *search* при неправильном задании исходных данных (например, при задании $\varepsilon < 0$). Описание П-Ф приведено на рис. 6.3.4. ♦

```

search(v, x, ε) :=  $\left\{ \begin{array}{l} j \leftarrow 1 \\ \text{while } |v_j - x| > \varepsilon \\ \quad j \leftarrow j + 1 \end{array} \right.$ 

```

$j := 1..500 \quad w_j := \text{ceil}(1000 \cdot \cos(j))$
 $\text{search}(w, 400, 10^{-3}) = \begin{pmatrix} 397 \\ 400 \end{pmatrix}$

Рис. 6.3.3. Реализация алгоритма примера 6.3.2

```

search_1(v, x, ε) :=  $\left\{ \begin{array}{l} \text{for } j \in 1..last(v) \\ \quad \text{break if } |v_j - x| \leq \varepsilon \end{array} \right.$ 

```

$\text{search}_1(w, 400, 10^{-3}) = \begin{pmatrix} 397 \\ 400 \end{pmatrix}$ ■

Рис. 6.3.4. Реализация алгоритма примера 6.3.2

В двух предыдущих примерах была показана возможность замены оператора *while* (используемого при программной реализации итерационного цикла) другими конструкциями. Это обусловлено тем, что в этих примерах параметры цикла изменялись по закону арифметической прогрессии. Однако встречаются циклы, параметры которых изменяются по более сложным соотношениям. Особенно это характерно для алгоритмов решения нелинейных уравнений. В примере 6.3.3 рассматривается один из таких алгоритмов — метод последовательных предложений.

Пример 6.3.3. Дано нелинейное уравнение

$$x^2 - e^x = 0. \quad (6.3.1)$$

Необходимо вычислить с точностью $\varepsilon = 10^{-5}$ вещественный корень, лежащий в интервале $[-4, 0]$, используя метод последовательных приближений (называемый также методом простой итерации). Не останавливаясь на теоретическом обосновании, приведем основные расчетные соотношения, необходимые для программной реализации этого метода.

Первоначально исходное уравнение $f(x) = 0$ заменяется эквивалентным ему уравнением

$$x = \varphi(x). \quad (6.3.2)$$

Затем выбираем начальное значение x_0 из некоторого интервала $[a, b]$, внутри которого находится только один корень уравнения $f(x) = 0$ (в нашем примере это интервал $[-4, 0]$). Подставляя это значение в правую часть (6.3.2), получаем первое приближение $x_1 = \varphi(x_0)$. Затем значение x_1 вновь подставляем в правую часть (6.3.2) и получаем $x_2 = \varphi(x_1)$.

Повторяя этот процесс, получаем последовательность чисел

$$x_n = \varphi(x_{n-1}) \quad (6.3.3)$$

(это оправдывает название — метод последовательных приближений). Возможны два случая.

Случай 1. Последовательность $x_0, x_1, x_2, \dots, x_n, \dots$ *сходится* к некоторому числу x^* , т.е. имеет предел, и тогда этот предел является решением нелинейного уравнения, т.е. $f(x^*) = 0$.

Случай 2. Последовательность $x_0, x_1, x_2, \dots, x_n, \dots$ **расходится**, т.е. не имеет предела, и в этом случае метод не применим.

Приведем условие, являющееся достаточным для сходимости метода последовательных приближений.

Пусть на отрезке $[a, b]$ имеется единственный корень уравнения $x = \phi(x)$, и во всех точках этого отрезка справедливо неравенство

$$|\phi'(x)| \leq q < 1. \quad (6.3.4)$$

Если при этом выполняется и условие

$$a \leq \phi(x) \leq b, \quad (6.3.5)$$

то итерационный процесс (6.3.3) сходится, а за нулевое приближение можно брать любое $x_0 \in [a, b]$.

Заметим, что чем меньше величина q в (6.3.4), тем быстрее сходится последовательность $x_0, x_1, x_2, \dots, x_n, \dots$ к своему пределу x^* . В качестве приближенного значения корня можно принять приближение x_n , удовлетворяющее условию

$$|x_n - x_{n-1}| \leq \frac{\varepsilon \cdot (1 - q)}{q} = \eta, \quad (6.3.6)$$

где ε — заданная точность вычисления корня, входящая в неравенство

$$|x_n - x^*| \leq \varepsilon, \quad (6.3.7)$$

x^* — точное значение корня.

Вернемся к уравнению нашего примера и преобразуем его к виду (6.3.2). Получаем $x = -e^{\frac{x}{2}}$ (знак минус появился из-за отрицательности искомого корня), т.е. $\phi(x) = -e^{\frac{x}{2}}$. Проверим выполнение условия (6.3.4) графическим способом, используя графику MathCAD (см. рис. 6.3.5).

$$\phi(x) := -e^{\frac{x}{2}} \quad d\phi(x) := \frac{d}{dx} \phi(x) \quad x := -4, -3.95 \dots 0$$

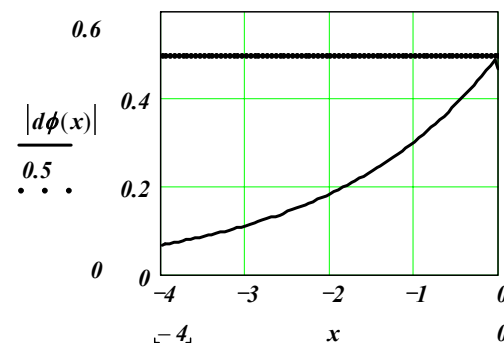


Рис. 6.3.5. Проверка условия (6.3.4)

Из графика видно, что в качестве величины q в условии (6.3.4) можно принять значение 0.5, которое удовлетворяет условию (6.3.4).

Для проверки условия (6.3.5) построим график функции $\phi(x)$ на интервале $[-4, 0]$ (см. рис. 6.3.6). Видно, что условие (6.3.5) также выполняется — на всем интервале $[-4, 0]$ функция $\phi(x)$ удовлетворяет условию $-4 < \phi(x) < 0$.

Вывод: метод последовательных приближений для любого $x_0 \in [-4, 0]$ сходится к точному решению нелинейного уравнения (6.3.1).

Определим величину η в условии окончания итераций (6.3.6):

$$\eta := \frac{10^{-6} \cdot (1 - 0.5)}{0.5} \quad \eta = 1 \times 10^{-6}$$

Описание П-Ф, реализующей метод последовательных приближений приведено на рис. 6.3.7. Здесь же показаны вычисленное приближенное значение корня и его проверка. Имя функции $\phi(x)$ является формальным параметром. Для предотвращения «зацикливания» итерационной процедуры (6.3.3) используется оператор цикла *for* (обсуждение см. в примере 5.3.4). Заметим, что переменные x_0, x_n , используемые в теле П-Ф, являются простыми переменными, имена

которых имеют нижние индексы (вводимые после нажатия клавиши [.] – «десятичная точка»). ♦

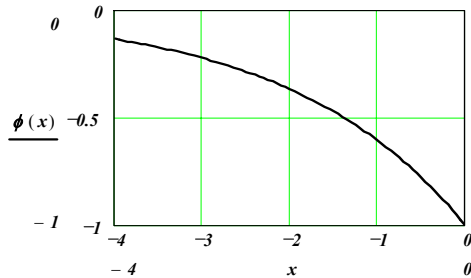


Рис. 6.3.6. Проверка условия (6.3.5)

```

postl_prib(x_0, phi, eta) := | ierr ← 1
                             | for n ∈ 1..10000
                             |   | x_n ← phi(x_0)
                             |   | if |x_n - x_0| ≤ eta
                             |   |   | ierr ← 0
                             |   |   | break
                             |   | x_0 ← x_n
                             | x_n ← "not solve" if ierr = 1
                             | x_n

f(x) := x^2 - e^x      phi(x) := -e^(x/2)    x_0 := -2    eta := 10^-6
x_app := postl_prib(x_0, phi, eta)
x_app = -0.70347      f(x_app) = -3.01 × 10^-7

```

Рис. 6.3.7. Программная реализация метода последовательных приближений

Задание 6.3.1. Дано нелинейное уравнение

$$f(x) = 5x^3 - 20x + 3 = 0.$$

Составьте П-Ф для вычисления корня уравнения, лежащего в интервале $[0, 1]$, методом последовательных приближений. Для организации итерационного цикла используйте оператор цикла *while* (см. пример 5.3.3). •

Задание 6.3.2. Дано нелинейное уравнение

$$f(x) = x^3 + 3x^2 - 24x + 1 = 0.$$

Составьте П-Ф для вычисления двух корней уравнения, лежащих в интервалах $[-7, -1]$ и $[2, 12]$, методом последовательных приближений с точностью 0.001. Для организации итерационного цикла используйте оператор цикла *while* (см. пример 5.3.3). •

ТЕМА 7. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ В MathCAD

В этом параграфе будут рассмотрены основные понятия модульного программирования и способы его реализации в MathCAD.

7.1. Преимущества модульного программирования

Модульное программирование появилось в конце шестидесятых годов XX столетия и способствовало резкому повышению производительности труда при разработке, тестировании и сопровождении программ. Вопросам модульного программирования было посвящено большое число публикаций (например, [10]). Основная идея модульного программирования заключается:

- в разбиении алгоритма решения той или иной задачи на слабо зависимые друг от друга фрагменты вычислений и реализации каждого такого фрагмента в виде *программных модулей*;
- в вызове в нужных местах «основной» программы соответствующих модулей с передачей необходимых данных.

Программа, реализующая алгоритм вычислений в виде модулей и обращения к ним, получила название *модульной программы*.

Что же дает модульное программирование? К основным преимуществам можно отнести следующее:

- малая зависимость модулей позволяет при необходимости существенно распараллелить разработку программы, поручив это разным программистам;

- модульную программу легче отлаживать, так как модули могут быть подвергнуты «автономному» тестированию и отладке, т.е. каждый модуль может проходить тестирование и отладку отдельно на подготовленном наборе тестовых данных;

- модульную программу легче сопровождать и модифицировать. В модуль можно ввести изменения, переписать его или заменить без внесения изменений в другие модули (это, отчасти, является признаком слабой зависимости между модулями);

- но самое главное – использование библиотек «готовых» модулей, ориентированных на решение определенного класса научно-технических задач.

Как же реализуется модульное программирование в MathCAD? Модуль реализуется в виде подпрограммы-функции (см. тему 4). По расположению описания П-Ф можно выделить два метода получения модульных программ:

- описание П-Ф и обращение к ней находятся в пределах одного документа. Назовем это модульным программированием в пределах одного документа;

- описание П-Ф и обращение к ней находятся в разных документах MathCAD – модульное программирование в нескольких документах.

7.2. Модульное программирование в пределах одного документа MathCAD

Этот метод характеризуется тем, что:

- для реализации простых вычислений используются функции пользователя, а для более сложных – П-Ф;

- описания функций пользователя, П-Ф и их вызов находятся в пределах одного документа и хранятся в одном файле. При этом в теле П-Ф могут находиться вызовы других функций пользователя и П-Ф.

Пример 7.2.1. Составим П-Ф вычисления определенного интеграла вида:

$$\int_a^b f(x)dx, \quad (7.2.1)$$

используя формулу Симпсона с автоматическим выбором числа интервалов, на которые делится отрезок интегрирования $[a, b]$.

Подпрограмма-функция *Simpson* (f, a, b, N) (приведенная на рис. 7.2.1) выделяет определенный интеграл по формуле Симпсона при фиксированном числе интервалов N , а подпрограмма-функция *Adapt-Simpson* (f, a, b, ε) увеличивает число интервалов до тех пор, пока не будет достигнута заданная точность вычисления интеграла ε , т. е. не выполнится условие:

$$|I_2 - I_1| < \varepsilon, \quad (7.2.2.)$$

где I_2 – значение интеграла, вычисленного по удвоенному (по сравнению с интегралом I_1) числу интегралов (длина которых стала в два раза меньше).

На рис. 7.2.2 показаны вызовы П-Ф для определенной подынтегральной функций $f(x)$. Для контроля вычисленных значений интегралов здесь же приводятся значения интегралов, вычисленных с помощью оператора Интеграл (палитра **МАТЕМАТИЧЕСКИЙ АНАЛИЗ**). Видно, что при $\varepsilon = 10^{-8}$ значения интегралов совпадают.

Задание 7.2.1. Составьте описания П-Ф вычисления интеграла вида (7.2.1.) с точностью ε по формуле трапеций с автоматическим выбором числа интервалов. Формула трапеций при делении отрезка $[a, b]$ на N равных интервалов имеет вид:

$$Int = \frac{b-a}{N} \left[\sum_{i=1}^{N-1} f(x_i) + \frac{f(x_0) + f(x_N)}{2} \right], \quad (7.2.3)$$

где $x_i = a + i \cdot \frac{b-a}{N}$, $i = 0, \dots, N$.

$ORIGIN := 0$

$$Simpson(f, a, b, N) := \left| \begin{array}{l} h \leftarrow \frac{b-a}{N} \\ S \leftarrow (f(a)) + f(b) \\ \text{for } i \in 0..N-1 \\ \quad S \leftarrow S + 4 \cdot f\left(a + i \cdot h + \frac{h}{2}\right) \\ \text{for } i \in 1..N-1 \\ \quad S \leftarrow S + 2 \cdot f(a + i \cdot h) \\ \frac{h}{6} \cdot S \end{array} \right|$$

$$Adapt_Simpson(f, a, b, \varepsilon) := \left| \begin{array}{l} Int1 \leftarrow Simpson(f, a, b, 5) \\ Int2 \leftarrow Simpson(f, a, b, 10) \\ Int2 \text{ if } |Int2 - Int1| < \varepsilon \\ \text{otherwise} \\ \quad Int \leftarrow Adapt_Simpson\left(f, a, \frac{a+b}{2}, \varepsilon\right) \\ \quad Int \leftarrow Int + Adapt_Simpson\left(f, \frac{a+b}{2}, b, \varepsilon\right) \\ Int \end{array} \right|$$

Рис. 7.2.1. Подпрограммы-функции вычисления интегралов

$$\begin{aligned} f_2(x) &:= e^{-x^2} \cdot \cos(2 \cdot x) \\ \int_0^1 f_2(x) dx &= 0.4290978 \blacksquare \\ Simpson(f_2, 0, 1, 10) &= 0.4290976 \blacksquare \\ Adapt_Simpson(f_2, 0, 1, 10^{-8}) &= 0.4290978 \blacksquare \end{aligned}$$

Рис. 7.2.2. Вычисление интеграла различными способами

Модульное программирование в одном документе имеет ряд недостатков:

- невозможность «автономной» отладки и тестирования П-Ф и их модификации в процессе эксплуатации программы;
- невозможность использования разработанных П-Ф в нескольких документах MathCAD без дублирования их описания;
- невозможность эффективного использования П-Ф из библиотек, ориентированных на решение определенного класса задач. Такие библиотеки размещаются на Web-сайтах в Internet (см. п. 7.4).

