СПРАВОЧНИК МАГНИТНОГО ДИСКА
(кафедра полупроводниковой электроники)

В. А. Буслов

ПАКЕТЫ ПРИКЛАДНЫХ МАТЕМАТИЧЕСКИХ ПРОГРАММ

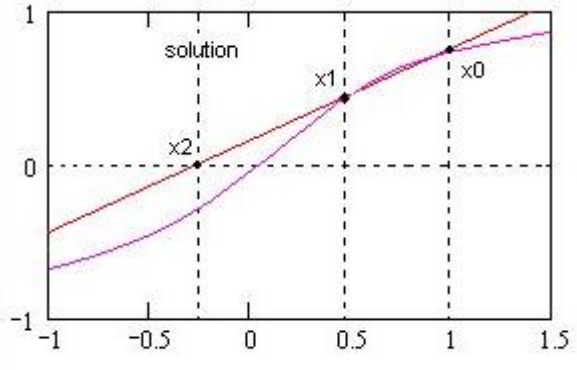
Учебное пособие

Приклпакеты.doc (наименование файла) 8,44 Мбайт (объем файла) 05.12.2006 (дата) 11,3 уч.-изд.л. (объем издания)

В. А. Буслов

ПАКЕТЫ ПРИКЛАДНЫХ МАТЕМАТИЧЕСКИХ ПРОГРАММ

Учебное пособие



Воронеж 2006

ГОУВПО «Воронежский государственный
технический университет»

В. А. Буслов

ПАКЕТЫ ПРИКЛАДНЫХ МАТЕМАТИЧЕСКИХ ПРОГРАММ

Утверждено Редакционно-издательским советом
университета в качестве учебного пособия

Воронеж 2006

УДК 22.183.4

Буслов В.А. Пакеты прикладных математических программ: учеб. пособие/ В.А. Буслов. Воронеж: ГОУВПО «Воронежский государственный технический университет», 2006. 243 с.

В учебном пособии рассматриваются основные характеристики прикладных математических пакетов MathCAD, Maple, MatLAB, STATISTICA, представлены методы решения наиболее распространенных в инженерной и учебной практике математических задач.

Учебное пособие соответствует требованиям Государственного образовательного стандарта высшего профессионального образования по направлению 140400 «Техническая физика», специальности 210104 “Микроэлектроника и твердотельная электроника”, дисциплине «Пакеты прикладных математических программ».

Учебное пособие подготовлено в электронном виде в текстовом редакторе MS Word for Windows и содержится в файле Приклматпакеты.doc.

Ил. 180 . Библиогр.: 7 назв.

Научный редактор д-р физ.-мат. наук, проф. С.И. Рембеза

Рецензенты: Кафедра физики полупроводников и микроэлектроники Воронежского государственного университета (зав. кафедрой д-р физ.-мат. наук, профессор Е.Н. Бормонтов);
канд. физ.-мат. наук, доц. Е.В. Бордаков

ВВЕДЕНИЕ

Компьютерные технологии всё шире внедряются в жизнь современного общества. Масштабы этого процесса всё увеличиваются и ускоряются, поскольку использование компьютеров позволяет экономить самый драгоценный и невосполнимый ресурс человека – его время. Компьютеры легко справляются там, где человек сталкивается с трудностями – там, где требуется однообразные длительные расчеты, обработка огромных массивов информации, выявление статистических закономерностей. Для решения подобных задач используются сложные специальные математические методы, грамотно запрограммировать которые сможет далеко не каждый пользователь.

Поэтому для нужд пользователя, которому необходимо просто решить задачу и получить ответ в удобной форме и не изучать для этого математику и программирование, были созданы пакеты прикладных программ, направленные на решение наиболее типичных часто встречающихся вычислительных задач. В настоящее время существует немало таких пакетов, каждый из которых, как правило, наиболее хорошо подходит для решения определенного типа задач, будь то решение систем уравнений, работа с данными в виде матриц большой размерности или статистическая обработка данных. От пользователя требуется просто ввести данные и указать, какого типа преобразование нужно сделать – и программа сама выберет оптимальный (с точки зрения ее разработчиков) математический метод решения. Однако не следует слепо полагаться на квалификацию разработчиков, все особенности учесть невозможно, поэтому необходимо тщательно проверять полученные результаты, чтобы избежать досадных ошибок.

В настоящем учебном пособии описаны приемы решения наиболее часто встречающихся задач при помощи мате-

матических пакетов MathCAD и Maple, работа с данными в матричном виде при помощи MATHLAB и статистическая обработка данных в среде STATISTICA.

Данное учебное пособие соответствует требованиям Государственного образовательного стандарта высшего профессионального образования по подготовке дипломированных специалистов по направлению 140400 «Техническая физика», специальности 210104 “Микроэлектроника и твердотельная электроника”, дисциплинам «Пакеты прикладных математических программ» и «Основы научных исследований и техника эксперимента».

1. ОБЩИЕ СВЕДЕНИЯ О МАТЕМАТИЧЕСКИХ ПАКЕТАХ ПЭВМ

История создания, назначение и сравнительная характеристика, требования к ресурсам ПЭВМ различных математических пакетов.

До широкого распространения персональных компьютеров вычислительные средства были доступны лишь специалистам, как правило, обладавшим высокой квалификацией в области математики. По мере распространения компьютеров, расширялся круг пользователей, которые не были одновременно специалистами в математике, вычислительных методах и программировании. Для нужд таких пользователей были разработаны программные продукты, предназначенные для решения некоторого набора наиболее часто встречающихся стандартных математических задач. Первые математические пакеты не отличались универсальностью и удобным интерфейсом, но по мере развития возможностей операционных систем происходит их совершенствование.

Современные математические пакеты обладают широким набором выполняемых операций и даже проявляют "искусственный интеллект" при решении наиболее сложных задач. В данном курсе будут рассмотрены некоторые их широко распространенных прикладных математических пакетов:

- **MathCAD** – позволяет решать очень широкий круг наиболее часто встречающихся математических задач, подходит как для оформления результатов расчетов, так и отлично взаимодействует и легко встраивается в популярные текстовые редакторы. Основными достоинствами является универсальность, стандартный интерфейс и простая запись условий и решений в общепринятом для математики виде, что позволяет легко освоить его начинающим пользователям и неспециалистам в области высшей математики;

- **Maple** – это программный пакет для автоматизации как численных, так и символьных вычислений. Основным достоинством данного пакета является представление вычислений в виде, похожем по внешнему виду и построению на входной язык многих популярных языков программирования. В настоящее время символьный процессор данного пакета, признанный одним из лучших в мире, частично интегрирован в MathCAD;

- **MATHLAB** (матричная лаборатория) – это интерактивная система, в которой основным элементом данных является массив. Это позволяет решать различные задачи, связанные с научными и техническими вычислениями, исходные данные которых представлены в виде больших массивов данных;

- **STATISTICA** – программный пакет, предназначенный в первую очередь для проведения статистических расчетов, статистической обработки данных, управления базами данных, с широкими возможностями графического отображения результатов, содержащий широчайший набор процедур и методов для нужд науки, бизнеса и управления производством.

Из представленных пакетов наиболее универсальными являются MathCAD и Maple. MATHLAB и STATISTICA более узко специализированы и предназначены для решения конкретных задач.

Данные прикладные пакеты отличаются умеренными требованиями к аппаратной части вычислительной системы: процессор Pentium 90 (или аналогичный по производительности), устройство чтения компакт-дисков, операционная система Windows (любая), 16 Мбайт оперативной памяти, SVGA видеокарта (желательно наличие поддержки OpenGL и 3D ускоритель).

2. ПАКЕТ MATHCAD

2.1. Основные понятия. Элементы окна MathCAD. Работа

с файлами MathCAD. Назначение команд меню и кнопок панелей инструментов. Печать документов.

MathCAD является математическим редактором, позволяющим проводить разнообразные научные и инженерные расчеты, начиная от элементарной арифметики и заканчивая сложными реализациями численных методов. MathCAD, в отличие от большинства других современных математических приложений, построен в соответствии с принципом WYSIWYG ("What You See Is What You Get" - "что Вы видите, то и получите"). Поэтому он очень прост в использовании, в частности, из-за отсутствия необходимости сначала писать программу, реализующую те или иные математические расчеты, а потом запускать ее на исполнение. Вместо этого достаточно просто вводить математические выражения с помощью встроенного редактора формул, причем в виде, максимально приближенном к общепринятому, и тут же получать результат. Кроме того, можно изготовить на принтере печатную копию документа или создать страницу в Интернете именно в том виде, который этот документ имеет на экране компьютера при работе с MathCAD.

Создатели MathCAD сделали все возможное чтобы пользователь, не обладающий специальными знаниями в программировании (а таких большинство среди ученых и инженеров), мог в полной мере приобщиться к достижениям современной вычислительной науки и компьютерных технологий. Для эффективной работы с редактором MathCAD достаточно базовых навыков пользователя. С другой стороны, профессиональные программисты могут извлечь из MathCAD намного больше, создавая различные программные решения, существенно расширяющие возможности, непосредственно заложенные в MathCAD. В соответствии с проблемами реальной жизни, математикам приходится решать одну или несколько из следующих задач:

- ввод на компьютере разнообразных математических выражений (для дальнейших расчетов или создания документов, презентаций, Web-страниц);
- проведение математических расчетов;
- подготовка графиков с результатами расчетов;
- ввод исходных данных и вывод результатов в текстовые файлы или файлы с базами данных в других форматах;
- подготовка отчетов работы в виде печатных документов;
- подготовка Web-страниц и публикация результатов в Интернете;
- получение различной справочной информации из области математики.

Таким образом, следует хорошо представлять себе, что в состав MathCAD входят несколько интегрированных между собой компонентов - это мощный текстовый редактор для ввода и редактирования как текста, так и формул, вычислительный процессор - для проведения расчетов согласно введенным формулам, и символьный процессор, являющийся, по сути, системой искусственного интеллекта. Сочетание этих компонентов создает удобную вычислительную среду для разнообразных математических расчетов и, одновременно, документирования результатов работы.

После того как MathCAD 2001 установлен на компьютере и запущен на исполнение, появляется основное окно приложения, показанное на рис. 1. Оно имеет ту же структуру, что и большинство приложений Windows. Сверху вниз располагаются заголовок окна, строка меню, панели инструментов (стандартная и форматирования) и *рабочий лист,* или *рабочая область,* документа (worksheet). Новый документ создается автоматически при запуске MathCAD. В самой нижней части окна находится строка состояния. Не забывая о сходстве редактора MathCAD с обычными текстовыми редакторами, можно интуитивно понять назначение большинства кнопок на панелях инструментов.

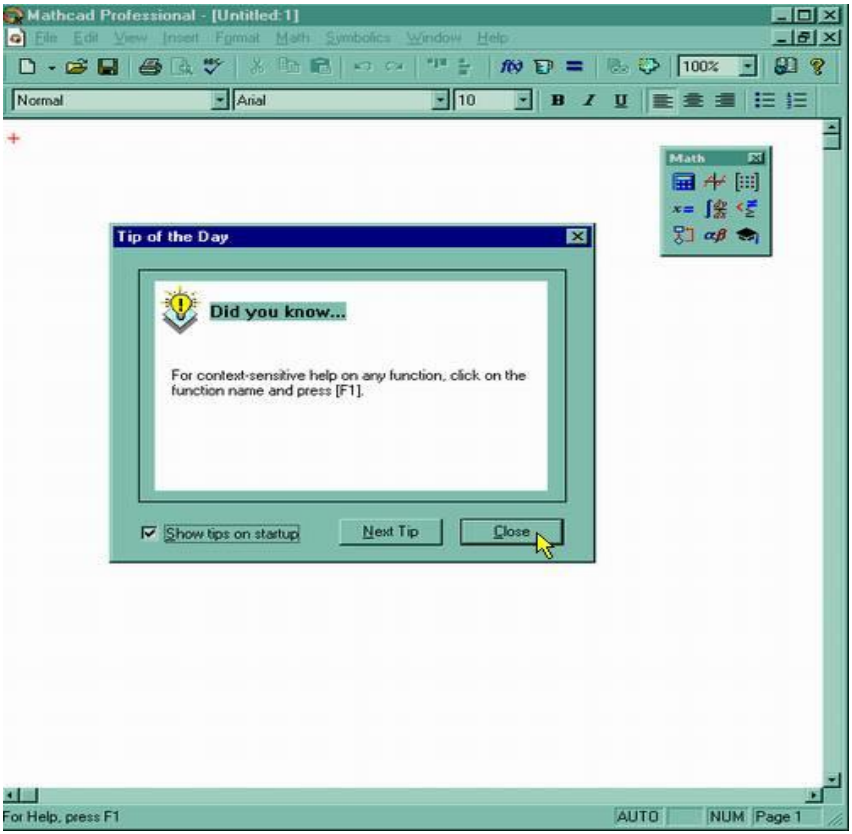


Рис. 1. Основное окно MathCAD.

В MathCAD интерфейс пользователя интуитивен и сходен с другими приложениями Windows. Его составные части:
- верхнее меню, или строка меню (menu bar);
- панели инструментов (toolbars) **Standard** (Стандартная) и **Formatting** (Форматирование);
- панель инструментов **Math** (Математика) и доступные через нее дополнительные математические панели инструментов;
- рабочая область (worksheet);
О строка состояния (status line, или status bar);

- всплывающие, или контекстные, меню (pop-up menus, или context menus);
- диалоговые окна, или диалоги (dialogs).

Большинство команд можно выполнить как с помощью меню (верхнего или контекстного), так и панелей инструментов или клавиатуры.

Меню. Строка меню располагается в самой верхней части окна MathCAD. Она содержит девять заголовков, щелчок мышью на каждом из которых приводит к появлению соответствующего меню с перечнем сгруппированных по действию команд:

- **File** (Файл) - команды, связанные с созданием, открытием, сохранением, пересылкой по электронной почте и распечаткой на принтере файлов с документами;
- **Edit** (Правка) - команды, относящиеся к правке текста (копирование, вставка, удаление фрагментов и т. п.);
- **View** (Вид) - команды, управляющие внешним видом документа в окне редактора MathCAD, а также команды, создающие файлы анимации;
- **Insert** (Вставка) - команды вставки различных объектов в документы;
- **Format** (Формат) - команды форматирования текста, формул и графиков;
- **Math** (Математика) - команды управления вычислительным процессом;
- **Symbolics** (Символика) - команды символьных вычислений;
- **Window** (Окно) - команды управления расположением окон с различными документами на экране;
- **Help** (Справка) - команды вызова контекстно-зависимой справочной информации, доступа к Центру Ресурсов, опции Совета Дня и сведений о версии программы.

Панели инструментов. Панели инструментов служат для быстрого (в один щелчок мыши) выполнения наиболее часто применяемых команд. Все действия, которые можно

выполнить с помощью панелей инструментов, доступны и через верхнее меню. На рис. 2 изображено окно MathCAD с тремя основными панелями инструментов, расположенными непосредственно под строкой меню.

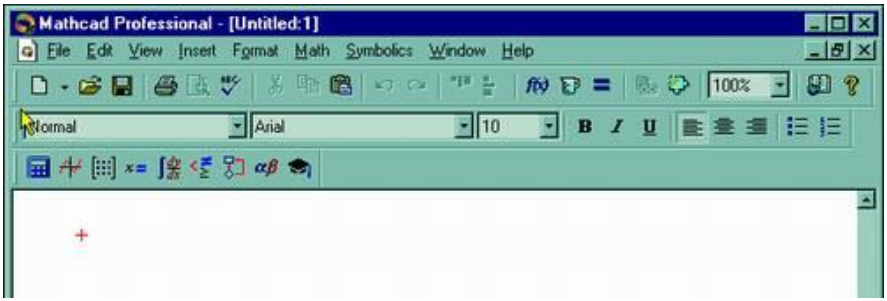


Рис. 2. Окно MathCAD с тремя основными панелями инструментов.

Кнопки в панелях сгруппированы по сходному действию команд:

- **Standard** (Стандартная) - служит для выполнения большинства операций, таких как действия с файлами, редакторская правка, вставка объектов и доступ к справочным системам;
- **Formatting** (Форматирование) - для форматирования (изменения типа и размера шрифта, выравнивания и т. п.) текста и формул;
- **Math** (Математика) - для вставки математических символов и операторов в документы.

Группы кнопок на панелях инструментов разграничены по смыслу вертикальными линиями - *разделителями.* При наведении указателя мыши на любую из кнопок рядом с кнопкой появляется *всплывающая подсказка* -короткий текст, поясняющий назначение кнопки. Наряду со всплывающей подсказкой, более развернутое объяснение готовящейся операции можно отыскать на строке состояния.

Панель **Math** (Математика) предназначена для вызова на экран еще девяти панелей (рис. 3), с помощью которых, собственно, и происходит вставка математических операций в документы. В прежних версиях MathCAD эти математические панели инструментов назывались *палитрами* (palettes) или *наборными панелями.* Чтобы показать какую-либо из них, нужно нажать соответствующую кнопку на панели **Math** (см. рис. 3). Перечислим назначение математических панелей:

- **Calculator** (Калькулятор) - служит для вставки основных математических операций, получила свое название из-за схожести набора кнопок с кнопками типичного калькулятора;
- **Graph** (График) - для вставки графиков;
- **Matrix** (Матрица) - для вставки матриц и матричных операторов;
- **Evaluation** (Выражения) - для вставки операторов управления вычислениями;
- **Calculus** (Вычисления) - для вставки операторов интегрирования, дифференцирования, суммирования;
- **Boolean** (Булевы операторы) - для вставки логических (булевых) операторов;
- **Programming** (Программирование) - для программирования средствами MathCAD;
- **Greek** (Греческие символы) - для вставки греческих символов;
- **Symbolic** (Символика) - для вставки символьных операторов.

Печать документа. Чтобы распечатать экземпляр активного документа на принтере, нажмите клавиши <Ctrl>+<P> или кнопку с изображением принтера на стандартной панели инструментов (приводит к мгновенной распечатке всего активного документа *с* текущими опциями печати и установками принтера).

Для более активного управления процессом печати служат следующие пункты меню **File** (Файл):

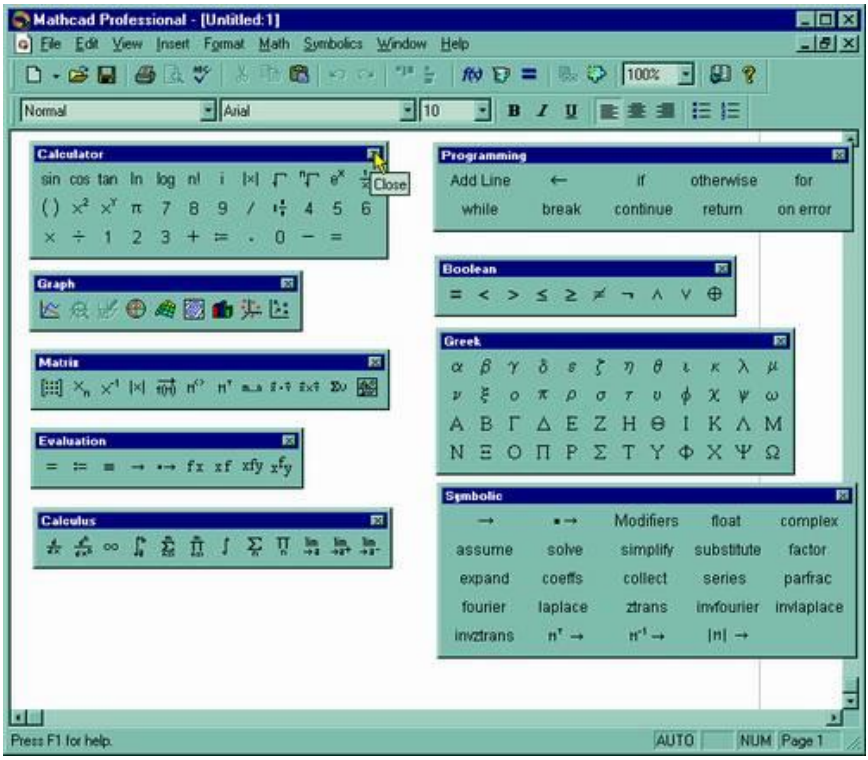


Рис. 3. Еще девять панелей инструментов MathCAD.

- **Page Setup** (Параметры страницы) - опции страницы вывода активного документа на печать (стандартный размер страницы, тип подачи бумаги, поля);
- **Print Preview** (Просмотр) - предварительный просмотр на экране вывода на печать активного документа;
- **Print** (Печать) - собственно печать активного документа с возможностью выбора принтера (если установлено несколько принтеров), смены установок принтера (таких, как качество печати, разрешение, количество печатных копий документа и диапазон печатаемых страниц).

Выбор любого из этих пунктов меню **File** приводит к

раскрытию одноименного диалогового окна, в котором следует задать соответствующие опции печати и начать саму распечатку документа. Все эти возможности реализованы в MathCAD так, как это принято в Windows.

2.2. Переменные, матрицы. Внутренние переменные, имена переменных пользователя, диапазон изменения переменных. Определение вектора, матрицы. Оператор вывода значений переменой, выражения. Оператор присваивания. Глобальный оператор присваивания.

Основные инструменты математика - это операции с переменными величинами и функциями. В MathCAD переменные, операторы и функции реализованы в интуитивной форме, т. е. выражения в редакторе вводятся и вычисляются так, как они были бы написаны на листе бумаги. Порядок вычислений в документе MathCAD также очевиден: математические выражения и действия воспринимаются процессором слева направо и сверху вниз.

Вот основные действия, которые пользователь может совершать для определения и вывода переменных и функций. Чтобы определить переменную, достаточно ввести ее имя и присвоить ей некоторое значение, для чего служит оператор присваивания:

Чтобы присвоить переменной новое значение, например переменную x сделать равной 10:

1. Введите в желаемом месте документа имя переменной, например x.

2. Введите оператор присваивания с помощью клавиши <Shift>+<:> или нажатием соответствующей кнопки **Definition** (Присваивание) на панели инструментов **Calculator** (Калькулятор) или **Evaluation** (Выражения), как показано на рис. 4.

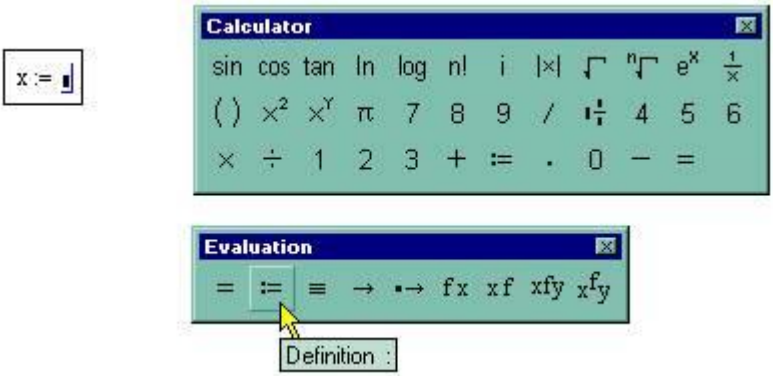


Рис. 4. Присваивание переменной x значения.

3. Введите в появившийся местозаполнитель новое значение переменной (10).

Ввести новое значение переменной возможно как в виде числа, так и в виде математического выражения, содержащего другие переменные:

$$x := 10$$

$$y := (x - 3)^2 + 1$$

а также в виде строкового выражения:

$$s := \text{"Hello,"}$$

Если переменная с некоторым именем создается в данном документе впервые, то для ввода оператора присваивания, вместо двоеточия, допускается использовать символ равенства "=", который MathCAD автоматически заменит символом присваивания.

Не вполне соответствующий общепринятому математическому стилю вид оператора присваивания (не =, а :=) является, на самом деле, компромиссом, связанным с назначением MathCAD как системы программирования. Этот оператор показывает, что он действует, в отличие от других, не слева направо, а справа налево, поскольку значение (справа) зада-

ется переменной (слева). И если непосвященного математика внешний вид этого оператора может ввести в некоторое заблуждение, то пользователю MathCAD он прямо говорит о действии, выполняемом в данном месте документа: значение переменной *не выводится* на экран (о чем говорит знак =), а некоторое значение *присваивается* (:=) данной переменной.

Глобальное присваивание **(Global Definition):** Рассмотрим различие между операторами обычного присваивания и *глобального присваивания* (процесс его вставки в документ показан на рис. 5).

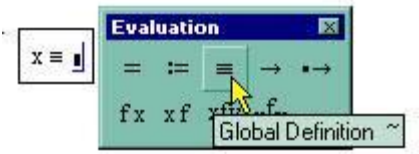


Рис. 5. Вставка оператора глобального присваивания

Для того чтобы вычислить выражение, содержащее некоторую переменную или функцию, необходимо, чтобы этой переменной ранее в документе было присвоено какое-либо значение. Иначе будет выдаваться сообщение об ошибке. Однако если в любой части документа (например, в самом низу) вставить оператор глобального присваивания, то переменная будет определена в любой части документа.

Переменная может быть задана в виде диапазона чисел (ранжированная переменная), а так же в виде вектора или массива. Матрицей называется совокупность переменных (массив), каждая из которых имеет уникальный набор индексов, однозначно определяющий ее положение в совокупности. Матрица, элементы которой имеют только один индекс, называется вектором. Матрицы, элементы которых имеют число индексов более 2 принято называть тензорами. В системе MathCAD существует большое число способов задания

ранжированных переменных и матриц. Рассмотрим основные.

Ранжированные переменные в MathCAD являются разновидностью векторов и предназначены, главным образом, для создания *циклов* или итерационных вычислений. Простейший пример ранжированной переменной - это массив с числами, лежащими в некотором диапазоне с некоторым шагом. Например, для создания ранжированной переменной s с элементами 0,1,2,3,4,5:

- Поместите курсор ввода в нужное место документа.
- Введите имя переменной (s) и оператор присваивания ":=".
- Нажмите кнопку **Range Variable** (Ранжированная переменная) на панели **Matrix** (Матрица), показанную на рис. 6, либо введите символ точки с запятой с клавиатуры.

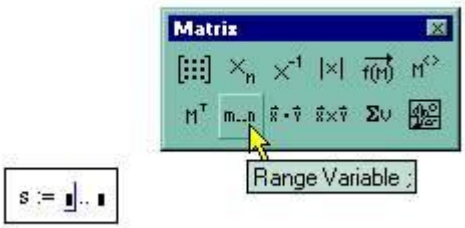


Рис. 6. Создание ранжированной переменной.

- В появившиеся местозаполнители введите левую и правую границы диапазона изменения ранжированной переменной 0 и 5.

Результат создания ранжированной переменной показан на рис. 7.

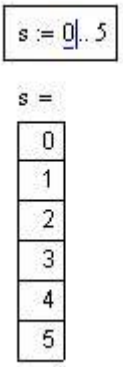


Рис. 7. Завершение создания и вывод ранжированной переменной.

Чтобы создать ранжированную переменную с шагом, не равным 1, например, 0, 2, 4, 5, 8:
- Создайте ранжированную переменную в диапазоне от 0 до 8 (см. рис. 6).
- Поместите линии ввода на значение начала диапазона (0).
- Введите запятую.

- В появившийся местозаполнитель введите значение шага изменения ранжированной переменной (2).
Созданная ранжированная переменная будет иметь значения от 0 до 8 включительно, с шагом, равным 2. То есть шаг будет равен разности между первым и вторым элементами, разделенными запятой.

Самый простой и наглядный способ создания вектора или матрицы заключается в следующем:
1. Нажмите кнопку **Matrix or Vector** (Матрица или вектор) на панели **Matrix** (Матрица) (рис. 8) либо клавиши <Ctrl>+<M>, либо выберите пункт меню **Insert / Matrix** (Вставка / Матри-

ца).
2. В диалоговом окне **Insert Matrix** (Вставка матрицы) задайте целое число столбцов и строк матрицы, которую хотите создать. Например, для создания вектора 3x1, введите показанные на рис. 8 значения.

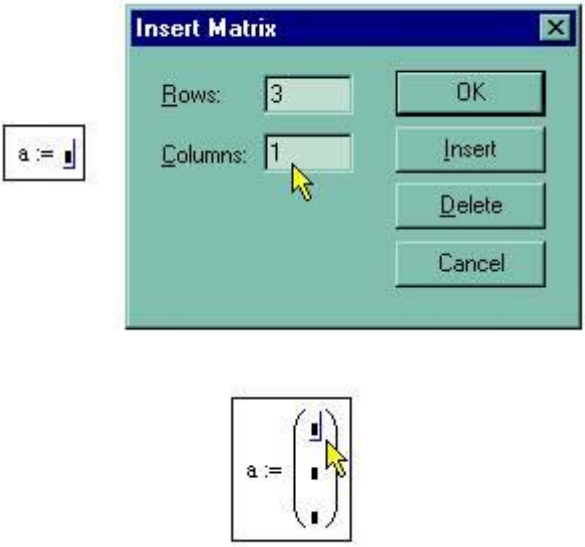


Рис. 8. Создание и ввод элементов матрицы

3. Нажмите кнопку **ОК** или **Insert** (Вставить) - в результате в документ будет вставлена заготовка матрицы с определенным числом строк и столбцов.
В данном случае был создан вектор – матрица из одного столбца. Добавление в уже созданную матрицу строк или столбцов производится точно так же:
- Выделите линиями ввода элемент матрицы, правее и ниже которого будет осуществлена вставка столбцов и (или) строк.
- Вставьте в него матрицу, как было описано выше. При этом допускается задание числа столбцов или строк равным нулю

(рис. 9).

- Заполните местозаполнители недостающих элементов матрицы.

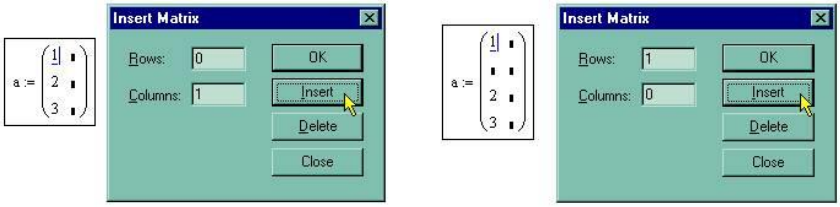


Рис. 9. Добавление столбцов и строк в имеющуюся матрицу.

В местозаполнители элементов матрицы можно вставлять не только числа (действительные или комплексные), но и любые математические выражения, состоящие из переменных, операторов, встроенных и пользовательских функций:

$$x := 3$$

$$A := \begin{pmatrix} \sin(x) \\ x \end{pmatrix}$$

$$A = \begin{pmatrix} 0.141 \\ 3 \end{pmatrix}$$

Другой способ создания матрицы заключается в определении любого количества его элементов. Это можно сделать:

- присваивая значения непосредственно отдельным элементам массива. Для ввода нижних индексов элемента надо нажать кнопку "[".;

- применяя ранжированные переменные. Любой из этих способов позволяет присвоить нужное значение как всем элементам массива (см. рис. 10), так и части из них, либо даже одному-единственному элементу. В последнем случае создается массив, размерность которого задается индексами введенного элемента, а неопределенным элементам по умолча-

нию присваиваются нулевые значения.

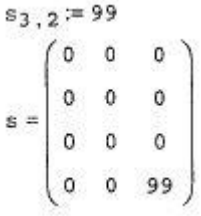


Рис. 10. Создание матрицы по описанию одного элемента

Системные переменные *(system variables)*
-TOL - точность численных методов;
-CTOL - точность выполнения выражений, используемая в некоторых численных методах;
-ORIGIN - номер начального индекса в массивах;
-PRNPRECISION - установка формата данных при выводе в файл;
- PRNCOLWIDTH - установка формата столбца при выводе в файл;
- CWD - строковое представление пути к текущей рабочей папке.
Предустановленные значения системных переменных перечислены рис. 11. Их можно поменять в любой части документа, присвоив соответствующей переменной новое значение. Кроме того, переопределение значения переменной для всего документа производится при помощи команды **Math / Options / Built-in Variables** (Математика / Опции / Встроенные переменные) в диалоговом окне **Math Options** (Опции математики), приведенном на рис. 11. Чтобы в любой момент вернуть значения по умолчанию, нажмите кнопку **Restore Defaults** (Восстановить установки по умолчанию).

Допустимые имена переменных и функций. Перечислим, какие символы можно, а какие нельзя применять

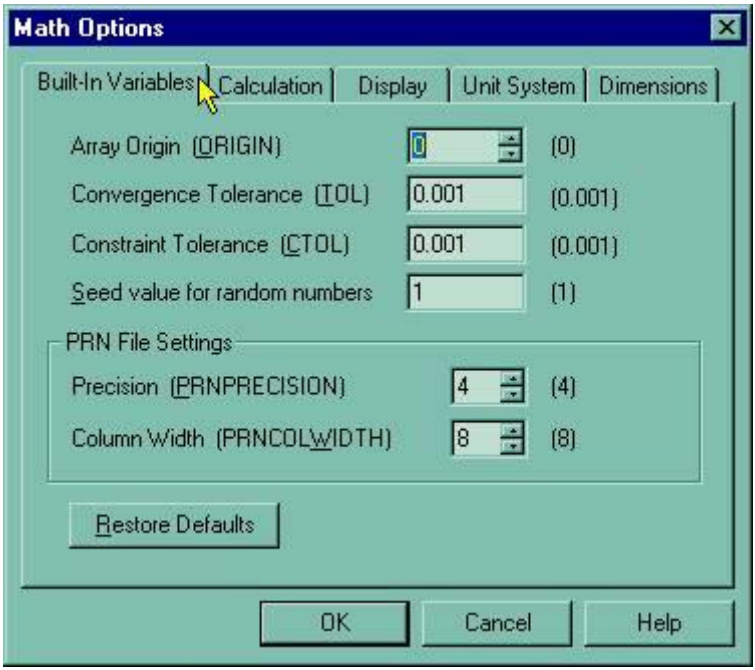


Рис. 11. Вкладка **Built-in Variables** диалога **Math Options**

в именах, которые пользователь дает переменным и функциям, и перечислим ряд ограничений на присваивание имен. Допустимые символы:

- большие и маленькие буквы - MathCAD различает регистр: так, имена X и x определяют разные переменные. Кроме того, различает и шрифт, например имена X и X воспринимаются как разные;
- числа от 0 до 9;
- символ бесконечности (клавиши <Ctrl>+<Shift>+<Z>);
- штрих (клавиши <Ctrl>+<F7>);
- греческие буквы - они вставляются с помощью панели **Greek** (Греческие символы);
- символ подчеркивания;

- символ процента;
- нижний индекс.