Эти недостатки устраняются при переходе к модульному программированию в нескольких документах, подробно рассмотренному в следующем пункте.

7.3. Модульное программирование в нескольких документах MathCAD

Такое модульное программирование подразумевает, что описания П-Ф выполнены в одном документе MathCAD, а их вызов помещен в другом документе (такой метод широко используется в современных алгоритмических языках высокого уровня). Однако при этом возникает вопрос: как «присоединить» документ с описанием П-Ф к документу, в котором вызывается эта П-Ф?

Для этого необходимо выполнить следующие шаги:

- 1) активизировать документ, в котором вызываются П-Ф, описанные в другом документе;
- 2) щелкнуть мышью в том месте документа (обязательно перед обращением к П-Ф), куда будет вставлена специальная конструкция **Ссылка**;
- 3) обратиться к пункту меню **Вставка** и выполнить команду **Ссылка**;
- 4) в появившемся диалоговом окне в поле ввести полное имя файла (используя кнопку **Обзор**), в котором находится описание вызываемых П-Ф. Затем щелкнуть на кнопке **ОК**;
- 5) в документе появится конструкция, подобная показанной на рис. 7.3.1.

➔ *Ref* *Reference:F:\COPY\P_Ф_Интегрирования.mcd*

Рис. 7.3.1. Ссылка на файл с описаниями подпрограмм-функций

Пример 7.3.1. Реализовать модульное программирование в нескольких документах для задачи примера 7.2.1.

На *первом этапе* составим описания П-Ф (приведены на рис. 7.2.1) и сохраним их в файле с именем *П_Ф_интегрирования* в папке с маршрутом *F:\Copy*.

На *втором этапе* в начале нового документа вставляем конструкцию *Ссылка*, показанную на рис. 7.3.1, а ниже записываем вызовы П-Ф, вычисляющих значения определенных интегралов (см. рис. 7.3.2).

➔ *Reference:F:\COPY\P_Ф_Интегрирования.mcd*

$$f_2(x) := e^{-x^2} \cdot \cos(2 \cdot x)$$

$$\text{Adapt_Simpson} \left(f_2, 0, 1, 10^{-8} \right) = 0.4290978$$

Рис. 7.3.2. Вызов подпрограмм-функций вычисления интегралов

Очевидно, что:

- после сохранения описания П-Ф в файле, можно вызывать эти П-Ф в различных документах MathCAD, не дублируя их описания в этих документах;
- при необходимости внести изменения в алгоритм вычислений редактированию подвергается только файл с описанием П-Ф. После открытия документа MathCAD, в котором вызываются отредактированные П-Ф, осуществляется пересчет результатов в соответствии с внесенными в П-Ф изменениями.

7.4. Программы MathCAD в Internet

Возможность использования в MathCAD П-Ф и целых программ, разработанных другими пользователями, вызвала появление в Internet Web-сайтов, на которых размещены файлы с документами

MathCAD для решения различных задач. Также на сайтах могут размещаться целые библиотеки П-Ф, ориентированные уже на решение определенного класса задач. Кратко остановимся на некоторых из таких ресурсов Internet.

Библиотека Web Library содержит в себе много интересных примеров применения пакета MathCAD в различных областях науки, техники и образования. Для доступа в библиотеку необходимо:

- открыть окно Resource Center (Центра ресурсов) (см. рис. 7.4.1);



Рис. 7.4.1. Окно Resource Center

- щелкнуть на кнопке **Web Library**;
- щелкнуть на интерактивной ссылке в нижней части появившегося окна программы-браузера (см. рис. 7.4.2);
- в появившемся окне (см. рис. 7.4.3) щелкнуть на интерактивной ссылке нужного документа.

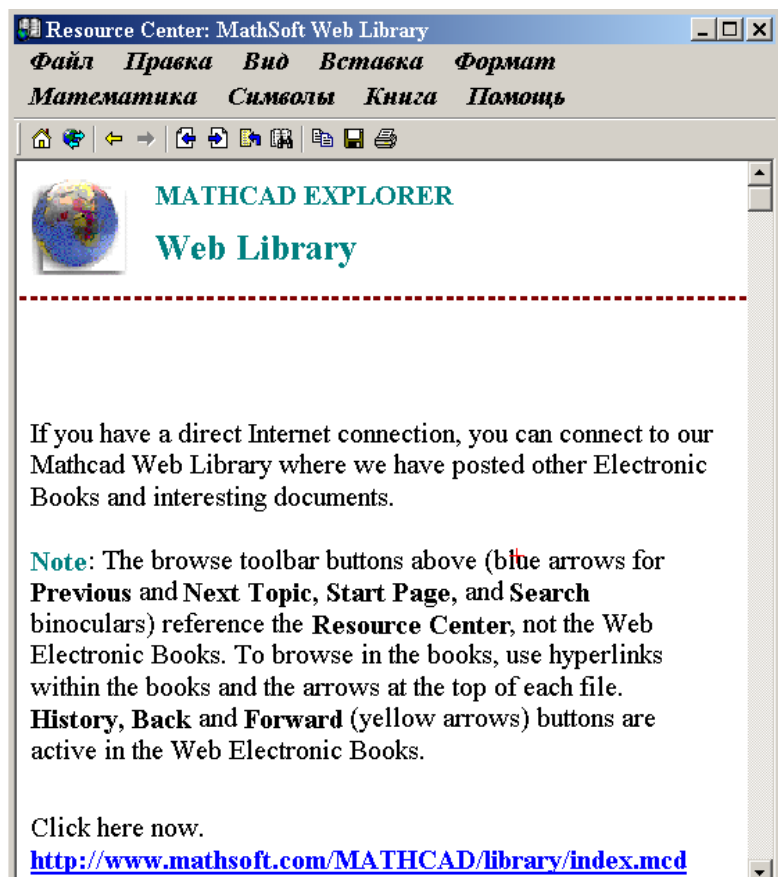


Рис. 7.4.2. Окно Web Library

На сайте фирмы MathSoft (разработчик пакета MathCAD) расположена тематическая библиотека файлов MathCAD для решения различных научных и технических задач (URL-адрес сайта http://www.mathcad.com/library/Mathcad_Files.asp). Оглавление библиотеки приведено на рис. 7.4.4. Для каждого тематического раздела указано количество файлов в этом разделе библиотеки. Например, в разделе *Civil and Mechanical Engineering* находится 94 файла (февраль 2002 года).



Welcome to the **Mathcad Web Library**, a part of the MathSoft WWW site consisting exclusively of linked Mathcad files.

This page is now several years out-of-date. We recommend that you visit instead <http://www.mathcad.com/library/> for the latest version of the Web Library, with improved searching capabilities and much more material.

Below you'll find a variety of Mathcad files and selections from our line of Electronic Books. If you want general MathSoft product and corporate information, including the latest Technical Support information, be sure to visit our home page at <http://www.mathsoft.com>.

Mathcad Electronic Books [About Ebooks in the Library](#)

- **Visual Electromagnetics**, Mathcad 8 or higher, by Keith W. Whites

Рис. 7.4.3. Страница Web Library

products
 upgrade
 buy/license
 download
 library
 community
 news
 support
 training

[Electronic Books](#) | [Mathcad Files](#) | [Gallery](#) | [Puzzles](#) | [Search Library](#)

Mathcad Files

Mathcad example files are available at the educational and professional levels. Select a discipline to browse the collection, or search for specific topics on the [library search](#) page.

Aerospace Engineering	31 files
Biology	41 files
Business and Economics	16 files
Chemistry and Chemical Engineering	66 files
Civil and Mechanical Engineering	94 files
Computer Science	23 files
Electrical Engineering	134 files
Environmental Engineering	13 files
Mathematics	128 files
Music	1 files
Physics	119 files
Probability and Statistics	23 files
Entertainment	19 files

Select Industry ▼

web store
Special Promotions (US & Canada only)
Mathcad Order Now!
quick links

Рис. 7.4.4. Оглавление тематической библиотеки файлов MathCAD

Щелкнув кнопкой мыши на выбранной гиперссылке нужного раздела, входим в него, и на экране появляется содержание этого раздела (см. рис. 7.4.5, на котором приведено оглавление *Civil and Mechanical Engineering*). Для загрузки нужного файла MathCAD достаточно щелкнуть мышью на его имени (в правой половине).

Civil and Mechanical Engineering

Title : Type	Author	Files
3-D Phase Plot for a Spring Anonymous		3dphase.mcd (24 KB) Mathcad 2000
A Spring that Compresses Anonymous		spring.avi (153 KB)
A Vibrating Drum Head Anonymous		drum.avi (162 KB)
A Waterwheel Design Tutorial Tommy L. Hollifeld		wtrwhl.mcd (39 KB) Mathcad 5

Рис. 7.4.5. Содержание тематического раздела *Civil and Mechanical Engineering*

Много методических разработок по изучению и использованию пакета MathCAD находится на образовательном сайте *Exponenta* (URL-адрес сайта www.exponenta.ru/soft/Mathcad). На рис. 7.4.6 показан пример оформления ссылки на методические материалы по программированию в пакете MathCAD.

■ Программирование в математическом пакете Mathcad

© Ю.Е.Воскобойников, Т.Н.Воскобойникова

Новосибирский государственный архитектурно-строительный университет,
кафедра прикладной математики, e-mail: voscob@mail.ru

Рис. 7.4.6. Пример методических материалов,
размещенных на сайте *Exponenta*

В Интернете существует много Web-страниц, на которых авторы размещают файлы MathCAD, посвященные решению различных научно-технических задач. На рис. 7.4.7 показан пример такой Web-страницы. Ниже приводятся адреса некоторых сайтов:

- collab.mathsoft.com/~mathcad2000 – англоязычный форум по проблемам MathCAD (русское описание форума – twt.mpei.ac.ru/ochkov/Collab/Collab.htm);
- www.mathsoft.com/mathcad/library/3Dplots/ – галерея трехмерной графики MathCAD (русское описание галереи – twt.mpei.ac.ru/ochkov/Lace/Lace.htm);
- www.mathsoft.com/appsindex.html – сборник прикладных файлов MathCAD;
- www.mathsoft.com/mathcad/library/world.html – всемирное собрание файлов MathCAD (MathCAD Files Around the World);
- www.mathsoft.com/books.html – книги по MathCAD;
- http://twt.mpei.ac.ru/ochkov/MC_ODE/Stiff_ODE/Stiff_DE.htm – статья «Жесткие дифференциальные уравнения (интегрирование в среде MathCAD)»;
- <http://petsu.karelia.ru/psu/Deps/IMO/Complex/> – учебно-методический комплекс «Численные методы с системой MathCAD» для изучения алгоритмов решения математических задач с использованием системы MathCAD;
- <http://www.keldysh.ru/comma> – Интернет-версия курса «Вычислительная физика» (автор Д.В. Кирьянов). Содержит много примеров, решенных в MathCAD;
- <http://www.mpei.ac.ru/homepages/mm/> – лабораторные работы по дисциплине «Вычислительная математика».

<http://www2.latech.edu/~dmg/#Civil Engineering>

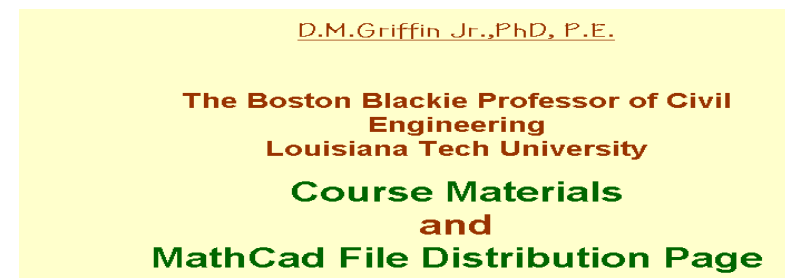


Рис. 7.4.7. Персональная страничка в Интернете

Одной из самых привлекательных возможностей, предоставляемых фирмой MathSoft, является *организация совместной работы по созданию программного обеспечения для пакета MathCAD*. Технически это реализуется с помощью специального раздела сайта этой фирмы, на котором хранятся тематические подборки проектов (разработок программного обеспечения) для совместной работы пользователей MathCAD из разных стран. Каждый проект имеет руководителя – обычно, первого автора, предложившего этот проект. Все остальные члены авторского коллектива могут принимать участие в доработке проекта. Таким образом обеспечивается объединение потенциалов различных организаций и научных школ в решении общих задач с использованием пакета MathCAD.

Для перехода в раздел совместной работы необходимо:

- в окне Resource Center (см. рис. 7.4.1) щелкнуть на кнопке Collaboratory;
- после установления связи появится окно раздела Collaboratory (см. рис. 7.4.7). В этом окне приводится перечень проектов и другая полезная информация.

После этого достаточно активизировать какую-либо гиперссылку тематического каталога и выбрать подходящий проект. Контекстное меню открывает доступ к ряду команд (например, для записи проекта на жесткий диск или загрузки его прямо в окно документов MathCAD).

Файл проекта (как и любого другого документа MathCAD) можно отправить по электронной почте нужному адресату. Для этого достаточно в окне программы MathCAD обратиться к пункту меню File команде Send. После этого появляется окно почтовой про-

граммы Outlook Express. Текущий документ MathCAD становится вложенным в электронное сообщение файлом с расширением *.mcd*. Щелчок на кнопке **Отправить** запускает обычный процесс отправки сообщения со вложенным файлом по электронной почте.

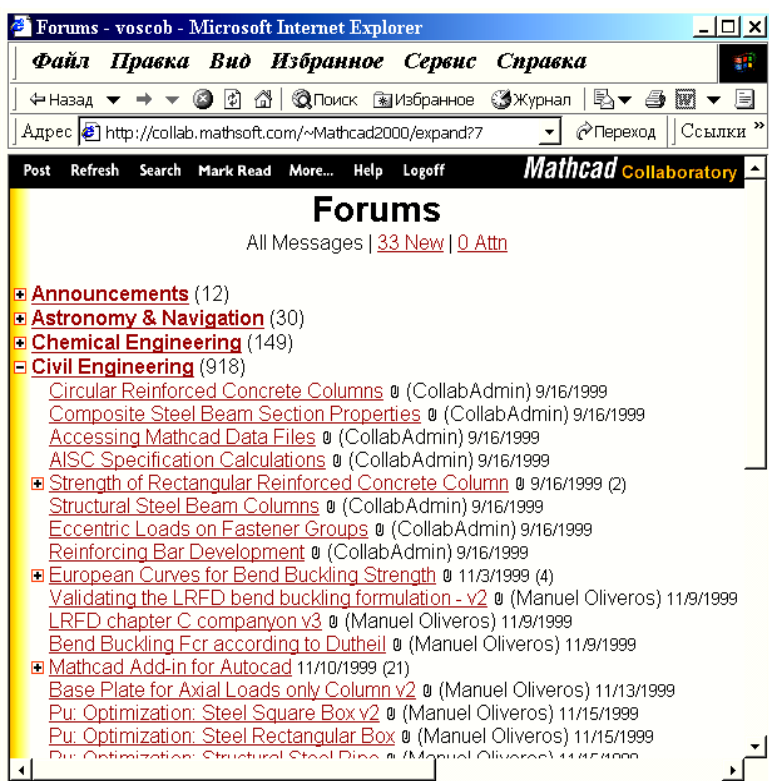


Рис. 7.4.7. Перечень проектов

РАЗДЕЛ 3. РЕШЕНИЕ НАУЧНО-ИНЖЕНЕРНЫХ ЗАДАЧ В ПАКЕТЕ MathCAD

В этом разделе рассматриваются три класса задач, часто возникающих при проектировании и расчетах строительных конструкций и сооружений. Это:

- решение нелинейных уравнений и систем уравнений;
- решение оптимизационных задач, включая задачи с ограничениями;
- обработка экспериментальных данных.

Для решения этих задач будут использоваться как «собственные» функции MathCAD, так и оригинальные подпрограммы-функции (П-Ф). Материал этого раздела, безусловно, будет полезен пользователю при решении «своих» задач в пакете MathCAD.

ТЕМА 8. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ В ПАКЕТЕ MathCAD

Рассматриваются функции пакета для решения нелинейных уравнений, а также систем уравнений, включая системы нелинейных уравнений. Кроме этого, некоторые алгоритмы реализуются в виде П-Ф.

8.1. Решение нелинейных уравнений

В этом параграфе приводится определение нелинейного уравнения, и описываются основные этапы его решения.

8.1.1. Определение нелинейного уравнения

Нелинейное уравнение с одним неизвестным можно записать в виде:

$$f(x) = 0. \quad (8.1.1)$$

Совокупность значений переменной x , при которых уравнение (8.1.1) превращается в тождество, называется *решением* этого уравнения, а каждое значение x из этой совокупности называется *корнем уравнения*. Например, уравнение $x^2 = 2 - x$ имеет два корня:

$x_1 = -2$, $x_2 = 1$. Для проверки этого утверждения необходимо каждое из этих двух значений подставить в уравнение и убедиться в том, что уравнение обращается в тождество. На графике функции $f(x)$ корнями являются те значения x , в которых функция обращается в ноль. На рис. 8.1.1 приведен график функции $f(x) = x^2 - 2 + x$, полученной после преобразования уравнения $x^2 = 2 - x$, на котором хорошо определяются значения корней уравнения $f(x) = 0$ (в этом и заключается *графический метод* нахождения корней нелинейного уравнения).

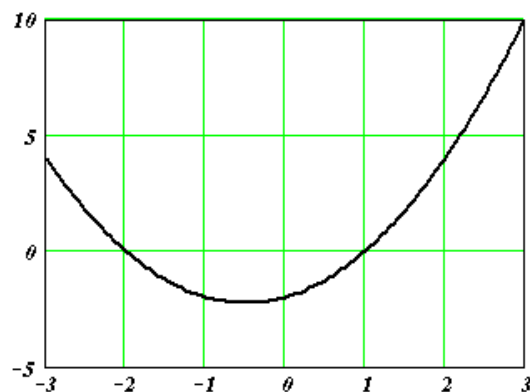


Рис. 8.1.1. К определению корней нелинейного уравнения

Если в запись уравнения входят только алгебраические функции (четыре арифметические операции и возведение в степень), то уравнение называется *алгебраическим*, и оно после необходимых преобразований может быть записано в виде:

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0. \quad (8.1.2)$$

Числа a_n , a_{n-1} , ..., a_1 , a_0 называются *коэффициентами уравнения*, и в дальнейшем будем полагать, что все коэффициенты являются вещественными (хотя они могут быть и комплексными). Корни уравнения при этом могут быть как *вещественными*, так и *мнимыми*.

Известно, что уравнение (8.1.2) n -го порядка имеет n корней и допускает представление

$$f(x) = (x - x_1) \cdot (x - x_2) \cdot \dots \cdot (x - x_n) = 0, \quad (8.1.3)$$

где $x_1, x_2, \dots, x_n$ – корни уравнения (8.1.2). Если поделить (8.1.2) на $(x - x_j)$, то порядок полученного алгебраического уравнения

$$\frac{a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0}{x - x_j} \quad (8.1.4)$$

будет на единицу меньше, и среди его корней не будет корня x_j . В справедливости нетрудно убедиться, используя формулу (8.1.3). Этот прием понижения порядка уравнения будет часто использоваться в дальнейшем.

Нелинейное уравнение, в которое входят *трансцендентные* функции (показательные, логарифмические, тригонометрические и обратные тригонометрические функции), называется *трансцендентным*. Примером может служить уравнение $\sin(x) = 0$, корни которого определяются как $x = \pi \cdot n$ ($n = 0, \pm 1, \pm 2, \dots$).

Решение уравнения можно найти, используя различные методы: *аналитические* (корни определяются аналитическим выражением), *графические* (приближенные значения корней находятся из графика – пример рис. 8.1.1), *численные* (приближенные численные значения корней определяются в результате некоторой вычислительной процедуры). В дальнейшем будут рассматриваться только численные методы.

Вычисление корней численными методами включает два основных этапа:

- *отделение корней*;
- *уточнение корней до заданной точности*.

Рассмотрим эти два этапа подробно.

8.1.2. Отделение корней нелинейного уравнения

Корень ξ уравнения $f(x) = 0$ считается *отделенным на отрезке* $[a, b]$, если на этом отрезке уравнение $f(x) = 0$ не имеет других корней. Такой отрезок $[a, b]$ можно назвать отрезком отделения корня. Следовательно, отделить корни, значит, разбить всю область допустимых значений корней на отрезки, в каждом из которых со-

держится только один корень уравнения. Отделение корней можно осуществить одним из двух методов: аналитическим или графическим.

Аналитическое отделение корней основано на следующем утверждении: если функция $f(x)$ непрерывна и монотонна на отрезке $[a, b]$ и принимает на концах отрезка значения разных знаков, то внутри отрезка $[a, b]$ содержится только один корень уравнения $f(x) = 0$.

Графическое отделение корней заключается в построении графика функции $f(x)$ и взятии в качестве отрезка отделения $[a, b]$ окрестность точки пересечения функции $f(x)$ с осью x .

Учитывая легкость построения графиков функций в MathCAD, в дальнейшем будет использоваться графический метод. Рассмотрим этот метод на конкретном примере.

Пример 8.1.1. Дано алгебраическое уравнение

$$x^3 + 3x^2 - 3 = 0. \quad (8.1.5)$$

Определить интервалы локализации корней этого уравнения.

На рис. 8.1.2 приведен график функции $f(x) = x^3 + 3x^2 - 3$, построенный в MathCAD. Видно, что в качестве интервалов локализации можно взять следующие интервалы: $[-3, -2]$, $[-2, -1]$, $[0, 2]$. Так, если алгебраическое уравнение третьей степени имеет три корня, то для всех трех корней определены интервалы локализации. ♦

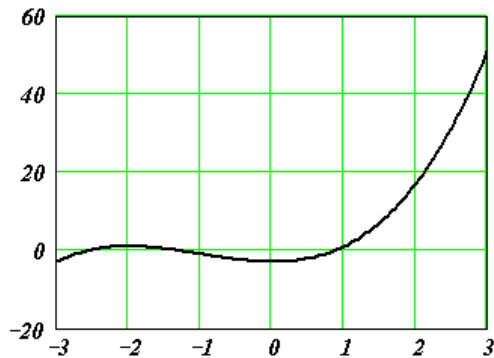


Рис. 8.1.2. Отделение корней уравнения (8.1.5)

Пример 8.1.2. Дано алгебраическое уравнение

$$x^3 - 6x^2 + 21x + 52 = 0. \quad (8.1.6)$$

Определить интервалы локализации корней этого уравнения.

На рис. 8.1.3 приведен график функции $f(x) = x^3 - 6x^2 + 21x + 52$, построенный в MathCAD. Видно, что в качестве интервала изоляции можно принять интервал $[-2, 0]$. Однако уравнение (8.1.6) имеет три корня. Следовательно, можно сделать вывод о наличии еще двух комплексных корней. ♦

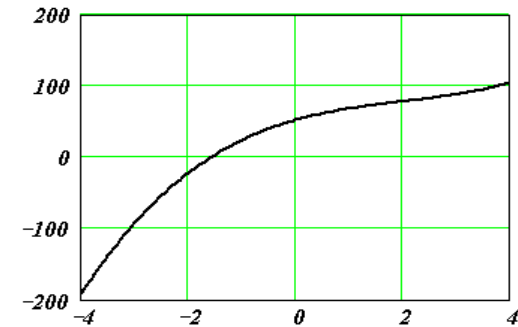


Рис. 8.1.3. Отделение корней уравнения (8.1.7)

8.1.3. Уточнение корней нелинейного уравнения

Под уточнением корня подразумевают процедуру уменьшения длины интервала локализации таким образом, чтобы корень уравнения всегда находился в новом интервале локализации (обозначим такой интервал локализации как $[\bar{a}, \bar{b}]$). Процедура уточнения завершается, если выполняется условие

$$|\bar{b} - \bar{a}| \leq \varepsilon, \quad (8.1.7)$$

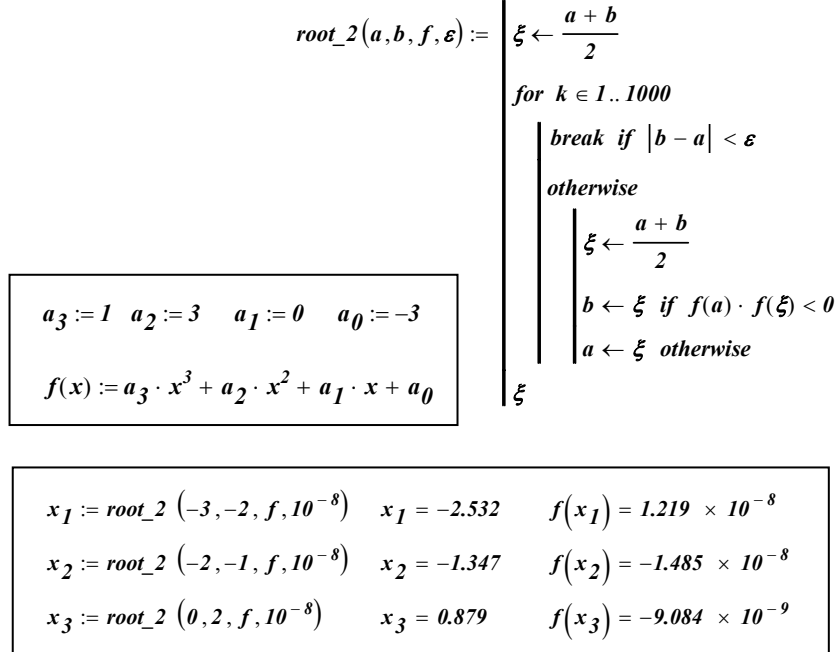
где ε – заданная точность вычисления корня. Для уточнения корня используются специальные вычислительные методы [11] такие, как метод деления отрезка пополам, метод хорд, метод касательных (метод Ньютона) и многие другие.

Пример 8.1.3. Используя метод деления отрезка пополам, уточнить корни уравнения (8.1.5). Реализовать метод в виде П-Ф.

Метод деления отрезка пополам заключается в делении длины «старого» интервала локализации пополам и принятии в качестве «нового» интервала локализации $[\bar{a}, \bar{b}]$ такого интервала из двух

полученных, у которого на концах функция $f(x)$ имеет разные знаки, т.е. $f(\bar{a}) \cdot f(\bar{b}) < 0$.

Описание П-Ф $root_2$ приведено на рис. 8.1.4. Здесь же показаны уточненные значения корней. Выполненная проверка показала высокую точность нахождения корней.