Внимание! *С осторожностью используйте нижний индекс в определении имен переменных и функций (вводится нажатием точки "."), не путая его с индексом векторной переменной (вводится нажатием квадратной скобки "["). Чтобы ввести имя с нижним индексом, введите букву "к", затем точку ".", после чего линии ввода опустятся чуть ниже, и только затем сам нижний индекс так.*

Рассмотрим **ограничения** на имена переменных и функций:

- имя не может начинаться с цифры, символа подчеркивания, штриха или процента. Использование букв русского языка не допускается;
- символ бесконечности должен быть только первым в имени;
- все буквы *в* имени должны иметь один стиль и шрифт;
- имена не могут совпадать с именами встроенных функций, констант и размерностей, например, sin или TOL. Тем не менее, допускается их переопределение, но тогда одноименная встроенная функция больше не будет использоваться по первоначальному назначению;
- MathCAD не различает имен переменных и функций: если сначала определить функцию f (x), а потом переменную f, то в оставшейся части документа будет утерян доступ к функции f(*x).*

Операторы. Каждый оператор в MathCAD обозначает некоторое математическое действие в виде символа. В полном согласии с терминологией, принятой в математике, ряд действий (например, сложение, деление, транспонирование матрицы и т. п.) реализован в MathCAD в виде встроенных операторов, а другие действия (например, sin, erf и т. п.) - в виде встроенных функций. Каждый оператор действует на одно или два числа (переменную или функцию), которые называют *операндами.* Если в момент вставки оператора одного или

обоих операндов не хватает, то недостающие операнды будут отображены в виде местозаполнителей. Символ любого оператора в нужное место документа вводится одним из двух основных способов:

- нажатием соответствующей клавиши (или сочетания клавиш) на клавиатуре;
- нажатием указателем мыши соответствующей кнопки на одной из математических панелей инструментов.

Большинство математических панелей содержат сгруппированные по смыслу математические операторы, а вызвать эти панели на экран можно нажатием соответствующей кнопки на панели **Math** (Математика).

Арифметические операторы, обозначающие основные арифметические действия, вводятся с панели **Calculator** (Калькулятор). Сюда также входят операторы обычного и глобального присваивания.

Из всего разнообразия рассмотрим наиболее распространенный оператор - численный вывод = . Если после имени переменной или функции ввести знак = то сразу будет выведено их текущее числовое значение.

Вычислительные операторы вставляются в документы при помощи панели инструментов **Calculus** (Вычисления). При нажатии любой из кнопок в документе появляется символ соответствующего математического действия, снабженный несколькими местозаполнителями. Количество и расположение местозаполнителей определяется типом оператора и в точности соответствует их общепринятой математической записи. Например, при вставке оператора суммы (рис. 12) необходимо задать четыре величины: переменную, по которой надо произвести суммирование, нижний и верхний пределы, а также само выражение, которое будет стоять под знаком суммы. Для того чтобы вычислить неопределенный интеграл, следует заполнить два местозаполнителя: подынтегрального выражения и переменной интегрирования.

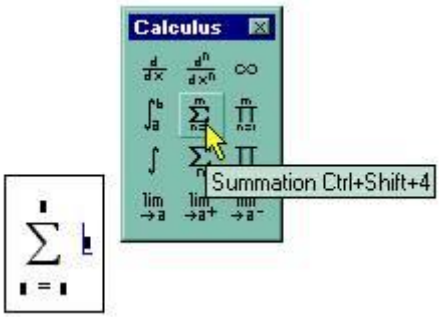


Рис. 12. Вставка вычислительного оператора суммирования.

Некоторые операторы реализуются при использовании вычислительных методов (например, вычисление интегралов). Перечислим остальные типы операторов:
- Логические операторы – панель инструментов **Boolean** (Булевы операторы) – для произведения логических действий;
- Операторы выражения – панель **Evaluation** (Выражения)
- Матричные операторы – панель **Matrix** (Матрица) – для ввода матриц и действий над ними.

2.3. Работа с текстом. Текстовая область и текстовый параграф. Установка шрифта, форматирование параграфа. Проверка орфографии. Гипертекст. Формат чисел. Вид курсора. Ввод формул и их редактирование.

MathCAD - это математический редактор. Основное его назначение заключается в редактировании математических формул и расчете по ним. Вместе с тем, наряду с формульным редактором, MathCAD обладает довольно развитыми средствами по оформлению текста. Назначение текстовых областей в документах MathCAD для разных пользователей и разных задач может быть различным. Качественно стоит различать подход к тексту, используемому:
- просто в качестве комментариев;

- как элемент оформления документов для создания качественных отчетов в печатной и электронной формах.

Текстовую область (или, по-другому, *регион с текстом* - text region) можно разместить в любом незанятом месте документа MathCAD. Однако, когда пользователь помещает курсор ввода в пустое место документа и просто начинает вводить символы, MathCAD по умолчанию интерпретирует их как начало формулы. Чтобы до начала ввода указать программе, что требуется создать не формульный, а текстовый регион, достаточно, перед тем как ввести первый символ, нажать клавишу <">. В результате на месте курсора ввода появляется новый текстовый регион, который имеет характерное выделение (рис. 13). Курсор принимает при этом вид вертикальной линии красного цвета, которая называется *линией ввода текста* и аналогична по назначению линиям ввода в формулах.

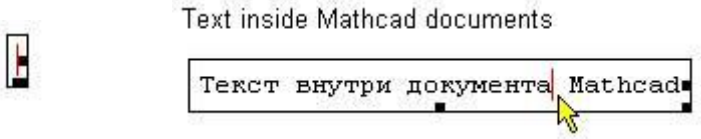


Рис. 13. Вновь создаваемый (слева) и редактируемый (справа) текстовый регион

Создать текстовый регион можно и эквивалентным способом, с помощью команды ***Insert / Text Region*** *(Вставка / Текстовая область).*

Теперь можно просто вводить любой текст в текстовый регион, причем очередной символ будет вставлен в позицию, обозначенную линией ввода текста.

Редактирование текста. Чтобы изменить какой-либо текст внутри документа:

1. Щелкните мышью на области текста - она приобретет характерный вид (рис. 13).
2. При необходимости переместите линию ввода текста внут-

ри текстовой области к символам, которые собираетесь изменить, щелкая мышью в нужном месте текста или нажимая клавиши со стрелками и клавиши <Home> и <End>.

3. Отредактируйте текст.

Для редактирования текста применяются те же средства, что и для редактирования формул:

- выделение части текста протаскиванием указателя мыши или нажатием клавиш-стрелок при удерживаемой клавише <Shift>;
- вырезка, копирование и вставка части текста либо сочетанием горячих клавиш <Ctrl>+<X>, <Ctrl>+<C>, <Ctrl>+<V> соответственно, либо при помощи меню **Edit** (Правка), контекстного меню или панели **Standard** (Стандартная).

MathCAD приемлет импорт фрагментов текста из других приложений (например, Notepad или Microsoft Word). Сделать это проще всего через буфер обмена, то есть скопировать кусок текса и вставить его в созданный текстовый регион. Если при этом не создавать текстового региона, а нажать <Ctrl>+<V>, то текст будет вставлен в виде объекта OLE, т. е. для его редактирования каждый раз будет вызываться то приложение, в котором он был создан.

Форматирование текста. Форматирование текста заключается в управлении его двумя основными составляющими:

- форматом шрифта;
- форматом абзаца.

Шрифт выделенного текста можно поменять при помощи панели **Formatting** (Форматирование), либо при помощи диалогового окна **Text Format** (Формат текста). Чтобы вызвать это диалоговое окно,следует выбрать пункт **Text** (Текст) в верхнем меню **Format** (Формат) илипункт **Font** (Шрифт) в контекстном меню. Перечислим параметры шрифта исоответствующие элементы этой панели, которыми допускается управлять:

- **Font** (Шрифт) - поле в окне **Text Format** (Формат текста);
- **Size** (Размер) - поле в окне **Text Format** (Формат текста) (на аналогичное поле панели форматирования (рис. 14) наведен курсор);

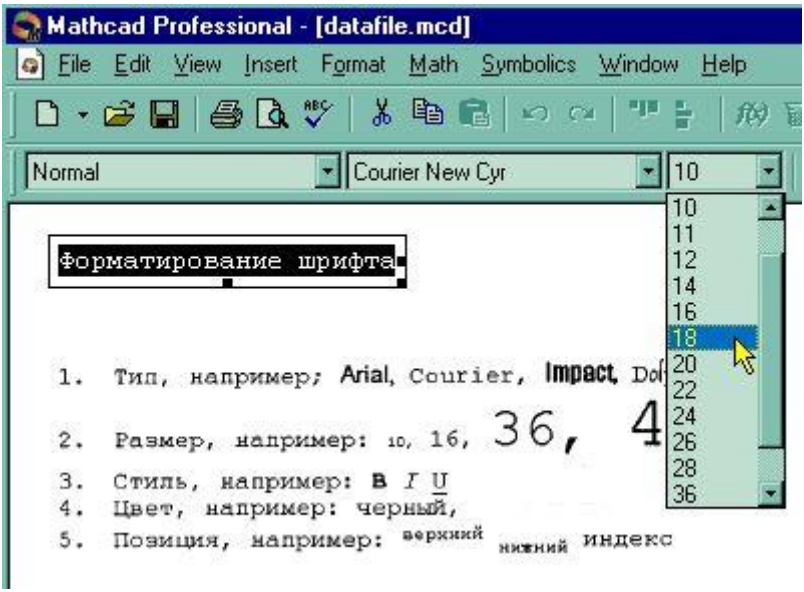


Рис. 14. Примеры шрифтового оформления текста.

- **Font Style** (Стиль шрифта)--поле в окне **Text Format** (Формат текста),ему соответствуют кнопки на панели форматирования:
- **Bold** (Полужирный);
- **Italic** (Наклонный);
- **Underlined** (Подчеркнутый).
- **Strikeout** (Зачеркнутый) - флажок в диалоге **Text Format;**
- **Color** (Цвет) - поле со списком в диалоге **Text Format;**
- **Position** (Позиция) - флажки в диалоге **Text Format:**
- **Superscript** (Верхний индекс);
- **Subscript** (Нижний индекс). Примеры форматирования

шрифта показаны на рис. 14.

Абзац. Для установки параметров абзаца применяются:
- абзацный отступ - три маркера на линейке в верхней части экрана задают левую границу первой строки абзаца (верхний левый маркер) и егоостальных строк (нижний левый маркер), а также правую границу абзаца(правый маркер);
- нумерованный и маркированный списки - две крайние правые кнопкина панели **Formatting** (Форматирование);
- выравнивание - задается кнопками на панели форматирования (рис. 15):
- по левому краю;
- по центру;
- по правому краю.

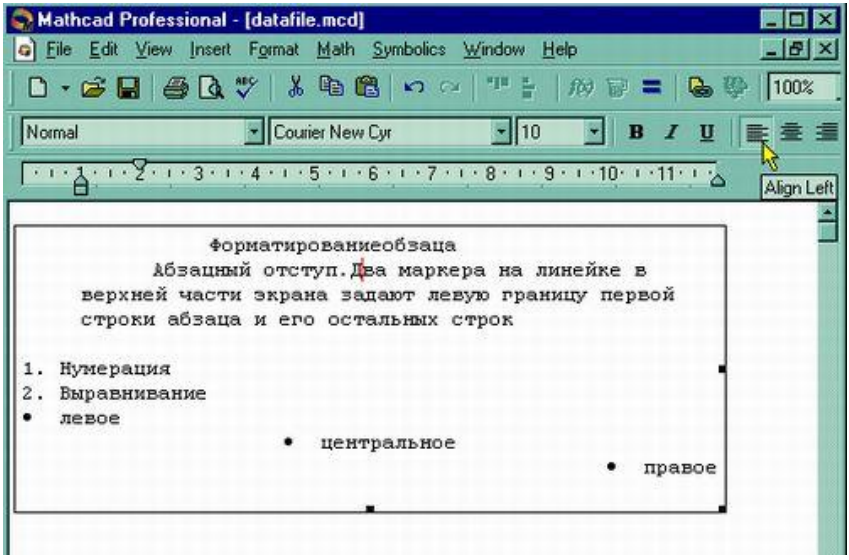


Рис. 15. Пример форматирования абзаца.

Все параметры абзаца можно изменить и в диалоговом окне **Paragraph Format** (Формат абзаца), которое вызывается вы-

бором пункта **Paragraph** (Абзац) меню **Format** (Формат) или одноименного пункта контекстного меню .

Проверка орфографии. Для проверки англоязычной орфографии выделите текстовые регионы, подлежащие проверке, и выполните команду **Edit / Check Spelling** (Правка / Проверка орфографии), либо нажмите кнопку с галочкой на стандартной панели инструментов. Если вы хотите проверить орфографию во всем документе, не выделяйте ни один текстовый регион, а поместите курсор ввода в точку, с которой требуется начать проверку. Если в процессе проверки MathCAD обнаружит слово, отсутствующее в его словаре, оно будет выделено в документе, а пользователь увидит диалоговое окно **Check Spelling** (Проверка орфографии), показанное на рис. 16.

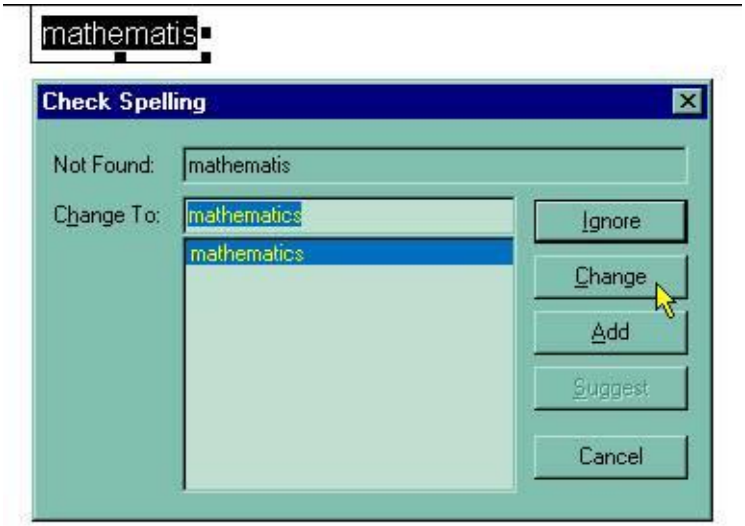


Рис. 16. Диалоговое окно **Check Spelling** (Проверка орфографии).

В диалоговом окне **Check Spelling** находятся следующие элементы:

- **Not Found** (Нет в словаре) - указание на то, что слово отсутствует *в* словаре. Проверить написание слова придется самостоятельно и затем ввести правильный вариант в поле ввода;
- **Change To** (Заменить на) - предложение наиболее близких слов из словаря для исправления. Выберите правильную замену из предложенного списка;
- **Change** (Заменить) - нажмите эту кнопку, чтобы заменить слово в документе на исправленное;
- **Ignore** (Пропустить) - оставить слово в документе таким неизменным;
- **Add** (Добавить) - оставить слово в документе и, кроме того, добавить его в словарь MathCAD, чтобы впоследствии он интерпретировал его как правильное;
- **Cancel** (Отмена) - оставить все как есть и выйти из диалогового окна, закончив проверку орфографии.

Гиперссылки. Иногда необходимо сделать текстовую область одновременно и гиперссылкой, переводящей курсор на какое-либо иное место в активном документе, другой документ MathCAD, либо на сайт в Интернете. Гиперссылки - это активные области в документах MathCAD, которые выводят на экран какое-либо другое место в активном документе, другой документ MathCAD или другого приложения, либо на сайт в Интернете. Для вставки гиперссылки используется команда **Insert/Hyperlink** (Вставка/Гиперссылка).

Прежде чем определить гиперссылку, следует задать место, на которое эта гиперссылка будет переводить курсор и которое в MathCAD называется *тегом* (tag). Для установки тега:

1. Щелкните на том месте, где вы хотите расположить тег, правой кнопкой мыши.
2. В контекстном меню выберите пункт **Properties** (Свойства).
3. В диалоге **Properties** (Свойства) перейдите на вкладку **Display** (Отображение).
4. В поле Tag (Тег) введите имя тега, которое будет иденти-

фицировать данное место в документе (рис. 17).

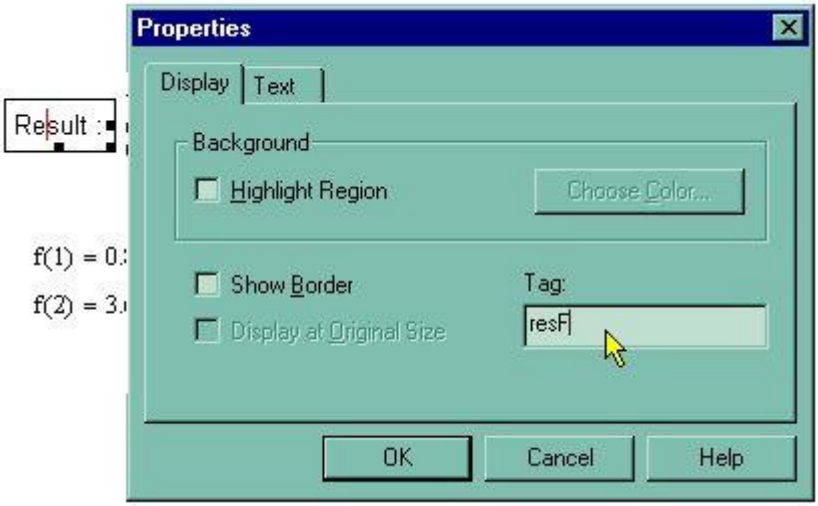


Рис. 17. Установка тега для гиперссылки.

5. Нажмите кнопку ОК.

Определив тег, можно создавать гиперссылку в любом месте любого документа, которая будет переводить курсор на место этого тега. Для вставки гиперссылки:

1. Щелкните на текстовой или формульной области документа, которую выхотите сделать гиперссылкой.
2. Выберите в меню **Insert** (Вставка) пункт **Hyperlink** (Гиперссылка);
3. В диалоговом окне **Edit Hyperlink** (Правка гиперссылки), в текстовом поле **Link to file or URL** (Связать с файлом или URL) определите путь к документу, в котором расположен тег и имя самого тега в формате filename#tag (рис. 18).
4. По желанию, в нижнем текстовом поле задайте текст, который будет появляться на строке состояния при наведении указателя мыши на гиперссылку.
5. Нажмите кнопку ОК.

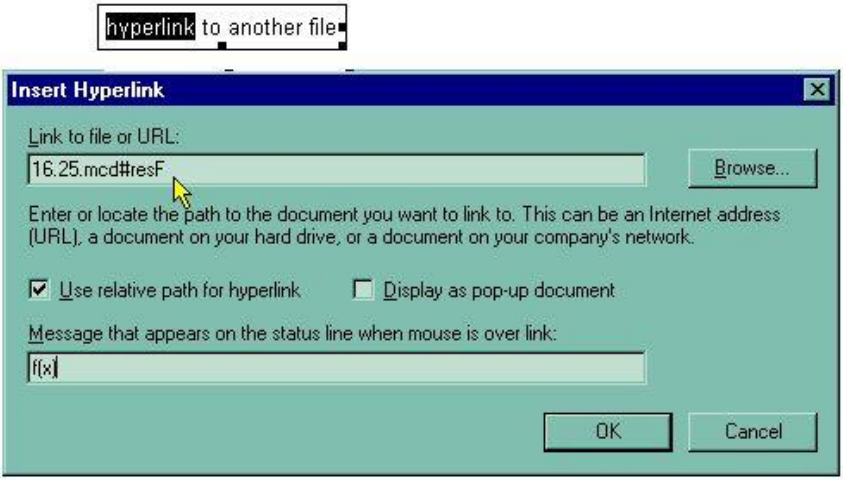


Рис.18. Заполнение свойств гиперссылки.

Примечание: *Имя файла необходимо указывать, даже если тег расположен в том же документе, что и гиперссылка.*

Если вы все сделали правильно, то при двойном щелчке на гиперссылке будет осуществлен переход на место, где расположен тег, т. е. в нашем примере на документ 16.25.mcd. Чтобы отредактировать гиперссылку, достаточно, находясь на ее области, выбрать тот же пункт меню **Insert / Hyperlink** (Вставка/ Гиперссылка). Появится диалог **Edit Hyperlink** (Правка гиперссылки), в котором можно исправить ее параметры. Удалить гиперссылку можно нажатием кнопки **Remove Link** (Удалить гиперссылку).

Помимо гиперссылок на документы MathCAD, допускается создавать гиперссылки на другие файлы (например, видеофайлы или HTML-файлы), в том числе, находящиеся в Интернете. Для этого достаточно указать соответствующий адрес URL в верхнем текстовом поле диалога **Edit Hyperlink** (Вставка гиперссылки).

Ссылки. Помимо гиперссылок, иногда стоит применять дру-

гие схожие с ними объекты, называемые *ссылками* (reference). Ссылка на документ А, вставленная в некоторое место документа В, приводит к расчету всего документа А внутри документа В. Таким образом, ссылки позволяют хранить вложенные друг в друга расчеты в разных файлах.

Совет: *Ссылки могут понадобиться, если группа разработчиков решает одну большую задачу. В этом случае после распределения задач и соглашения об именах глобальных и локальных переменных каждый разработчик создает свой файл с расчетами.*

Для установки ссылки достаточно выбрать команду **Insert / Reference** (Вставка / Ссылка) и затем в диалоговом окне **Insert Reference** (Вставка ссылки) определить путь к имени файла-ссылки. В примере, показанном на рис. 19, в файле отсутствует описание функции f(x), зато оно есть в файле, на который оформляется ссылка.

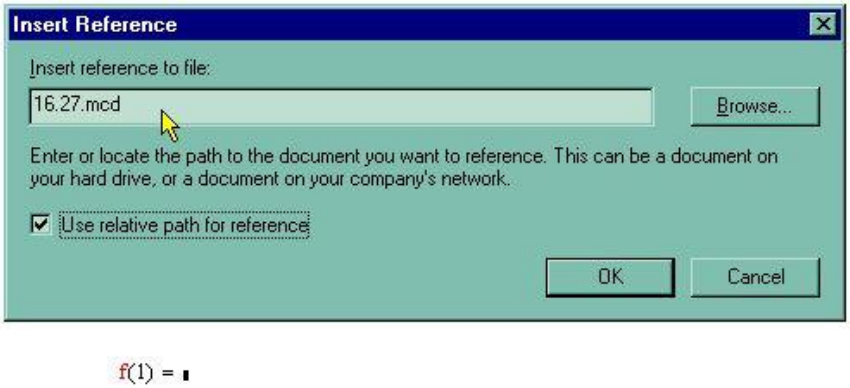


Рис. 19. Создание ссылки.

Поэтому после нажатия кнопки **ОК** на месте курсора ввода появится информация о файле-ссылке, а результат функции f(1) будет рассчитан в соответствии с формулами этого файла.

Форматы чисел. Действительные числа. Любое выражение, начинающееся с цифры, MathCAD интерпретирует как число. Поэтому для ввода числа просто начните его набирать на клавиатуре. Несмотря на то, что MathCAD хранит все числа в одинаковом формате, вводить их можно в наиболее подходящем *представлении* (notation), исходя из контекста документа:

- как *целое число;*
- как *десятичное число* (decimal notation) с любым количеством десятичных цифр после точки;
- в представлении *с порядком* (exponential notation) - в так называемом *научном формате* или *представлении* (scientific notation), для чего после ввода числа напечатайте символ умножения и введите 10 в нужной степени;
- как число в другой системе счисления.

Три первых представления иллюстрируются содержанием соответствующей строки рис. 20.

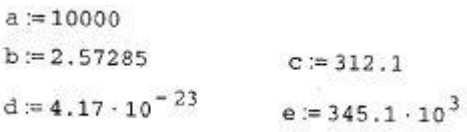


Рис. 20. Ввод действительных чисел.

При вводе целых чисел, больших или равных 1000, все цифры пишутся слитно (как показано в первой строке рис.20), и ни в коем случае не разделяются на порядки запятыми. Например, ввод числа 1000 как 1,000 или 1.000 недопустим.

Для ввода числа в других системах счисления: *двоичной* (binary), *восьмеричной* (octal) или *шестнадцатеричной* (hexadecimal), сделайте следующее:

1. Введите его представление в соответствующей системе, применяя лишь корректные символы (для двоичной системы допустимы только цифры 0 и 1; для восьмеричной - цифры от

0 до 7, для шестнадцатеричной - цифры от 0 до 9 и буквы от a до f) Например, число 34 в двоичной системе представлено такой последовательностью: 100010.
2. После ввода последнего символа числа введите b (для двоичного числа), o (для восьмеричного числа), или h (для шестнадцатеричного).
Использование чисел в других системах счисления иллюстрируется рис. 21. Обратите внимание, что вывод осуществляется все равно в десятичной системе.

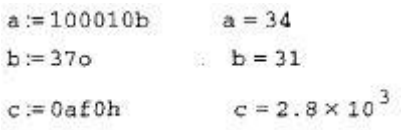


Рис. 21. Ввод чисел в других системах исчисления

Комплексные числа. Большинство операций в среде MathCAD по умолчанию осуществляются над *комплексными числами.* Комплексное число является суммой действительного и *мнимого числа,* получающегося путем умножения любого действительного числа на *мнимую единицу* (imaginary unit) i. По определению, i = корень -l ИЛИ i^2=-l.
Чтобы ввести мнимое число, например 3i:
1. Введите действительный сомножитель (3).
2. Введите символ "i" или "j" непосредственно после него. Для ввода мнимой единицы надо нажать клавиши <1>, <i.>.
Если просто ввести символ "i", то MathCAD интерпретирует его как переменную i. Кроме того, мнимая единица имеет вид 1i, только когда соответствующая формула выделена. В противном случае мнимая единица отображается просто как i (рис. 22).
Комплексное число можно ввести в виде обычной суммы действительной и мнимой частей, или в виде любого выражения, содержащего мнимое число. Примеры ввода и вывода комплексных чисел иллюстрируются рис. 22 справа.

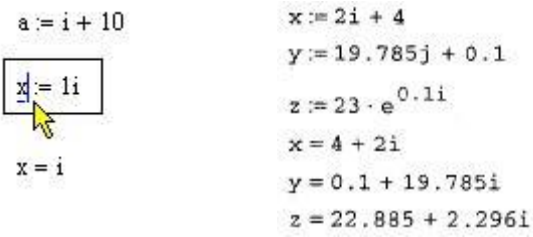


Рис. 22. Ввод мнимой единицы (слева) и примеры ввода и вывода комплексных чисел (справа).

Формат результата. Управление представлением числа в десятичном представлении или представлении с порядком осуществляется при помощи следующих параметров:

- количество отображаемых десятичных знаков *(decimal places)* после точки. Например, число 122,5587 с четырьмя десятичными знаками при отображении с двумя знаками будет выглядеть как 122,56;
- отображение или скрытие незначащих нулей (trailing zeros) - опция, позволяющая показывать или скрывать незначащие нули в десятичном представлении числа, т. е. выводить, к примеру, "1,5" вместо "1,500" (даже если установлено количество десятичных знаков, равное 3);
- порядковый порог (exponential threshold), при превышении степени 10 которого число будет показываться с порядком. Например, при пороге 3 число 122,56 будет отображаться как десятичное, а при пороге 2 – уже как "$1{,}23 \times 10^2$";
- кроме того, число с порядком может представляться в эквивалентных видах: "$1{,}23 \times 10^2$" или с порядком в инженерном формате (engineering format): "1.23E+002".
- MathCAD имеется несколько типов форматов, в каждом из которых разрешается изменение различных параметров представления числа. Формат выбирается на вкладке **Number Format** (Формат числа) диалогового окна **Result Format** (Формат результата) (рис. 23).

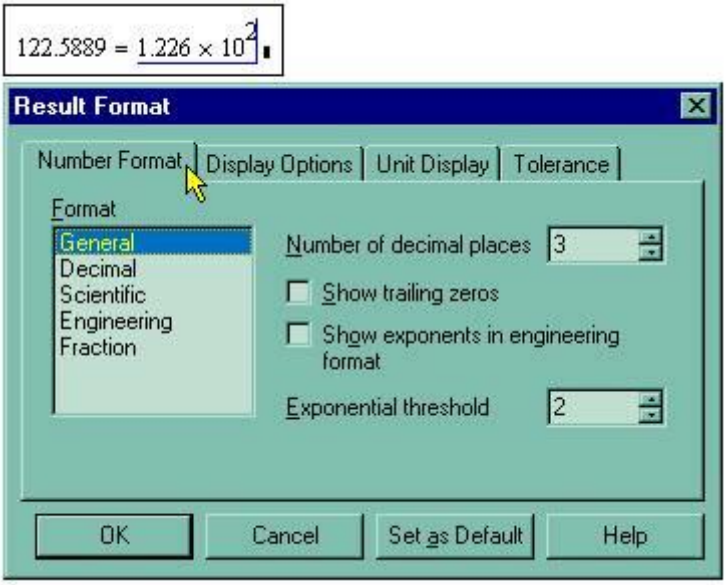


Рис. 23. Выбор формата вывода числа

Основной *(general)* формат: Этот формат принят при выводе чисел по умолчанию. Можно управлять и количеством отображаемых десятичных знаков (поле **Number of decimal places),** и порядковым порогом (поле **Exponential threshold).** При превышении порога число отображается с порядком.

Десятичный *(decimal)* формат: Числа отображаются только в десятичном представлении и никогда - в представлении с порядком.

Научный *(scientific)* формат: Числа отображаются только с порядком, причем количество десятичных знаков левого сомножителя, как и отображение незначащих нулей, определяется пользователем.

Инженерный *(engineering)* формат: Числа отображаются только с порядком, причем обязательно кратным 3; как и в

научном формате, пользователю разрешается изменять количество десятичных знаков.

Дробный *(fraction)* формат: Этот формат сильно отличается от предыдущих, представляя число в виде дроби. Причем можно управлять как точностью представления числа с помощью поля **Level of accuracy** (Уровень точности), так и задать модификацию этого формата - отображение числа в виде целой и дробной посредством установки флажка **Use mixed numbers** (Смешанные числа).

Округление малых чисел до нуля. MathCAD автоматически округляет малые числа до нуля (см. рис. 24).

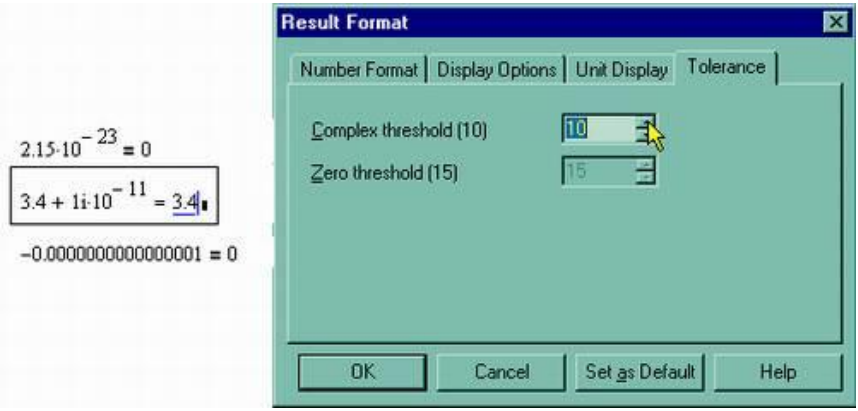


Рис. 24. Задание порога мнимого нуля

Допускается установка порогового значения округления (в степенях 10), отдельно для действительной и мнимой части числа. При этом числа, по модулю меньшие порога, отображаются в виде нуля. Помните, что это касается только отображения чисел. В памяти компьютера они хранятся корректно.

Аналогично вводу чисел в других системах счисления, вывести результат также возможно в виде десятичного, двоичного, восьмеричного или шестнадцатеричного числа (рис.

25).

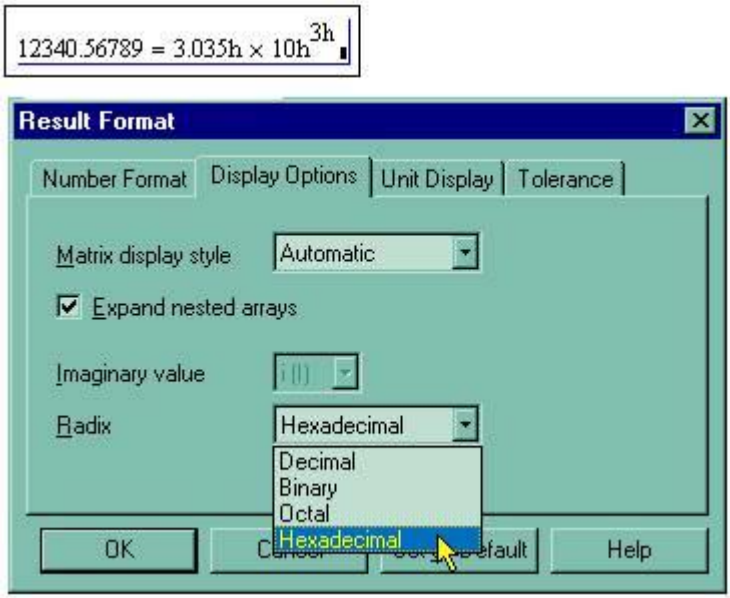


Рис. 25. Задание вывода результата в других системах счисления

Чтобы задать систему счисления, выберите команду **Format / Result / Display Options** (Формат / Результат / Опции отображения), а затем желаемый элемент списка **Radix** (Система счисления) (рис. 25). При отображении чисел в других системах счисления также доступно форматирование их представления на вкладке **Number Format** (Формат числа) того же диалога **Result Format** (Формат результата).

Ввод и редактирование формул. Формульный редактор MathCAD позволяет быстро и эффективно вводить и изменять математические выражения. Тем не менее, некоторые аспекты его применения не совсем интуитивны, что связано с необходимостью избежать ошибок при расчетах по этим формулам.

Поэтому не пожалейте немного времени на знакомство с особенностями формульного редактора, и впоследствии при реальной работе вы сэкономите гораздо больше.