Заметим, что в теле П-Ф для организации итерационного цикла использовался оператор цикла *for*, что предотвращает «зацикливание» при неправильно заданных исходных данных. ♦

Функция $root$. В MathCAD для уточнения корней любого нелинейного уравнения (не обязательно только алгебраического) введена функция $root$, которая может иметь два или четыре аргумента, т.е. $root(f(x), x)$ или $root(f(x), x, a, b)$, где $f(x)$ – имя функции или арифметическое выражение, соответствующее решаемому нелинейному уравнению, x – скалярная переменная, относительно которой решается уравнение, a, b – границы интервала локализации корня.

Пример 8.1.4. Используя функцию $root(f(x), x)$, найти все три корня уравнения (8.1.6), включая и два комплексных (см. пример 8.1.2). Фрагмент документа MathCAD, вычисляющий эти корни приведен на рис. 8.1.5.

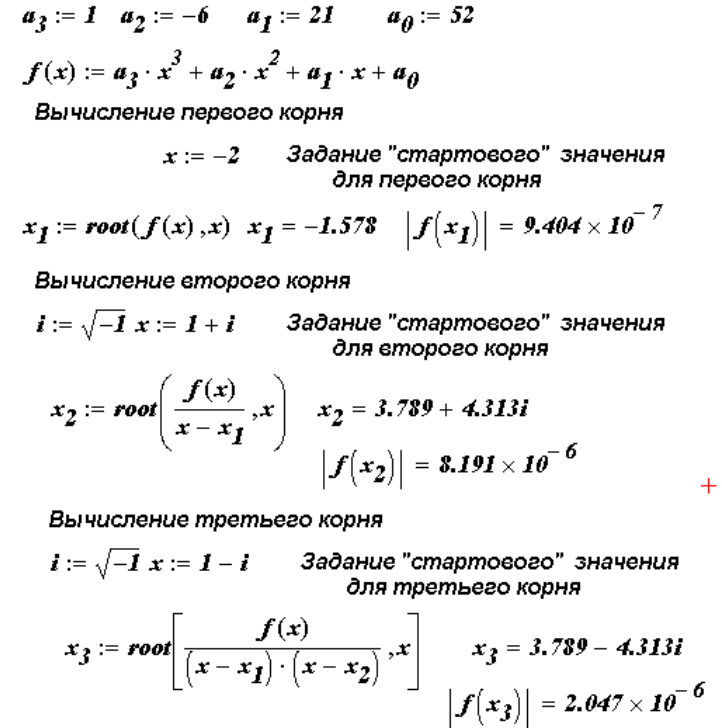


Рис. 8.1.5. Вычисление корней уравнения примера 8.1.2

Заметим, что для вычисления *всех трех корней* использовался прием понижения порядка алгебраического уравнения, рассмотренный в п. 8.1.1. ♦

Функция $root$ с двумя аргументами требует задания (до обращения к функции) переменной x начального значения корня из интервала локализации.

Рекомендации по использованию функции root:

- в функции *root* реализован метод секущих [11], поэтому требуется предварительная локализация корней решаемого нелинейного уравнения;

- для изменения точности, с которой функция *root* ищет корень, нужно изменить значение системной переменной TOL. Если значение TOL увеличивается, функция *root* будет сходиться быстрее, но ответ будет менее точен. Если значение TOL уменьшается, то функция *root* будет сходиться медленнее, но ответ будет более точен. Чтобы изменить значение TOL в определенной точке документа MathCAD, используйте определение вида $TOL := 0.01$. Чтобы изменить значение TOL для всего рабочего документа, необходимо обратиться к пункту меню **Математика** команда **Опции** в появившемся диалоговом окне в закладке *Встроенные переменные* изменить значение системной переменной TOL (по умолчанию она равна 0.001);

- если два корня расположены близко друг от друга, следует уменьшить TOL, чтобы различить их;

- если функция $f(x)$ имеет малый наклон около искомого корня, функция $root(f(x), x)$ может *сходиться* к значению r , отстоящему от корня достаточно далеко. В таких случаях для нахождения более точного значения корня необходимо уменьшить значение TOL. Другой вариант заключается в замене уравнения $f(x) = 0$ на эквивалентное $g(x) = 0$, где

$$g(x) = \frac{f(x)}{\frac{d}{dx} f(x)}.$$

Пример 8.1.5. Используя функцию *root*, вычислить изменения корня нелинейного уравнения $e^x - ax^2 = 0$ при изменении коэффициента a от 1 до 10 с шагом 1.

Фрагмент документа приведен на рис. 8.1.6. ♦

$$solve_root(a, x) := root(e^{-x} - a \cdot x^2, x)$$

$$a := 1..10 \quad x_0 := 0$$

$$x_a := solve_root(a, x_{a-1})$$

$$x_a =$$

0.704
0.54
0.459
0.408
0.371
0.344
0.322
0.304
0.289
0.276

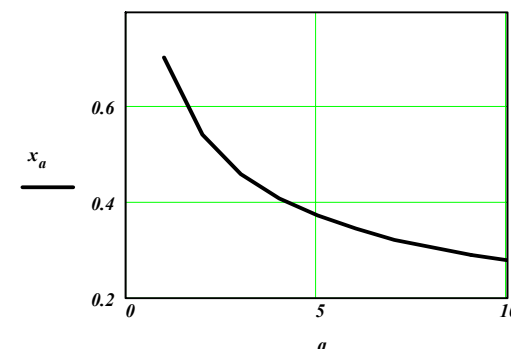


Рис. 8.1.6. Вычисление корней уравнения в цикле

Функция polyroots. Для вычисления всех корней алгебраического уравнения вида (8.1.2) порядка n (не выше 5) рекомендуется использовать функцию *polyroots*. Обращение к этой функции имеет вид $polyroots(v)$, где v – вектор, состоящий из $n+1$ проекций, равных коэффициентам алгебраического уравнения, т.е. $v_0 = a_0, v_1 = a_1, \dots, v_n = a_n$. Эта функция не требует проведения процедуры локализации корней.

Пример 8.1.6. Используя функцию *polyroots*, найти все три корня уравнения (8.1.6), включая и два комплексных (см. пример 8.1.2). Фрагмент документа MathCAD, вычисляющий эти корни приведен на рис. 8.1.7. ♦

Задание 8.1.1. Используя функцию *root*, вычислить корень из указанного интервала локализации следующих нелинейных уравнений:

- $3\sin(\sqrt{x}) + 0.35x - 3.8 = 0, \quad x \in [2, 3];$

- $3x - 14 + e^x - e^{-x} = 0, \quad x \in [1, 3].$ •

$$v := \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad \text{polyroots}(v) = \begin{pmatrix} -1.578 \\ 3.789 - 4.313i \\ 3.789 + 4.313i \end{pmatrix}$$

Рис. 8.1.7. Вычисление корней уравнения с использованием функции *polyroots*

Блок Given. При уточнении корня нелинейного уравнения можно использовать специальный вычислительный блок *Given*, имеющий следующую структуру:

$$\begin{aligned} &\langle \text{Начальные условия} \rangle \\ &\text{Given} \\ &\langle \text{Равенство} \rangle \\ &\langle \text{Ограничения} \rangle \\ &\langle \text{Вызов функции Find или Minerr} \rangle \end{aligned} \quad (8.1.8)$$

Решаемое уравнение задается в виде равенства, в котором используется «жирный» знак равно, вводимый с палитры ЛОГИЧЕСКИЙ. Ограничения содержат равенства или неравенства, которым должен удовлетворять искомый корень.

Функция *Find* уточняет корень уравнения, вызов этой функции имеет вид *Find(x)*, где *x* — переменная, по которой уточняется корень. Если корня уравнения на заданном интервале не существует, то следует вызвать функцию *Minerr(x)*, которая возвращает *приближенное значение корня*.

Для уточнения корней в функциях *Find(x)*, *Minerr(x)* реализованы различные численные алгоритмы и пользователю предоставляется возможность либо задать один из реализованных алгоритмов (к которому он имеет «максимальное доверие»), либо предоставить право выбора алгоритма пакету MathCAD. Для этого необходимо щелкнуть правой кнопкой мыши на имени функции *Find(x)* и в появившемся контекстном меню (см. рис. 8.1.8) выбрать подходящий алгоритм. Пользователю, не «искушенному» в вычислительных ме-

тодах рекомендуется предоставить выбор MathCAD, включив для этого команду **Автосбор**. В контекстном меню можно увидеть выбранный алгоритм.

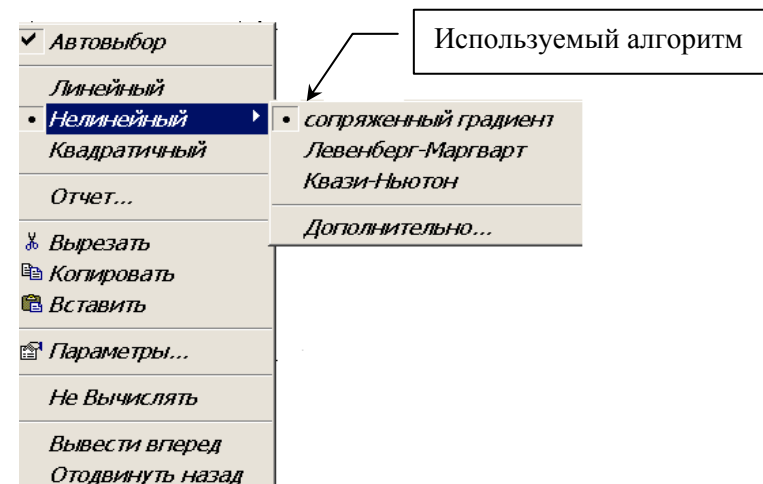


Рис. 8.1.8. Задание алгоритма функции *Find*

Аналогично можно задать алгоритм решения и для функции *Minerr(x)*.

Замечание 8.1.1. Использование численных методов в функциях *Find(x)*, *Minerr(x)* требует перед блоком *Given* задать начальные значения переменным, по которым осуществляется поиск корней уравнения.

Пример 8.1.7. Используя блок *Given*, вычислите корень уравнения примера 8.1.2 в интервале отделения $[-5, -1]$. Фрагмент документа приведен на рис. 8.1.9. ♦

Задание 8.1.2. Используя блок *Given*, вычислить корень из указанного интервала отделения следующих нелинейных уравнений и выполнить проверку найденных корней:

- $\sqrt{1-x} - \lg(x) = 0, \quad x \in [0, 1];$
- $1 - x + \sin x - \ln(1+x) = 0, \quad x \in [0, 2];$
- $3x - 4 \ln(x) - 5 = 0, \quad x \in [2, 4].$ •

$$a_3 := 1 \quad a_2 := -6 \quad a_1 := 21 \quad a_0 := 52$$

$$f(x) := a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x + a_0$$

$$x := -2 \quad \text{Задание начального значения}$$

Given

$$f(x) = 0 \quad \text{Задание нелинейного уравнения}$$

$$-4 \leq x \leq -1 \quad \text{Задание интервала отделения}$$

$$\text{Find}(x) = -1.578 \quad \text{Вычисленное значение корня}$$

Рис. 8.1.9. Решение уравнения с использованием *Given*

8.2. Решение систем уравнений

В этом параграфе приводится определение системы уравнений, и описываются основные этапы построения решения системы уравнений и конструкции MathCAD для этого.

8.2.1. Системы уравнений

Совокупность нескольких уравнений с несколькими неизвестными называется *системой уравнений*. Решением системы уравнений с несколькими неизвестными называется совокупность значений этих неизвестных, обращающая каждое уравнение в тождество. Например, система

$$\begin{cases} x^2 + y = 5 \\ x + y^2 = 3 \end{cases} \quad (8.2.1)$$

имеет решение $x = 2, y = 1$, так как при этих значениях уравнения обращаются в тождество (убедитесь в этом).

В зависимости от того, какие функции входят в систему уравнений, можно выделить два класса систем:

- алгебраические системы уравнений;
- трансцендентные системы уравнений.

Среди алгебраических систем уравнений особое место занимают системы линейных алгебраических уравнений (СЛАУ).

Рассмотрим этот класс систем подробнее.

8.2.2. Системы линейных алгебраических уравнений

Системой линейных алгебраических уравнений (СЛАУ) называется система вида:

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,m}x_m &= b_1; \\ a_{2,1}x_1 + a_{2,2}x_2 + \dots + a_{2,m}x_m &= b_2; \\ &\vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \dots + a_{n,m}x_m &= b_n, \end{aligned} \quad (8.2.2)$$

состоящая из n уравнений относительно m неизвестных. В матричном виде систему можно записать как

$$Ax = b, \quad (8.2.3)$$

где A — матрица размерности $n \times m$, b — вектор с m проекциями. В дальнейшем будем предполагать, что: а) матрица A является квадратной, т.е. $n = m$; б) матрица A является невырожденной, т.е. ее определитель не равен нулю.

При выполнении этих условий для любого вектора b всегда существует единственное решение — вектор с n проекциями.

Для вычисления решения СЛАУ следует использовать функцию *lsolve*, обращение к которой имеет вид: *lsolve(A,b)*, где A — матрица системы, b — вектор правой части.

Пример 8.2.1. Дана система уравнений

$$\begin{aligned} 2x_1 + 12x_2 - 4 &= 40; \\ x_2 + 7x_3 &= 4; \\ 10x_1 + 5x_2 &= -20. \end{aligned} \quad (8.2.4)$$

Используя функцию *lsolve*, найти решение этой системы и выполнить проверку найденного решения.

Фрагмент программы документа MathCAD приведен на рис. 8.2.1. Здесь же вычислен определитель матрицы. ♦

$$A := \begin{pmatrix} 2 & 12 & -4 \\ 0 & 1 & 7 \\ 10 & 5 & 0 \end{pmatrix} \quad b := \begin{pmatrix} 40 \\ 4 \\ -20 \end{pmatrix}$$

$$x := \text{Isolve}(A, b) \quad |A| = 810$$

$$x = \begin{pmatrix} -4 \\ 4 \\ 0 \end{pmatrix} \quad A \cdot x - b = \begin{pmatrix} 0.000000 \\ 0.000000 \\ 0.000000 \end{pmatrix}$$

$$|A \cdot x - b| = 0.000000000$$

Рис. 8.2.1. Решение системы уравнений

Задание 8.2.1. Используя функцию *Isolve*, вычислить решение следующих систем уравнений:

$$A \begin{cases} 100x_1 + 6x_2 - 2x_3 = 200; \\ 6x_1 + 200x_2 - 10x_3 = 600; \\ x_1 - 2x_2 - 100x_3 = 500. \end{cases} \quad (8.2.5)$$

$$B \begin{cases} 10x_1 + x_2 + x_3 = 12, \\ 2x_1 + 10x_2 + x_3 = 13, \\ 2x_1 + 2x_2 + 10x_3 = 14. \end{cases} \quad (8.2.6)$$

8.2.3. Решение нелинейных систем уравнений

MathCAD дает возможность находить решение системы уравнений численными методами, при этом максимальное число уравнений в MathCAD2001i доведено до 200.

Для решения системы уравнений необходимо выполнить следующие этапы.

Задание начального приближения для всех неизвестных, входящих в систему уравнений. При небольшом числе неизвестных этот этап можно выполнить графически, как показано в примере 8.2.2.

Пример 8.2.2. Дана система уравнений:

$$\begin{aligned} y &= x^2; \\ y &= 8 + 3x. \end{aligned} \quad (8.2.7)$$

Определить начальные приближения для решений этой системы. Фрагмент документа MathCAD, приведенный на рис. 8.2.2, содержит построение графиков двух функций, входящих в систему (8.2.7), координаты точек пересечения этих графиков могут быть взяты в качестве начальных приближений.

$$\begin{aligned} y_1(x) &:= x^2 & x &:= -3, -2.95.. 6 \\ y_2(x) &:= 8 + 3 \cdot x \end{aligned}$$

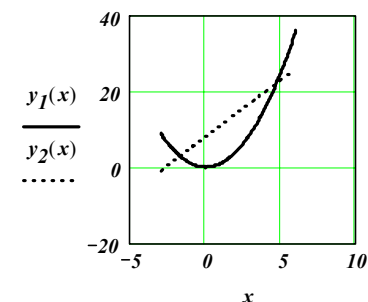


Рис. 8.2.2. Определение начальных приближений

Видно, что система имеет два решения: для первого решения в качестве начального приближения может быть принята точка $(-2, 2)$, а для второго решения – точка $(5, 20)$. ♦

Вычисление решения системы уравнений с заданной точностью (т.е. уточнение начального приближения). Для этого используется уже известный вычислительный блок *Given*, имеющий структуру (8.1.8). В этом блоке решаемые уравнения задаются в виде равенств, в которых используется «жирный» знак равно, вводимый с палитры ЛОГИЧЕСКИЙ. Ограничения содержат равенства или неравенства, которым должно удовлетворять искомое решение.

Функция *Find* вычисляет решение системы уравнений с заданной точностью, и вызов этой функции имеет вид *Find(x)*, где x – список переменных, по которым ищется решение. Начальные значения этим переменным задаются в блоке < Начальные условия >. Число аргументов функции должно быть равно числу неизвестных.

Следующие выражения недопустимы внутри блока решения:

- ограничения со знаком $\neq$;

- дискретная переменная или выражения, содержащие дискретную переменную в любой форме;

- блоки решения уравнений не могут быть вложены друг в друга, каждый блок может иметь только одно ключевое слово *Given* и имя функции *Find* (или *Minerr*).

Заметим, что, используя функцию *Find*, можно определить другую функцию пользователя. Например,

$$f(a, b, c, \dots) := \text{Find}(x, y, z, \dots).$$

Эта конструкция удобна для многократного решения системы уравнений для различных значений некоторых параметров $a, b, c, \dots$, непосредственно входящих в систему уравнений.

Сообщение об ошибке No solution was found. Try changing the guess value or the value of TOL or CTOL. (решение не найдено) при вычислении решения уравнений появляется, когда:

- поставленная задача может не иметь решения;
- для уравнения, которое не имеет вещественных решений, в качестве начального приближения взято вещественное число и наоборот;
- в процессе поиска решения последовательность приближений попала в точку локального минимума функционала невязки. Для поиска искомого решения нужно задать другие начальные приближения;

- возможно, поставленная задача не может быть решена с заданной точностью. Попробуйте увеличить значение TOL. Для этого необходимо обратиться к пункту меню **Математика** команда **Опции** в появившемся диалоговом окне в закладке *Встроенные переменные* изменить значение системной переменной TOL (по умолчанию она равна 0.001).

Если в процессе вычисления решения не может быть получено дальнейшее уточнение текущего приближения к решению, то функция *Minerr* возвращает это приближение. Функция *Find* в этом случае возвращает сообщение об ошибке.

Пример 8.2.3. Используя блок *Given*, вычислить все решения системы (8.2.7) примера 8.2.2. Выполнить проверку найденных решений. Фрагмент документа приведен на рис. 8.2.3. ♦

$$\begin{array}{ll}
 x := -2 \quad y := 2 & \text{Начальное приближение для} \\
 \text{Given} & \text{первого решения} \\
 y = 8 + 3 \cdot x & \\
 y = x^2 & \\
 s_A := \text{Find}(x, y) & \\
 s_A = \begin{pmatrix} -1.702 \\ 2.895 \end{pmatrix} & \text{Проекция первого решения} \\
 \\
 x := 5 \quad y := 20 & \text{Начальное приближение для} \\
 \text{Given} & \text{второго решения} \\
 y = 8 + 3 \cdot x & \\
 y = x^2 & \\
 x > 0 & \text{Ограничение на положительность} \\
 & \text{проекции } x \text{ второго решения} \\
 s_B := \text{Find}(x, y) & \\
 s_B = \begin{pmatrix} 4.702 \\ 22.105 \end{pmatrix} & \text{Проекция второго решения}
 \end{array}$$

Рис. 8.2.3. Решение системы уравнений примера 8.2.2

Задание 8.2.1. Используя блок *Given*, вычислить все решения следующих систем уравнений:

$$\begin{array}{ll}
 \text{А} \begin{cases} \sin x + 2y = 2, \\ \cos(y - 1) + x = 0.7. \end{cases} & \text{Б} \begin{cases} \sin y + x = -0.4, \\ 2y - \cos(x + 1) = 0. \end{cases} \\
 \text{В} \begin{cases} -\sin(x + 1) + y = 0.8, \\ \sin(y - 1) + x = 1.3. \end{cases} & \text{Г} \begin{cases} \sin(x) - 2y = 1, \\ \sin(y - 1) + x = 1.3. \end{cases}
 \end{array}$$

Проверить найденные решения. •

Пример 8.2.4. Используя функцию *Minerr*, вычислите решение системы уравнений

$$x + y = 0.95;$$

$$(x^2 + 1)^2 + (y^2 + 1)^2 = 5.5.$$

Фрагмент документа MathCAD показан на рис. 8.2.4. ♦

$$x := 0 \quad y := 1$$

Given

$$(x^2 + 1)^2 + (y^2 + 1)^2 = 5.5$$

$$x + y = 0.95$$

$$z := \text{Minerr}(x, y)$$

$$z = \begin{pmatrix} -0.106 \\ 1.056 \end{pmatrix} \quad \text{Найденное решение}$$

$$z_0 + z_1 = 0.95$$

Проверка найденного решения

$$[(z_0)^2 + 1]^2 + [(z_1)^2 + 1]^2 = 5.5$$

Рис. 8.2.4. Вычисление решения системы примера 8.2.4

ТЕМА 9. РЕШЕНИЕ ОПТИМИЗАЦИОННЫХ ЗАДАЧ В ПАКЕТЕ MathCAD

В ряде задач проектирования строительных конструкций существует необходимость определить значения параметров, которые доставляют максимум или минимум некоторому функционалу (или целевой функции), зависящему от этих параметров. Если на значения этих параметров не заданы какие-либо ограничения (например, требование положительности), то приходим к задаче *безусловной оптимизации* (или *оптимизации без ограничений*). Если заданы ограничения, определяющие *допустимые значения* параметров, то приходим к задаче *условной оптимизации* (*оптимизации с ограничениями*). Вторая задача отличается от первой тем, что решение ищется только *среди допустимых значений* или, иначе, на *допустимом множестве* параметров.

9.1. Решение оптимизационных задач без ограничений

Для этого используются две функции MathCAD:

- $\text{Maximize}(f, \langle \text{список параметров} \rangle)$ – вычисление точки максимума;
- $\text{Minimize}(f, \langle \text{список параметров} \rangle)$ – вычисление точки минимума, где f – имя минимизируемого функционала, определенного до обращения к функции; $\langle \text{список параметров} \rangle$ – содержит перечисление (через запятую) имен параметров, относительно которых решается оптимизационная задача.

Внимание! Перед обращением к функциям Maximize , Minimize (имена которых начинаются прописными буквами) обязательно задать начальное значение параметров оптимизации.

Пример 9.1.1. Дан функционал:

$$g(x, y, z) = 10\sqrt{x^2 - 2x + 36 + y^2 + 4y + 3z^2 - 18z}. \quad (9.1.1)$$

Определить значения x, y, z , при которых $g(x, y, z)$ достигает минимального значения.

Документ MathCAD, решающий эту задачу, приведен на рис. 9.1.1. ♦

$$g(x, y, z) := 10\sqrt{x^2 - 2 \cdot x + 36 + y^2 + 4 \cdot y + 3 \cdot z^2 - 18 \cdot z}$$

$$x := 1 \quad y := 1 \quad z := 1 \quad \text{Задание точки "старта"}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} := \text{Minimize}(g, x, y, z) \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} \quad g(x, y, z) = 20$$

Рис. 9.1.1. Пример минимизации функционала (9.1.1)

Пример 9.1.2. Дан функционал:

$$f(u, v) = \frac{1}{4\pi} e^{\frac{-41 - 32u - 16u^2 - 4v^2 + 20v}{32}}. \quad (9.1.2)$$

Определить значения u, v , при которых $f(u, v)$ достигает максимального значения.

Документ MathCAD, решающий эту задачу приведен на рис. 9.1.2. В последних строках документа выполнена проверка найденного решения на максимум. ♦

$$d(u, v) := \frac{1}{4 \cdot \pi} \cdot e^{\frac{-41 - 32 \cdot u - 16 \cdot u^2 - 4 \cdot v^2 + 20 \cdot v}{32}}$$

$$u := 0 \quad v := 0 \quad \text{Задание точки "старта"}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} := \text{Maximize}(d, u, v) \quad \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -1 \\ 2.5 \end{pmatrix} \quad d(u, v) = 0.0795775$$

$$d(u + 0.01 \cdot u, v + 0.01 \cdot v) = 0.0795673$$

$$d(u - 0.001 \cdot u, v - 0.001 \cdot v) = 0.0795774$$

Рис. 9.1.2. Максимизация функционала (9.1.2)

Задание 9.1.1. Дан функционал:

$$\varphi(x, y, z) = (\cos(x \cdot y) + \cos(y \cdot z)) \sin(x \cdot y \cdot z). \quad (9.1.3)$$

Определить точки минимума и максимума этого функционала. ●

9.2. Решение оптимизационных задач с ограничениями

Используются те же функции *Maximize*, *Minimize*, но они входят уже в блок решения *Given* (см. (8.1.8)) и перед ними размещаются ограничения в виде равенств или неравенств, определяющие допустимую область значений параметров оптимизации.

Пример 9.2.1. Дан функционал:

$$F(a, b) = 100(a - b)^2 - 50 \frac{a}{b} \quad (9.2.1)$$

и ограничения в виде

$$a + 2b \leq 5; \quad b \geq 1; \quad a \geq 0. \quad (9.2.2)$$

Определить значения a, b , доставляющие максимальное значение функционала (9.2.1) и удовлетворяющие неравенствам (9.2.2).

Документ MathCAD, решающий эту задачу, показан на рис. 9.2.1. Точка «старта» алгоритма берется из допустимой области, определяемой ограничениями (9.2.2).

$$F(a, b) := 100 \cdot (a - b)^2 - 50 \cdot \frac{a}{b}$$

$$a := 1 \quad b := 1$$

Given

$$a + 2 \cdot b \leq 5 \quad b \geq 1 \quad a \geq 0$$

$$\begin{pmatrix} a \\ b \end{pmatrix} := \text{Maximize}(F, a, b) \quad \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 2.5 \end{pmatrix} \quad F(a, b) = 625$$

$$a + 2 \cdot b = 5 \quad \text{Проверка ограничений}$$

$$b = 2.5$$

Рис. 9.2.1. Условная максимизация функционала (9.2.1)

Замечание 9.2.1. В оптимизационных задачах с ограничениями решение целесообразно определять из необходимых условий экстремума. Эти условия порождают систему уравнений (чаще всего нелинейных), которые располагаются в блоке *Given*, вместе с ограничениями, определяющими допустимую область. Само решение ищется с помощью функций *Find*, *Minerr* (подробно рассмотренных в п. 8.2.2).

Пример 9.2.2. В качестве тестового функционала при поиске точки минимума часто используется функционал Розенброка:

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2. \quad (9.2.3)$$

«Поверхность» этого функционала напоминает глубокий овраг, что сильно осложняет работу многих алгоритмов минимизации. Требуется вычислить точку минимума функционала при ограничениях:

$$x \geq 0; \quad y \geq 0; \quad y \leq 9 - x. \quad (9.2.4)$$

Документ MathCAD решения этой задачи показан на рис. 9.2.2.

$$f(x, y) := 100 \cdot (y - x^2)^2 + (1 - x)^2$$

$$x := 2 \quad y := 3$$

Given

$$\frac{d}{dx} f(x, y) = 0 \quad \frac{d}{dy} f(x, y) = 0 \quad \text{Условия минимума}$$

$$x \geq 0 \quad y \geq 0 \quad y \leq 9 - x \quad \text{Ограничения}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} := \text{Minerr}(x, y) \quad \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$f(x, y) = 3.538 \times 10^{-8}$$

Рис. 9.2.2. Минимизация функции Розенброка

Пример 9.2.3 (задача линейного программирования). Цех малого предприятия должен изготовить 100 изделий трех типов (x_1, x_2, x_3) и не менее 20 штук изделий каждого типа. На изделия уходит 4, 3.4 и 2 кг металла соответственно, при его общем запасе 340 кг, а также расходуются по 4.75, 11 и 2 кг пластмассы, при ее общем запасе 400 кг. Прибыль, полученная от каждого изделия равна 4, 3 и 2 р. Определить сколько изделий каждого типа необходимо выпустить для получения максимальной прибыли в рамках установленных запасов металла и пластмассы.