Перечислим еще раз элементы интерфейса и виды курсора редактора MathCAD (рис. 26):

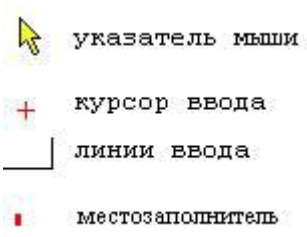


Рис. 26. Интерфейс редактирования

- указатель мыши (mouse pointer) - играет обычную для приложений Windows роль, следуя за движениями мыши;
- курсор - обязательно находится внутри документа в одном из трех видов:

курсор ввода (crosshair) - крестик красного цвета, который отмечает пустое место в документе, куда можно вводить текст или формулу;

линии ввода (editing lines) - горизонтальная (underline) и вертикальная (insertion line) линии синего цвета, выделяющие в тексте или формуле определенную часть;

линия ввода текста (text insertion point) - вертикальная линия, аналог линий ввода для текстовых областей.

- местозаполнители (placeholders) - появляются внутри незавершенных формул в местах, которые должны быть заполнены символом или оператором:

местозаполнитель символа - черный прямоугольник;

местозаполнитель оператора - черная прямоугольная рамка.

Ввод формул. Ввести математическое выражение можно

в любом пустом месте документа MathCAD. Для этого поместите курсор ввода в желаемое место документа, щелкнув в нем мышью, и просто начинайте вводить формулу, нажимая клавиши на клавиатуре. При этом в документе создается *математическая область* (math region), которая предназначена для хранения формул, интерпретируемых процессором MathCAD.

Продемонстрируем последовательность действий на примере ввода выражения x^{5+x} (рис. 27):

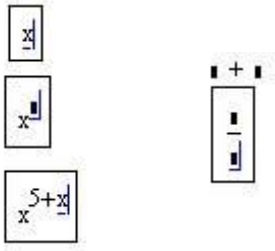


Рис. 27. Примеры ввода формулы (слева) и начала ввода операторов (справа).

1. Щелкните мышью, обозначив место ввода.
2. Нажмите клавишу <x> - в этом месте вместо курсора ввода появится регион с формулой, содержащей один символ x, причем он будет выделен линиями ввода.
3. Введите оператор возведения в степень, нажав клавишу <^>, либо выбрав кнопку возведения в степень на панели инструментов **Calculator** (Калькулятор) - в формуле появится местозаполнитель для введения значения степени, а линии ввода выделят этот местозаполнитель.
4. Последовательно введите остальные символы <5>, <+>, <x>.

Таким образом, поместить формулу в документ можно, просто начиная вводить символы, числа или операторы,

например + или /. Во всех этих случаях на месте курсора ввода создается математическая область, иначе называемая *регионом,* с формулой, содержащей и линии ввода. В последнем случае, если пользователь начинает ввод формулы с оператора, в зависимости от его типа, автоматически появляются и местозаполнители, без заполнения которых формула не будет восприниматься процессором MathCAD.

Чтобы изменить формулу, щелкните на ней мышью, поместив таким образом в ее область линии ввода, и перейдите к месту, которое хотите исправить. Перемещайте линии ввода в пределах формулы одним из двух способов:

- щелкая в нужном месте мышью;
- нажимая на клавиатуре клавиши - со стрелками, пробел и <Ins>:

клавиши со стрелками имеют естественное назначение, переводя линии ввода вверх, вниз, влево или вправо;

клавиша <Ins> переводит вертикальную линию ввода с одного конца горизонтальной линии ввода на противоположный;

пробел предназначен для выделения различных частей формулы.

Если раз за разом нажимать клавишу пробела в формуле, пример ввода которой рассмотрен выше (см. рис. 27), то линии ввода будут циклически изменять свое положение, как это показано на рис. 28. Если в ситуации, показанной сверху на этом рисунке, нажать стрелку <-, то линии ввода переместятся влево (рис. 28). При нажатии пробела теперь линии ввода будут попеременно выделять одну из двух частей формулы.

Таким образом, комбинация клавиш со стрелками и пробела позволяет легко перемещаться внутри формул. Иногда поместить линии ввода в нужное место формулы с помощью указателя мыши непросто. Поэтому в MathCAD для этого лучше использовать клавиатуру.

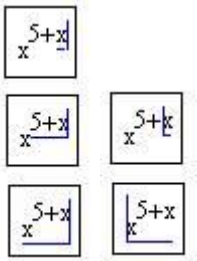


Рис. 28. Изменение положения линий ввода с помощью пробела (слева) и пробелом после сдвига стрелкой (справа)

Изменение формул. Редактируйте формулы в MathCAD так, как подсказывают вам интуиция и опыт работы с другими текстовыми редакторами. Большинство операций правки формул реализованы естественным образом, однако некоторые из них несколько отличаются от общепринятых, что связано с особенностью MathCAD как вычислительной системы. Рассмотрим основные действия по изменению формул.

Вставка оператора. Операторы могут быть унарными (действующими на один операнд, как, на пример, оператор транспонирования матрицы или смены знака числа), так и бинарными (например + или /, действующими на два операнда). При вставке нового оператора в документ MathCAD определяет, сколько операндов ему требуется. Если в точке вставки оператор один или оба операнда отсутствуют, MathCAD автоматически помещает рядом с оператором один или два местозаполнителя.

Последовательность вставки оператора в формулу такова:

1. Поместите линии ввода на часть формулы, которая должна стать первым операндом.
2. Введите оператор, нажав кнопку на панели инструментов или сочетание клавиш.

MathCAD сам расставляет, если это необходимо, скобки,

чтобы часть формулы, отмеченная линиями ввода, стала первым слагаемым. Некоторые операторы MathCAD вставит в правильное место независимо от положения линий ввода. Таков, например, оператор численного вывода =, который по смыслу выдает значение всей формулы в виде числа.

Выделение части формулы. Чтобы выделить часть формулы в некоторой математической области:

1. Поместите ее между линиями ввода, пользуясь, при необходимости, клавишами-стрелками и пробелом.
2. Поместите указатель мыши на вертикальную линию ввода, нажмите и удерживайте левую кнопку мыши.
3. Удерживая кнопку мыши, протащите указатель мыши вдоль горизонтальной линии ввода, при этом часть формулы будет выделяться обращением цвета.
4. Отпустите кнопку мыши, когда будет выделена нужная часть формулы.

Правка формулы. Для правки части формулы:

1. Выделите ее, либо просто поместите между линиями ввода, пользуясь либо мышью, либо клавишами-стрелками и пробелом.
2. Воспользуйтесь либо верхним меню **Edit** (Правка), либо контекстным меню, либо кнопкой на панели инструментов, либо соответствующим сочетанием горячих клавиш:
Cut (Вырезать), или <Ctrl>+<X> - для вырезки части формулы в буфер;
Copy (Копировать), или <Ctrl>+<C> - для копирования в буфер;
Paste (Вставить), или <Ctrl>+<V> - для вставки из буфера предварительно помещенной туда части формулы.
3. Чтобы удалить часть формулы нажмите клавишу <Del>.

Для того чтобы удалить оператор, поместите его перед вертикальной линией ввода и нажмите клавишу <BackSpace>. В результате оператор либо исчезнет (а операнды слева и справа сольются в одно имя), либо (в сложных формулах) по-

явится местозаполнитель оператора в виде черной рамки. При желании можно удалить и этот местозаполнитель повторным нажатием <BackSpace>.

2.4. Вычисления в MathCAD. Приближенные вычисления. Счет по формулам. Встроенные функции и функции пользователя. Численное дифференцирование и интегрирование.

Основные инструменты математика - это операции с переменными величинами и функциями. В MathCAD переменные, операторы и функции реализованы в интуитивной форме, т. е. выражения в редакторе вводятся и вычисляются так, как они были бы написаны на листе бумаги. Порядок вычислений в документе MathCAD также очевиден: математические выражения и действия воспринимаются процессором слева направо и сверху вниз.

Вообще говоря, имеется два режима вычислений:
- *автоматический режим* (automatic mode) - все вычисления проводятся автоматически по мере ввода формул;
- *ручной режим* (manual mode) - старт вычислений каждой формулы или всего документа производится пользователем.

Режим вычислений можно выбрать с помощью команды **Math / Automatic Calculation** (Математика / Считать автоматически), как показано на рис. 29.

Если в этой строке меню установлен флажок проверки, значит включен автоматический режим, если флажка нет, то редактируется документ в ручном режиме вычислений. Чтобы сменить режим, просто выберите этот пункт.

MathCAD осуществляет вычисления документа, как это принято в большинстве сред программирования: сверху вниз и слева направо. Пока очередное выражение находится в процессе расчета (вычислительным или символьным процессором), оно выделяется рамкой зеленого цвета, а любые

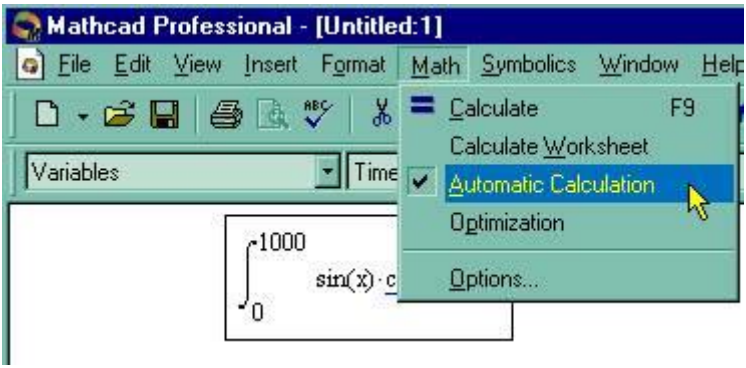


Рис. 29. Выбор режима вычислений

действия пользователя по дальнейшему редактированию документа блокируются.

Если у вас не слишком быстрый компьютер, а формулы достаточно сложные, то можно наблюдать, как зеленая рамка перескакивает с одного выражения на другое.

Чтобы прервать затянувшийся процесс вычислений, нажмите клавишу <Esc>. Появится диалоговое окно, в котором нужно подтвердить прерывание вычислений (ОК). В этом случае выражения, которые MathCAD не успел вычислить, будут помечены в документы красным цветом. Прерванные вычисления возобновляются нажатием клавиши <F9> или командой **Math/Calculate** (Математика/Пересчитать).

Если флажок в строке команды **Math/Automatic Calculation** (Математика/Считать автоматически) снят, пользователь должен запускать вычисления самостоятельно.
- Для того чтобы вычислить все формулы во всем документе, выполните команду **Math/Calculate Worksheet** (Математика/Пересчитать все).
- Для вычисления всех формул в видимой части документа выберите пункт **Math/Calculate** (Математика/Пересчитать), либо нажмите клавишу <F9>, либо щелкните на кнопке с изображением знака равенства **(Calculate)** на стандартной па-

нели инструментов.
- Прервать вычисления можно обычным образом, нажав клавишу <Esc>.

MathCAD позволяет отключить вычисление какой-либо формулы. При этом она не будет влиять на последующие вычисления. Чтобы не вычислять определенную формулу в документе:

1. Щелкните правой кнопкой мыши на формуле.
2. Выберите в контекстном меню пункт **Disable Evaluations** (Выключить вычисления).

Эквивалентный способ выключения вычисления отдельной формулы заключается в вызове диалогового окна Properties (Свойства) через одноименный пункт контекстного меню или главного меню **Format** (Формат). В диалоге **Properties** следует перейти на вкладку **Calculations** (Вычисления) и установить там флажок **Disable Evaluations** (Выключить вычисления).

В MathCAD имеется возможность выполнения обычных числовых вычислений с повышенной точностью – до 20 знаков после запятой. Для перехода в такой режим числовые константы в вычисляемых объектах нужно задавать с указанием десятичной точки, например 10.0 или 3.0, а не 10 и 3. Этот признак является указанием на произведение вычислений такого типа. При помощи функций символьного редактора можно добиться получения результата с 4000 верными знаками после запятой, что практически никогда не требуется в обычной жизни. Нужда в таких вычислениях может потребоваться при решении специальных вычислительных задач, например, в теории кодирования информации. Но для решений таких задач существуют специализированные программные продукты, возможности которых неизмеримо превосходят заложенные в MathCAD. Однако следует всегда помнить, что даже у самых совершенных алгоритмов всегда существует погрешность вычисления, которая может привести к неточно-

му или неверному результату.

Для того чтобы выполнить простые расчеты по формулам, проделайте следующее:

- определите место в документе, где должно появиться выражение, щелкнув мышью в соответствующей точке документа;
- введите левую часть выражения;
- введите знак равенства <=>.

Для вычисления синуса какого-нибудь числа достаточно ввести с клавиатуры выражение типа sin (1/4)=. После того как будет нажата клавиша со знаком равенства, с правой стороны выражения, как по мановению волшебной палочки, появится результат:

$$\sin\left(\frac{1}{4}\right) = 0.247 \qquad (*)$$

Подобным образом можно проводить и более сложные и громоздкие вычисления, пользуясь при этом всем арсеналом специальных функций, которые встроены в MathCAD. Легче всего вводить их имена с клавиатуры, как в примере с вычислением синуса, но, чтобы избежать возможных ошибок в их написании, лучше выбрать другой путь. Чтобы ввести встроенную функцию в выражение:

1. Определите место в выражении, куда следует вставить функцию.
2. Нажмите кнопку с надписью *f(x)* на стандартной панели инструментов (на нее указывает курсор на рис. 30).
3. В списке **Function Category** (Категория функции) появившегося диалогового окна **Insert Function** (Вставить функцию) выберите категорию, к которой принадлежит функция, - в нашем случае это категория **Trigonometric** (Тригонометрические).
4. В списке **Function Name** (Имя функции) выберите имя встроенной функции, под которым она фигурирует в MathCAD (sin). В случае затруднения с выбором ориентируйтесь на подсказку, появляющуюся при выборе функции в

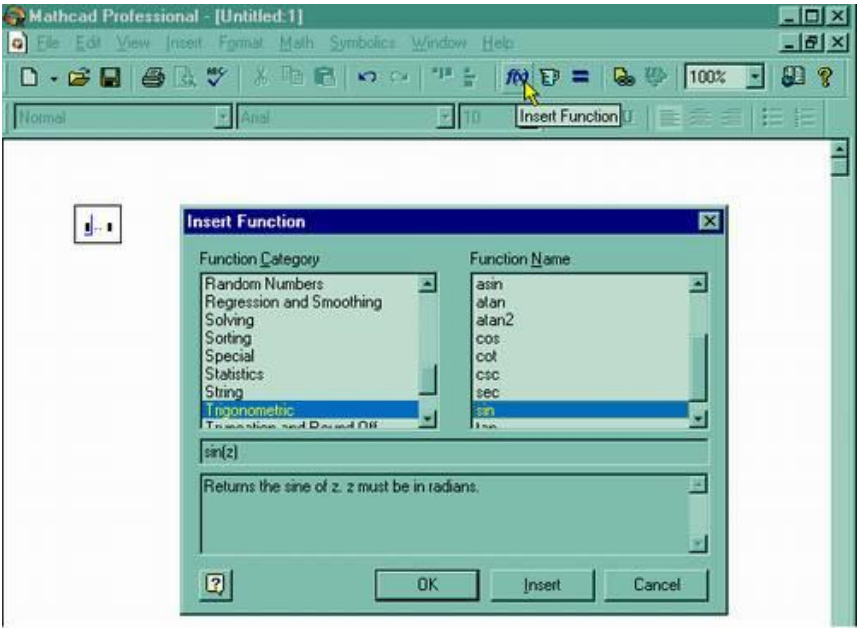


Рис. 30. Вставка встроенной функции

нижнем текстовом поле диалогового окна **Insert Function.**

5. Нажмите кнопку OK - функция появится в документе.

6. Заполните недостающие аргументы введенной функции (в нашем случае это 1/4).

Результатом будет введение выражения (*), для получения значения которого осталось лишь ввести знак равенства

Функции. Функции в MathCAD записываются в обычной для математика форме:

- f (x,...) - функция;

  f - имя функции;

  x,... - список переменных.

Легче всего ввести написание функции в документ при помощи клавиатуры. В MathCAD формально можно разделить

функции на два типа:
- встроенные функции;
- функции, определенные пользователем.

Применение функций обоих типов в расчетах совершенно одинаково, с тем исключением, что любую встроенную функцию можно сразу использовать в любом месте документа, а пользовательскую функцию необходимо предварительно определить в документе до момента вычисления ее значения.

Для того чтобы определить функцию пользователя, например $f(x,y) = x^2-\cos(x+y)$:

1. Введите в желаемом месте документа имя функции (например f).
2. Введите левую скобку "(", имена переменных через запятую x, y и правую скобку ")". При вводе левой скобки и запятых автоматически будут появляться соответствующие местозаполнители.
3. Введите оператор присваивания с панели инструментов или нажатием клавиши <:>.
4. Введите в появившийся местозаполнитель выражение, определяющее функцию $x^2 \cdot \cos(x+y)$, пользуясь клавиатурой или панелями инструментов.

Результат ввода функции и вывод значений при определенных значений переменных иллюстрируется на рис. 31:

$$f(x,y) := x^2 \cdot \cos(x+y)$$
$$f(2,5.99) = -0.542$$
$$f(1.3,7) = -0.729$$
$$x := 1.3$$
$$y := 7$$
$$f(x,y) = -0.729$$

Рис. 31. Определение функции пользователя и вывод значения функции при определенных значений переменных.

Численное интегрирование и дифференцирование функций. Рассмотрим основные математические операции, к которым относятся численное дифференцирование и интегрирование функций.

Интегрирование и дифференцирование - самые простые, с вычислительной точки зрения, операции, реализованные в MathCAD в виде операторов. Тем не менее, если расчеты выполняются с помощью вычислительного процессора, необходимо хорошо представлять себе особенности численных алгоритмов, действие которых остается для пользователя "за кадром".

Интегрирование. Интегрирование в MathCAD реализовано в виде вычислительного оператора. Допускается вычислять интегралы от скалярных функций в пределах интегрирования, которые также должны быть скалярами. Несмотря на то, что пределы интегрирования обязаны быть действительными, подынтегральная функция может иметь и комплексные значения, поэтому и значение интеграла может быть комплексным. Если пределы интегрирования имеют размерность, то она должна быть одной и той же для обоих пределов.

Интегрирование, дифференцирование, как и множество других математических действий, устроено в MathCAD по принципу "как пишется, так и вводится". Чтобы вычислить определенный интеграл, следует напечатать его обычную математическую форму в документе. Делается это с помощью панели **Calculus** (Вычисления) нажатием кнопки со значком интеграла или вводом с клавиатуры сочетания клавиш <Shift>+<7> (или символа "&", что то же самое). Появится символ интеграла с несколькими местозаполнителями (рис.32), в которые нужно ввести нижний и верхний интервалы интегрирования, подынтегральную функцию и переменную интегрирования. Можно вычислять интегралы с одним или обоими бесконечными пределами. Для этого на месте

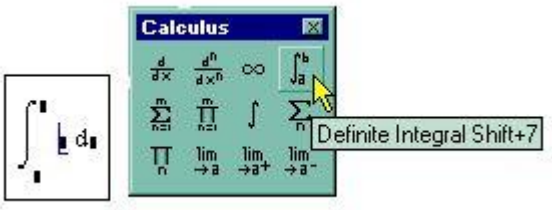


Рис. 32. Оператор интегрирования

соответствующего предела введите символ бесконечности, воспользовавшись, например, той же самой панелью **Calculus** (Вычисления). Чтобы ввести -∞ (минус бесконечность), добавьте знак минус к символу бесконечности, как к обычному числу.

Чтобы получить результат интегрирования, следует ввести знак равенства или символьного равенства. В первом случае интегрирование будет проведено численным методом, во втором - в случае успеха, будет найдено точное значение интеграла с помощью символьного процессора MathCAD. Эти два способа иллюстрирует рис. 33:

$$\int_0^{\pi} \sin(x)\,dx = 2$$

$$\int_0^{\pi} \sin(x)\,dx \rightarrow 2$$

Рис. 33. Численное и символьное вычисление определенного интеграла

Конечно, символьное интегрирование возможно только для небольшого круга несложных подынтегральных функций.

Подынтегральная функция может зависеть от любого количества переменных. Именно для того чтобы указать, по ка-

кой переменной MathCAD следует вычислять интеграл, и нужно вводить ее имя в соответствующий местозаполнитель. Помните, что для численного интегрирования по одной из переменных предварительно следует задать значение остальных переменных, от которых зависит подынтегральная функция и для которых вы намерены вычислить интеграл.

Об алгоритмах интегрирования. Результат численного интегрирования - это не точное, а приближенное значение интеграла, определенное с погрешностью, которая зависит от встроенной константы TOL. Чем она меньше, тем с лучшей точностью будет найден интеграл, но и тем больше времени будет затрачено на расчеты. По умолчанию TOL=0.001. Для того чтобы ускорить вычисления, можно установить меньшее значение TOL.

Если скорость расчетов имеет для вас принципиальное значение, например, при многократном вычислении интеграла внутри цикла, проявите осторожность, выбирая значение точности. Обязательно поэкспериментируйте на тестовом примере с характерной для ваших расчетов подынтегральной функцией. Посмотрите, как уменьшение константы TOL сказывается на погрешности интегрирования, вычислив интеграл для разных ее значений и выбрав оптимальное, исходя из соотношения точность / скорость вычислений.

Отдавайте себе отчет в том, что при вводе в редакторе MathCAD оператора численного интегрирования, вы, фактически, создаете самую настоящую программу. В большинстве случаев об этом не приходится специально задумываться, можно полностью положиться на MathCAD. Но иногда может потребоваться умение управлять параметрами этой программы, как мы уже рассмотрели на примере выбора константы TOL. Кроме нее, пользователь имеет возможность выбирать сам алгоритм численного интегрирования. Для этого:

1. Щелкните правой кнопкой мыши в любом месте на левой части вычисляемого интеграла.

2. В появившемся контекстном меню выберите один из четырех численных алгоритмов (рис. 34).

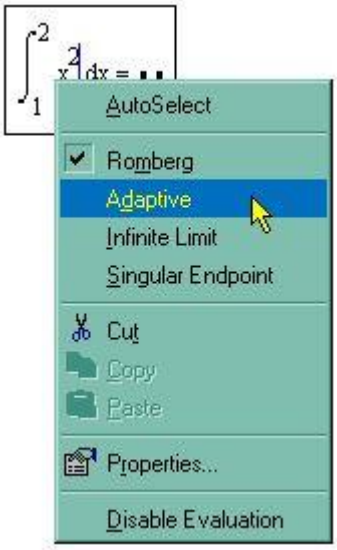


Рис. 34. Выбор алгоритма численного интегрирования

Обратите внимание, что перед тем как один из алгоритмов выбран впервые, как показано на рис. 33, флажок проверки в контекстном меню установлен возле пункта **AutoSelect** (Автоматический выбор). Это означает, что алгоритм определяется MathCAD, исходя из анализа пределов интегрирования и особенностей подынтегральной функции. Как только один из алгоритмов выбран, этот флажок сбрасывается, а избранный алгоритм отмечается точкой.

Разработчиками MathCAD 2001 запрограммированы четыре численных метода интегрирования:

- **Romberg** (Ромберга) - для большинства функций, не содержащих особенностей;
- **Adaptive** (Адаптивный) - для функций, быстро меняющихся

на интервале интегрирования;
- **Infinite Limit** (Бесконечный предел) - для интегралов с бесконечными пределами ();
- **Singular Endpoint** - для интегралов с сингулярностью на конце. Используется модифицированный алгоритм Ромберга для функций, не определенных на одном или обоих концах интервала интегрирования.

Старайтесь все-таки оставить выбор численного метода за MathCAD, установив флажок **AutoSelect** (Автоматический выбор) в контекстном меню. Попробовать другой метод можно, например, чтобы сравнить результаты расчетов в специфических случаях, когда у вас закрадываются сомнения в их правильности.

По аналогичной схеме проводится вычисление кратных интегралов (двойных и тройных).

Дифференцирование. С помощью MathCAD можно вычислять производные скалярных функций любого количества аргументов, от 0-го до 5-го порядка включительно. И функции, и аргументы могут быть как действительными, так и комплексными. Невозможно дифференцирование функций только вблизи точек их сингулярности.

Вычислительный процессор MathCAD обеспечивает превосходную точность численного дифференцирования. Но больше всего пользователь оценит возможности символьного процессора, который позволяет с легкостью осуществить рутинную работу вычисления производных громоздких функций, поскольку, в отличие от всех других операций, символьное дифференцирование выполняется успешно для подавляющего большинства аналитически заданных функций. В MathCAD 2001 для ускорения и повышения точности численного дифференцирования функций, заданных аналитически, автоматически задействуется символьный процессор

Для того чтобы продифференцировать функцию f(x) в некоторой точке:

1. Определите точку x, в которой будет вычислена производная, например, x:=1.
2. Введите оператор дифференцирования нажатием кнопки **Derivative** (Производная) на панели **Calculus** (Вычисления) или введите с клавиатуры вопросительный знак <?>.
3. В появившихся местозаполнителях (рис. 35) введите функцию, зависящую от аргумента x, т. е. f(x), и имя самого аргумента x.

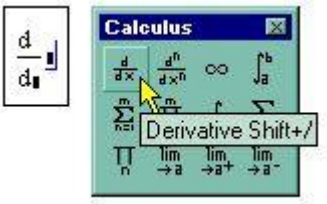


Рис. 35. Оператор дифференцирования

4. Введите оператор <=> численного или <- символьного вывода для получения ответа.
Пример дифференцирования функции f (x)=cos(x)-in(x) приведен на рис. 36.

$$x := 0.01$$

$$\frac{d}{dx}\cos(x)\cdot\ln(x) = 100.041$$

$$\frac{d}{dx}\cos(x)\cdot\ln(x) \rightarrow -\sin\left(1.\cdot10^{-2}\right)\cdot\ln\left(1.\cdot10^{-2}\right)+1.\cdot10^{2}\cdot\cos\left(1.\cdot10^{-2}\right)$$

Рис. 36. Численное и символьное дифференцирование.

Не забывайте предварительно определять точку, в которой производится численное дифференцирование, как это сделано на рис. 36 в первой строчке.

Как вы заметили, оператор дифференцирования, в основном, соответствует его общепринятому математическому обозначению. Однако в некоторых случаях при его вводе следует проявить осторожность. Рассмотрим один показательный пример, приведенный в листинге на рис. 37.

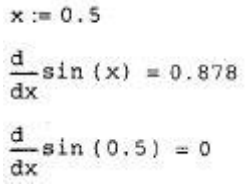


Рис. 37. Пример правильного (вверху) и неправильного (внизу) применения дифференцирования.

Его первые две строки вычисляют производную sin(x) в точке x=0.5. Последняя строка демонстрирует неправильное применение оператора дифференцирования. Вместо вычисления производной sin(x) в той же точке, как этого можно было ожидать, получено нулевое значение. Это случилось потому, что аргумент функции sin(x) введен не в виде переменной x, а в виде числа. Поэтому MathCAD воспринимает последнюю строку как вычисление сначала значения синуса в точке x=0.5, а затем дифференцирование этого значения (т. е. константы) также в точке x=0.5, в соответствии с требованием первой строки листинга. Поэтому ответ, на самом деле, неудивителен - в какой точке ни дифференцируй константу, результатом будет ноль.

Для численного дифференцирования MathCAD применяет довольно сложный алгоритм, вычисляющий производную с колоссальной точностью до 7-8-го знака после запятой. Этот алгоритм (метод Риддера) описан во встроенной справочной системе MathCAD, доступной через меню **Help** (Справка). Погрешность дифференцирования не зависит от констант TOL

или CTOL, в противоположность большинству остальных численных методов, а определяется непосредственно алгоритмом.

Исключение составляют функции, которые дифференцируются в окрестности сингулярной точки; например для рассмотренной нами функции f(x)=l/x это будут точки вблизи x=0. При попытке найти ее производную при x=0 будет выдано сообщение об одной из ошибок деления на ноль **"Can't divide by zero"** (Деление на ноль невозможно) или **"Found a singularity while evaluating this expression. You may be dividing by zero"** (Найдена сингулярность при вычислении этого выражения. Возможно, вы делите на ноль) Если попробовать численно определить производную очень близко к нулю, например, при $x=10^{100}$, то может появиться сообщение об ошибке **"Can't converge to a solution"** (Невозможно найти решение). Встретившись с одной из упомянутых ошибок, присмотритесь повнимательнее к дифференцируемой функции и убедитесь, что вы не имеете дело с точкой сингулярности.

MathCAD позволяет численно определять производные высших порядков, от 0-го до 5-го включительно. Чтобы вычислить производную функции f(x) N-го порядка в точке x, нужно проделать те же самые действия, что и при взятии первой производной, за тем исключением, что вместо оператора производной необходимо применить оператор N-й производной (Nth **Derivative).** Этот оператор вводится с той же панели **Calculus** (Вычисления), либо с клавиатуры нажатием клавиш <Ctii>+<?>, и содержит еще два местозаполнителя, в которые следует поместить число N. В полном соответствии с математическим смыслом оператора, определение порядка производной в одном из местозаполнителей приводит к автоматическому появлению того же числа в другом из них."Производная" при N=0 по определению равна самой функции, при N=1 получается обычная первая производная.

Рис. 38 демонстрирует численное и символьное вычисление второй производной.

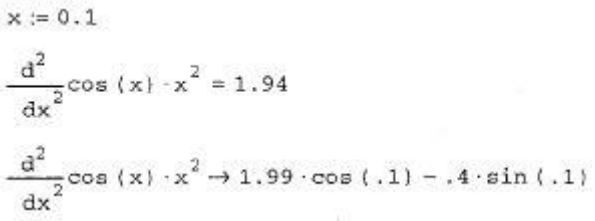


Рис. 38. Численное и символьное вычисление второй производной

Обратите внимание, что, как и при вычислении обычной производной, необходимо перед оператором дифференцирования присвоить аргументу функции значение, для которого будет вычисляться производная.

Убедиться в том, что символьный процессор MathCAD в последней строке рис. 38. дает тот же результат, что и вычислительный процессор в предыдущей строке, можно, упростив его. Для этого следует выделить полученное по следнее выражение и выбрать в меню **Symbolics** (Символика) пункт **Symplify** (Упростить). После этого ниже появится еще одна строка с численным результатом выделенного выражения.

Чтобы вычислить производную порядка выше 5-го, следует последовательно применить несколько раз оператор m-й производной подобно тому, как вводились операторы кратного интегрирования. Однако для символьных вычислений этого не потребуется - символьный процессор умеет считать производные порядка выше 5-го.

Сказанное иллюстрирует рис. 39, в котором сначала численно, а затем символьно вычисляется седьмая производная синуса в точке x=0.1.

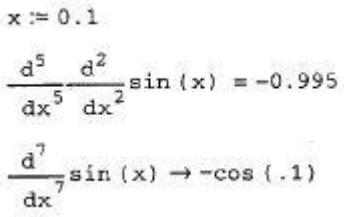


Рис. 39. Численное и символьное вычисление седьмой производной.

Расчет производных высших порядков производится тем же вычислительным методом Риддера, что и расчет первых производных. Причем для первой производной этот метод обеспечивает точность до 7-8 значащих разрядов числа, а при повышении порядка производной на каждую единицу точность падает примерно на один разряд. Из сказанного ясно, что падение точности при численном расчете высших производных может быть очень существенно. В частности, если попытаться определить девятую производную синуса, подобно идее листинга на рис. 39, то в качестве результата будет выдан нуль, в то время, как истинное значение девятой производной равно cos(0.1).

С помощью обоих процессоров MathCAD можно вычислять производные функций любого количества аргументов. В этом случае, как известно, производные по разным аргументам называются *частными.* Чтобы вычислить частную производную, необходимо, как обычно, ввести оператор производной с панели **Calculus** (Вычисления) и в соответствующем местозаполнителе напечатать имя переменной, по которой должно быть осуществлено дифференцирование. Пример приведен на рис. 40, в первой строке которого определена функция двух переменных, а в двух следующих строках символьным образом вычислены ее частные производные по обеим переменным - x и y - соответственно.

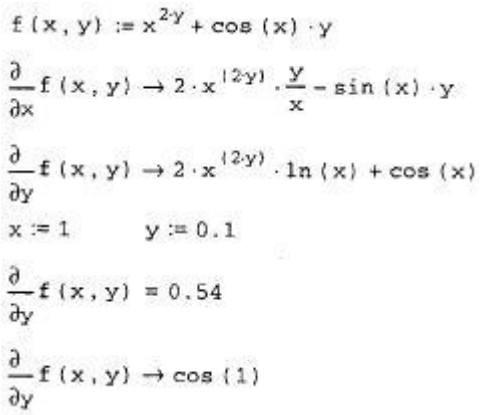


Рис. 40. Символьное и численное вычисление частных производных.

Чтобы определить частную производную численным методом, необходимо предварительно задать значения всех аргументов, что и сделано в следующих двух строках листинга. Последнее выражение на рис. 40 снова (как и в третьей строке) определяет символьно частную производную по y. Но, поскольку переменным x и y уже присвоено конкретное значение, то в результате получается число, а не аналитическое выражение.

Частные производные высших порядков рассчитываются точно так же, как и обычные производные высших порядков.

2.5. Решение уравнений в MathCAD. Поиск корня уравнения. Влияние точности вычисления на результат. Нахождение вектора корней алгебраического уравнения. Решение системы уравнений на основе FIND, MinErr и путем обращения матрицы. Режим OPTIMIZE. Решение дифференциальных уравнений.

Здесь рассматривается решение алгебраических нелинейных уравнений и систем таких уравнений.

Задача ставится следующим образом. Пусть имеется одно алгебраическое уравнение с неизвестным x. f(x)=0, или система N алгебраических уравнений, где f(x) - некоторая функция. Требуется найти корни уравнения, т. е. все значения x, которые переводят уравнение (или, соответственно, систему уравнений) в верное равенство (равенства).

Как правило, отыскание корней численными методами связано с несколькими задачами:

- исследование существования корней в принципе, определение их количества и примерного расположения;
- отыскание корней с заданной погрешностью TOL.