Документ MathCad, решающий эту задачу приведен на рис. 9.2.3.

$$f(x_1, x_2, x_3) := 4 \cdot x_1 + 3 \cdot x_2 + 2 \cdot x_3$$

$$x_1 := 1 \quad x_2 := 1 \quad x_3 := 1$$

Given

$$x_1 \geq 20 \quad x_2 \geq 20 \quad x_3 \geq 20$$

$$4 \cdot x_1 + 3.4 \cdot x_2 + 2 \cdot x_3 \leq 340 \quad \text{Ограничения}$$

$$4.75 \cdot x_1 + 11 \cdot x_2 + 2 \cdot x_3 \leq 700$$

$$x_1 + x_2 + x_3 = 100$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} := \text{Maximize} \left(f, x_1, x_2, x_3 \right) \quad \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 56 \\ 20 \\ 24 \end{pmatrix}$$

$$4 \cdot x_1 + 3.4 \cdot x_2 + 2 \cdot x_3 = 340$$

$$4.75 \cdot x_1 + 11 \cdot x_2 + 2 \cdot x_3 = 534 \quad \text{Проверка Ограничений}$$

$$x_1 + x_2 + x_3 = 100$$

Рис. 9.2.3. Решение задачи линейного программирования

Пример 9.2.4 (задача нелинейного программирования). Пусть вектор v состоит из трех проекций и дан функционал:

$$N(v) = \|v\|^2 + 2v_1 - v_2 + 2v_3.$$

Вычислить точку минимума этого функционала при ограничениях:

$$\sum_{i=1}^3 v_i = 1, \quad v_i \geq 0.2, \quad i = \overline{1, 3}.$$

Документ MathCAD, решающий эту задачу, показан на рис. 9.2.4.

$$N(v) := (|v|)^2 + v \cdot \begin{pmatrix} 2 \\ -1 \\ 2 \end{pmatrix}$$

$$v_2 := 1$$

Given

$$\sum v = 1 \quad v > \frac{1}{5}$$

$$v := \text{Minimize}(N, v)$$

$$v = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \text{Точка старта}$$

$$v = \begin{pmatrix} 0.2 \\ 0.6 \\ 0.2 \end{pmatrix} \quad \text{Решение}$$

$$N(v) = 0.64 \quad \sum v = 1 \quad \text{Проверка решения}$$

Рис. 9.2.4. Решение задачи нелинейного программирования

Задание 9.2.1 (задача линейного программирования). Дан функционал:

$$F(x) = 2x_0 + 9x_1 + 15x_2.$$

Определить точку максимума этого функционала при ограничениях:

$$x_0 \geq 0; \quad x_1 \geq 0; \quad x_2 \geq 0;$$

$$7x_0 + 3x_1 + x_2 \leq 47;$$

$$0.5x_0 - 3x_1 + 8x_2 \leq 25;$$

$$9x_0 + 2x_1 - 10x_2 \leq 29.$$

Вычислить значения функционала в этой точке.

Ответ: максимум функционала достигается в точке (0, 13, 8). •

Задание 9.2.2 (задача квадратичного программирования). Дан функционал:

$$Q(u, v, \omega) = u(15 - u) + 5v(20 - v) + 2\omega(12 - \omega).$$

Определить точку максимума этого функционала при ограничениях:

$$u \geq 0; \quad v \geq 0; \quad \omega \geq 0;$$

$$3u + 2v + 4\omega \leq 100;$$

$$u + 7v + \omega \leq 90.$$

Ответ: максимум функционала достигается в точке (7.5, 10, 6). •

ТЕМА 10. ОБРАБОТКА ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ В ПАКЕТЕ MathCAD

Рассматривается решение в пакете MathCAD наиболее распространенных задач статистической обработки данных: вычисление числовых характеристик, построение гистограмм и эмпирических зависимостей, сглаживание экспериментальных данных.

10.1. Моделирование и обработка статистических данных

При использовании в научных и инженерных исследованиях метода статистического моделирования (метод Монте-Карло) необходимо генерировать случайные (точнее, псевдослучайные) числа, распределенные по определенному закону.

Функции MathCAD для вычисления плотности распределения. В табл. 10.1.1 приведены функции (имена начинаются с буквы d), позволяющие вычислить значения функции плотности наиболее используемых распределений (x – значение, для которого вычисляется плотность распределения).

Таблица 10.1.1

Распределение	Функция MathCAD
Нормальное распределение $\frac{1}{\sqrt{2\pi} \cdot \sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \sigma > 0$	$dnorm(x, \mu, \sigma)$ $qnorm(\alpha, \mu, \sigma)$ $vnorm(m, \mu, \sigma)$
Распределение Пуассона $\frac{\lambda^x}{x!} e^{-\lambda}$ (x – целое неотрицательное число, $\lambda > 0$)	$dpois(x, \lambda)$ $qpois(\alpha, \lambda)$ $rpois(m, \lambda)$
Равномерное распределение $\frac{1}{b-a}$, если $x \in [a, b]$ 0, если $x \notin [a, b]$	$dunif(x, a, b)$ $qunif(\alpha, a, b)$ $runif(m, a, b)$
Биноминальное распределение $C_n^x p^x (1-p)^{n-x}, 0 \leq x \leq n$	$dbinom(x, n, p)$ $qbinom(\alpha, n, p)$ $rbinom(m, n, p)$

Окончание табл. 10.1.1

Распределение	Функция MathCAD
χ^2 – распределение 0, если $x \leq 0$; $\frac{1}{2^{n/2} \Gamma(n/2)} e^{-\frac{x}{2}} x^{n/2-1}$, если $x > 0$ ($n > 0$ – число степеней свободы)	$dchisq(x, n)$ $qchisq(\alpha, n)$ $rchisq(m, n)$
Распределение Стьюдента $\frac{\Gamma(\frac{n+1}{2})}{\sqrt{n\pi} \cdot \Gamma(\frac{n}{2})} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}$ ($n > 0$ – число степеней свободы)	$dt(x, n)$ $qt(\alpha, n)$ $rt(m, n)$

Функции MathCAD для вычисления квантилей распределения. Число x_α называется квантилем уровня α распределения с плотностью $p(x)$, если оно является решением следующего нелинейного уравнения:

$$\int_{-\infty}^{x_\alpha} p(x) dx = \alpha.$$

В табл. 10.1.1 в правой колонке второй строкой приведены функции MathCAD (имена начинаются с буквы *q*) вычисления квантилей соответствующих вычислений.

Функции MathCAD генерирования случайных векторов. Вектор, проекции которого являются случайными числами, распределенными по определенному закону, называется случайным вектором.

В табл. 10.1.1 в правой колонке третьей строкой приведены имена функций (начинаются с буквы *r*), вычисляющих случайный вектор с соответствующим распределением его проекций. Параметр m – размерность случайного вектора.

Функция $rnd(x)$ генерирует одно случайное число, равномерно распределенное в интервале $[0, 1]$.

Пример 10.1.1. На рис. 10.1.1 приведен фрагмент документа MathCAD, в котором генерируются два случайных вектора: ξ_N – проекции имеют нормальное распределение (математическое ожидание равно -20 , дисперсия 100); ξ_χ – проекции имеют χ^2 – распределение (с числом степеней свободы 10). Размерность векторов равна 100 . ♦

Функции MathCAD вычисления выборочных значений числовых характеристик. К числовым характеристикам случайной величины относятся: математическое ожидание (или среднее), дисперсия, среднеквадратическое отклонение и т.д. Часто возникает необходимость оценить эти характеристики по выборке значений случайной величины объема m . Такие оценки называют выборочными значениями числовых характеристик.

$$\xi_N := rnorm(100, -20, 10) \quad i := 1..100$$

$$\xi_\chi := rchisq(100, 10)$$

$$mean(\xi_N) = -21.505$$

$$var(\xi_N) = 95.484$$

$$mean(\xi_\chi) = 10.312$$

$$var(\xi_\chi) = 25.356$$

$$corr(\xi_N, \xi_\chi) = -0.014$$

$$cvar(\xi_N, \xi_\chi) = -0.704$$

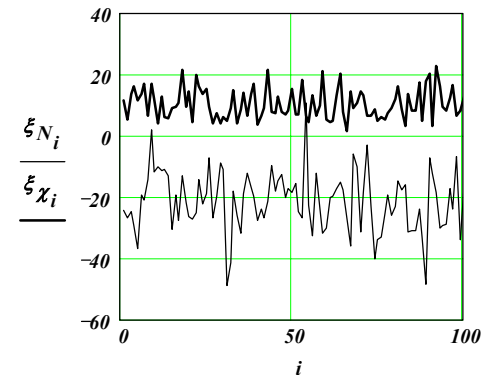


Рис. 10.1.1. Моделирование и обработка статистических данных

В табл. 10.1.2 приведены имена функций, вычисляющих выборочное значение часто используемых числовых характеристик.

Здесь X, Y – векторы размерности m , составленные из значений случайной величины X и Y .

Таблица 10.1.2

Числовые характеристики	Функция MathCAD
Математическое ожидание случайной величины X	$mean(X)$
Дисперсия случайной величины X	$var(X)$
Среднеквадратическое отклонение случайной величины X	$side(X)$
Медиана случайной величины X	$median(X)$
Мода случайной величины X	$mode(X)$
Корреляционный момент двух случайных величин X, Y	$cvar(X, Y)$

Пример 10.1.2. На рис. 10.1.1 приведен фрагмент документа MathCAD, в котором вычисляются выборочные значения некоторых характеристик по выборкам, полученным в примере 10.1.1. ♦

Функции MathCAD вычисления частот значений случайной величины (построение гистограмм). Введём некоторые определения.

Предположим, что дана выборка $\{x_i\}$, $i = \overline{1, N}$ случайной величины X (N – объём выборки). Введём $L+1$ точку $z_1 < z_2 < \dots < z_L < z_{L+1}$, при этом:

$$z_1 \leq \min\{x_i\}; \quad z_{L+1} \geq \max\{x_i\}. \quad (10.1.1)$$

Тогда число значений x_i , попавших в интервал $[z_k, z_{k+1}]$, $k = 1, \dots, L$, обозначим через n_k и назовём частотой.

Очевидно, что

$$\sum_{k=1}^L n_k = N. \quad (10.1.2)$$

Величину

$$w_k = \frac{n_k}{N} \quad (10.1.3)$$

назовём относительной частотой, для которой выполняется условие

$$\sum_{k=1}^L w_k = 1. \quad (10.1.4)$$

В качестве оценки плотности распределения вероятности непрерывной случайной величины X используют гистограмму относительных частот, т.е. систему прямоугольников, k -й из которых основанием имеет $z_{k+1} - z_k$, а высота p_k^* определяется по формуле

$$p_k^* = \frac{w_k}{z_{k+1} - z_k}, \quad k = 1, \dots, L \quad (10.1.5)$$

и имеет место приближенное тождество

$$p_k^* \approx p(x^*), \quad (10.1.6)$$

где x^* – некоторое число из интервала $[z_k, z_{k+1}]$. Поэтому при больших объёмах выборки и удачном выборе длин интервалов $[z_k, z_{k+1}]$ гистограмма является ступенчатой аппроксимацией плотности распределения $p(x)$.

Возникает вопрос: как сформировать интервалы $[z_k, z_{k+1}]$? Количество интервалов L рекомендуется вычислять по формуле $L = [1 + 3.222 \lg(N)] + 1$, где $[Q]$ – целая часть числа Q . Обычно интервалы берут равной длины $h = z_{k+1} - z_k$, и тогда узлы z_k определяются выражением:

$$z_k = \min\{x_i\} + h \cdot (k - 1), \quad k = 1, \dots, L + 1, \quad (10.1.7)$$

где

$$h = \frac{\max_i\{x_i\} - \min_i\{x_i\}}{L}.$$

Значения w_k , p_k^* вычисляются по частотам n_k . Поэтому для определения n_k по выборке $\{x_i\}$ в MathCAD включены две функции: $hist(int, X)$, $histogram(int, X)$.

Параметры функции $hist(int, X)$:

- int – массив длины $(L+1)$, составленный из значений z_k , $k = 1, \dots, L + 1$. Если параметр int задать целым числом, равным

числу интервалов L , то при выполнении функции формируется рабочий массив узлов $\{z_k\}$ по формулам (10.1.7), (10.1.8);

- X – массив длиной N , составленный из значений выборки $\{x_i\}$.

Результатом работы функции является одномерный массив $\{n_k\}$, $k=1, \dots, L$.

Параметры функции $histogram(int, X)$:

- int – массив длины $(L+1)$, составленный из значений z_k , $k=1, \dots, L+1$. Если int задать целым числом, равным числу интервалов L , то при выполнении функции формируется рабочий массив узлов $\{z_k\}$ по формулам (10.1.7), (10.1.8);

- X – массив длиной N , составленный из значений выборки $\{x_i\}$.

Результатом работы функций является матрица размером $L \times 2$, первый столбец содержит значения d_k (середины отрезков $[z_k, z_{k+1}]$, $k=1, \dots, L$, а второй столбец – значения n_k .

На наш взгляд, более предпочтительной является функция $histogram$, так как значения d_k более удобны для дальнейшего графического отображения вычисленных величин n_k , w_k , p_k .

Пример 10.1.3. Построить гистограммы относительных частот по выборкам случайных величин ξ_N , ξ_χ , определенных в примере 10.1.1. Объём выборки $N=1000$.

На рис. 10.1.2 показано построение гистограммы для случайной величины ξ_N , а на рис. 10.1.3 – для случайной величины ξ_χ с использованием функции $histogram$ при $L=11$. Середины отрезков d_k «откладываются» по оси абсцисс, а для отображения гистограммы задаётся параметр `solidbar` (команда **Формат** контекстного меню, закладка *Метки*). Точками на рисунках показаны значения соответствующих плотностей распределений, вычисленных при $x=d_k$.

$ORIGIN := 1 \quad N := 1000 \quad \xi_N := rnorm(N, -20, 10)$

$L := round(1 + 3.222 \cdot \log(N)) \quad L = 11$

$H_N := histogram(L, \xi_N) \quad d_N := H_N^{(1)} \quad n_N := H_N^{(2)}$

$h := d_{N_2} - d_{N_1} \quad h = 5.909 \quad p_N := \frac{1}{N \cdot h} \cdot n_N \quad k := 1..L$

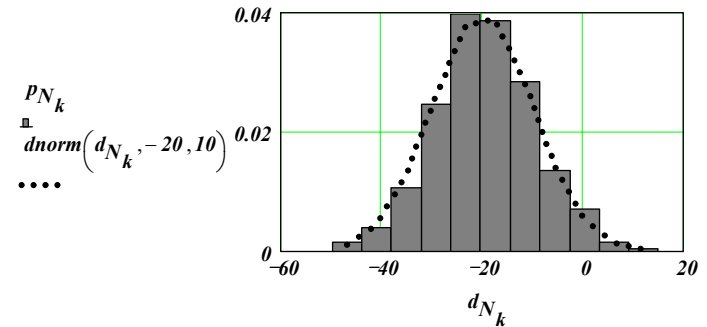


Рис. 10.1.2. Гистограмма выборки с нормальным распределением

$ORIGIN := 1 \quad N := 1000 \quad \xi_\chi := rchisq(N, 10)$

$L := round(1 + 3.222 \cdot \log(N)) \quad L = 11$

$H_\chi := histogram(L, \xi_\chi) \quad d_\chi := H_\chi^{(1)} \quad n_\chi := H_\chi^{(2)}$

$h := d_{\chi_2} - d_{\chi_1} \quad h = 3.545 \quad p_\chi := \frac{1}{N \cdot h} \cdot n_\chi \quad k := 1..L$

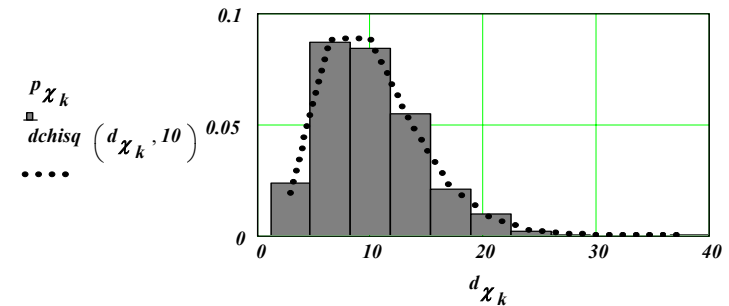


Рис. 10.1.3. Гистограмма выборки с χ^2 – распределением

10.2. Построение эмпирических зависимостей

Эмпирические зависимости. В научных и инженерных исследованиях часто возникает задача математического описания некоторого исследуемого процесса или объекта. Для определенности предположим, что связь независимой переменной x и зависимой переменной y выражается в общем случае нелинейной функций $y = f(x)$. Функция $f(x)$ является неизвестной, но значения переменных x , y могут быть измерены и результаты таких измерений представлены таблицей $\{x_i, \tilde{y}_i\}, i = 1, \dots, n$, где $\tilde{y}_i = f(x_i) + \eta_i$, η_i – погрешности измерений, имеющие случайную природу.

Задача построения эмпирической зависимости состоит в нахождении функции $S(x)$, как можно точнее аппроксимирующей неизвестную $f(x)$.

Решение этой задачи включает два этапа.

Этап 1. Выбор класса функций, из которого выбирается $S(x)$. Например, $S(x)$ выбирается из класса линейных функций вида $S(x) = a_0 + a_1x$, где a_0, a_1 – параметры. Таким образом, на этом этапе эмпирическая зависимость выбирается с «точностью» до её параметров. Выбор класса функций осуществляется на основе анализа диаграммы рассеяния (в декартовой системе координат наносятся точки $\{x_i, \tilde{y}_i\}, i = 1, \dots, n$) или на основе достоверной априорной информации. Например, из курса физики известно, что сопротивление металлического проводника линейно зависит от его температуры.

Этап 2. Вычисление неизвестных параметров функции $S(x)$ с использованием таблицы измерений. Параметры вычисляются из условия минимума некоторого функционала $F(a)$. Наиболее часто в качестве такого функционала используется функционал метода наименьших квадратов, имеющий вид:

$$F(a) = \sum_{i=1}^n (\tilde{y}_i - S(a, x_i))^2, \quad (10.2.1)$$

где запись $S(a, x)$ указывает на наличие у функции $S(x)$ параметров $\{a_j\}, j = 1, 2, \dots, M$.

Функции MathCAD для построения эмпирических зависимостей. В пакет MathCAD включены функции для вычисления па-

раметров различных функций $S(x)$. Обращения к этим функциям приведены в табл. 10.2.1, где X – вектор, составленный из значений $\{x_i\}, i = 1, \dots, n$, а Y – вектор, сформированный из значений $\{\tilde{y}_i\}, i = 1, \dots, n$.

Таблица 10.2.1

Функция	Назначение функции
$slope(X, Y)$	Вычисляет параметр a_1 линейной функции $S(x) = a_0 + a_1x$
$intercept(X, Y)$	Вычисляет параметр a_0 линейной функции $S(x) = a_0 + a_1x$
$regress(X, Y, m)$	Вычисляет параметры $ a_0, a_1, \dots, a_m ^T$ полиномиальной функции $S(x) = a_0 + a_1x + \dots + a_mx^m$ для любого m (на практике $m \leq 5$). Вычисленные параметры размещаются в результирующем векторе, начиная с четвертой проекции (см. пример 10.2.2)
$line(X, Y)$	Вычисляет параметры a_0, a_1 линейной зависимости $S(x) = a_0 + a_1x$
$linfit(X, Y, \Phi)$	Вычисляет параметры $a_1, a_2, \dots, a_m$ линейной комбинации $a_1\varphi_1(x) + a_2\varphi_2(x) + \dots + a_m\varphi_m(x)$. Базисные функции $\varphi_1(x), \dots, \varphi_n(x)$ являются проекциями вектора-функции $\Phi(x)$, формируемого до обращения к функции $linfit$
$genfit(X, Y, a_0, F)$	Вычисляет параметры $a_1, a_2, \dots, a_m$ нелинейной функции $S(x)$. До обращения формируется $F(x, a)$ – вектор-функция размерности $(m+1)$, составленный из самой функции $S(x)$ и частных производных $\frac{\partial S(x)}{\partial a_j}, j = 1, \dots, m$, a_0 – вектор размерности m , составленный из «стартовых» значений параметров $a_j, j = 1, \dots, m$ (см. пример 10.2.3)

Замечание 10.2.1. Для вычисления значения эмпирической зависимости в точке x с параметрами $\{a_j\}$, найденными с помощью

функции *regress*, используется функция *interp*(*v*, *X*, *Y*, *x*), где *v* – вектор, результат работы функции *regress* (см. пример 10.2.2). Вычисление других эмпирических зависимостей осуществляется с использованием функций пользователя (см. примеры 10.2.1, 10.2.3).

Замечание 10.2.2. Разница между функциями *linfit* и *genfit* заключается в том, что параметры $a_1, \dots, a_m$ функции *linfit* находятся из решения линейной системы уравнений, а в функции *genfit* из нелинейной системы, и поэтому задаётся точка старта (вектор *ao*) итерационной процедуры построения решения этой нелинейной системы.

Пример 10.2.1. Значения линейной зависимости $f(x) = 2 + 0.5 \cdot x$ измерялись в $n + 1$ узлах $\{x_i\}$ с относительным уровнем шума δ . Построение линейной эмпирической зависимости показано на рис. 10.2.1. ♦

ORIGIN := 0 *n* := 20 $a_0 := 2$ $a_1 := 0.5$ Точные значения параметров
i := 0..*n* $\delta := 0.1$ Задаваемый уровень шума измерен

$x_i := -2 + 0.2 \cdot i$ $y_i := a_0 + a_1 \cdot x_i$
 $\eta := \text{rnorm}(n + 1, 0, \delta \cdot \max(y))$ $y_\eta := y + \eta$ $\frac{|\eta|}{|y|} = 0.139$
 $A := \text{line}(x, y_\eta)$ $S(x) := A_0 + A_1 \cdot x$

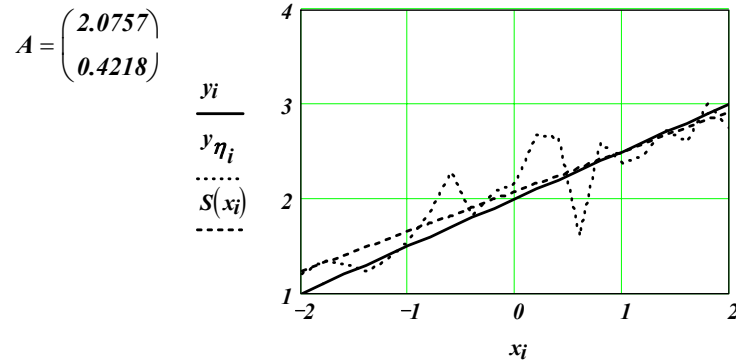


Рис. 10.2.1. Построение линейной эмпирической зависимости

Пример 10.2.2. По исходным данным, приведенным на рис. 10.2.2, построить полиномиальные эмпирические зависимости вида $S_m(x) = a_0 + a_1 x + \dots + a_m x^m$ при $m = 2$ и $m = 3$. Построение

этих эмпирических зависимостей показано в документе, приведенном на рис. 10.2.2. Видно, что при $m = 3$ эмпирическая зависимость в большей степени адекватна исходным экспериментальным данным, т.е. более близко подходит к измеренным значениям $\{\tilde{y}_i\}$. ♦

$x := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{pmatrix}$ $y := \begin{pmatrix} 0.8 \\ 3.5 \\ 8 \\ 15 \\ 19 \\ 15 \end{pmatrix}$ *ORIGIN* := 0 *i* := 0..5
 $v2 := \text{regress}(x, y, 2)$
 $\text{coef2} := \text{submatrix}(v2, 3, 5, 0, 0)$ $\text{coef2} = \begin{pmatrix} -8.15 \\ 7.995 \\ -0.634 \end{pmatrix}$
 $S_2(xI) := \text{interp}(v2, x, y, xI)$
 $v3 := \text{regress}(x, y, 3)$ $S_3(xI) := \text{interp}(v3, x, y, xI)$
 $\text{coef3} := \text{submatrix}(v3, 3, 6, 0, 0)$

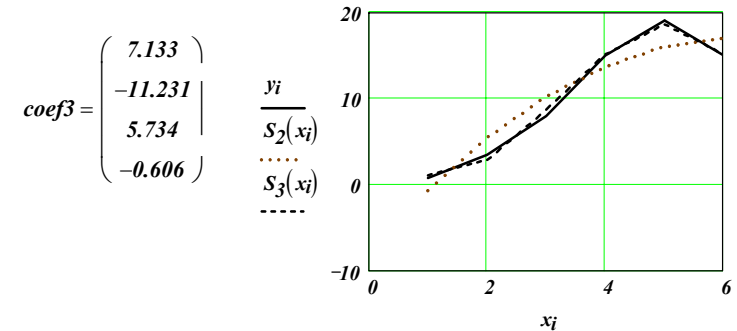


Рис. 10.2.2. Построение полиномиальных зависимостей

Пример 10.2.3. По данным, приведенным на рис. 10.2.3 (векторы *X* и *Y*), построить эмпирическую зависимость вида

$$S(x) = \exp(a_0 + a_1 \cdot x + a_2 \cdot x^2).$$

Построение выполнено в документе MathCAD, показанном на рис. 10.2.3, с использованием функции *genfit*. Перед обращением к этой функции формируется вектор-функция $\Phi(x, a)$, первая проекция которого содержит формулу эмпирической зависимости, а три других проекции частные производные от этой формулы по пара-

метрам a_0, a_1, a_2 соответственно. Окружностями отмечены измеренные значения $\tilde{y}_i$. ♦

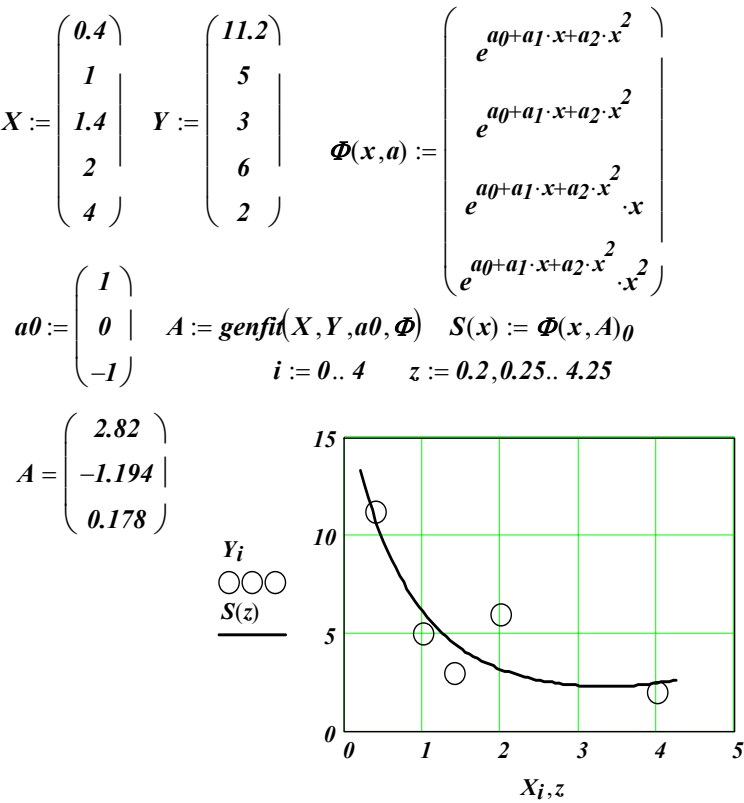


Рис. 10.2.3. Построение нелинейной зависимости

Кроме функций вычислений параметров «универсальных» эмпирических зависимостей, приведенных в табл. 10.2.1, определены функции, вычисляющие параметры «специальных» эмпирических зависимостей. В табл. 10.2.2 приведено обращение к этим функциям.