Последнее означает, что надо найти значения x_0, при которых $f(x_0)$ отличается от нуля не более чем на TOL. Почти все встроенные функции системы MathCAD, предназначенные для решения нелинейных алгебраических уравнений, нацелены на решение второй задачи, т. е. предполагают, что корни уже приблизительно локализованы. Чтобы решить первую задачу (предварительной локализации корней), можно использовать, например, графическое представление f(x), или последовательный поиск корня, начиная из множества пробных точек, покрывающих расчетную область (так называемое *сканирование*). MathCAD предлагает несколько встроенных функций, которые следует применять в зависимости от специфики уравнения, т. е. свойств f(x). Для решения одного уравнения с одним неизвестным служит функция root, реализующая "метод секущих"; для решения системы - вычислительный блок Given/Find, сочетающий различные "градиентные" методы. Если f(x) - это полином, то вычислить все его корни можно также с помощью функции. Кроме того, в некоторых случаях приходится сводить решение уравнений к задаче поиска экстремума. Различные приемы нахождения экстремумов функций реализуются при помощи встроенных функций Minerr,

Maximize и Minimize. В конце данной главы рассказывается о символьном решении уравнений о возможной программной реализации эффективного метода решения серии алгебраических уравнений или задач оптимизации, зависящих от параметра.

Одно уравнение с одним неизвестным. Рассмотрим одно алгебраическое уравнение с одним неизвестным x. f(x)=0, например, sin(x)=0.

Для решения таких уравнений MathCAD имеет встроенную функцию root, которая, в зависимости от типа задачи, может включать либо два, либо четыре аргумента и, соответственно, работает несколько по-разному.

- root(f(x),x);
- root(f(x),x,a,b);

где: f (x) - скалярная функция, определяющая уравнение;

x - скалярная переменная, относительно которой решается уравнение;

a,b - границы интервала, внутри которого происходит поиск корня.

Первый тип функции root требует дополнительного задания *начального значения* (guess value) переменной x. Для этого нужно просто предварительно присвоить x некоторое число. Поиск корня будет производиться вблизи этого числа. Таким образом, присвоение начального значения требует априорной информации о примерной локализации корня.

Приведем пример решения очень простого уравнения sin(x)=0, корни которого известны заранее.

График функции f(x)=sin(x) и положение найденного корня показаны на рис. 41. Обратите внимание, что хотя уравнение имеет бесконечное количество корней, MathCAD находит (с заданной точностью) только один из них, x_0, лежащий наиболее близко к x=0.5. Если задать другое начальное значение, например, x=3, то решением будет другой корень уравнения x_i=n и т. д. Таким образом, для поиска корня средствами

$x := 0.5$

$f(x) := \sin(x)$

$solution := root(f(x), x)$

$solution = -6.2 \times 10^{-7}$

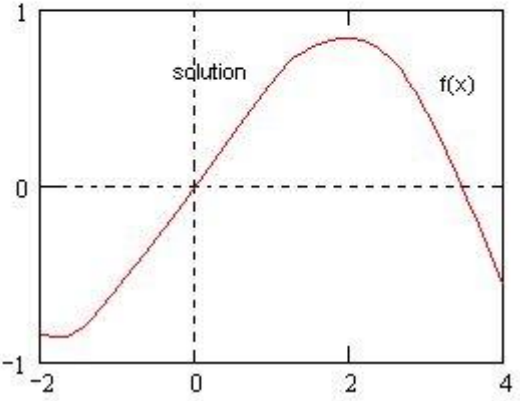


Рис. 41. Поиск корня нелинейного алгебраического уравнения и графическое решение уравнения sin(x)=0

MathCAD требуется его предварительная локализация. Это связано с особенностями выбранного численного метода, который называется *методом секущих* и состоит в следующем (рис. 42):

1. Начальное приближение принимается за нулевое приближение к корню: $x_0=x$.
2. Выбирается шаг и определяется первое приближение к корню $x_1=x_0+h$.
3. Через эти две точки проводится секущая - прямая линия, которая пересекает ось x в некоторой точке x_2. Эта точка принимается за второе приближение.
4. Новая секущая проводится через первую и вторую точки, тем самым определяя третье приближение, и т. д.

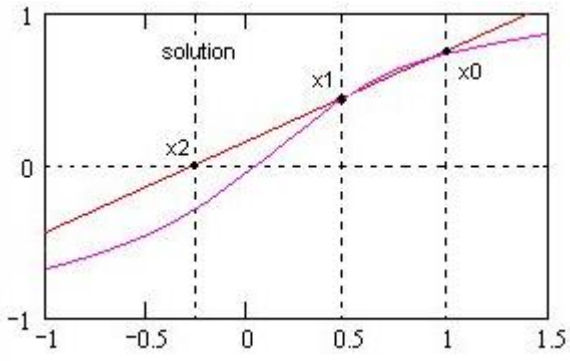


Рис. 42. Иллюстрация метода секущих.

5. Если на каком-либо шаге оказывается, что уравнение выполнено, т. е. |f (x)|<TOL, то итерационный процесс прерывается, и x выдается в качестве решения.

Влияние точности на результат. Результат, показанный на рис. 42, получен для погрешности вычислений, которой в целях иллюстративности предварительно присвоено значение TOL=0.5. Поэтому для поиска корня с такой невысокой точностью оказалось достаточно одной итерации. В вычислениях, приведенных на рис. 41, погрешность TOL=0.001 была установлена по умолчанию, и решение, выданное численным методом, лежало намного ближе к истинному положению корня x=0. Иными словами, чем меньше константа TOL, тем ближе к нулю будет значение f(x) в найденном корне, но тем больше времени будет затрачено вычислительным процессором MathCAD на его поиск.

Если уравнение неразрешимо, то при попытке найти его корень будет выдано сообщение об ошибке. Кроме того, к ошибке или выдаче неправильного корня может привести и попытка применить метод секущих в области локального максимума или минимума f(x). В этом случае секущая может иметь направление, близкое к горизонтальному, выводя точку

следующего приближения далеко от предполагаемого положения корня. Для решения таких уравнений лучше применять другую встроенную функцию Minerr. Аналогичные проблемы могут возникнуть, если начальное приближение выбрано слишком далеко от настоящего решения, или f(x) имеет особенности типа бесконечности.

Иногда удобнее задавать не начальное приближение к корню, а интервал [a,b], внутри которого корень заведомо находится. В этом случае следует использовать функцию root с четырьмя аргументами, а присваивать начальное значение x не нужно, как показано на рис. 43.

```
solution := root(sin(x), x, -1, 1)
solution = 0
```

Рис. 43. Поиск корня алгебраического уравнения в заданном интервале.

Поиск корня будет осуществлен в промежутке между a и b альтернативным численным методом (Риддера или Брента). Обратите внимание, что явный вид функции f(x) может быть определен непосредственно в теле функции root.

Когда root имеет четыре аргумента, следует помнить о двух ее особенностях:

- внутри интервала [a,b] не должно находиться более одного корня, иначе будет найден один из них, заранее неизвестно какой именно;
- значения f(a) и f(b) должны иметь разный знак, иначе будет выдано сообщение об ошибке.

Если уравнение не имеет действительных корней, но имеет мнимые, то их также можно найти.

Остается добавить, что f(x) может быть функцией не только x, а любого количества аргументов. Именно поэтому в самой функции root необходимо определить, относительно какого из аргументов следует решить уравнение. Эта возмож-

ность проиллюстрирована на рис. 44 на примере функции двух переменных $f(x,y)=x^2-y^2+3$.

$$f(x,y) := x^2 - y^2 + 3$$
$$x := 1$$
$$y := 0$$
$$root(f(x,y),x) = -1.732i$$
$$root(f(x,y),y) = 2$$

Рис. 44. Поиск корня уравнения, заданного функцией двух переменных.

В нем сначала решается уравнение f(x,0)=0 относительно переменной x, а потом - другое уравнение f(1,y)=0 относительно переменной y. В первой строке листинга определяется функция *f(x,y),* во второй и третьей - значения, для которых будет производиться решение уравнения по y их соответственно. В четвертой строке решено уравнение f(x,0)=0, а в последней - уравнение f(1,y)=0. Не забывайте при численном решении уравнений относительно одной из переменных предварительно определить значения остальных переменных. Иначе попытка вычислить уравнения приведет к появлению ошибки **"This variable or function is not defined above",** в данном случае говорящей о том, что другая переменная ранее не определена. Конечно, можно указать значение других переменных непосредственно внутри функции root, беспрепятственно удалив, например, вторую и третью строки листинга 8.4, и введя его последние строки в виде root(f(x,0),x)= и root(f(1, y),y) =, соответственно.

Вектор корней полинома. Если функция f(x) является *полиномом,* то все его корни можно определить, используя встроенную функцию polyroots(v), где v - вектор, составленный из коэффициентов полинома.

Поскольку полином N-Й степени имеет ровно N корней

(некоторые из них могут быть кратными), вектор v должен состоять из N+I элемента. Результатом действия функции polyroots является вектор, составленный из N корней рассматриваемого полинома. При этом нет надобности вводить какое-либо начальное приближение, как для функции root. Пример поиска корней полинома четвертой степени иллюстрируется рис. 45.

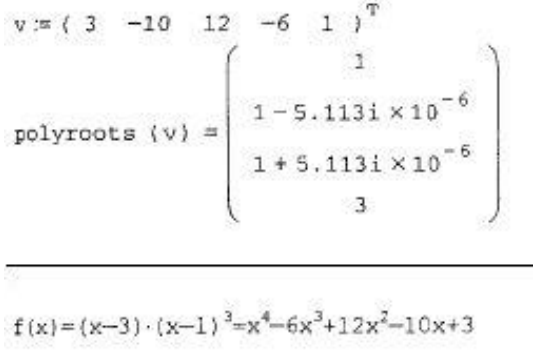


Рис. 45. Поиск корня полинома.

Коэффициенты рассматриваемого в примере полинома f (x) = (x-3)-(x-l)^3=x^4-6x^3+12x^2-10x+3 записаны в виде вектора в первой строке листинга. Первым в векторе должен идти свободный член полинома, вторым - коэффициент при x^1 и т. д. Соответственно, последним N+I элементом вектора должен быть коэффициент при старшей степени X^N.

Совет: *Иногда исходный полином имеется не в развернутом виде, а, например, как произведение нескольких полиномов. В этом случае определить все его коэффициенты можно, выделив его и выбрав в меню Symbolics (Символика) пункт Expand (Разложить). В результате символьный процессор MathCAD сам преобразует полином в нужную форму, пользователю надо будет только корректно ввести ее в аргументы функции poiyroots.*

Во второй строке на рис. 45 показано действие функции polyroots. Обратите внимание, что численный метод вместо двух из трех действительных единичных корней (иными словами, кратного корня 1) выдает два мнимых числа. Однако малая мнимая часть этих корней находится в пределах погрешности, определяемой константой TOL, и не должна вводить пользователей в заблуждение. Просто нужно помнить, что корни полинома могут быть комплексными, и ошибка вычислений может сказываться как на действительной, так и на комплексной части искомого корня.

Для функции polyroots можно выбрать один из двух численных методов - метод полиномов Лаггера (он установлен по умолчанию) или метод парной матрицы.

Для смены метода:

1. Вызовите контекстное меню, щелкнув правой кнопкой мыши на слове polyroots.
2. В верхней части контекстного меню выберите либо пункт **LaGuerre** (Лаггера), либо **Companion Matrix** (Парная матрица).
3. Щелкните вне действия функции polyroots - если включен режим автоматических вычислений, будет произведен пересчет корней полинома в соответствии с вновь выбранным методом.

Для того чтобы оставить за MathCAD выбор метода решения, установите флажок **AutoSelect** (Автоматический выбор), выбрав одноименный пункт в том же самом контекстном меню.

Системы уравнений. Рассмотрим решение системы N нелинейных уравнений с m неизвестными. Некоторые скалярные функции от скалярных переменных и, возможно, от еще каких-либо переменных. Уравнений может быть как больше, так и меньше числа переменных. Заметим, что любую систему можно формально переписать в виде f(x)=0, где x - вектор, составленный из переменных $x_1, x_2, \ldots, x_м$, а f(x) - соответствующая векторная функция.

Для решения систем имеется специальный *вычислительный блок,* состоящий из трех частей, идущих последовательно друг за другом:

- Given - ключевое слово;
- система, записанная логическими операторами в виде равенств и, возможно, неравенств;
- Find (x_1,... ,XM) - встроенная функция для решения системы относительно переменных x_1,... ,$x_м$.

Вставлять логические операторы следует, пользуясь панелью инструментов **Boolean** (Булевы операторы). Если вы предпочитаете ввод с клавиатуры, помните, что логический знак равенства вводится сочетанием клавиш<Ctrl>+<=>. Блок Given/Find использует для поиска решения итерационные методы, поэтому, как и для функции root, требуется задать начальные значения для всех X_1,...,X_M. Сделать это необходимо до ключевого слова Given. Значение функции Find есть вектор, составленный из решения по каждой переменной. Таким образом, число элементов вектора равно числу аргументов Find. На рис. 46 приведен пример решения системы двух уравнений.

В первых двух строках листинга вводятся функции, которые определяют систему уравнений. Затем переменным x и y, относительно которых она будет решаться, присваиваются начальные значения. После этого следует ключевое слово Given и два логических оператора, выражающих рассматриваемую систему уравнений. Завершает вычислительный блок функция Find, значение которой присваивается вектору v.

Следующая строка показывает содержание вектора v, т. е. решение системы. Первый элемент вектора есть первый аргумент функции Find, второй элемент - ее второй аргумент. В последних двух строках осуществлена проверка правильности решения уравнений.

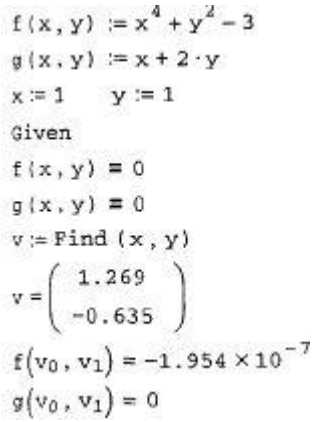


Рис. 46. Решение системы уравнений.

Совет: *Часто бывает очень полезно проверить точность решения уравнений, вычислив значения образующих их функций в найденных вычислительным процессором корнях, как это сделано в конце рис. 46.*

Пока мы рассмотрели пример системы из двух уравнений и таким же числом неизвестных, что встречается наиболее часто. Но число уравнений и неизвестных может и не совпадать. Более того, в вычислительный блок можно добавить дополнительные условия в виде неравенств. Например, введение ограничения на поиск только отрицательных значений x в рассмотренный выше рис. 46 приведет к нахождению другого решения, как это показано в рис. 47.

Обратите внимание, что, несмотря на те же начальные значения, что и в листинге рис. 46, мы получили в рис. 47 другой корень. Это произошло именно благодаря введению дополнительного неравенства, которое определено в блоке Given в предпоследней строке рис. 47.

Если предпринять попытку решить несовместную систему, MathCAD выдаст сообщение об ошибке, гласящее, что ни одного решения не найдено, и предложение попробовать

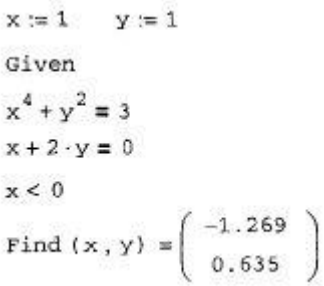


Рис. 47. Решение системы уравнений и неравенств

поменять начальные значения или значение погрешности.

Примечание: *Вычислительный блок использует константу CTOL в качестве погрешности выполнения уравнений, введенных после ключевого слова Given. Например, если CTOL:=0.001, то уравнение x=0 будет считаться выполненным и при x=10.001,и при x=9.999. Другая константа TOL определяет условие прекращения итераций численным алгоритмом (см. выше). Значение CTOL может быть задано пользователем также, как и TOL, например, CTOL:=0.01. По умолчанию принято, что CTOL=TOL=0.001, но вы по желанию можете переопределить их.*

Вычислительным блоком с функцией Find можно найти и корень уравнения с одним неизвестным. Действие Find в этом случае совершенно аналогично уже рассмотренным в данном разделе примерам. Задача поиска корня рассматривается как решение системы, состоящей из одного уравнения. Единственным отличием будет скалярный, а не векторный тип числа, возвращаемого функцией Find. Пример решения уравнения из предыдущего раздела приведен в на рис. 48.

В чем же отличие приведенного решения на рис. 41 с функцией root ? Оно состоит в том, что одна и та же задача решена различными численными методами. В данном случае выбор метода не сильно влияет на окончательный результат,

```
x := 0.5
Given
sin(x) = 0
Find(x) = -3.814 × 10^-7
```

Рис. 48. Поиск корня уравнения с одним неизвестным с помощью функции Find.

но бывают ситуации, когда применение того или иного метода имеет решающее значение.

Рассмотрим в данном разделе некоторые особенности численных методов и возможности установки их различных параметров, которые предоставляет MathCAD. Функция Find реализует градиентные численные методы. Покажем их основную идею на примере уравнения с одним неизвестным f(x)=0 для функции $f(x)=x^2+5x+2$, график которой показан на рис. 49.

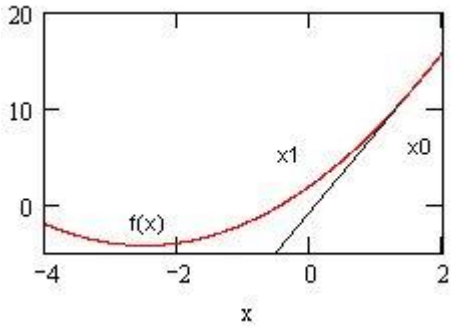


Рис. 49. Иллюстрация метода Ньютона.

Основная идея градиентных методов состоит в последовательных приближениях к истинному решению уравнения, которые вычисляются с помощью производной от f(x).

Приведем наиболее простую форму алгоритма, называемого методом Ньютона:

1. За нулевую итерацию принимается введенное пользователем начальное значение x_0=x.
2. В точке xo методом конечных разнос и вычисляется производная $f(x_0)$.
3. Пользуясь разложением Тейлора, можно заменить f(x) в окрестности x_0 касательной - прямой линией $f(x)=f(x_0)+f(x_0)\cdot(x-x_0)$.
4. Определяется точка x_i, в которой прямая пересекает ось x (см. рис. 49).
5. Если $f(x_1)<$ TOL, то итерации прерываются, и значение x_i выдается в качестве решения. В противном случае x_i принимается за новую итерацию, и цикл повторяется: строится касательная к f(x) в точке x_l, определяется x_2 - точка ее пересечения с осью x и т. д.

Модификация алгоритма Ньютона для решения системы нескольких уравнений заключается в линеаризации соответствующих функций многих переменных, т. е. аппроксимации их линейной зависимостью с помощью частных производных. Например, для нулевой итерации в случае системы двух уравнений используются выражения типа:

$$f_1(x, y) = f_1(x0, y0) + \frac{\partial f_1(x0, y0)}{\partial x}(x - x0) + \frac{\partial f_1(x0, y0)}{\partial y}(y - y0),$$

$$f_2(x, y) = f_2(x0, y0) + \frac{\partial f_2(x0, y0)}{\partial x}(x - x0) + \frac{\partial f_2(x0, y0)}{\partial y}(y - y0).$$

Чтобы отыскать точку, соответствующую каждой новой итерации, требуется приравнять оба равенства нулю, т. е. решить на каждом шаге полученную систему линейных уравнений.

MathCAD предлагает три различных вида градиентных методов. Чтобы поменять численный метод:
1. Щелкните правой кнопкой мыши на названии функции Find.
2. Наведите указатель мыши на пункт **Nonlinear** (Нелинейный) в контекстном меню.

3. В появившемся подменю (рис. 50) выберите один из трех методов:**Conjugate Gradient** (Сопряженных градиентов), **Quasi-Newton** (Квази-Ньютоновский) или **Levenberg-Marquardt** (Левенберга-Марквардта).

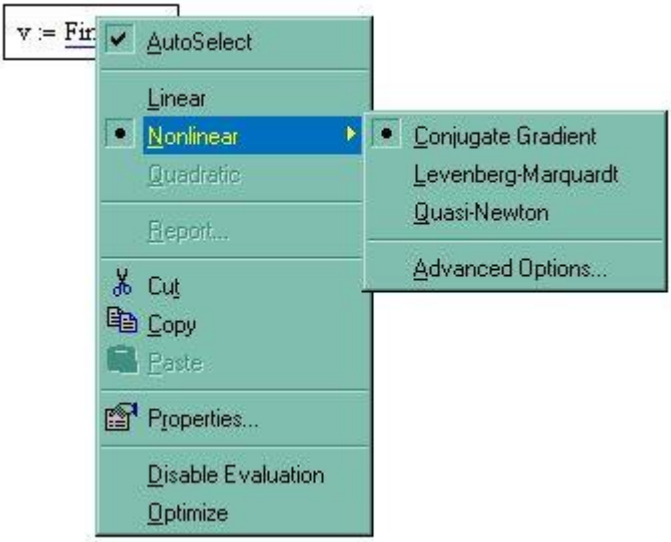


Рис. 50. Смена численного метода.

Чтобы вернуть автоматический выбор типа численного метода, в контекстном меню надо выбрать пункт **AutoSelect** (Автоматический выбор). Если установлена опция автоматического выбора (о чем говорит флажок, установленный в пункте **AutoSelect),** то текущий тип численного метода можно узнать, вызвав то же самое подменю и посмотрев, который из них отмечен точкой. Два последних метода являются квази-Ньютоновскими, основная идея которых была рассмотрена выше. Первый из них, метод сопряженных градиентов, является двухшаговым, - для поиска очередной итерации он использует как текущую, так и предыдущую итерации. Алго-

ритм Левенберга-Марквардта подробно описан в справочной системе MathCAD, а подробную информацию о методах Ньютона и сопряженных градиентов можно найти в большинстве книг по численным методам.

Помимо выбора самого метода, имеется возможность устанавливать их некоторые параметры. Для этого нужно вызвать с помощью того же контекстного меню диалоговое окно **Advanced Options** (Дополнительные параметры),выбрав в контекстном меню пункты **Nonlinear/Advanced options** (Нелинейный/Дополнительные параметры). В этом диалоговом окне (рис. 51) имеются три группы переключателей, по два в каждой.

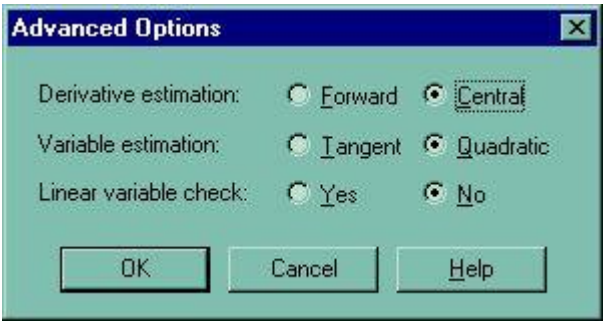


Рис. 51. Диалоговое окно **Advanced Options**

В первой строке **Derivative estimation** (Аппроксимация производной) определяется метод вычисления производной **Forward** (Вперед) или **Central** (Центральная). Они соответствуют аппроксимации производной либо правой (двухточечная схема "вперед"), либо центральной (трехточечная симметричная схема) конечной разностью.

Во второй строке **Variable estimation** (Аппроксимация переменных) можно определить тип аппроксимации рядом Тейлора. Для рассмотренного нами в этом разделе случая аппроксимации касательной прямой линией выберите переключатель

Tangent (Касательная), для более точной квадратичной аппроксимации (параболой), выберите **Quadratic** (Квадратичная). Наконец, последняя группа переключателей **Linear variable check** (Проверка линейности) позволяет в специфических задачах сэкономить время вычислений. Если вы уверены, что нелинейности всех функций, входящих в уравнение, мало сказываются на значениях всех их частных производных, то установите переключатель **Yes** (Да). В этом случае производные будут приняты равными константам и не будут вычисляться на каждом шаге.

Совет: *С осторожностью изменяйте параметры численных методов. Пользуйтесь ими, когда решение не находится при выставленных по умолчанию параметрах, или когда расчеты занимают очень продолжительное время.*

Приближенное решение уравнений: Иногда приходится заменять задачу отделения корней системы уравнений задачей поиска экстремума функции многих переменных. Например, когда невозможно найти решение с помощью функции Find, можно попытаться потребовать вместо точного выполнения уравнений условий минимизировать их невязку. Для этого следует в вычислительном блоке вместо функции Find использовать функцию Minerr, имеющую тот же самый набор параметров. Она также должна находиться в пределах вычислительного блока:

1. x_1:=d ... $x_м$: =$c_м$ - начальные значения для неизвестных.
2. Given - ключевое слово.
3. Система алгебраических уравнений и неравенств, записанная логическими операторами.
4. Minerr (x_1,... ,$x_м$) - приближенное решение системы относительно переменных x_1,... ,$x_м$, минимизирующее невязку системы уравнений.

В функции Minerr реализованы те же самые алгоритмы, что и в функции Find, иным является только условие завершения работы численного метода. Поэтому пользователь может тем

же самым образом, с помощью контекстного меню (см. рис. 48 и 49), выбирать численный алгоритм приближенного решения для функции Minerr.
Пример использования функции Minerr показан на рис. 52.

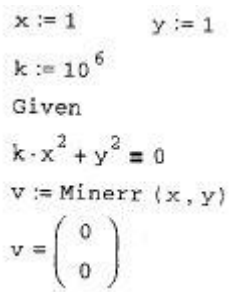


Рис. 52. Приближенное решение уравнения, имеющего корень (x=0, y=0)

Как видно, достаточно заменить в вычислительном блоке имя функции на Minerr, чтобы вместо точного (с точностью до TOL) получить приближенное решение уравнения, заданного после ключевого слова Given.

На рис. 52 мы рассмотрели пример нахождения существующего решения уравнения. Приведем пример нахождения функцией Minerr приближенного решения несовместной системы уравнений и неравенств (рис. 53). Решение, выдаваемое функцией Minerr, минимизирует невязку данной системы.

Как видно из рис. 53, в качестве результата выдаются значения переменных, наилучшим образом удовлетворяющие уравнению и неравенствам внутри вычислительного блока.

Численное решение систем уравнений путем обращения матрицы. Векторные и матричные операторы и функции системы MathCAD позволяют решать широкий круг задач линейной алгебры. К примеру, если задана матрица A и вектор B для системы линейных уравнений в матричной форме A·X=B,

$x := 1 \qquad y := 1$

Given

$x^2 + y^2 = -1$

$x > 0.1$

$y \leq -0.2$

$$\text{Minerr}(x, y) = \begin{pmatrix} -0.042 \\ -0.085 \end{pmatrix}$$

Рис. 53. Приближенное решение несовместной системы уравнений и неравенств.

то вектор решения можно получить из выражения $X=A^{-1}\cdot B$. На рис. 54 приведен пример решения системы линейных уравнений путем обращения матрицы.

Поскольку решение систем линейных уравнений довольно распространенная задача, то для этого в MathCAD введена встроенная функция lsolve(A,B), которая возвращает вектор X для системы линейных уравнений $A\cdot X=B$ при заданной матрице коэффициентов A и векторе свободных членов B. Если число уравнений n, размер вектора B должен быть n, а матрицы A - $n\times n$. Пример применения этой функции так же дан на рис. 54.

Оптимизация вычислений. Отличительная черта новых версий MathCAD - улучшенные возможности ускорения численных вычислений за счет применения элементов символьной математики. Непосредственно перед численным расчетом MathCAD автоматически пытается упростить выражение, используя символьный процессор. Это называется *оптимизацией.* За счет того, что от версии к версии качество работы символьного процессора улучшается, символьное преобразование зачастую существенно ускоряет расчеты. Режим оптимизации включается либо в документе целиком, либо для отдельных формул.

$$x_1 \cdot (4 + i) + x_2 \cdot (0.24) + x_3 \cdot (-0.08) := 8$$
$$x_1 \cdot (0.09) + x_2 \cdot (3) + x_3 \cdot (-0.15) := 9$$
$$x_1 \cdot (0.04) + x_2 \cdot (-0.08) + x_3 \cdot (4 + i) := 20$$

$$i := \sqrt{-1}$$

$$A := \begin{pmatrix} 4+i & 0.24 & -0.08 \\ 0.09 & 3 & -0.15 \\ 0.04 & -0.08 & 4+i \end{pmatrix} \qquad B := \begin{pmatrix} 8 \\ 9 \\ 20 \end{pmatrix}$$

$$X := A^{-1} \cdot B$$

$$X = \begin{pmatrix} 1.787 - 0.468i \\ 3.184 - 0.045i \\ 4.75 - 1.184i \end{pmatrix}$$

$$X1 := \text{lsolve}(A, B) \qquad X1 = \begin{pmatrix} 1.787 - 0.468i \\ 3.184 - 0.045i \\ 4.75 - 1.184i \end{pmatrix}$$

Рис. 54. Пример решения системы линейных уравнений путем обращения матрицы и при помощи встроенной функции **lsolve**.

Чтобы включить или отключить режим оптимизации всех выражений в активном документе, выберите команду **Math / Optimization** (Математика / Оптимизация), как показано на рис. 55.

Содержание документа, изображенного на этом же рисунке, помогает понять математический смысл режима оптимизации: для ускорения вычисления нижнего (определенного) интеграла выгодно использовать его аналитическое решение, определенное символьным процессором (верхний неопределенный интеграл).

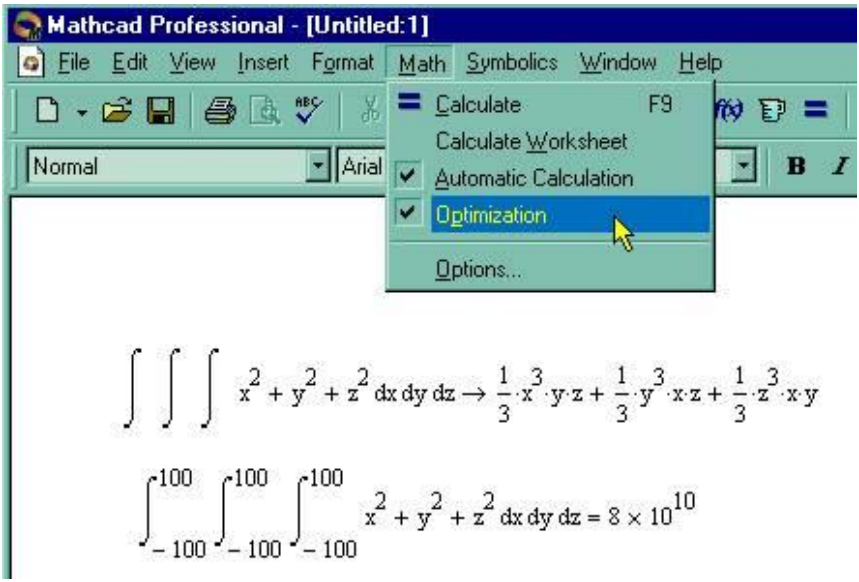


Рис. 55. Режим оптимизации вычислений.

Чтобы изменить режим оптимизации для отдельной формулы, не меняя выбранного режима для остальных формул документа, выберите элемент **Optimize** (Оптимизация) контекстного меню. Кроме того, для той же цели можно воспользоваться известным нам по предыдущему разделу диалоговым окном **Properties** (Свойства). Оно вызывается командой **Format/Properties/Calculations** (Формат/Свойства/ Вычисления) или через аналогичный пункт контекстного меню **Properties / Calculations** (Свойства / Вычисления).

Вызывая на какой-либо формуле контекстное меню, пользователь видит, включен или нет режим оптимизации, по наличию или отсутствию флажка проверки в пункте **Optimize** (Оптимизация). Можно узнать, удалось ли символьному процессору упростить выражение, приглядевшись к формуле с включенным режимом оптимизации. В ее конце присутствует символ в виде звездочки. Если звездочка красная, то символь-

ный процессор упростил выражение, если нет - то его усилия были напрасными.

Дифференциальные уравнения - это уравнения, в которых неизвестными являются не переменные (т. е. числа), а функции одной или нескольких переменных. Эти уравнения (или системы) включают соотношения между искомыми функциями и их производными. Если в уравнения входят производные только по одной переменной, то они называются *обыкновенными дифференциальными уравнениями* (далее чаще используется сокращение *ОДУ).*В противном случае говорят об *уравнениях в частных производных* .Таким образом, решить (иногда говорят *проинтегрировать)* дифференциальное уравнение - значит определить неизвестную функцию на определенном интервале изменения ее переменных.

Как известно, одно обыкновенное дифференциальное уравнение или система ОДУ имеет единственное решение, если помимо уравнения определенным образом заданы *начальные* или *граничные* условия. В соответствующих курсах высшей математики доказываются теоремы о существовании и единственности решения в зависимости от тех или иных условий. Имеются два типа задач, которые возможно решать с помощью MathCAD 2001:

- *задачи Коши* - для которых определены начальные условия на искомые функции, т. е. заданы значения этих функций в начальной точке интервала интегрирования уравнения;
- *краевые задачи* - для которых заданы определенные соотношения сразу на обеих границах интервала.

Как правило, решение задач Коши для ОДУ и их систем - задача хорошо разработанная и с вычислительной точки зрения не слишком сложная. Большое значение здесь имеет представление результатов и анализ зависимостей решения от различных параметров системы. Между тем, имеется целый класс ОДУ, называемых *жесткими,* который не поддается решению стандартными методами, типа методов Рунге-Кутты.

Для них в MathCAD имеются специальные возможности.

ОДУ первого порядка. Дифференциальное уравнение первого порядка может по определению содержать помимо самой искомой функции y(t) только ее первую производную y'(t). В подавляющем большинстве случаев дифференциальное уравнение можно записать в *стандартной форме (форме Коши):*

$$y'(t)=f(y(t),t) \qquad (**)$$

– и только с такой формой умеет работать вычислительный процессор MathCAD. Правильная с математической точки зрения постановка соответствующей задачи Коши для ОДУ первого порядка должна, помимо самого уравнения, содержать одно начальное условие - значение функции $y(t_0)$ в некоторой точке t_0. Требуется явно определить функцию y(t) на интервале от t_0 до t_1. По характеру постановки задачи Коши называют еще *задачами с начальными условиями* (initial value problem), в отличие от краевых задач.

Для численного интегрирования одного ОДУ у пользователя версии MathCAD 2001 (точнее, начиная с версии MathCAD 2000 Pro) имеется выбор -либо использовать вычислительный блок Given/Odesoive, либо встроенные функции, как в прежних версиях MathCAD. Первый путь предпочтительнее из соображений наглядности представления задачи и результатов, а второй дает пользователю больше рычагов воздействия на параметры численного метода. Рассмотрим последовательно оба варианта решения.