Таблица 10.2.2

Функция	Назначение функции
---------	--------------------

<i>expfit</i> (X,Y,ao)	Вычисляет параметры a_1, a_2, a_3 экспоненциальной зависимости $S(x) = a_1 e^{a_2 x} + a_3$. Вектор ao (размерности 3) определяет точку старта, т.е. задает «начальное» значение для a_1, a_2, a_3
<i>lgffit</i> (X,Y,ao)	Вычисляет параметры a_1, a_2, a_3 зависимости $S(x) = \frac{a_1}{1 + a_2 e^{-a_3 x}}$. Вектор ao (размерности 3) определяет «стартовые» значения для a_1, a_2, a_3
<i>lnfit</i> (X,Y)	Вычисляет параметры a_1, a_2 зависимости $S(x) = a_1 \cdot \ln(x) + a_2$
<i>logfit</i> (X,Y, ao)	Вычисляет параметры a_1, a_2, a_3 зависимости $S(x) = a_1 \cdot \ln(x + a_2) + a_3$. Вектор ao (размерности 3) задаёт «стартовые» значения для a_1, a_2, a_3
<i>pwrfit</i> (X,Y,ao)	Вычисляет параметры степени зависимости $S(x) = a_1 x^{a_2} + a_3$. Вектор ao (размерности 3) задаёт «стартовые» значения для a_1, a_2, a_3
<i>sinfit</i> (X,Y,ao)	Вычисляет параметры синусоидальной зависимости $S(x) = a_1 \cdot \sin(x + a_2) + a_3$. Вектор ao (размерности 3) задаёт «стартовые» значения для a_1, a_2, a_3

Пример 10.2.4. По данным, приведенным на рис. 10.2.4, построить эмпирическую зависимость вида

$$S(x) = a_1 e^{a_2 x} + a_3.$$

Построение этой эмпирической зависимости показано в документе MathCAD, приведенном на рис. 10.2.4. ♦

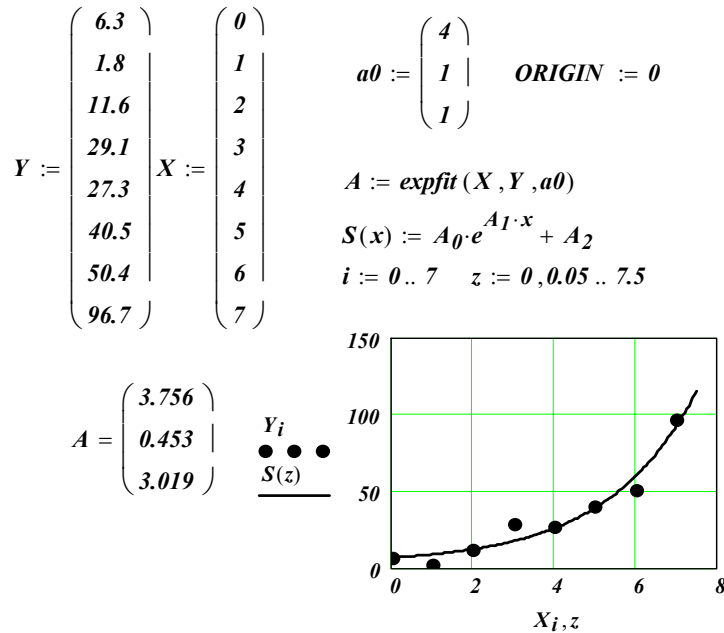


Рис. 10.2.4. Построение экспоненциальной зависимости

Приведенные в табл. 10.2.1 и 10.2.2 функции осуществляют построение эмпирических зависимостей, которые можно назвать «глобальными», так как при вычислении их параметров используется вся таблица исходных данных. В ряде экспериментов возникает необходимость построить эмпирическую зависимость по некоторому небольшому набору исходных данных, которые находятся в окрестности точки x , для которой будет вычисляться значение эмпирической формулы. Такие эмпирические формулы можно назвать «локальными».

Для построения «локальных» эмпирических зависимостей в MathCAD включена функция $\text{loess}(X, Y, d)$, приближающая исходные данные полиномом второй степени. Формальные параметры X, Y — это массивы с исходными данными, а параметр $d > 0$ определяет размер области приближаемых данных (рекомендуемое начальное значение 0.75 — 0.85). Чем больше величина d , тем больше сглаживание исходных данных. При больших значениях d работа функции $\text{loess}(X, Y, d)$ становится аналогичной функции $\text{regress}(X, Y, 2)$. Результатом работы функции $\text{loess}(X, Y, d)$ является вектор v , используемый

функцией $\text{interp}(v, X, Y, z)$, которая вычисляет значение построенной эмпирической формулы в точке z .

Пример 10.2.5. На рис. 10.2.5 приведен документ MathCAD, реализующий вычислительный эксперимент по построению локальной эмпирической формулы с помощью функции $\text{loess}(X, Y, d)$ при $d = 0.8$ и $d = 3$. Из графиков видно, что формула при $d = 3$ достаточно хорошо приближает полином второго порядка. ♦

$\text{ORIGIN} := 0 \quad n := 20$

$a_0 := 2 \quad a_1 := 0.5 \quad a_2 := -0.3$ Точные значения параметров

$i := 0..n \quad \delta := 0.1$ Задаваемый уровень шума измерения

$x_i := -2 + 0.2 \cdot i \quad y_i := a_0 + a_1 \cdot x_i + a_2 \cdot (x_i)^2$

$\eta := \text{rnorm}(n + 1, 0, \delta \cdot \max(y)) \quad y_\eta := y + \eta \quad \frac{|\eta|}{|y|} = 0.152$

$d_1 := 0.4 \quad d_2 := 3.2 \quad v1 := \text{loess}(x, y_\eta, d_1) \quad v2 := \text{loess}(x, y_\eta, d_2)$

$S_1(x1) := \text{interp}(v1, x, y_\eta, x1) \quad S_2(x1) := \text{interp}(v2, x, y_\eta, x1)$

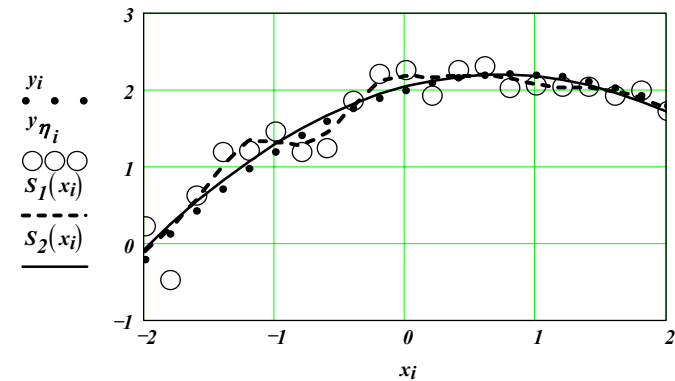


Рис. 10.2.5. Построение локальных эмпирических зависимостей

Сглаживание экспериментальных данных. В большинстве случаев результаты измерения содержат случайные погрешности, которые необходимо «отфильтровать». Это можно сделать путем построения эмпирических зависимостей. Однако в MathCAD опре-

делены специальные «сглаживающие» функции, имена которых содержат слово *smooth* (гладкий). Таких функций три:

- **medsmooth(Y,m)** – сглаживает массив Y , состоящий из n элементов с использованием скользящего медианного фильтра с размером окна сглаживания m (m должно быть нечетным и $m < n$). Результатом работы является массив, содержащий n сглаженных значений массива Y ;

- **supsmooth(X,Y)** – сглаживает массив Y , состоящий из n значений, измеренных в точках x_i , которые являются элементами массива X . Сглаживание осуществляется построением линейной регрессии по k – ближайшим точкам x_i с адаптивным выбором «размера» k окна сглаживания. Элементы массива X должны быть упорядочены по возрастанию;

- **ksmooth(X,Y,b)** – сглаживает массив Y , состоящий из n значений, измеренных в точках x_i , которые являются элементами массива X . Сглаживание осуществляется на основе формулы

$$\hat{f}_i = \frac{\sum_{j=1}^n h\left(\frac{x_i - x_j}{b}\right) \cdot \tilde{y}_j}{\sum_{j=1}^n h\left(\frac{x_i - x_j}{b}\right)}, \text{ где } h(t) = \frac{1}{\sqrt{2\pi} \cdot 0.37} \cdot \exp\left(-\frac{t^2}{2 \cdot (0.37)^2}\right).$$

Видно, что сглаженное значение $\hat{f}_i$ есть сумма измерений $\tilde{y}_j$ с экспоненциальными весами, величина которых зависит от параметра b – чем больше величина параметра, тем в большей степени сглаживаются «зашумленные» значения $\tilde{y}_j$.

Пример 10.2.6. На рис. 10.2.6 приведен документ MathCAD, реализующий вычислительный эксперимент по сглаживанию «зашумленных» данных (относительный уровень шума равен 38%) с помощью функции *ksmooth* при двух значениях параметра b ($b=0.3$ – относительная ошибка сглаживания равна 0.161 и $b=0.9$ – относительная ошибка сглаживания равна 0.141).

формирование "точных" данных

$$i := 0..n \quad x_i := i \cdot 0.1 \quad y_i := f(x_i)$$

зашумление "точных" данных

$$\delta := 0.2 \quad \eta := \text{rnorm}(n+1, 0, \delta \cdot \max(y)) \quad y_\eta := y + \eta$$

сглаживание фильтром ksmooth

$$Y1 := \text{ksmooth}(x, y_\eta, 3 \cdot 0.1)$$

$$Y2 := \text{ksmooth}(x, y_\eta, 9 \cdot 0.1) \quad \frac{|Y1 - y|}{|y|} = \blacksquare \quad \frac{|Y2 - y|}{|y|} = \blacksquare$$

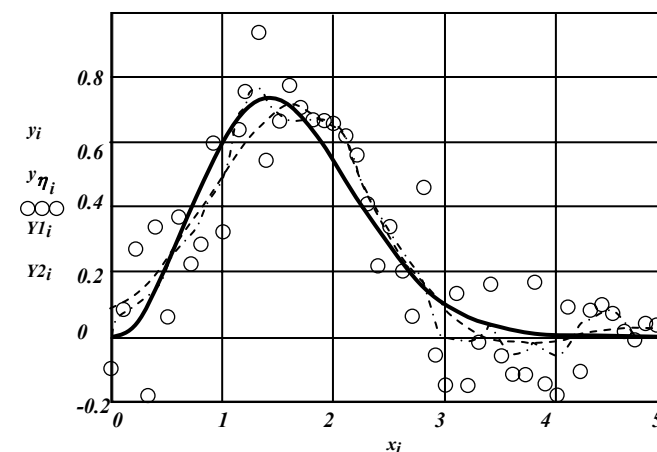


Рис. 10.2.6. Сглаживание данных функций *ksmooth*

ЗАКЛЮЧЕНИЕ

В данном пособии авторы стремились изложить основы программирования в пакете MathCAD и продемонстрировать решение часто встречающихся задач в этом пакете. К сожалению, ограниченный объем учебного пособия не позволил уделить внимание ряду важных возможностей пакета. Это прежде всего относится к символьным вычислениям, к решению обыкновенных дифференциальных уравнений и дифференциальных уравнений в частных производных. Некоторым «оправданием» служит хорошее освещение этих тем в литературе [2, 3, 9,10].

Авторы желают читателям (особенно студентам и аспирантам), чтобы пакет MathCAD стал для них «дружелюбным и незаменимым» помощником в решении широкого круга научно-инженерных задач.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Дьяконов В.П. Компьютерная математика. Теория и практика / В.П. Дьяконов. – М.: Изд-во Нолидж, 2001. – 296 с.
2. Дьяконов В.П. MathCAD2000: Учебный курс / В.П. Дьяконов. – СПб: Питер, 2000. – 596 с.
3. Дьяконов В.П. MathCAD 8 Professional в математике, физике и Internet / В.П. Дьяконов, И.В. Абраменкова. – М.: Нолидж, 1999. – 512 с.
4. Дьяконов В.П. Matlab: Учебный курс / В.П. Дьяконов. – СПб: Питер, 2001. – 560 с.
5. Потемкин В.Г. Система инженерных и научных расчетов Matlab 5.x.: В 2-х т. – М. ДИАЛОГ–МИФИ, 1999. – 672 с.
6. Дьяконов В.П. Mathematica 4: Учебный курс / В.П. Дьяконов. – СПб: Питер, 2000 – 482 с.
7. Дьяконов В.П. Математическая система Maple V R3/R4/R5 / В.П. Дьяконов. – М.: Солон, 1998. – 381 с.
8. Воскобойников Ю.Е. Программирование в математическом пакете MathCAD. Методические указания / Ю.Е. Воскобойников, Т.Н. Воскобойникова. – Новосибирск: Изд-во НГАСУ, 1999. – 32 с.
9. Кирьянов Д.В. Самоучитель MathCAD2001 / Д.В. Кирьянов. – СПб.: БХВ-Петербург, 2002. – 459 с.
10. Очков В.Ф. Советы пользователям MathCAD / В.Ф. Очков. – М.: Изд-во МЭИ, 2001. – 196 с.
11. Демидович В.П. Основы вычислительной математики / В.П. Демидович, И.А. Марон. – М.: Наука, 1970. – 486 с.