Вычислительный блок *Given/Odesolve.* Вычислительный блок для решения одного ОДУ, реализующий численный метод Рунге-Кутты, состоит из трех частей:

Given - ключевое слово;

ОДУ и начальное условие, записанное с помощью логических операторов, причем начальное условие должно быть в форме

$y(t_0)=b$;
odesoive(t,t_l) - встроенная функция для решения ОДУ относительно переменной t на интервале (t_0,t_1).
Примечание. *Допустимо, и даже часто предпочтительнее, задание функции Odesolve (t, t1, step) с тремя параметрами, где step - внутренний параметр численного метода, определяющий количество шагов, в которых метод Рунге-Кутты, будет рассчитывать решение дифференциального уравнения. Чем больше step, тем с лучшей точностью будет получен результат, но тем больше времени будет затрачено на его поиск. Помните, что подбором этого параметра можно заметно (в несколько раз) ускорить расчеты без существенного ухудшения их точности.*

Пример решения задачи Коши для ОДУ первого порядка $y' = y - y^2$ посредством вычислительного блока приведен на рис. 56:

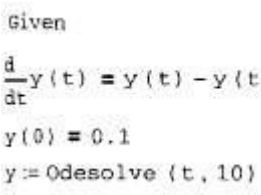


Рис. 56. Решение задачи Коши для ОДУ первого порядка.

Не забывайте о том, что вставлять логические операторы следует при помощи панели инструментов **Boolean** (Булевы операторы). При вводе с клавиатуры помните, что логическому знаку равенства соответствует сочетание клавиш <Ctrl>+<=>. Символ производной можно ввести как средствами панели **Calculus** (Вычисления), как это сделано на рис. 54, так и в виде штриха, набрав его с помощью сочетания клавиш <Ctrl>+<F7>. Выбирайте тот или иной способ представления производной из соображений наглядности представления ре-

зультатов – на ход расчетов он не влияет.

MathCAD требует, чтобы конечная точка интегрирования ОДУ лежала правее начальной: $t_0<t_1$ (на рис. 56 $t_0=0$, $t_1=10$), иначе будет выдано сообщение об ошибке. Как можно заметить, результатом применения блока Given/odesoive является функция y(t), определенная на промежутке (t_0,t_1). Следует воспользоваться обычными средствами MathCAD, чтобы построить ее график или получить значение функции в какой-либо точке указанного интервала, например: y(3) =0.691.

Пользователь имеет возможность выбирать между двумя модификациями численного метода Рунге-Кутты. Для смены метода необходимо нажатием правой кнопки мыши на области функции odesoive вызвать контекстное меню и выбрать в нем один из двух пунктов: **Fixed** (Фиксированный шаг) или **Adaptive** (Адаптивный). По умолчанию применяется первый из них, т. е.метод Рунге-Кутты с фиксированным шагом.

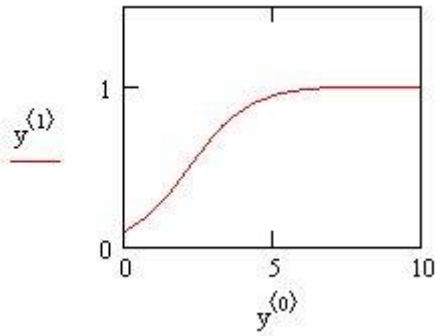
Встроенные функции *rkfixed, Rkadapt, Bulstoer.* Альтернативный метод решения ОДУ перешел из прежних версий MathCAD. Он заключается в использовании одной из встроенных функций rkfixed, Rkadapt или Bulstoer. Этот способ несколько проигрывает первому и в простоте, и в наглядности. Поэтому я советую предпочесть вычислительный блок Given/odesoive.

Приведем пример использования данного метода для решения задачи, уже решенной на рис. 56, с применением одной из трех существующих для этих целей встроенных функций rkfixed (рис. 57).

В MathCAD 2001 имеются три встроенные функции, которые позволяют решать поставленную в задачу Коши различными численными методами.

- rkfixed(y_0,t_0,t_1,M,D) - метод Рунге-Кутты с фиксированным шагом;
- Rkadapt(y_0,t_0,t_1,M,D) - метод Рунге-Кутты с переменным шагом;

```
y := 0.1
D(t, y) := y - y^2
M := 100
y := rkfixed(y, 0, 10, M, D)
```



y =

	0	1
0	0	0.1
1	0.1	0.109
2	0.2	0.119
3	0.3	0.13
4	0.4	0.142
5	0.5	0.155
6	0.6	0.168
7	0.7	0.183
8	0.8	0.198
9	0.9	0.215
10	1	0.232
11	1.1	0.25
12	1.2	0.269
13	1.3	0.29
14	1.4	0.311
15	1.5	0.332

Рис. 57. Решение задачи Коши для ОДУ первого порядка вторым способом и график решения. *Пример взят из области математической экологии и описывает динамику популяций с внутривидовой конкуренцией. Сначала происходит рост численности популяции, близкий к экспоненциальному, а затем выход на стационарное состояние.*

- Buistoer (y_0,t_0,t_1,M,D) - метод Булирша-Штера;

yo - вектор начальных значений в точке to размера NXI;

t_0 - начальная точка расчета;

t_1 - конечная точка расчета;

M - число шагов, на которых численный метод находит решение;

D - векторная функция размера Nx1 двух аргументов – скалярного t и векторного y. При этом y - искомая векторная функция аргумента t того же размера Nx1.

Соблюдайте регистр первой буквы рассматриваемых функций, поскольку это влияет на выбор алгоритма счета, в отличие от многих других встроенных функций MathCAD, например Find - find.

Обратите внимание только на необходимость явного задания количества точек интегрирования ОДУ M=100 в третьей строке листинга, а также на получение результата, в отличие от вычислительного блока, не в виде функции, а в виде матрицы размерности Mx2. Она состоит из двух столбцов: в одном находятся значения аргумента t (от t_0 до t_1 включительно), а в другом соответствующие значения искомой функции y(t). График решения рассматриваемого уравнения показан на рис. 57. Обратите внимание, что он соответствует получению решения в матричном виде, поэтому по осям отложены соответствующие столбцы, выделенные из матрицы **y** оператором < >.

ОДУ высшего порядка. Обыкновенное дифференциальное уравнение с неизвестной функцией y(t), в которое входят производные этой функции вплоть до $y^{(N)}(t)$, называется ОДУ N-го порядка. Если имеется такое уравнение, то для корректной постановки задачи Коши требуется задать N начальных условий на саму функцию y(t) и ее производные от первого до (N-l)-го порядка включительно. В MathCAD 2001 можно решать ОДУ высших порядков как с помощью вычислительного блока Given/odesoive, так и путем сведения их к системам уравнений первого порядка.

Внутри вычислительного блока:

- ОДУ должно быть линейно относительно старшей производной, т. е. фактически должно быть поставлено в стандартной форме;
- начальные условия должны иметь форму y(t)=b или y'(t)=b, а не более сложную (как например, встречающаяся в некоторых математических приложениях форма y(t)+y'(t)=b).

В остальном, решение ОДУ высших порядков ничем не отли-

чается от решения уравнений первого порядка что иллюстрируется рис. 58.

Given

$$\frac{d^2}{dt^2}y(t) + 0.1 \cdot \frac{d}{dt}y(t) + 1 \cdot y(t) = 0$$

$y(0) = 0.1$

$y'(0) = 0$

$y := Odesolve(t, 50)$

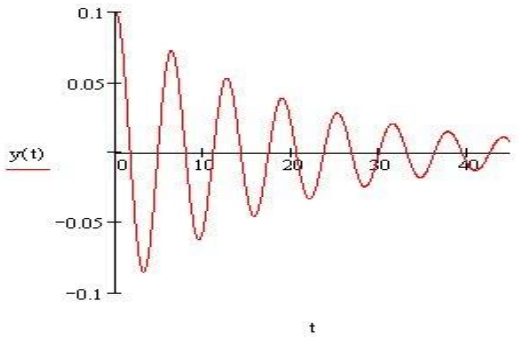
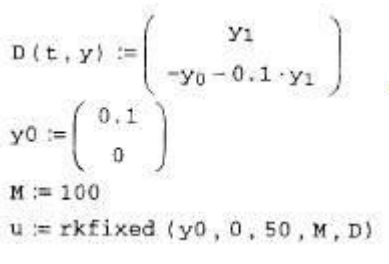


Рис. 58. Решение задачи Коши для ОДУ второго порядка. *Здесь решено уравнение затухающего гармонического осциллятора, которое описывает, например, колебания маятника. Для модели маятника y(t) описывает изменения угла его отклонения от вертикали, y'(t) - угловую скорость маятника, y''(t) - ускорение, а начальные условия, соответственно, начальное отклонение маятника y(0)=0.1 и начальную скорость y'(0) = 0.*

Как вы помните, допустимо написание производной как в виде знака дифференциала (так на рис. 58 введено само уравнение), так и с помощью штриха (так введено начальное условие для первой производной). Не забывайте пользоваться булевыми операторами при вводе уравнений и начальных условий. Полученное решение y(t) показано на рис. 58.

Второй способ решения ОДУ высшего порядка связан со сведением его к эквивалентной **системе ОДУ** первого порядка ("понизить порядок" уравнения). Покажем на том же примере из рис. 58, как решается система ОДУ. Действительно, если формально обозначить $y_0(t)=y(t)$, $y_1(t)=y'(t)$, то исходное уравнение на рис. 56 примет вид, сходный с (**): $y''(t) = -y_0(t) - 0.1\cdot y_1(t)$, а вместе с условиями запишется через функции $y_0(t)$ и $y_1(t)$ в виде системы двух ОДУ. Именно эта система решается в качестве примера ниже на рис. 59.



u =

	0	1	2
0	0	0.1	0
1	0.5	0.088	-0.047
2	1	0.056	-0.08
3	1.5	0.011	-0.093
4	2	-0.033	-0.082
5	2.5	-0.068	-0.053
6	3	-0.084	-0.013
7	3.5	-0.08	0.029
8	4	-0.057	0.062
9	4.5	-0.021	0.078
10	5	0.018	0.075
11	5.5	0.051	0.054
12	6	0.07	0.021
13	6.5	0.071	-0.015
14	7	0.056	-0.046
15	7.5	0.028	-0.064

Рис. 59. Решение системы двух ОДУ, полученных преобразованием уравнения второго порядка с примера на рис. 56.

Таким образом, любое ОДУ N-го порядка, линейное относительно высшей производной, можно свести к эквивалентной системе N дифференциальных уравнений.

В подавляющем большинстве случаев достаточно использовать первую функцию rkfixed, как это показано на рис. 58 на примере решения системы ОДУ модели осциллятора с затуханием.

Самая важная - это первая строка листинга, в которой, собственно, определяется система ОДУ. Во-первых, функция D(t,y), входящая в число параметров встроенных функций для решения ОДУ, должна быть функцией обязательно двух аргу-

ментов. Во-вторых, второй ее аргумент должен быть вектором того же размера, что и сама функция D(t,y). В-третьих, точно такой же размер должен быть и у вектора начальных значений y0 (он определен во второй строке листинга).

Не забывайте, что векторную функцию D(t,y) следует определять через компоненты вектора y с помощью кнопки нижнего индекса **(Subscript)**с наборной панели **Calculator** (Калькулятор) или нажатием клавиши <[>. В третьей строке листинга определено число шагов M, на которых рассчитывается решение, а его последняя строка присваивает матричной переменной **u** результат действия функции rkfixed. Решение системы ОДУ будет осуществлено на промежутке (0,50).

Как выглядит все решение, показано на рис. 59 справа. Размер полученной матрицы будет равен (M+1)x(2+1), т. е. 101×3. Просмотреть все компоненты матрицы и, которые не помещаются на экране, можно с помощью вертикаль ной полосы прокрутки.

В заключение следует сказать несколько слов об особенностях различных численных методов. Все они основаны на аппроксимации дифференциальных уравнений разностными аналогами. В зависимости от конкретной формы аппроксимации, получаются алгоритмы различной точности и быстродействия. В MathCAD использован наиболее популярный алгоритм Рунге-Кутты четвертого порядка, описанный в большинстве книг по методам вычислений. Он обеспечивает малую погрешность для широкого класса систем ОДУ за исключением жестких систем. Поэтому в большинстве случаев стоит применять функцию rkfixed. Если по различным причинам время расчетов становится критичным или точность неудовлетворительна, стоит попробовать вместо rkfixed другие функции, благо сделать это очень просто, благодаря одинаковому набору параметров. Для этого нужно только поменять имя функции в программе.

Функция Rkadapt может быть полезна в случае, когда из-

вестно, что решение на рассматриваемом интервале меняется слабо, либо существуют участки медленных и быстрых его изменений. Метод Рунге-Кутты с переменным шагом разбивает интервал не на равномерные шаги, а более оптимальным способом. Там, где решение меняется слабо, шаги выбираются более редкими, а в областях его сильных изменений - частыми. В результате для достижения одинаковой точности требуется меньшее число шагов, чем для rkfixed. Метод Булирша-Штера Bulstoer часто оказывается более эффективным для поиска гладких решений.

Решение систем ОДУ в одной заданной точке. Зачастую при решении дифференциальных уравнений требуется определить значения искомых функций не на всем интервале (t_0,t_1), а только в одной его последней точке. Например, весьма распространены задачи поиска аттракторов динамических систем. Известно, что для широкого класса ОДУ одна и та же система при разных (или даже любых, как рассмотренный выше пример осциллятора с затуханием) начальных условиях при t приходит в одну и ту же точку (аттрактор). Поэтому часто нужно определить именно эту точку.

Такая задача требует меньше ресурсов компьютера, чем решение системы ОДУ на всем интервале, поэтому в MathCAD 2001 имеются модификации встроенных функций Rkadapt и Buistoer. Они имеют несколько другой набор параметров и работают быстрее своих аналогов.

- rkadapt (y_0, t_0, t_1, acc, D, k, s) - метод Рунге-Кутты с переменным шагом;
- buistoer (y_0, t_0, t_1, acc, D, k, s) - метод Булирша-Штера;

y_0 - вектор начальных значений в точке to;

t_0, t_1 - начальная и конечная точки расчета;

acc - погрешность вычисления (чем она меньше, тем с лучшей точностью будет найдено решение; рекомендуется выбирать значения погрешности в районе 0.001);

D - векторная функция, задающая систему ОДУ;

k - максимальное число шагов, на которых численный метод будет находить решение;

s - минимально допустимая величина шага.

Как легко заметить, вместо числа шагов на интервале интегрирования ОДУ, в этих функциях необходимо задать точность расчета численным методом значения функций в последней точке. В этом смысле параметр **acc** похож на константу TOL, которая влияет на большинство встроенных численных алгоритмов MathCAD. Количество шагов и их расположение определяется численным методом автоматически, чтобы обеспечить эту точность. Два последних параметра нужны для того, чтобы пользователь мог искусственно повлиять на разбиение интервала на шаги. Параметр **k** служит для того, чтобы шагов не было чрезмерно много, причем нельзя сделать **k**>1000. Параметр **s** - для того, чтобы ни один шаг не был слишком малым для появления больших погрешностей при разностной аппроксимации дифференциальных уравнений внутри алгоритма. Эти параметры следует задавать явно, исходя из свойств конкретной системы ОДУ. Как правило, проведя ряд тестовых расчетов, можно подобрать их оптимальный набор для каждого конкретного случая. Пример использования функции buistoer для того же примера (рис. 59) приведен на рис. 60.

В его первых двух строках, как обычно, определяется система уравнений и начальные условия; в следующей строке матрице **u** присваивается решение, полученное с помощью buistoer. Структура этой матрицы в точности такая же, как и в случае решения системы ОДУ посредством уже рассмотренных нами ранее встроенных функций. Однако в данном случае нам интересна только последняя точка интервала. Поскольку сделанное численным методом количество шагов, т. е. размер матрицы и, заранее неизвестно, то его необходимо предварительно определить.

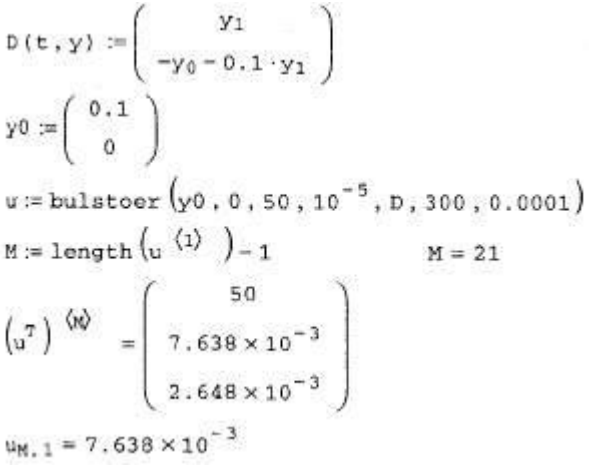


Рис. 60. Решение системы двух ОДУ.

Это сделано в четвертой строке листинга, присваивающей это число переменной M, в этой же строке оно выведено на экран. В предпоследней строке листинга осуществлен вывод решения системы ОДУ на конце интервала, т. е. в точке t=50 в виде вектора. В последней строке для примера еще раз выводится искомое значение первой функции из системы ОДУ(сравните его с соответствующим местом вектора из предыдущей строки).

Внимание! Функции bulstoer и rkadapt (те, что пишутся с маленькой буквы) не предназначены для нахождения решения в промежуточных точках интервала, хотя они и выдают их в матрице-результате.

Жесткие системы ОДУ. До сих пор мы имели дело с "хорошими" уравнениями, которые надежно решались численными методами Рунге-Кутты. Однако имеется класс так называемых *жестких* (stiff) систем ОДУ, для которых стандартные методы практически неприменимы, поскольку их решение требует исключительно малого значения шага численного метода. Некоторые из специальных алгоритмов, раз-

работанных для этих систем, реализованы в MathCAD.

Строгого общепринятого математического определения жестких ОДУ нет. В рамках этой книги будем считать, что жесткие системы - это те уравнения, решение которых получить намного проще с помощью определенных неявных методов, чем с помощью явных методов (типа тех, что были рассмотрены в предыдущих разделах). Примерно такое определение было предложено в 1950-х годах классиками в этой области Кертиссом и Хиршфельдером. Исторически, интерес к жестким системам возник в середине XX века при изучении уравнений химической кинетики с одновременным присутствием очень медленно и очень быстро протекающих химических реакций. Тогда неожиданно оказалось, что считавшиеся исключительно надежными методы Рунге-Кутты стали давать сбой при расчете этих задач.

Решение жестких систем дифференциальных уравнений можно осуществить только с помощью встроенных функций, аналогичных по действию семейству рассмотренных выше функций для обычных ОДУ. Отличием является добавление к стандартному набору параметров дополнительной матричной функции, задающей якобиан системы ОДУ. Решение выдается в виде матрицы, по форме идентичной аналогичным функциям решения нежестких задач Коши.

- stiffb(y_0, t_0, t_1,M, F,J) - метод Булирша-Штера для жестких систем ОДУ;
- stiffr (y_0, t_0, t_1,M,F, J) - метод Розенброка для жестких систем ОДУ;

y_0 - вектор начальных значений в точке to;

t_0, t_1, - начальная и конечная точки расчета;

M - число шагов численного метода;

F - векторная функция F(t,y) размера 1×N, задающая систему ОДУ;

J - матричная функция J(t,y) размера (N+1J×N, составленная из вектора производных функции F(t,y) no t (правый столбец)

и ее якобиана (N левых столбцов). Матрица Якоби (якобиан) векторной функции F(t,y), есть функциональная матрица, составленная из частных производных F(t,y). Чем вырожденнее матрица Якоби, тем жестче система уравнений.

Покажем действие этих алгоритмов на примере жесткой системы ОДУ химической кинетики (рис. 61).

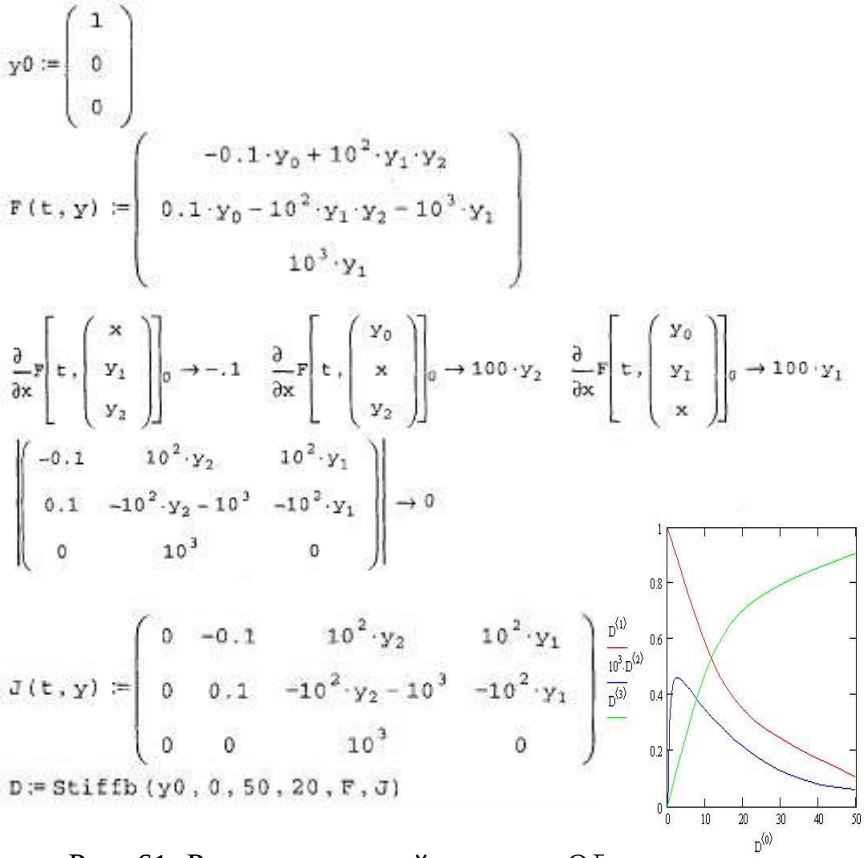


Рис. 61. Решение жесткой системы ОДУ химическои кинетики F(t,y) с нахождением Якобиана J(t,y) путем нахождения частных производных от функции в матричной форме.

Пусть вещество "0" медленно превращается в "1": "0">>"1" (со скоростью 0.l), вещество "1" при каталитическом воздействии самого себя превращается очень быстро в вещество "2": "1"+"1">>"2"+"1" (10^3). И, наконец, подобным образом (но со средней скоростью) реагируют вещества "2" и "1": "2"+"1">>"2" (10^2). Обратите внимание, как следует представлять в данном случае якобиан, сравнив задание матричной функции в предпоследней строке листинга.

В заключение приведем встроенные функции, которые применяются для решения жестких систем ОДУ не на всем интервале, а только в одной заданной точке ti.
- stiffb(y_0, t_0, t_1, acc, F, J, k, s) - метод Булирша-Штера;
- stiffr (y_0, t_0, t_1, acc, F, J, k, s) - метод Розенброка.

Имена этих функций пишутся с маленькой буквы, а их действие и набор параметров полностью аналогичны рассмотренным нами ранее для функций, относящихся к решению в заданной точке нежестких систем (см. выше). Отличие заключается в специфике применяемого алгоритма и необходимости задания матричной функции якобиана J(t,y).

Регрессия. Задачи математической регрессии имеют смысл приближения выборки данных (x_i,y_i) некоторой функцией f(x), определенным образом минимизирующей совокупность ошибок $|f(x_i)-Y_i|$. Регрессия сводится к подбору неизвестных коэффициентов, определяющих аналитическую зависимость f(x).

В силу производимого действия большинство задач регрессии являются частным случаем более общей проблемы сглаживания данных. Как правило, регрессия очень эффективна, когда заранее известен (или, по крайней мере, хорошо угадывается) закон распределения данных (x_i,y_i). В MathCAD существует большое количество функций, осуществляющих регрессию кривыми различного вида.

Но, как правило, для обработки экспериментальных научных данных они оказываются бесполезны, так как полу-

ченные зависимости зачастую имеют сложный вид и представляют собой комбинацию специальных функций. Для обработки таких данных применяют регрессию общего вида. Получив путем регрессии значения коэффициентов в известном выражении, описывающем совокупность экспериментальных данных, можно составить представление о процессах, протекающих в объекте исследования.

В MathCAD можно осуществить регрессию в виде линейной комбинации $F(x)=c_1f_1(x)+c_2f_2(x) + ...$, где $f_i(x)$ - любые функции пользователя, c_i - подлежащие определению коэффициенты. Такая регрессия осуществляется при помощи функции linfit. Рассмотрим пример, регрессии, приведенный на рис. 62.

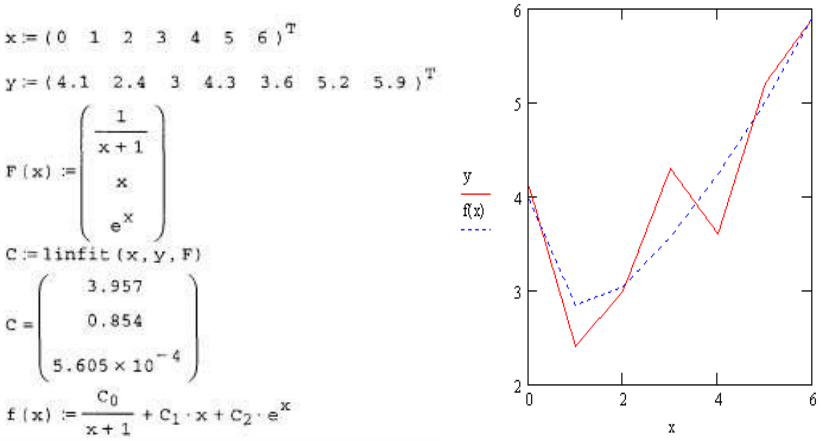


Рис. 62. Регрессия линейной комбинацией функций пользователя.

В начале задаются два вектора значений x и y экспериментальной зависимости, которая будет исследоваться. Вообще-то данные должны представлять собой матрицу из одного столбца, но для экономии места на странице данные были

написаны в строчку и поставлен оператор транспонирования. Значения в векторе x обязательно должны располагаться либо в порядке возрастания, либо в порядке убывания. Далее располагается функция F(x), которая представляет собой матрицу из одного столбца, каждый элемент которой представляет собой функцию $f_i(x)$, при которой будет найден коэффициент c_i. Затем располагается функция linfit(x,y,F), вычисляющая вектор C - параметров линейной комбинации функций пользователя, осуществляющей регрессию данных. Полученный результат приведен ниже, там же приведена полученная функция f(x) для графической проверки качества регрессии.

Кроме того, имеется путь проведения регрессии более общего вида, когда комбинацию функций и искомых коэффициентов задает сам пользователь. Приведем встроенные функции для регрессии общего вида и пример их использования (рис. 63).

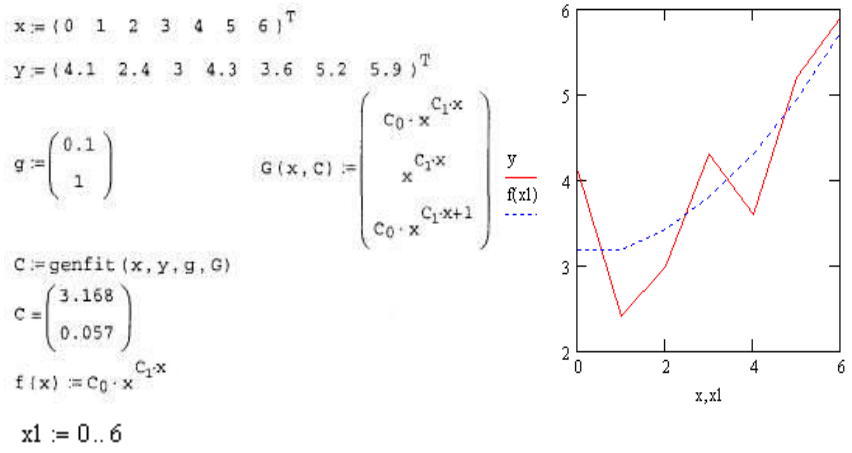


Рис. 63. Регрессия общего вида.

В начале, так же как и предыдущем примере, задаются два вектора значений x и y экспериментальной зависимости, которая будет исследоваться. Значения в векторе x обязатель-

но должны располагаться либо в порядке возрастания, либо в порядке убывания. Затем определяется g - вектор начальных значений параметров регрессии размерности N. От того, насколько удачно выбраны начальные значения, сильно зависит время расчетов и даже полученный результат. Далее располагается функция G(x,C), которая представляет собой матрицу из одного столбца, первым элементом которой является сама функция пользователя, а остальными – частные производные этой функции по искомым коэффициентам, то есть по C_0 и C_1. (Следует обратить внимание, что искомые коэффициенты C_0 и C_1 являются элементами матрицы, и чтобы ввести матричные индексы используется клавиша с изображением квадратной скобки **[**.) Таким образом функция G(x,C) имеет размерность N+1. Затем располагается функция genfit(x,y,g,G), вычисляющая вектор параметров C, реализующих регрессию данных с помощью функций пользователя общего вида. Полученный результат приведен ниже, там же приведена полученная функция f(x) для графической проверки качества регрессии.

Как видно из приведенных выше примеров, применение функции linfit более простое, не требует задания начального приближения и всегда дает однозначный результат. Результат, полученный при помощи genfit, сильно зависит от выбора начальных условий в векторе g и установленной точности проведения расчетов. В ряде случаев экспериментальная кривая может быть одинаково успешно описана несколькими наборами числовых коэффициентов, поэтому проблема корректного определения начальных условий стоит очень остро и остается на полное усмотрение пользователя.

2.6. Символьные вычисления в MathCAD. Разложение на множители, упрощение выражений. Разложение в ряд. Дифференцирование и интегрирование. Поиск корня уравнения. Решение системы уравнений. Обращение мат-

рицы. Интегральные преобразования – Фурье, Лапласса, Z-преобразование

Здесь рассматриваются возможности символьного процессора MathCAD. Он позволяет решить многие задачи математики аналитически, без применения численных методов и, соответственно, без погрешностей вычислений.

Способы символьных вычислений. Символьные вычисления в MathCAD можно осуществлять в двух различных вариантах:

- с помощью команд меню;
- с помощью оператора символьного вывода "→", ключевых слов символьного процессора и обычных формул (в справочной системе MathCAD этот способ называется *символьными вычислениями в реальном времени* – live symbolic evaluation).

Первый способ более удобен, когда требуется быстро получить какой-либо аналитический результат для однократного использования, не сохраняя сам ход вычислений. Второй способ более нагляден, т. к. позволяет записывать выражения в традиционной математической форме и сохранять символьные вычисления в документах MathCAD. Кроме того, аналитические преобразования, проводимые через меню, касаются только одного, выделенного в данный момент, выражения. Соответственно, на них не влияют формулы, находящиеся в документе MathCAD выше этого выделенного выражения (например, операторы присваивания значений каким-либо переменным). Оператор символьного вывода, напротив, учитывает все предыдущее содержимое документа и выдает результат с его учетом.

Для символьных вычислений при помощи команд предназначено главное меню **Symbolics** (Символика), объединяющее математические операции, которые MathCAD умеет выполнять аналитически (рис. 62). Для реализации второго способа применяются все средства MathCAD, пригодные для

численных вычислений (например, панели **Calculator, Evaluation** и т. д.), и специальная математическая панель инструментов, которую можно вызвать на экран нажатием кнопки **Symbolic Keyword Toolbar** (Панель символики) на панели **Math** (Математика) (рис. 64).

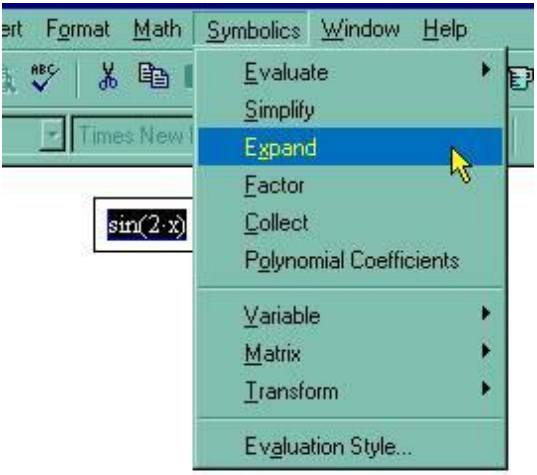


Рис. 64. Меню Symbolics.

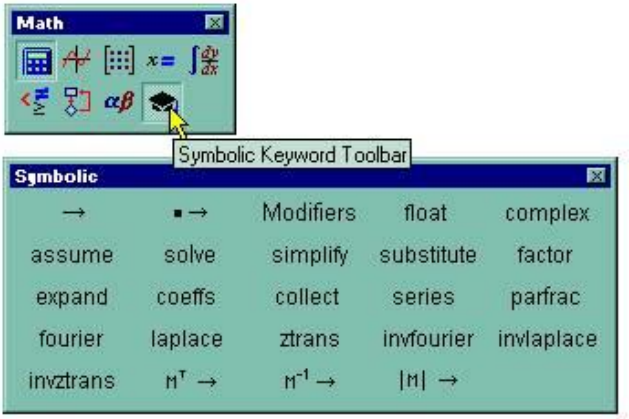


Рис. 65. Панель Symbolic

На панели **Symbolic** (Символика) находятся кнопки, соответствующие специфическим командам символьных преобразований. Например, таким как разложение выражения на множители, расчет преобразования Лапласа и другим операциям, которые в MathCAD нельзя проводить численно, и для которых, соответственно, не предусмотрены встроенные функции.

Рассмотрим оба типа символьных вычислений на простом примере разложения на сомножители выражения sin (2-x).

Первый способ (с помощью меню):

1. Введите выражение sin (2-x).
2. Выделите его целиком (см. рис. 64).
3. Выберите в главном меню пункты **Symbolics / Expand** (Символика / Разложить).

После этого результат разложения выражения появится чуть ниже в виде еще одной строки (рис. 66).

$$\sin(2 \cdot x)$$

$$2 \cdot \sin(x) \cdot \cos(x)$$

Рис. 66. Результат применения команды меню Symbolics/Expand

Внимание! *Свольные операции с помощью меню возможны лишь над каким-либо объектом (выражением, его частью или отдельной переменной). Для того чтобы правильно осуществить желаемое аналитическое преобразование, предварительно необходимо выделить тот объект, к которому оно будет относиться. В данном случае преобразование было применено ко всему выражению sin (2-x). Если же выделить часть формулы, как показано на рис. 67, то соответствующее преобразование будет отнесено к выделенной части*

(нижняя строка на этом рисунке).

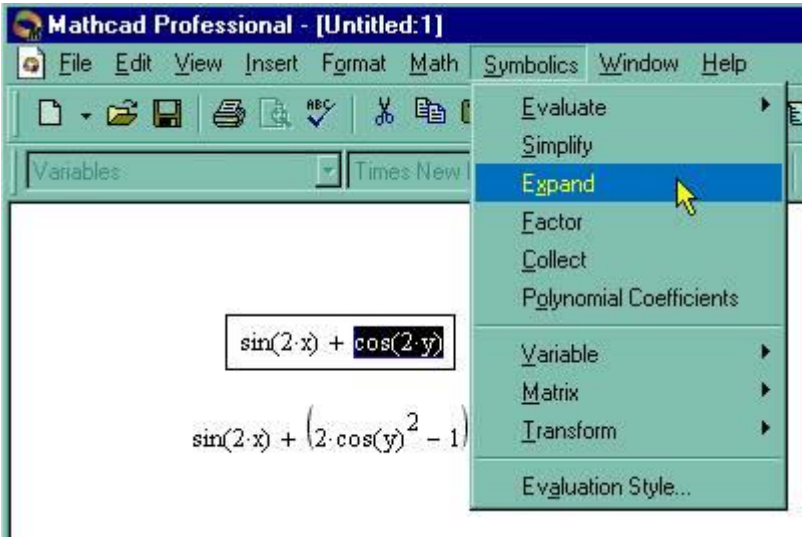


Рис. 67. Символьное разложение части выражения и его результат.

Второй способ символьных преобразований (с помощью оператора "→").
1. Введите выражение sin (2-x).
2. Нажмите кнопку **Expand** (Разложить) на панели **Symbolic** (Символика) (см. рис. 65):

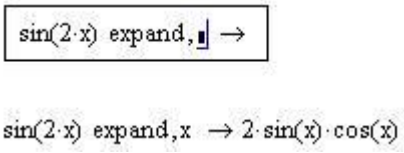


Рис. 68. Символьное разложение выражения.

3. Введите в местозаполнитель после появившегося ключево-

го слова expand (рис. 68, сверху) имя переменной x, либо нажмите клавишу <Del>, чтобы просто удалить местозаполнитель.
4. Введите оператор символьного вывода
5. Нажмите клавишу <Enter>, либо просто щелкните мышью за пределами выражения.

Оператор символьного вывода, как вы помните, можно ввести в редакторе MathCAD несколькими способами: нажатием кнопки "→" на любой из панелей **Evaluation** (Выражения) или **Symbolic** (Символика) либо сочетания клавиш <Ctrl>+<.>. Результат символьного разложения выражения показан на рис. 68, снизу.

Внимание! *Если символьные вычисления осуществляются вторым способом, символьный процессор учитывает все формулы, предварительно введенные в документе (рис. 69, снизу). Но если те же преобразования выполняются при помощи меню, символьный процессор "не видит" ничего, кроме одной формулы и воспринимает все ее переменные аналитически, даже если им предварительно были присвоены какие-то значения (рис. 69, сверху). По этой причине, например, символьным преобразованиям через меню недоступны предварительные определения функций пользователя.*

Не всякое выражение поддается аналитическим преобразованиям. Если это так (либо в силу того, что задача вовсе не имеет аналитического решения, либо она оказывается слишком сложной для символьного процессора MathCAD), то в качестве результата выводится само выражение:

$$\cos(2\cdot x)\ \text{expand},x \rightarrow 2\cdot\cos(x)^2-1$$
$$\cos(x)\ \text{expand},x \rightarrow \cos(x)$$

Символьный процессор MathCAD умеет выполнять основные алгебраические преобразования, такие как упрощение выражений, разложение их на множители, символьное суммирование и перемножение.

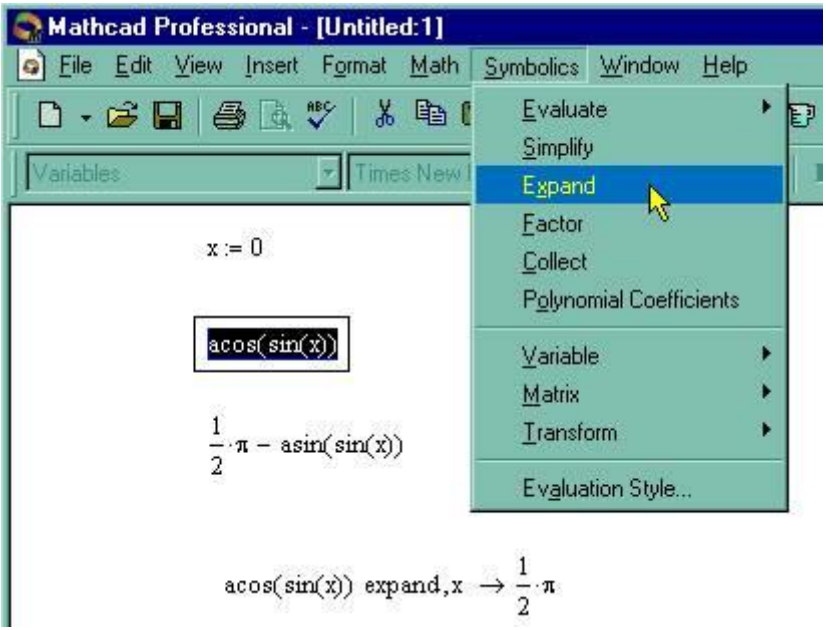


Рис. 69. Различие в символьных вычислениях при помощи меню (сверху) и оператора "→" (снизу)

Упрощение выражений (Simplify). *Упрощение выражений* - наиболее часто применяемая операция. Символьный процессор MathCAD стремится так преобразовать выражение, чтобы оно приобрело более простую форму. При этом используются различные арифметические формулы, приведение подобных слагаемых, тригонометрические тождества, пересчет обратных функций и др. Чтобы упростить выражение с помощью меню (рис. 70):

1. Введите выражение.
2. Выделите выражение целиком или его часть, которую нужно упростить.
3. Выберите команду **Symbolics / Simplify** (Символика / Упро-

стить).

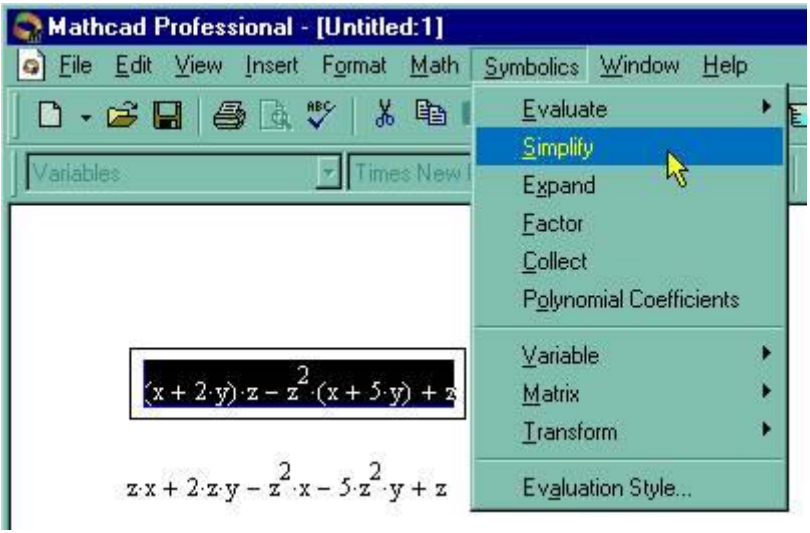


Рис. 70. Упрощение выражения.

Для упрощения выражения при помощи оператора символьного вывода используйте ключевое слово simplify.

$$(x+2\cdot y)\cdot z - z^2\cdot(x+5\cdot y)+z \text{ simplify } \rightarrow z\cdot x+2\cdot z\cdot y - z^2\cdot x - 5\cdot z^2\cdot y + z$$

Не забывайте, если некоторым переменным, входящим в выражение, ранее были присвоены некоторые значения, то они будут подставлены в него при выполнении символьного вывода:

$$x := 10 \quad y := 1$$

$$(x+2\cdot y)\cdot z - z^2\cdot(x+5\cdot y)+z \text{ simplify } \rightarrow 13\cdot z - 15\cdot z^2$$

Упрощение выражений, содержащих числа, производится по-

разному, в зависимости от наличия в числах десятичной точки. Если она есть, то выполняется непосредственное вычисление выражения:

$\sqrt{3}$ simplify $\rightarrow \sqrt{3}$

$\sqrt{3.01}$ simplify $\rightarrow 1.7349351572897472412$

Разложение выражений (Expand). Операция символьного *разложения,* или *расширения,* выражений противоположна по смыслу операции упрощения. В ходе разложения раскрываются все суммы и произведения, а сложные тригонометрические зависимости разлагаются с помощью тригонометрических тождеств. Разложение выражений производится путем выбора команды **Symbolics / Expand** (Символика / Разложить), либо использованием вместе с оператором символьного вывода ключевого слова expand. *Применение операции разложения было подробно рассмотрено выше* (см. рис. 67-69).

Разложение на множители (Factor). Разложение выражений на простые множители производится при помощи команды **Symbolics / Factor** (Символика / Разложить на множители) (рис. 71), либо использованием вместе с оператором символьного вывода ключевого слова factor:

$x^4 - 16$ factor $\rightarrow (x-2)\cdot(x+2)\cdot\left(x^2+4\right)$

28 factor $\rightarrow 2^2\cdot 7$

Эта операция позволяет разложить полиномы на произведение более простых полиномов, а целые числа - на простые сомножители. Применяя команду меню, не забывайте перед ее вызовом выделить все выражение или его часть, которую планируете разложить на множители.

Разложение в ряд (Expand to Series). С помощью символьного процессора MathCAD возможно получить разложение выражения в *ряд Тейлора* по любой переменной x в точке

x=0, т.е. представить выражение в окрестности точки x

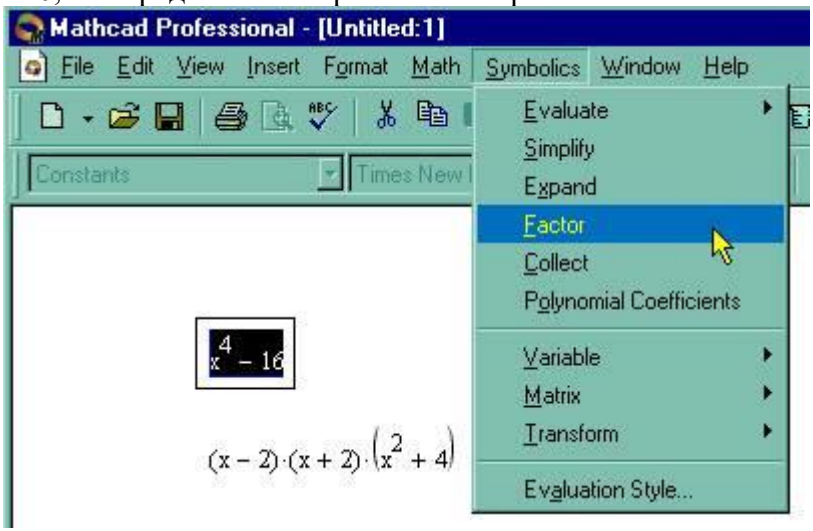


Рис. 71. Разложение выражения на множители.

суммой вида $a_0+a_1x+a_2x^2+a_3x^3+...$ Здесь a - некоторые коэффициенты, не зависящие от x, но, возможно, являющиеся функциями других переменных, входящих в исходное выражение. Если выражение имеет в точке x=0 особенность, то соответствующее разложение называют *рядом Лорана.*

Чтобы разложить выражение в ряд:

1. Введите выражение.
2. Выделите значение переменной, по которой требуется получить разложение в ряд.
3. Выполните команду **Symbolics / Variable / Expand to Series** (Символика / Переменная / Разложить в ряд) (рис. 72):
4. В появившемся диалоговом окне (рис. 73) введите желаемый порядок аппроксимации **(Order of Approximation)** и нажмите кнопку OK:

Результат разложения появится под выражением (рис. 74): Не забывайте, что разложение строится только в точке x=0.

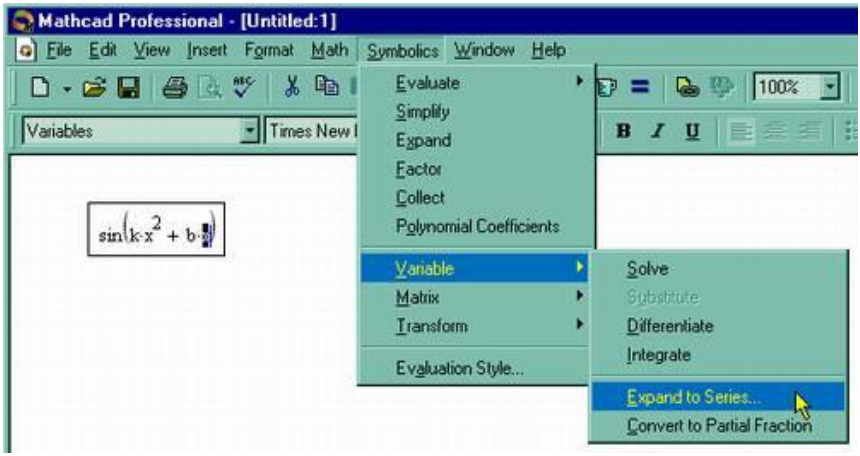


Рис. 72. Подготовка выражения для разложения в ряд по переменной x.

Рис. 73. Разложение в ряд Тейлора.

$$\sin\left(k\cdot x^2 + b\cdot x\right)$$

$$b\cdot x + k\cdot x^2 + \frac{-1}{6}\cdot b^3\cdot x^3 + \frac{-1}{2}\cdot k\cdot b^2\cdot x^4 + \left(\frac{1}{120}\cdot b^5 - \frac{1}{2}\cdot k^2\cdot b\right)\cdot x^5 + O\left(x^6\right)$$

Рис. 74. Результат разложения в ряд Тейлора.

Чтобы получить разложение в другой точке x=a, можно, к

примеру, подставить вместо переменной x значение x-а.

Для разложения в ряд альтернативным способом, с помощью оператора символьного вывода, используйте ключевое слово series, вставляя его одноименной кнопкой панели **Symbolic** (Символика). После ключевого слова series, через запятую, указывается имя переменной, по которой производится разложение, и порядок аппроксимации (рис. 75):

$$\sin(k\cdot x^2+b\cdot x)\ \text{series},x,2 \rightarrow b\cdot x$$
$$\sin(k\cdot x^2+b\cdot x)\ \text{series},x,3 \rightarrow k\cdot x^2+b\cdot x$$
$$\sin(k\cdot x^2+b\cdot x)\ \text{series},x,4 \rightarrow b\cdot x+k\cdot x^2-\frac{1}{6}\cdot b^3\cdot x^3$$
$$\sin(k\cdot x^2+b\cdot x)\ \text{series},x,5 \rightarrow b\cdot x+k\cdot x^2-\frac{1}{6}\cdot b^3\cdot x^3-\frac{1}{2}\cdot k\cdot b^2\cdot x^4$$

Рис. 75. Разложение выражения в ряд с разным порядком аппроксимации

$$\sin(k\cdot x^2+b\cdot x)\ \text{series},k,3 \rightarrow \sin(b\cdot x)+\cos(b\cdot x)\cdot x^2\cdot k-\frac{1}{2}\cdot\sin(b\cdot x)\cdot x^4\cdot k^2$$
$$\sin(k\cdot x^2+b\cdot x)\ \text{series},b,3 \rightarrow \sin(k\cdot x^2)+\cos(k\cdot x^2)\cdot x\cdot b-\frac{1}{2}\cdot\sin(k\cdot x^2)\cdot x^2\cdot b^2$$

Рис. 76. Разложение выражения в ряд по разным переменным.

Сравнение функции и ее разложений в ряды с разными порядками аппроксимации (для k=b=1) иллюстрируется рис. 77.

Видно, что разложение в ряд хорошо работает в окрестности точки x=0, а по мере удаления от нее все сильнее и сильнее отличается от функции.

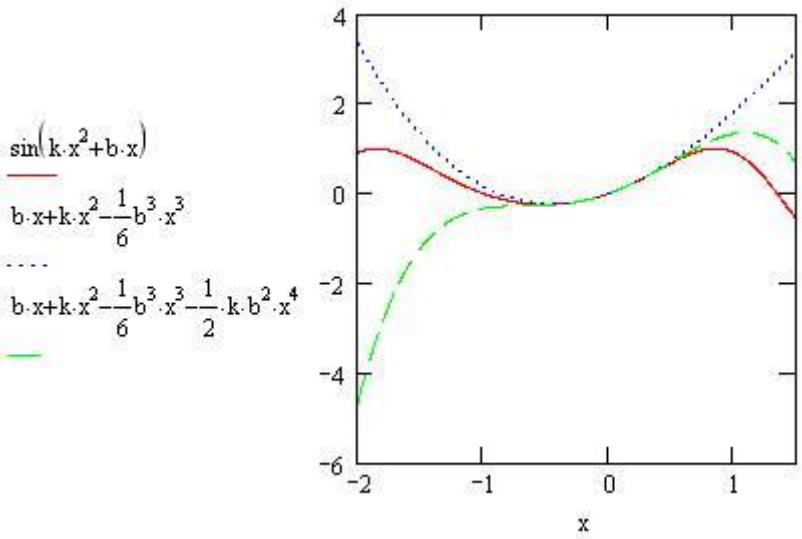


Рис. 77. Функция и ее разложения в ряды Тейлора.

Решение уравнений (Solve). С помощью символьного процессора можно вычислить аналитически значение переменной, при котором выражение обращается в ноль. Для этого:

1. Введите выражение.
2. Выделите переменную, относительно которой будет решаться уравнение, приравнивающее выражение нулю.
3. Выберите в меню **Symbolics** (Символика) пункт **Variable / Solve** (Переменная / Решить) (рис. 78).

Дифференцирование (Differentiate). Чтобы аналитически продифференцировать выражение по некоторой переменной, выделите в нем эту переменную и выберите команду **Symbolics / Variable /Differentiate** (Символика/Переменная/Дифференцировать) (рис. 79).

В результате в следующей строке за выражением появится значение ее производной. Для того чтобы найти вторую производную, повторно примените эту последовательность

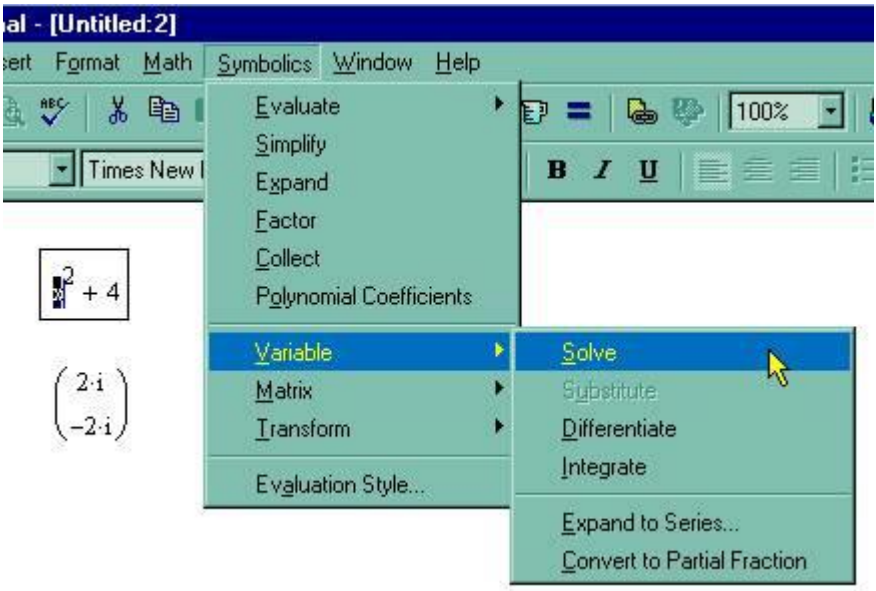


Рис. 78. Символьное решение уравнения.

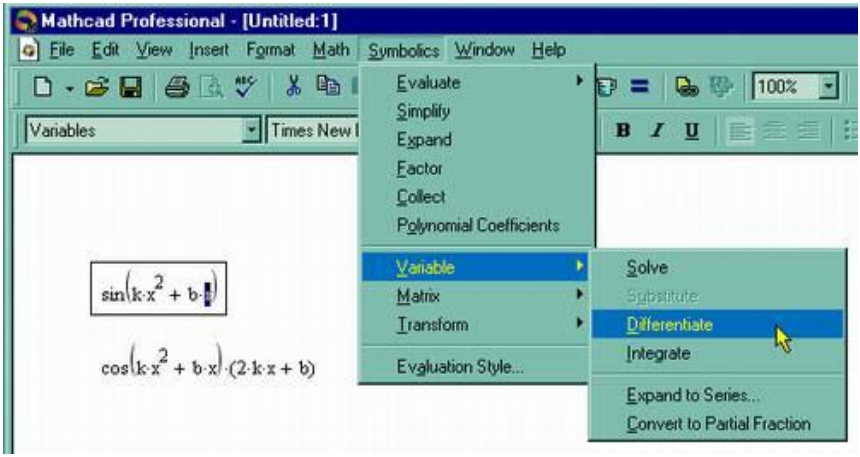


Рис. 79. Дифференцирование по переменной.

действий, но уже к полученному результату дифференцирования. Так же находятся и производные высших порядков.

Интегрирование (Integrate). Для вычисления неопределенного интеграла от некоторого выражения по определенной переменной выделите в выражении переменную и выполните команду **Symbolics / Variable / Integrate** (Символика / Переменная / Интегрировать) (рис. 80). Вычисленное аналитическое представление неопределенного интеграла появится ниже. При этом результат может содержать как встроенные в MathCAD функции, так и другие спецфункции, которые нельзя непосредственно рассчитать в MathCAD, но символьный процессор "умеет" выдавать их в качестве результата некоторых символьных операций.

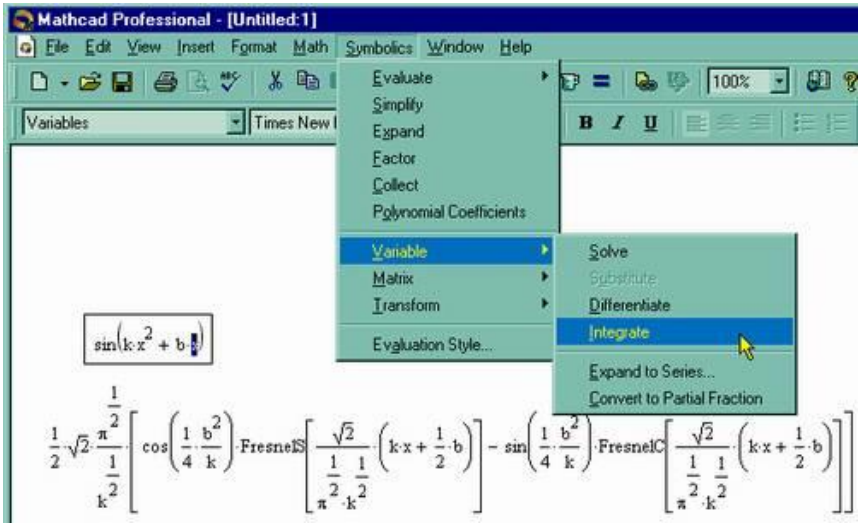


Рис. 80. Интегрирование по переменной.

Символьное решение уравнений. Некоторые уравнения можно решить точно с помощью символьного процессора MathCAD. Делается это очень похоже на численное решение

уравнений с применением вычислительного блока. Присваивать неизвестным начальные значения нет необходимости. Рис. 81 и 82 демонстрируют символьное разрешение уравнения с одним неизвестным и системы двух уравнений соответственно.

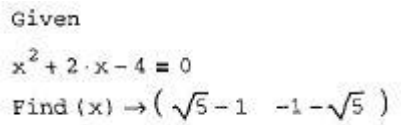


Рис. 81. Символьное решение алгебраического уравнения с одним неизвестным.

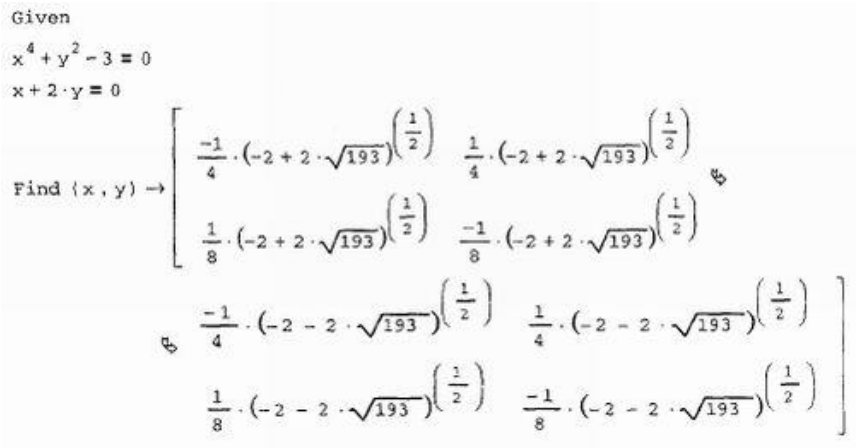


Рис. 82. Символьное решение системы алгебраических уравнений.

Как видно, вместо знака равенства после функции Find в листингах следует знак символьных вычислений, который можно ввести с панели **Symbolic** (Символика) или, нажав клавиши <Ctrl>+<.>. Не забывайте, что сами уравнения должны иметь вид логических выражений, т. е. знаки равенства нужно вво-

дить с помощью панели **Booleans** (Булевы операторы). Обратите внимание, что на рис. 82 вычислены как два первых действительных корня, которые мы уже находили численным методом. С помощью символьного процессора решить уравнение с одним неизвестным можно и по-другому:

1. Введите уравнение, пользуясь панелью **Booleans** (Булевы операторы) илинажав клавиши <Ctrl>+<> для получения логического знака равенства, например, $x^2+2-x-4=0$.
2. Щелчком мыши выберите переменную, относительно которой вы собираетесь решить уравнение.
3. Выберите в меню **Symbolics** (Символика) пункт **Variable / Solve** (Переменная / Решить).

После строки с уравнением появится строка с решением или сообщение о невозможности символьного решения этого уравнения. В данном примере после осуществления описанных действий появляется вектор, состоящий из двух корней уравнения.

Символьные вычисления могут производиться и над уравнениями, в которые, помимо неизвестных, входят различные параметры. На рис. 83 приведен пример решения уравнения четвертой степени с параметром a. Как видите, результат получен в аналитической форме:

Рис. 83. Символьное решение уравнения, зависящего от параметра.

Символьные операции с матрицами. Все матричные и векторные операторы, о которых шла речь выше, допустимо использовать в символьных вычислениях. Мощь символьных операций заключается в возможности проводить их не только

над конкретными числами, но и над переменными. Несколько примеров приведены на рис. 84. Первое выражение - пример символьного перемножения матриц. Второе – пример символьного обращения матрицы. Рядом справа – пример сильного вычисления определителя матрицы. Последняя строка – символьное вычисление суммы элементов матрицы.

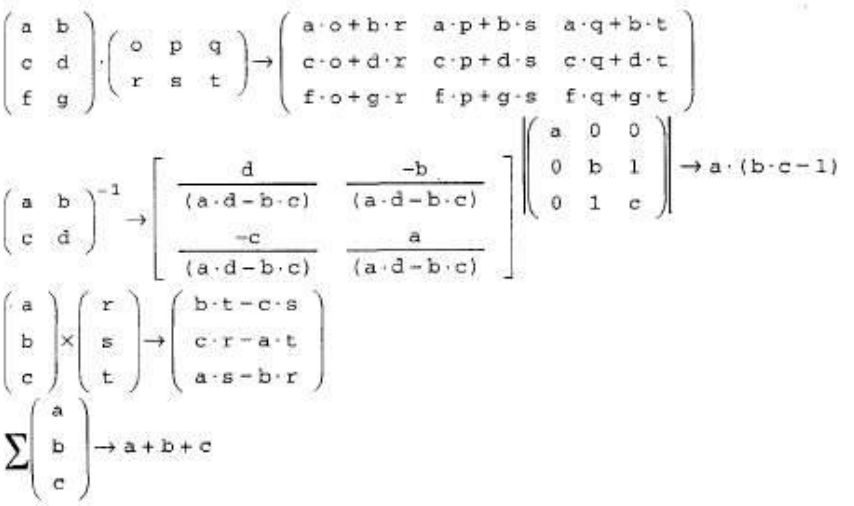


Рис. 84. Примеры символьных операций над векторами и матрицами.

Интегральные преобразования. Интегральные преобразования, по определению, ставят в соответствие некоторой функции f(x) другую функцию от другого аргумента Причем это соответствие f(X)→F(CO) задается интегральной зависимостью. Символьный процессор MathCAD позволяет осуществлять три вида интегральных преобразований функций - преобразование Фурье, Лапласа и Z-преобразование. Наряду с прямыми преобразованиями, имеется возможность совершать любое из этих трех обратных преобразований, т. е. F(co)→f(x).

Выполняются все символьные интегральные преобразования аналогично уже рассмотренным операциям. Для вычисления преобразования выражения выделяется переменная, по которой будет осуществляться преобразование, и затем выбирается соответствующий пункт меню. Преобразования с применением оператора символьного вывода используются с одним из соответствующих ключевых слов, вслед за которым требуется указать имя нужной переменной.

Преобразование Фурье (Fourier). Преобразование Фурье представляет функцию f (x) в виде интеграла по гармоническим функциям, называемого *интегралом Фурье:*

$$F(\omega) = \int_{-\infty}^{\infty} f(x) \cdot \exp(-i\omega x)\, dx.$$

Аналитический расчет преобразования Фурье при помощи меню показан на рис. 85.

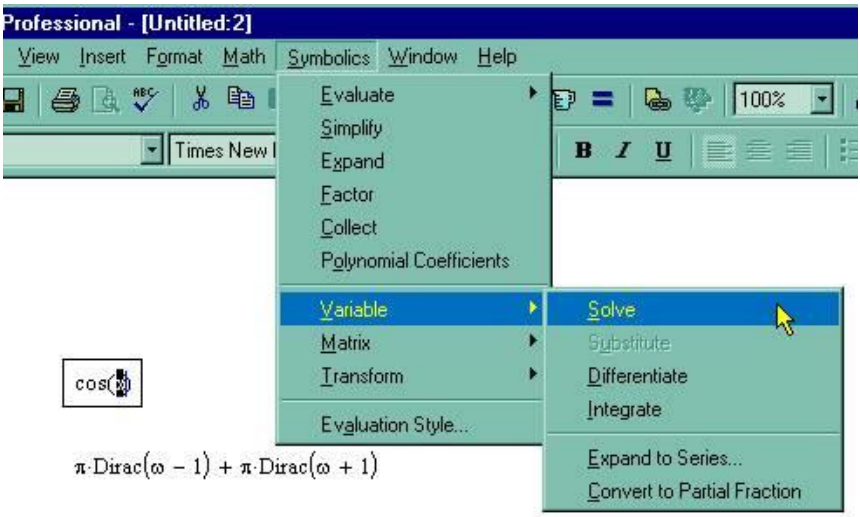


Рис. 85. Расчет Фурье-преобразования при помощи меню.

На рис. 86 приведены два примера вычисления прямого преобразования Фурье с применением ключевого слова fourier и оператора символьного вывода.

$$\cos(x)\ \text{fourier},x \rightarrow \pi\cdot\text{Dirac}(\omega-1)+\pi\cdot\text{Dirac}(\omega+1)$$
$$\left(x^2+4\right)\ \text{fourier},x \rightarrow -2\cdot\pi\cdot\text{Dirac}(2,\omega)+8\cdot\pi\cdot\text{Dirac}(\omega)$$

Рис. 86. Прямое преобразование Фурье.

На рис. 87 иллюстрируется обратное преобразование Фурье одной из функций предыдущего листинга.

$$-2\cdot\pi\cdot\text{Dirac}(2,\omega)+8\cdot\pi\cdot\text{Dirac}(\omega)\ \text{invfourier},\omega \rightarrow t^2+4$$

Рис. 87. Обратное преобразование Фурье.

В MathCAD преобразование Фурье также можно вычислить и с помощью численного процессора, использующего популярный алгоритм быстрого преобразования Фурье (БПФ).
Преобразование Лапласа (Laplace). Преобразованием Лапласа называют интеграл от f(x) следующего вида:

$$F(s)=\int_0^{\infty} f(x)\cdot\exp(-sx)\,dx.$$

Рассчитывается преобразование Лапласа совершенно аналогично Фурье-преобразованию *(см. предыдущий раздел).* Примеры преобразования Лапласа приведены рис. 88.

Z-преобразование (Z). Z-преобразование функции f(x) определяется через бесконечную сумму ряда. Пример Z-преобразования приведен на рис. 89.

$$x^2 + 4 \text{ laplace}, x \rightarrow \frac{2}{s^3} + \frac{4}{s}$$

$$\frac{2}{s^3} + \frac{4}{s} \text{ invlaplace}, s \rightarrow t^2 + 4$$

Рис. 88. Прямое и обратное преобразование Лапласа

$$x^2 + 4 \text{ ztrans}, x \rightarrow z \cdot \frac{\left(-7 \cdot z + 5 + 4 \cdot z^2\right)}{(z-1)^3}$$

$$z \cdot \frac{\left(-7 \cdot z + 5 + 4 \cdot z^2\right)}{(z-1)^3} \text{ invztrans}, z \rightarrow 4 + n^2$$

Рис. 89. Прямое и обратное Z-преобразование.

2.7. Построение графиков в MathCAD. Построение X-Y графика в декартовых и полярных системах. Семейства графиков. Изменение параметров графика. Цифровое измерение точек графика. Построение трехмерных графиков поверхности. Построение линий уровня, векторного поля. Настройка параметров графиков.

Создание графиков. В MathCAD 2001 встроено несколько различных типов графиков, которые можно разбить на две большие группы.

- Двумерные графики:
  XY (декартовый) график (XY Plot);
  полярный график (Polar Plot).- Трехмерные графики:
  график трехмерной поверхности (Surface Plot);
  график линий уровня (Contour Plot);
- трехмерная гистограмма (3D Bar Plot);
  трехмерное множество точек (3D Scatter Plot);
  векторное поле (Vector Field Plot).

Деление графиков на типы несколько условно, т. к., управляя установками многочисленных параметров, можно создавать комбинации типов графиков, а также новые типы (например, двумерная гистограмма распределения является разновидностью простого XY-графика).

Все графики создаются совершенно одинаково, с помощью панели инструментов **Graph** (График), различия обусловлены отображаемыми данными.

Некорректное определение данных приводит, вместо построения графика, к выдаче сообщения об ошибке.

Чтобы создать график, например двумерный декартов:

1. Поместите курсор ввода в то место документа, куда требуется вставить график.
2. Если на экране нет панели **Graph** (График), вызовите ее нажатием кнопки с изображением графиков на панели **Math** (Математика).
3. Нажмите на панели **Graph** (График) кнопку **X-Y Plot** для создания декартового графика (рис. 90) или другую кнопку для иного желаемого типа графика.

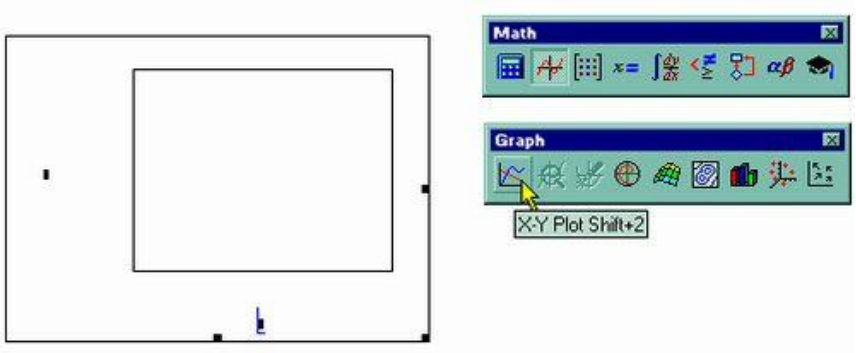


Рис. 90. Создание декартового графика.

4. В результате в обозначенном месте документа появится пустая область графика с одним или несколькими местозаполни-

телями (рис. 90, слева). Введите в местозаполнители имена переменных или функций, которые должны быть изображены на графике. В случае декартова графика это два местозаполнителя данных, откладываемых по осям x и y.

Если имена данных введены правильно, нужный график появится на экране. Созданный график можно изменить, в том числе, меняя сами данные, форматируя его внешний вид или добавляя дополнительные элементы оформления.

Самый наглядный способ создания графика - с помощью панели инструментов **Graph** (График). Однако точно так же создаются графики путем выбора соответствующего элемента подменю **Insert / Graph** (Вставка / График), показанного на рис. 91, либо нажатием соответствующей типу графика горячей клавиши.

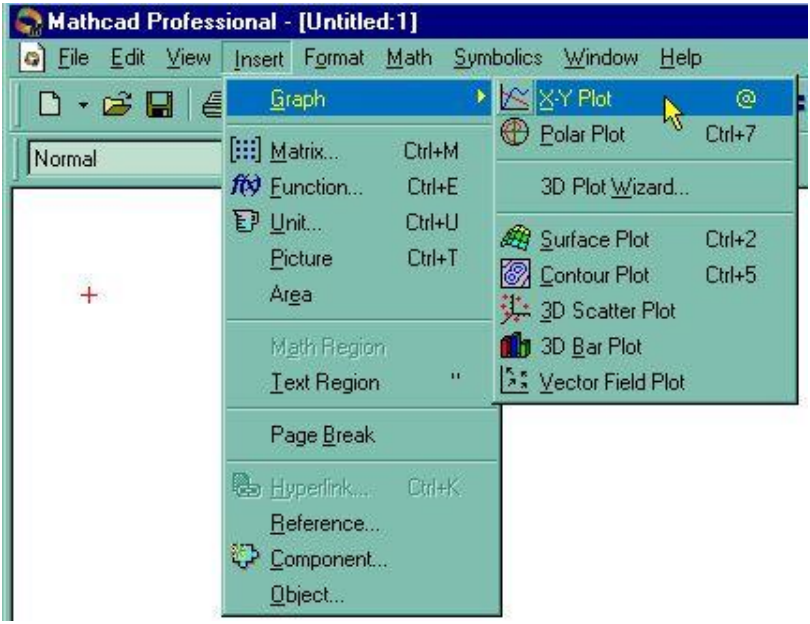


Рис. 91. Создание графика посредством меню.

Чтобы удалить график, щелкните в его пределах и выберите в верхнем меню **Edit** (Правка) пункт **Cut** (Вырезать) или **Delete** (Удалить).

Нарисовать график любой скалярной функции f(x) можно двумя способами. Первый заключается в дискретизации значений функции, присвоении этих значений вектору и прорисовке графика вектора. Второй, более простой способ, называемый *быстрым построением графика,* заключается во введении функции в один из местозаполнителей (например, у оси Y), а имени аргумента - в местозаполнитель у другой оси (рис. 92).

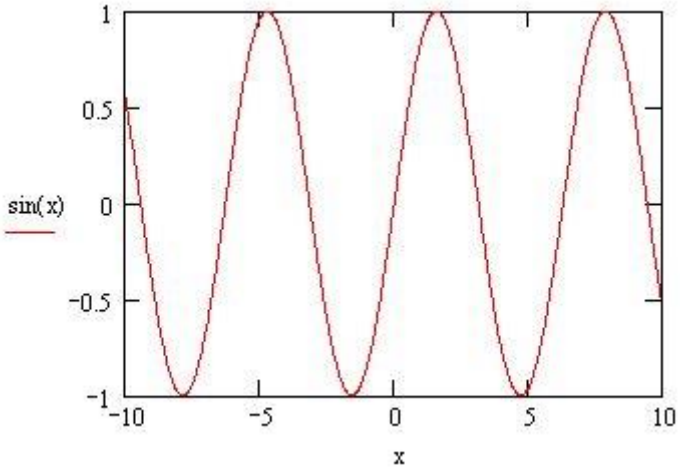


Рис. 92. Быстрое построение графика функции.

В результате MathCAD сам создает график функции в пределах значений аргумента, по умолчанию принятых равными от -10 до 10. Разумеется, впоследствии можно поменять диапазон значений аргумента, и график автоматически подстроится под него.

Полярный график. Для создания полярного графика необходимо нажать кнопку **Polar Plot** на панели **Graph** (Гра-

фик) (рис. 91) и вставить в местозаполнители имена переменных и функций, которые будут нарисованы в полярной системе координат: угол (нижний местозаполнитель) и радиус-вектор (левый местозаполнитель).

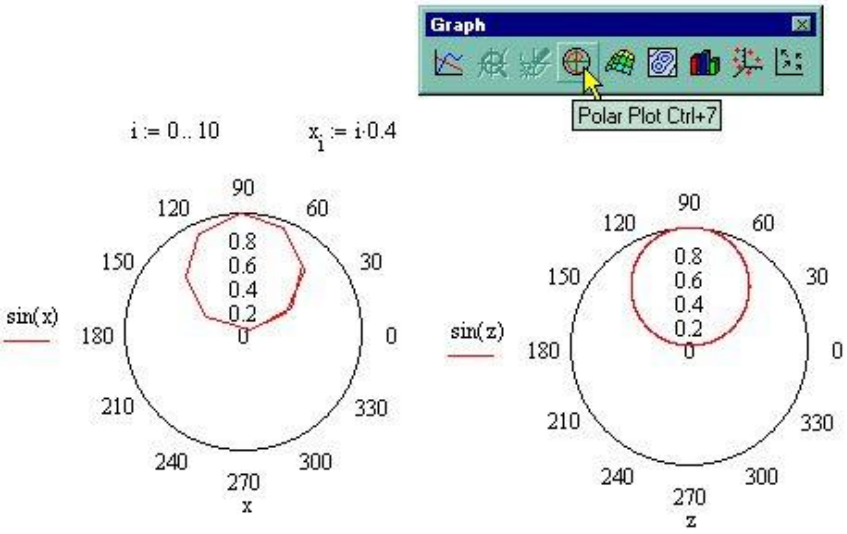


Рис. 93. Полярные графики.

Форматирование полярных графиков практически идентично форматированию декартовых, поэтому все, сказанное ниже об оформлении двумерных графиков на примере XY-графиков, в полной мере относится и к полярным.

Построение нескольких рядов данных. На одном графике может быть отложено до 16 различных зависимостей. Чтобы построить на графике еще одну кривую, необходимо выполнить следующие действия:

1. Поместите линии ввода таким образом, чтобы они целиком захватывали выражение, стоящее в надписи координатной оси y (рис. 94).
2. Нажмите клавишу <,>.

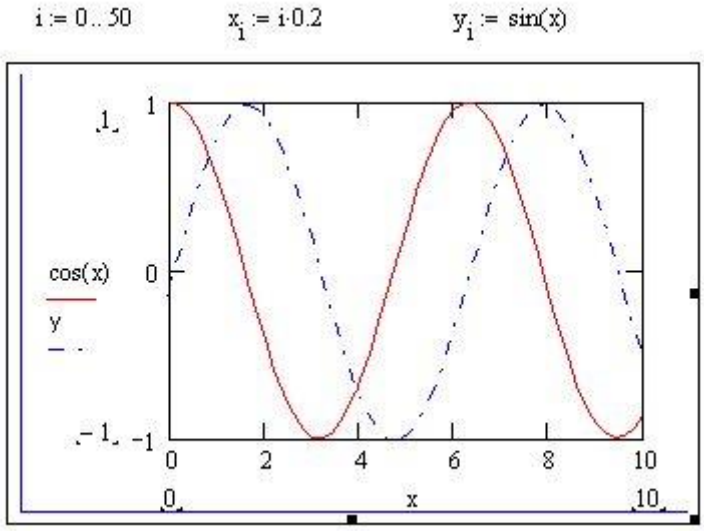


Рис. 94. Построение нескольких зависимостей на одном графике.

3. В результате появится местозаполнитель, в который нужно ввести выражение для второй кривой.
4. Щелкните в любом месте вне этого выражения (на графике или вне его).

После этого вторая кривая будет отображена на графике. На рис. 94 уже нарисованы два ряда данных, а нажатие клавиши с запятой <,> приведет к появлению третьего местозаполнителя, с помощью которого можно определить третий ряд данных. Чтобы убрать один или несколько рядов данных с графика, удалите клавишами<BackSpace> или <Del> соответствующие им надписи у координатных осей.

Изменение параметров графика. Форматирование осей. Возможности форматирования координатных осей графиков включают в себя управление их внешним видом, диапазоном, шкалой, нумерацией и отображением некоторых зна-

чений на осях при помощи маркеров.

Изменение диапазона осей. Когда график создается впервые, MathCAD выбирает представленный диапазон для обеих координатных осей автоматически. Чтобы изменить этот диапазон:

1. Перейдите к редактированию графика, щелкнув в его пределах мышью.
2. График будет выделен, а вблизи каждой из осей появятся два поля с числами, обозначающими границы диапазона.

Щелкните мышью в области одного из полей, чтобы редактировать соответствующую границу оси (например, верхнего предела оси x, как показано на рис. 95):

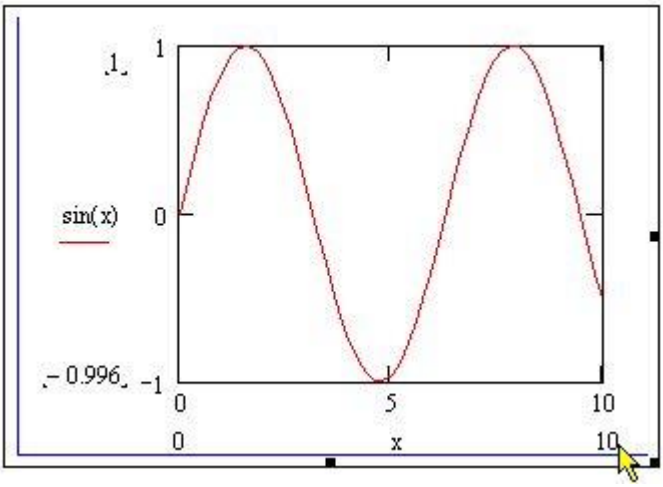


Рис. 95. Изменение диапазона оси X.

3. Пользуясь клавишами управления курсором и клавишами <BackSpace> и <Del>, удалите содержимое поля.
4. Введите новое значение диапазона (например, 20).
5. Щелкните за пределами поля, и график будет автоматически перерисован в новых пределах.

Изменения на оси У производятся точно так же.

Чтобы вернуть автоматический выбор какого-либо диапазона, удалите число из соответствующего поля и щелкните вне его. Граница шкалы будет выбрана MathCAD, исходя из значений данных, представляемых на графике.

Форматирование шкалы. Изменение внешнего вида шкалы, нанесенной на координатную ось, производится с помощью диалогового окна **Formatting Currently Selected X-Y Plot**(Форматирование выбранного графика), в котором следует перейти навкладку **X-YAxes** (Оси X-Y) (рис. 96).

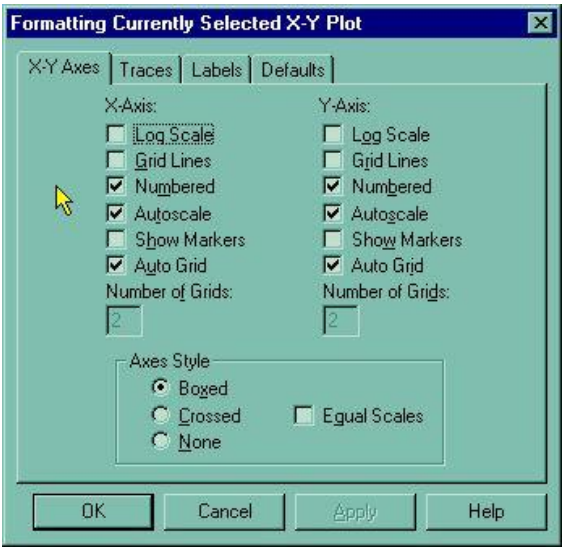


Рис. 96. Диалоговое окно Formatting Currently Selected XY Plot.

Вызвать диалог можно двойным щелчком мыши в области графика или выполнением команды **Format /Graph / X-Y Plot** (Формат / График / X-Y График), или выбором в контекстном меню команды **Format** (Формат).

С помощью флажков и переключателей легко поменять

внешний вид каждой из осей. Перечислим доступные опции и поясним их действие:

- **Log Scale** (Логарифмический масштаб) - график по данной оси будет нарисован в логарифмическом масштабе. Это полезно, если данные разнятся на несколько порядков;
- **Grid Lines** (Линии сетки) - показать линии сетки (пример на рис. 97);
- **Numbered** (Нумерация) - показать нумерацию шкалы. Если убрать этот флажок, то числа, размечающие шкалу, пропадут (пример см. ниже на рис. 97);

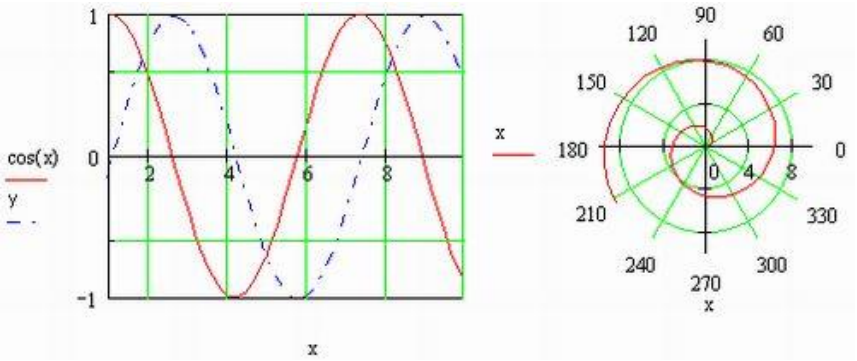


Рис. 97. Линии сетки на декартовом и полярном графиках, вид осей – **Crossed**.

- **Autoscale** (Автоматический масштаб) - выбор диапазона оси производится автоматически процессором MathCAD;
- **Show Markers** (Показать маркеры) - выделение значений на осях.
- **AutoGrid** (Автоматическая шкала) - разбиение шкалы производится автоматически процессором MathCAD. Если этот флажок снят, в поле ввода рядом с ним следует указать желаемое количество меток шкалы;
- **Equal Scales** (Одинаковый масштаб) - оси x и Y принудительно рисуютсяв одинаковом масштабе;

- Axes Style (Вид оси) - можно выбрать один из трех видов системы координат:

Boxed (Прямоугольник) - как показано на рис. 95;

Crossed (Пересечение) - координатные оси в виде двух пересекающихся прямых;

None (Нет) - координатные оси не показываются на графике.

Для полярного графика предусмотрены другие виды осей: **Perimeter** (Периметр), **Crossed** (Пересечение) и **None** (Нет). Первый тип оси показан выше (см. рис. 93), а второй вы видите на рис. 97.

Изменить описанные параметры можно и в диалоговом окне **Axis Format** (Формат оси), которое появляется, если щелкнуть дважды на самой оси (рис. 98).

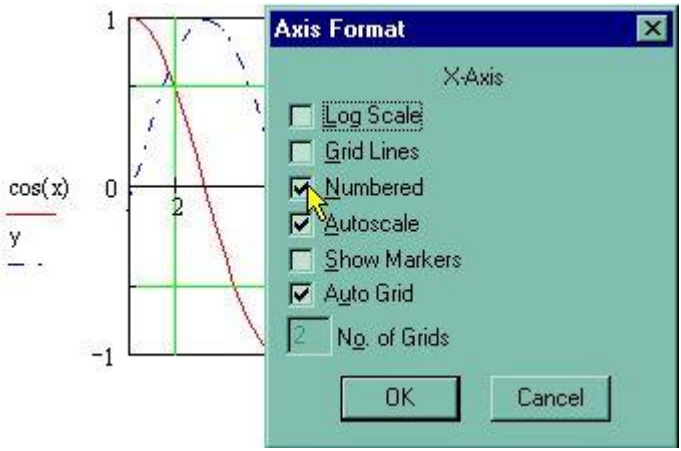


Рис. 98. Диалоговое окно Axis Format.

Форматирование рядов данных. С помощью вкладки Traces (Ряды данных) диалогового окна **Formatting Currently Selected X-Y Plot** (Форматирование выбранного графика) легко установить комбинацию параметров линии и точек для

каждого из рядов данных, представленных на графике. Пользователю требуется выделить в списке нужный ряд данных (его положение в списке соответствует положению метки зависимости у оси у) и изменить в списках в середине диалогового окна желаемые установки (рис. 99).

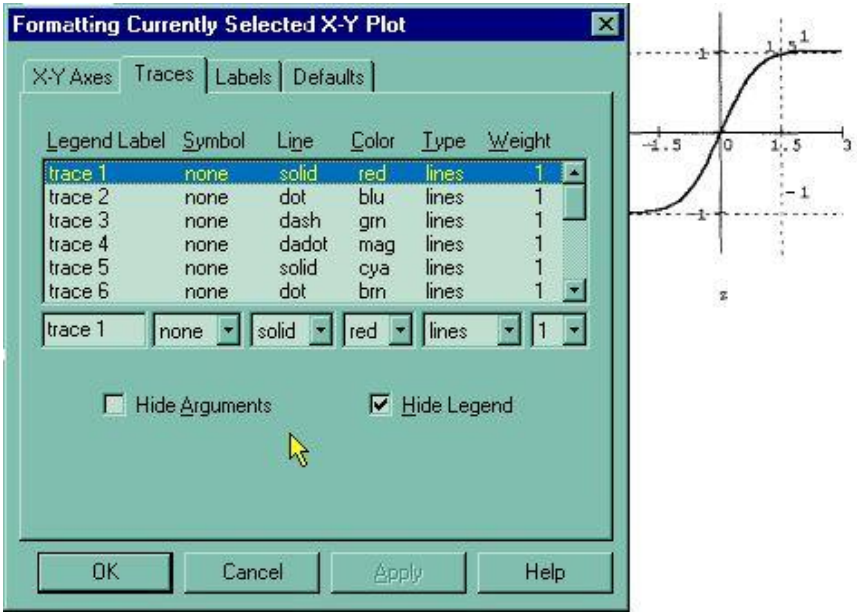


Рис. 99. Вкладка Traces диалога Formatting Currently Selected X-Y Plot.

На вкладке **Traces** (Ряды данных) регулируются следующие параметры:

- **Legend Label** (Легенда) - текст легенды, описывающий ряд данных;
- **Symbol** (Символ) - символ, которым обозначаются отдельные точки данных;
- **Line** (Линия) - Изменяя параметры линии (Стиль, толщина и цвет), можно добиться наилучшего восприятия разных зави-

симостей на одном графике. стиль линии:

solid (сплошная);
dot (пунктир);
dash (штрих);
dashdot (штрихпунктир).

- **Color** (Цвет) - цвет линии и точек данных;
- **Weight** (Толщина) - толщина линии и точек данных;
- **Type** (Тип) - тип представления ряда данных:

lines (линии);
points (точки);
error (ошибки);
bar (столбцы);
step (шаг);
draw (рисунок);
stem (стержень);
solid bar (гистограмма).

Для некоторых типов графиков те или иные параметры недоступны (например, нельзя задать символ для шаговой кривой).

Трассировка – цифровое измерение точек графика. *Трассировка* позволяет очень точно изучить строение графика.

Для того чтобы включить режим трассировки, щелкните в области графика правой кнопкой мыши и выберите в контекстном меню пункт **Trace** (Трассировка).

В результате появится окно трассировки (рис. 100), а в поле графика вы увидите две пересекающиеся пунктирные линии.

Перемещая указатель мыши по графику, вы тем самым передвигаете точку пересечения линий трассировки. При этом координаты точки указываются с высокой точностью в окне трассировки в полях **X-Value** (Значение X) и **Y-Value** (Значение Y). Нажатие кнопки **Copy X** (Копировать X) или **Copy Y** (Копировать Y) копирует соответствующее число в буфер обмена. В дальнейшем его можно вставить в любое место

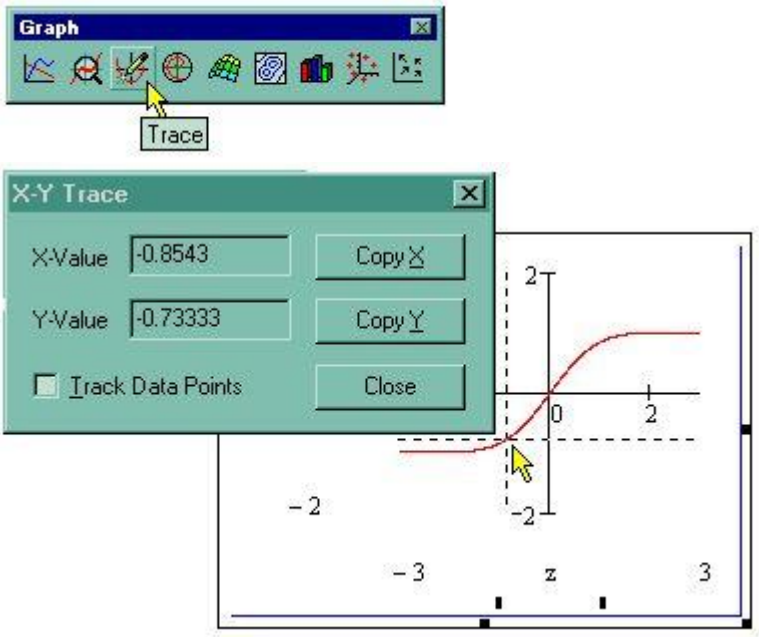


Рис. 100. Кнопка трассировки на панели **Graph** и трассировка графика.

документа или в маркер, нажав клавиши <Ctrl>+<V>. Если установлен флажок **Track Data Points** (Следовать за рядом данных), как это показано на рис. 100, то линии трассировки следуют точно вдоль графика. Если нет, то они могут перемещаться по всей области графика.

Трехмерные графики. Коллекция трехмерных графиков - настоящее чудо, которое MathCAD дарит пользователю. За считанные секунды вы можете создать великолепную презентацию результатов своих расчетов.

Создание трехмерных графиков. Чтобы создать трехмерный график, требуется нажать кнопку с изображением любого из типов трехмерных графиков на панели инструментов **Graph** (График). В результате появится пустая область графи-

ка с тремя осями (рис. 101) и единственным местозаполнителем в нижнем левом углу.

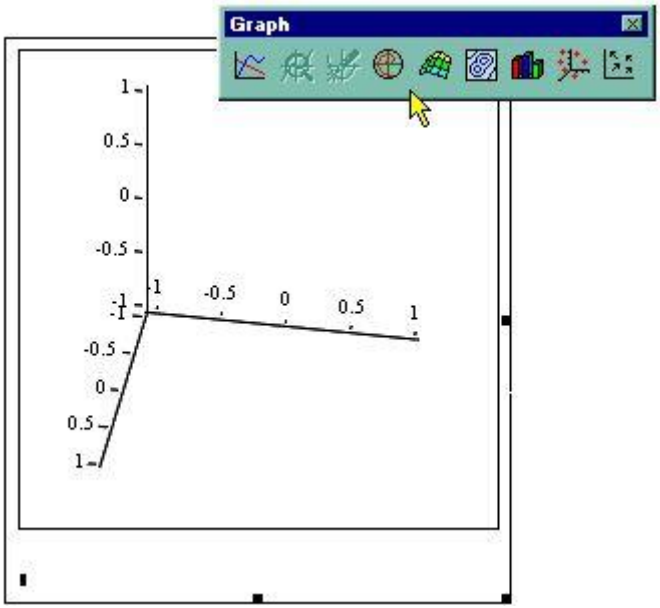


Рис. 101. Создание трехмерного графика.

В этот местозаполнитель следует ввести либо имя z функции z(x,y) двух переменных для *быстрого построения трехмерного графика,* либо имя матричной переменной Z, которая задаст распределение данных $z_{x,y}$ на плоскости XY. Рассмотрим на простом примере функции z(x, y) и матрицы Z, (они заданы на рис. 102).

Еще раз отметим, что для получения графиков не требуется никакого текста, кроме введения имени соответствующей функции или матрицы в местозаполнитель.

Для графиков, задаваемых матрицами, шкалу плоскости XY приходится задавать вручную. MathCAD просто рисует

$$z(x, y) := x^2 + y^2$$

$$Z := \begin{pmatrix} 1 & 1 & 0 & 1.1 & 1.2 \\ 1 & 2 & 3 & 2.1 & 1.5 \\ 1.3 & 3.3 & 5 & 3.7 & 2 \\ 1.3 & 3 & 5.7 & 4.1 & 2.9 \\ 1.5 & 2 & 6.5 & 4.8 & 4 \end{pmatrix}$$

Рис. 102. Функция и матрица для отображения на трехмерном графике.

поверхность, точки в пространстве или линии уровня, основываясь на двумерной структуре этой матрицы. При быстром же построении графиков имеется возможность строить их в различном диапазоне аргументов, подобно двумерным графикам.

При другом способе построения трехмерных графиков используется специальная функция **CreateMesh(F)**, где F – ранее заданная функция двух переменных (пример применения смотри ниже на рис. 106)

Примеры построения трехмерных графиков различных типов, создаваемых нажатием той или иной кнопки на панели **Graph** (График) (рис 103 и 104).

График векторного поля несколько отличается от остальных типов двумерных графиков (рис. 105). Его смысл заключается в построении некоторого вектора в каждой точке плоскости XY. Чтобы задать вектор на плоскости, требуются два скалярных числа. Поэтому в MathCAD принято, что векторное поле задает комплексная матрица. Действительные части каждого ее элемента задают проекцию вектора на ось x, а мнимые - на ось y.

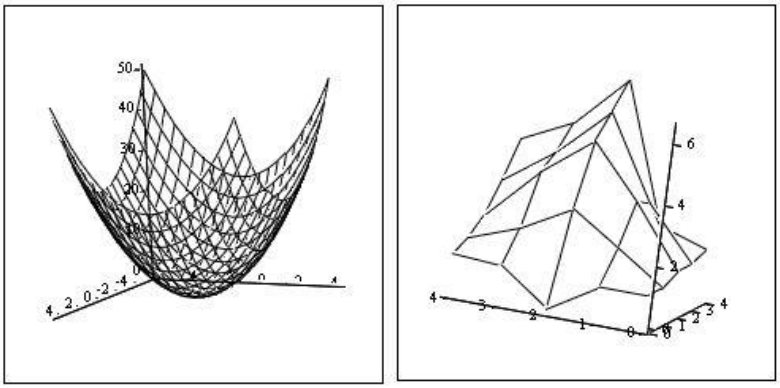


Рис. 103. Быстрое построение графика поверхности **Surface Plot** функции (слева) и график поверхности, заданный матрицей (справа).

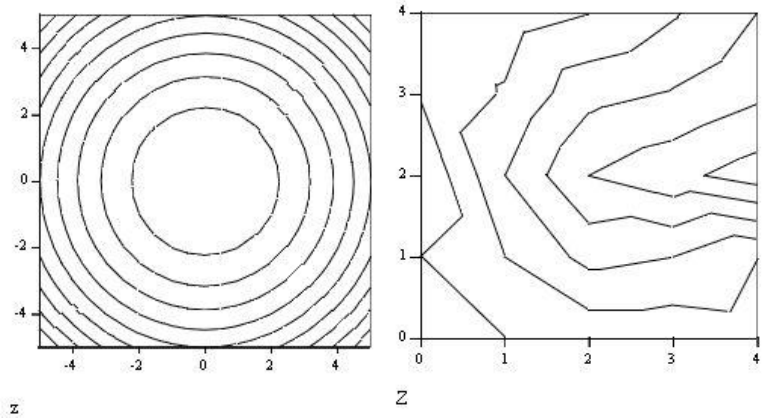


Рис. 104. Быстрое построение графика линий уровня **Contour Plot** функции (слева) и заданный матрицей (справа).

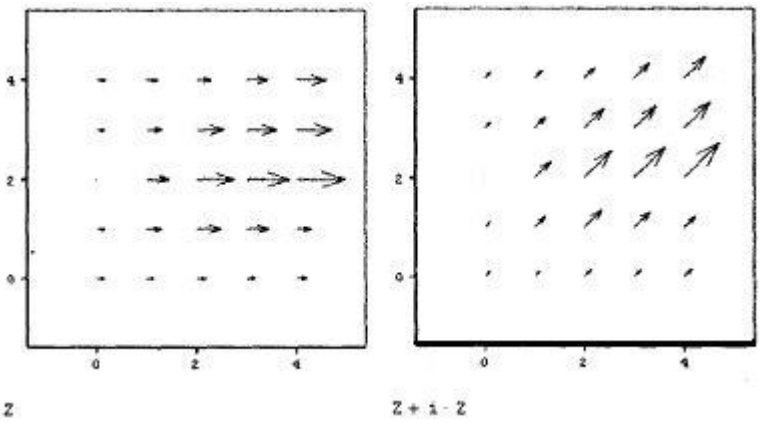


Рис. 105. Графики векторного поля, заданные матрицами.

Форматирование трехмерных графиков. Форматирование трехмерных графиков выполняется с помощью диалогового окна **3-D Plot Format** (Форматирование 3-D графика), которое вызывается двойным щелчком мыши в области графика (рис. 106).

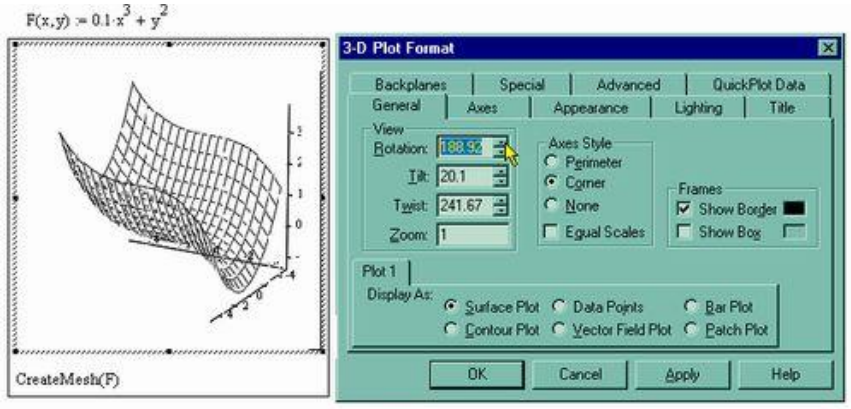


Рис. 106. Диалоговое окно **3-D Plot Format**

Изменение типа графика.

Параметры трехмерных графиков всех типов устанавливаются посредством этого диалогового окна.
В диалоге **3-D Plot Format** (Форматирование 3-D графика) доступно большое количество параметров, изменение которых способно очень сильно повлиять на внешний вид графика. Они сгруппированы по принципу действия на нескольких вкладках. Остановимся коротко на возможностях оформления трехмерных графиков, поясняя их, главным образом, примерами.

Чтобы поменять тип уже имеющегося графика (например, построить вместоповерхности график линий уровня и т. д.), просто установите соответствующий переключатель в нижней части вкладки **General** (Общие) и нажмитекнопку ОК. График будет перерисован.

Вращение графика. Самый простой способ ориентации системы координат с графиком в трехмерном пространстве - это перетаскивание ее указателем мыши. Попробуйте перемещать при нажатой левой кнопке мыши указатель в пределах графика, и вы увидите, как поворачивается график.

Разумеется, поворачивать можно лишь графики в трехмерном пространстве; векторное поле и линии уровня строятся, по определению, на прямоугольном участке плоскости.

Другой способ изменения ориентации графика - с помощью полей **Rotation** (Вращение), **Tilt** (Наклон) и **Twist** (Поворот) на вкладке **General** (Общие) (см. рис. 96), которые в совокупности определяют соответствующие углы (в градусах) и тем самым задают направление всех трех осей координат в пространстве. Сравнивая рис. 106 и рис. 107, вы разберетесь, как эти углы влияют на ориентацию графика.

Стиль осей. С помощью группы переключателей **Axes Style** (Стиль осей) можно задать один из следующих стилей осей координат:
Corner (Углом) – как на рис. 107;
- **Perimeter** (Периметр) – на рис. 108;

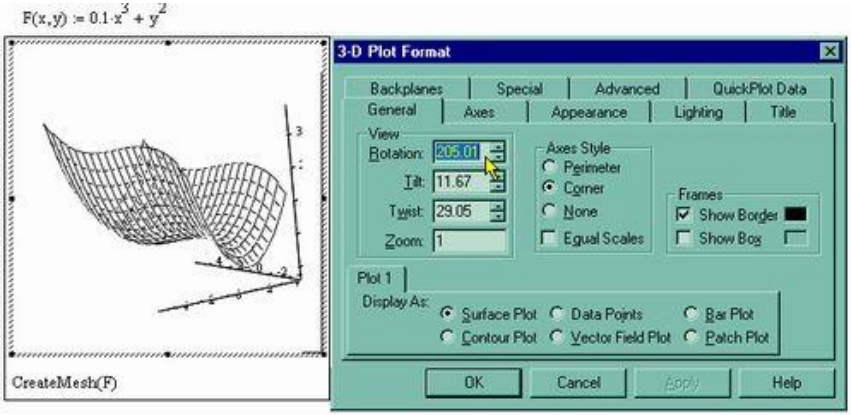


Рис. 107**.** Изменение параметра **Rotation** (сравните с рис. 104)

- **None** (Нет) - оси отсутствуют.

Если установить флажок **Show Box** (Показать куб), то координатное пространство будет изображено в виде куба (рис. 108).

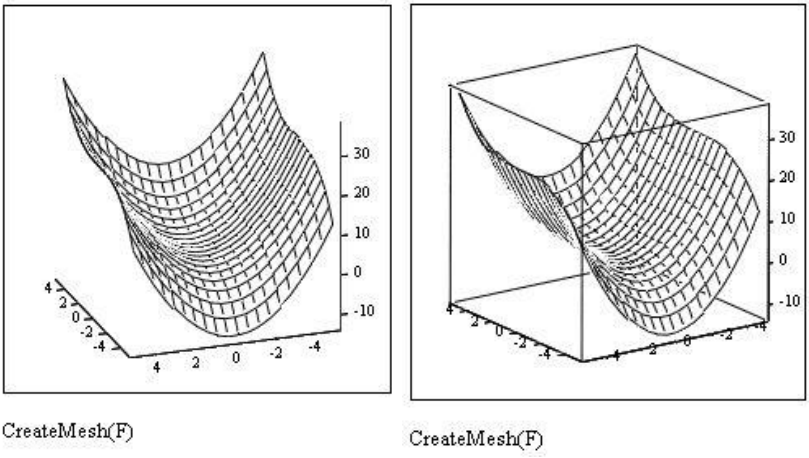


Рис. 108. Расположение координатных осей по периметру (слева) и установлен флажок **Show Box** (справа).

Форматирование осей. Вкладка **Axes** (Оси) содержит три вложенных вкладки, в которых задаются параметры для каждой из трех координатных осей. В частности, можно включить или отключить показ линий сетки, нумерации и задать диапазон по каждой из осей (рис. 109).

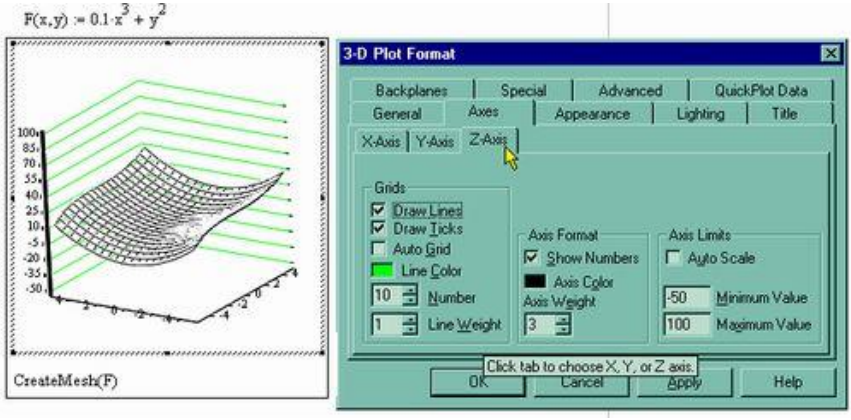


Рис. 109. Форматирование осей координат.

Смысл этих операций сходен с аналогичными операциями для двумерных графиков. При помощи еще одной вкладки - **Backplanes** (Плоскости заднего плана) задается показ проекций координатной сетки на три скрытые плоскости трехмерного графика (пример форматирования плоскости XY показан на рис. 108).

Стиль заливки и линий. На рис. 110 показано влияние различного стиля задания заливки и линий с помощью вкладки **Appearance** (Появление) для контурного и поверхностного графиков. При выборе переключателя **Fill Surface** (Заливка поверхности) из группы **Fill Options** (Опции заливки) вы получаете доступ к опциям цвета (в группе **Color Options).** Если выбрать переключатель **Solid Color** (Один цвет), то получится однотонная заливка поверхности.

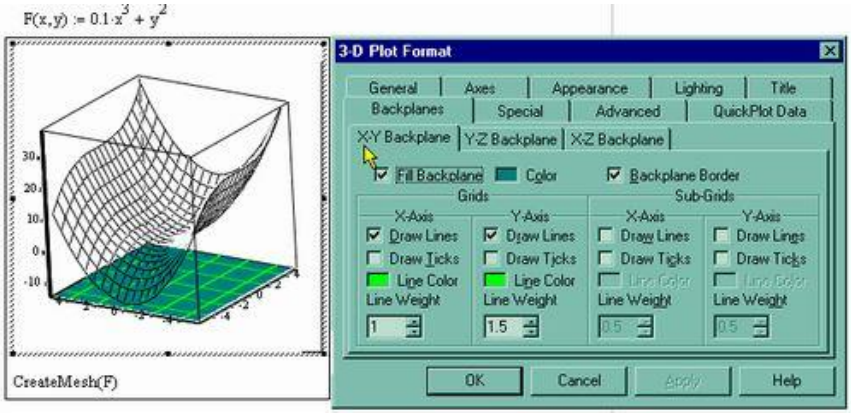


Рис. 110. Форматирование скрытых плоскостей графика.

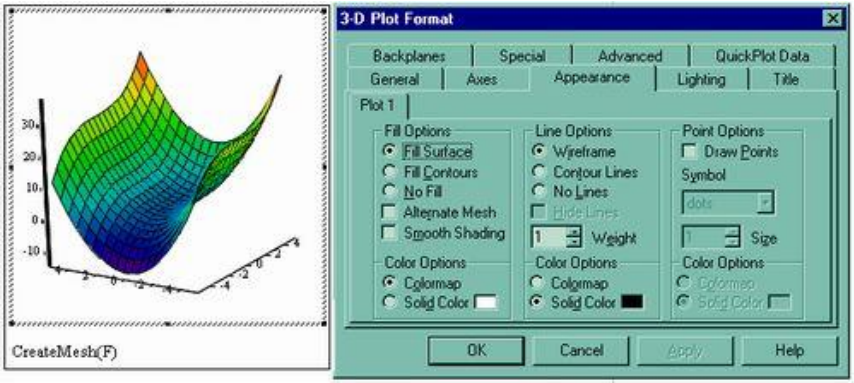


Рис. 111. Заливка графика поверхности.

Если установить переключатель **Color-map** (Цветовая схема), то поверхность или контурный график будут залиты разными цветами и оттенками (рис. 111), причем выбрать цветовую схему можно на вкладке **Advanced** (Дополнительно) (рис. 112).

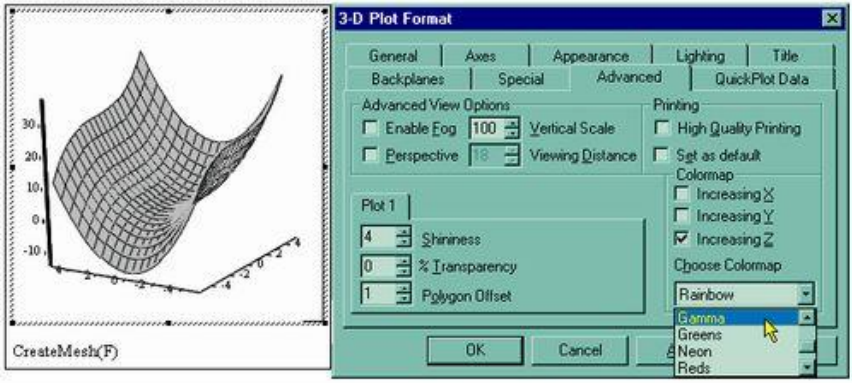


Рис. 112. Выбор цветовой схемы.

Поэкспериментируйте с разными цветовыми схемами и представлениями заливки и линий, задаваемых полем **Line Options** (Опции линии) (рис. 111), чтобы представлять себе богатство возможностей MathCAD. Некоторые параметры, влияющие на представление контуров графиков, доступны на вкладке **Special** (Специальные) (рис. 113).

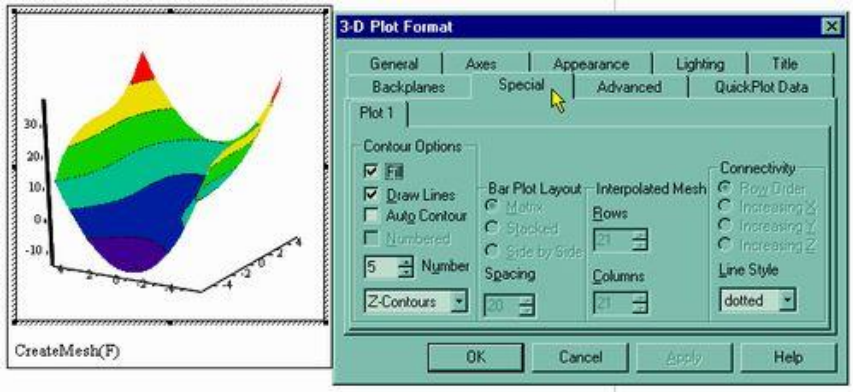


Рис. 113. Вкладка Special.

Сочетаний различных цветовых схем, заливок и других параметров настолько много, что лучше предоставить читателю самому попробовать применить их различные комбинации и выбрать из них наиболее понравившиеся.

Спецэффекты. В той же вкладке **Advanced** (Дополнительно) (рис. 112) имеется доступ к управлению несколькими специальными эффектами оформления графиков, благодаря которым они смотрятся более красиво.

Перечислим эти эффекты:

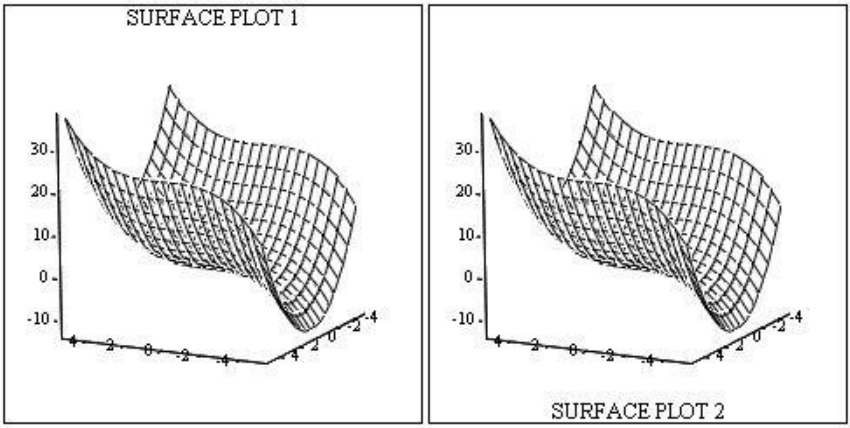
- **Shininess** (Сияние) - имеется возможность регулировать сияние в пределах от 0 до 128;
- **Fog** (Туман) - эффект тумана;
- **Transparency** (Прозрачность) - задается процент прозрачности графика;
- **Perspective** (Перспектива) - показ перспективы с определением видимости расстояния.

Еще один спецэффект подсветки графика задается на вкладке **Lighting** (Подсветка), причем имеются как встроенные схемы подсветки так и возможность задавать ее цвет и направление самому пользователю.

Заголовок графика. Заголовок графика определяется на вкладке **Title** (Заголовок) и может быть расположен как сверху, так и снизу графика (рис. 114).

2.8. Основы программирования. Панель программирования. Условное ветвление, цикл. Примеры программирования различных задач.

MathCAD - это система, ориентированная на пользователя, который не обязан знать абсолютно ничего о программировании. Создатели MathCAD изначально поставили перед собой такую задачу, чтобы дать возможность профессионалам-математикам, физикам и инженерам самостоятельно проводить сложные расчеты, не обращаясь за помощью к програм-



F F

Рис. 114. Графики с заголовками.

мистам.

Несмотря на блестящее воплощение этих замыслов, выяснилось, что вовсе без программирования MathCAD серьезно теряет в своей силе, в основном, из-за недовольства пользователей, знакомых с техникой создания программ и желающих осуществить свои расчеты в привычном для себя программистском стиле.

Вместо знакомых принципов программирования, пользователям старых версий MathCAD предлагалось комбинировать несколько специфичных встроенных функций и ранжированные переменные.

Последние версии MathCAD имеют не очень мощный, но весьма элегантный собственный язык. С одной стороны, он дает возможность программисту эффективно применять программный код в документах MathCAD. С другой, простота и интуитивность языка программирования позволяет быстро ему обучиться. Наконец, программные модули внутри документа MathCAD сочетают в себе и обособленность (поэтому

их легко отличить от остальных формул), и простоту смыслового восприятия.

Несмотря на небольшое число операторов, язык программирования MathCAD позволяет решать самые различные, в том числе и довольно сложные, задачи и является серьезным подспорьем для расчетов.

Программирование без программирования. В ранних версиях MathCAD встроенного языка программирования не было. Чтобы применять привычные операции проверки условий и организовывать циклы, приходилось изобретать причудливую смесь из встроенных функций условия if и until и комбинаций ранжированных переменных (рис. 115).

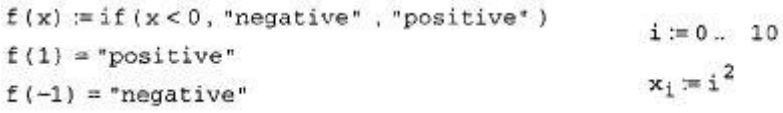


Рис. 115. Функция условия (слева) и организация цикла при помощи ранжированной переменной (справа).

Фактически, использование ранжированных переменных - мощный аппарат MathCAD, похожий на применение циклов в программировании. В подавляющем большинстве случаев намного удобнее организовать циклы (в том числе вложенные) с помощью ранжированных переменных, чем заниматься для этого программированием. Полезнее освоить технику, связанную с ранжированными переменными, векторами и матрицами, поскольку на ней основаны главные принципы расчетов в MathCAD, в частности подготовка графиков.

Язык программирования MathCAD. Для вставки программного кода в документы в MathCAD имеется специальная панель инструментов **Programming** (Программирование), которую можно вызвать на экран нажатием кнопки **Programming Toolbar** на панели Math (Математика), как по-

казано на рис. 116. большинство кнопок этой панели выполнено в виде текстового представления операторов программирования, поэтому их смысл легко понятен.

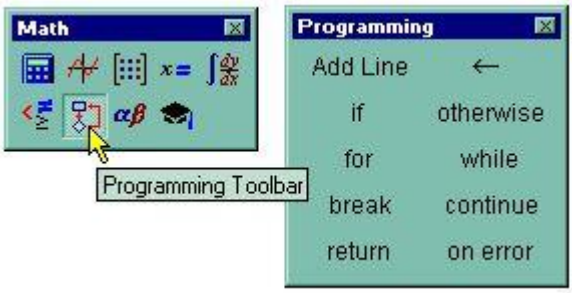


Рис. 116. Панель инструментов **Programming.**

Изложим последовательно основные составные части языка программирования MathCAD и рассмотрим примеры его использования.

Основными инструментами работы в MathCAD являются математические выражения, переменные и функции. Нередко записать формулу, использующую ту или иную внутреннюю логику (например, возвращение различных значений в зависимости от условий), в одну строку не удается. Назначение программных модулей как раз и заключается в определении выражений, переменных и функций в несколько строк, часто с применением специфических программных операторов. Сравните определение функции f(x) на рис. 115 с определением f(x) с помощью программного модуля (рис. 117).

Создание программы (Add Line). Чтобы создать программный модуль, например, представленную на рис. 117, проделайте следующее:

1. Введите часть выражения, которая будет находиться слева от знака присваивания и сам знак присваивания. В нашем примере это имя функции f(x).

```
f(x) := | "negative"   if x < 0
        | "positive"   if x > 0
        | "zero"   otherwise
f(1) = "positive"
f(-1) = "negative"
f(0) = "zero"
```

Рис. 117. Функция условия, определенная с помощью программы.

2. При необходимости вызовите на экран панель инструментов **Programming** (Программирование) (см. рис. 118).
3. Нажмите на этой панели кнопку **Add Line** (Добавить линию).
4. Если приблизительно известно, сколько строк кода будет содержать программа, можно создать нужное количество линий повторным нажатием кнопки **Add Line** (Добавить линию) соответствующее число раз (на рис. 118 показан результат трехкратного нажатия).

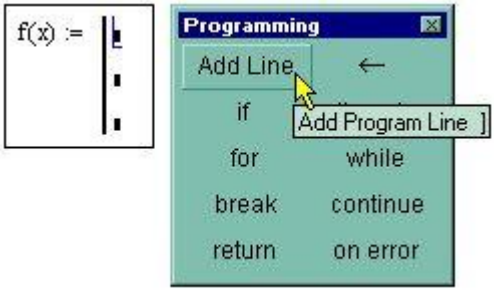


Рис. 118. Начало создания программного модуля.

5. В появившиеся местозаполнители введите желаемый программный код, используя программные операторы. В рас-

сматриваемом примере в каждый местозаполнитель вводится строка, например, "positive" (рис. 119), затем нажимается кнопка **If** (Если) на панели **Programming** (Программирование) и в возникший местозаполнитель вводится выражение x>0 (рис. 120).

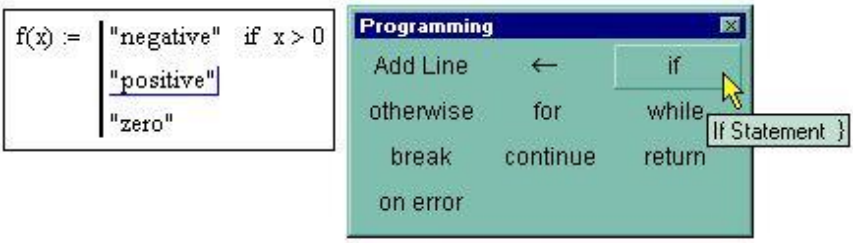


Рис. 119. Вставка программного оператора.

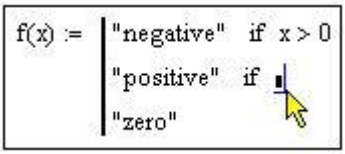


Рис. 120. Вставка условия в программу.

После того как программный модуль полностью определен, и ни один местозаполнитель не остался пустым, функция может использоваться обычным образом, как в численных, так и в символьных расчетах.

Не вводите с клавиатуры имена программных операторов. Для их вставки можно применять лишь сочетания клавиш, которые приведены в тексте всплывающей подсказки (рис. 119).

Вставить строку программного кода в уже созданную программу можно в любой момент с помощью той же самой кнопки **Add Line** (Добавить линию). Для этого следует пред-

варительно поместить на нужное место внутри программного модуля линии ввода.

В режиме выполнения программы, а это происходит при любой попытке вычислить f(x), выполняется последовательно каждая строка кода. Таким образом, основной принцип создания программных модулей заключается в правильном расположении строк кода. Ориентироваться в их действии довольно легко, т. к. фрагменты кода одного уровня сгруппированы в программе с помощью вертикальных черт.

Локальное присваивание (←). Язык программирования MathCAD не был бы эффективным, если бы не позволял создавать внутри программных модулей локальные переменные, которые "не видны" извне, из других частей документа. Присваивание в пределах программ, в отличие от документов MathCAD, производится с помощью оператора **Local Definition** (Локальное присваивание), который вставляется нажатием кнопки с изображением стрелки ← на панели **Programming** (Программирование).

Ни оператор присваивания :=, ни оператор вывода = в пределах программ **не применяются**.

Локальное присваивание иллюстрируется на рис. 121

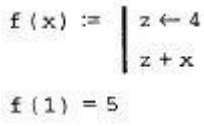


Рис. 121. Локальное присваивание в программе.

Переменная z существует только внутри программы, выделенной вертикальной чертой. Из других мест документа получить ее значение невозможно.

Условные операторы *(if, otherwise).* Действие условного оператора if состоит из двух частей. Сначала проверяется логическое выражение (условие) справа от него. Если оно ис-

тинно, выполняется выражение слева от оператора if. Если ложно - ничего не происходит, а выполнение программы продолжается переходом к ее следующей строке. Вставить условный оператор в программу можно следующим образом (рис. 122):

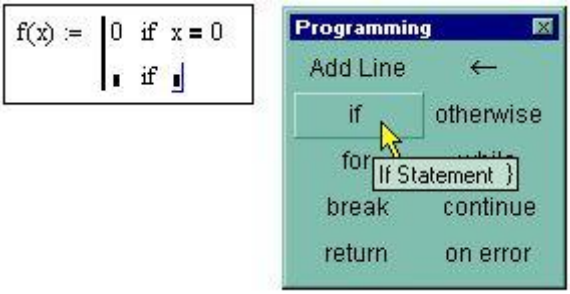


Рис. 122. Вставка условного оператора.

1. Если необходимо, введите левую часть выражения и оператор присваивания.
2. Создайте новую строку программного кода, нажав на панели **Programming** (Программирование) кнопку **Add Line** (Добавить строку).
3. Нажмите кнопку условного оператора if.
4. Справа от оператора if введите условие. Пользуйтесь логическими операторами, вводя их с панели **Boolean** (Булевы операторы).
5. Выражение, которое должно выполняться, если условие оказывается выполненным, введите слева от оператора if.
6. Если в программе предусматриваются дополнительные условия, добавьте в программу еще одну строку нажатием кнопки **Add Line** и введите их таким же образом, используя оператор if или otherwise.

Оператор otherwise используется совместно с одним или несколькими условными операторами *if* и указывает на выра-

жение, которое будет выполняться, если ни одно из условий не оказалось истинным. Примеры использования операторов if и otherwise приведены на рис. 117.

Операторы цикла *(for, while, break, continue).* В языке программирования MathCAD имеются два оператора цикла: for и while. Первый из них дает возможность организовать цикл по некоторой переменной, заставляя ее пробегать некоторый диапазон значений. Второй создает цикл с выходом из него по некоторому логическому условию. Чтобы вставить в программный модуль оператор цикла:

1. Создайте в программном модуле новую линию.
2. Вставьте один из операторов цикла for или while нажатием одноименной кнопки на панели **Programming** (Программирование).
3. Если выбран оператор for (рис. 123), то вставьте в соответствующие местозаполнители имя переменной и диапазон ее значений (рис. 124), а если while - то логическое выражение, при нарушении которого должен осуществляться выход из цикла (рис. 125).

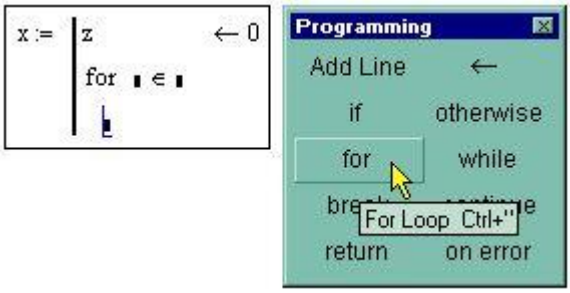


Рис. 123. Вставка оператора цикла.

Рис. 124. Оператор цикла for с ранжированной переменной (слева) и оператор цикла for с вектором (справа).

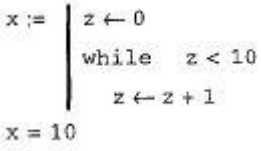


Рис. 125. Оператор цикла while.

4. В нижний местозаполнитель введите тело цикла, т. е. выражения, которые должны выполняться циклически.
5. При необходимости дополните программу другими строками и введите в них нужный код.

Иногда необходимо досрочно завершить цикл, т. е. не по условию в его заголовке, а в некоторой строке в теле цикла. Для этого предназначен оператор break. Модификации рис. 124 (слева) и рис. 125 с прерыванием цикла оператором break приведены на рис. 126.

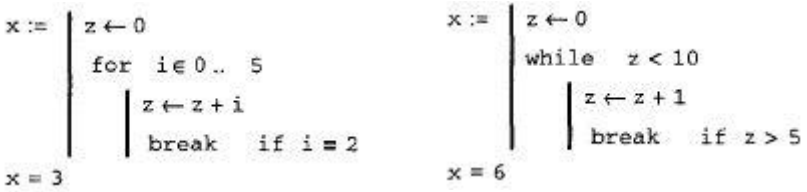


Рис. 126. Оператор break внутри цикла for (слева) и оператор break внутри цикла while (справа).

Например, в программе на рис. 126 (слева), как только значение переменной цикла i достигает 2, цикл, благодаря оператору break в последней строке программного модуля, прерывается. Соответственно, значение переменной x остается равным 0+1+2=3.

Чтобы четче обозначить границы завершения тела цикла, в его конце может использоваться дополнительная строка с оператором continue, который вводится одноименной кнопкой панели **Programming.** Примеры, модернизирующие примеры на рис. 124 (справа) и рис. 125, иллюстрируются на рис. 127, слева и справа, соответственно. Как видно, на результат программы наличие оператора continue не влияет.

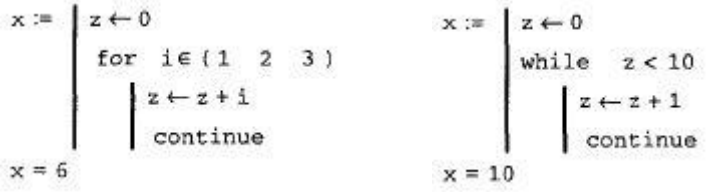


Рис. 127. Оператор continue в конце цикла for (слева) и оператор continue в конце цикла while (справа).

Возврат значения *(return).* Если для определения переменной или функции применяется программный модуль, то его строки исполняются последовательно при вычислении в документе этой переменной или функции. Соответственно, по мере выполнения программы рассчитываемый результат претерпевает изменения. В качестве окончательного результата выдается последнее присвоенное значение (смотри примеры на рис. 117, 121, 124-127). Чтобы подчеркнуть возврат программным модулем определенного значения, можно взять за правило делать это в последней строке программного модуля (рис. 128).

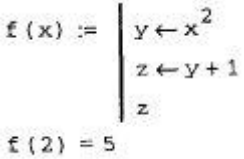


Рис. 128. Возврат значения обозначен явно в последней строке программы.

Вместе с тем, можно прервать выполнение программы в любой ее точке (например, с помощью условного оператора) и выдать некоторое значение, применив оператор return. В этом случае при выполнении указанного условия (рис. 129) значение, введенное в местозаполнитель после return, возвращается в качестве результата, а никакой другой код больше не выполняется. Вставляется в программу оператор return с помощью одноименной кнопки панели **Programming** (Программирование).

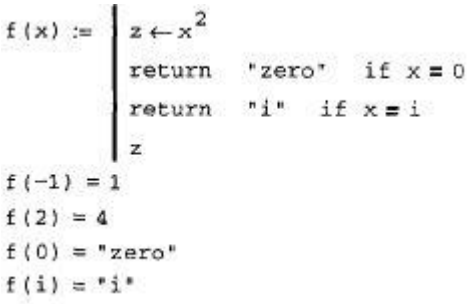


Рис. 129. Применение оператора return.

Примеры программирования. Рассмотрим два простых примера использования программных модулей в MathCAD

для численных (рис. 130) и символьных (рис. 131) расчетов.

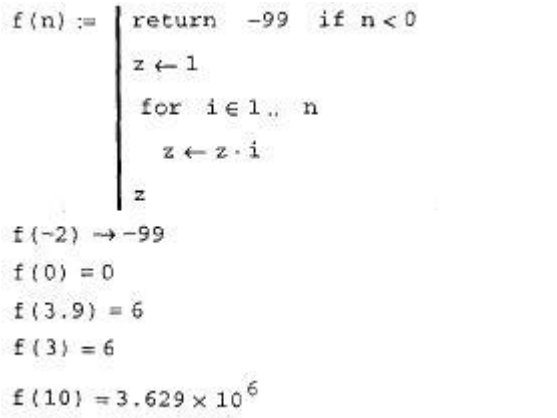


Рис. 130. Программирование в численных расчетах.

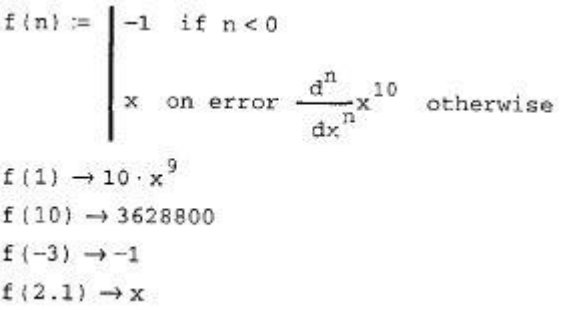


Рис. 131. Программирование в символьных расчетах.

В двух приведенных примерах используется большинство операторов, рассмотренных выше. Когда вы станете сами разрабатывать свои программные модули в MathCAD, не забывайте, что операторы программирования вставляются в текст программы с помощью кнопок панели инструментов

Programming (Программирование). Их имена нельзя ни в коем случае просто набирать на клавиатуре, поскольку они не будут восприняты MathCAD корректно. /1/

Рассмотрим несколько более сложный пример программирования численного решения уравнения диффузии примеси в полупроводнике в режиме “загонка” и “разгонка”.

Твердотельная диффузия – это физический процесс, вызывающий перераспределение примеси внутри полупроводникового кристалла в ходе высокотемпературной термообработки. Вместе с ионной имплантацией и эпитаксией диффузия является одним из основных методов, управляющих типом, уровнем концентрации и распределением примеси внутри определенных участков полупроводниковой подложки.

Для того, чтобы сформировать диффузионный слой, атомы примеси вводятся в приповерхностную область либо путем предварительного этапа их осаждения (этап загонки), либо путем ионной имплантации. На ранних этапах развития диффузионной технологии загонку проводили из паровой фазы или нанесенного на подложку слоя. В обоих случаях количество диффузанта вне подложки значительно превышало количество вошедшее в подложку, ограниченное предельной растворимостью атомов примеси при данной температуре. При этом можно было считать, что диффузант поступает из неограниченного источника. В технологии формирования СБИС предпочтение отдают методу ионной имплантации, поскольку он позволяет более точно управлять количеством вводимой примеси и обладает большими возможностями по расположению этой примеси по глубине за счет соответствующего выбора энергии имплантированных ионов.

Для того, чтобы активировать имплантированные ионы и удалить введенные в процессе ионной имплантации радиационные повреждения, как правило, необходим этап высокотемпературного отжига (этап разгонки). Так как количество атомов примеси, введенное при загонке, ограничено, данный

процесс описывается законами диффузии из ограниченного источника.

В 1855 году Фик предложил теорию диффузии. В основу этой теории положена аналогия между процессами переноса в жидких растворах и тепла за счет теплопроводности:

$$J = -D \cdot \text{grad}\, C(x, t) \text{ (Первый закон Фика)}$$

$$\partial C/\partial t = D \cdot \nabla^2 C(x, t) \text{ (Второй закон Фика)}$$

Первый закон связывает плотность потока атомов J с концентрацией диффузанта C, которая, как предполагается, зависит только от координаты x, отсчитываемой от поверхности образца, и времени t. Второй закон дает зависимость концентрации C в любой точке образца x с прошедшим временем t. Коэффициент пропорциональности D [см^2/с] – называется коэффициентом диффузии.

Задачей моделирования диффузии является решение второго уравнения Фика, то есть получение зависимости распределения примеси по толщине образца. Для этого необходимо решать дифференциальное уравнение при следующих граничных и начальных условиях. В случае загонки:

Начальные условия: при t=0 C(x,0)=0;

Граничные условия: $C(0,t)=C_S$ и $C(\infty,t)=0$.

В случае разгонки:

Начальные условия: при t=0 C(x,0)= C(x);

Граничные условия: $\int_0^x C(x,t)=S$ и $C(x,\infty)=0$.

Здесь: C_S – концентрация примеси на поверхности, см$^{-3}$, C(x) – начальный профиль, полученный при загонке, S – общее количество введенной в ходе загонки примеси. /2/

Несмотря на то, что существуют явные решения второго уравнения Фика (при некоторых допущениях), нашей целью

будет провести численное решение с использованием программных модулей в MathCAD.

Решение данного дифференциального уравнения будет проводиться с использованием так называемых разностных схем представления производной. В данном случае будет использована так называемая трехточечная схема представления для аппроксимации второй производной, и двухточечная для первой производной. Существуют и схемы с большим количеством используемых узлов сетки, позволяющие достичь большей точности, их применение в принципе будет аналогично рассмотренному. Таким образом, с использованием разностных схем записи второй закон Фика будет иметь следующий вид:

$$(C_i^{t+1} - C_i^t)/h^t = D_i^t \cdot (C_{i+1}^t - 2 \cdot C_i^t + C_{i-1}^t)/(h_x)^2$$

где: t – номер шага по временной сетке;
i – номер шага по пространственной сетке;
h^t – шаг по временной сетке;
h_x – шаг по пространственной сетке.

или в явном виде:

$$C_i^{t+1} = C_i^t + (D_i^t \cdot h^t/(h_x)^2) \cdot (C_{i+1}^t - 2 \cdot C_i^t + C_{i-1}^t)$$

Таким образом, программа для моделирования диффузии разностными методами будет выглядеть следующим образом:

ЗАГОНКА

$\tau := 0.0000005$ - шаг по временной сетке

$M1 := 20$ $\Delta 1 := \frac{1}{M1}$ - Коэффициент диффузии (в данном случае const, но может быть функцией от концентрации - переменная u)

$D1(u) := 1$

$$\sqrt{2 \cdot D1(1) \cdot \tau} = 0.01$$

- **Условие устойчивости метода**, это значение должно быть меньше чем Δl

$$Init1(x) := \begin{cases} 10 & \text{if } x < 0.001 \\ 0 & \text{otherwise} \end{cases}$$

- Функция, задающая начальные условия в виде отрезка функции

$$m := 0..M1$$

- формирование пространственной сетки

$$u_m := Init1(m \cdot \Delta 1)$$

- Перевод функции - начального условия в сеточный вид

$$F1(v) := \begin{cases} v1_0 \leftarrow 10 \\ v1_{M1} \leftarrow 0 \\ \text{for } m \in 1..M1 - 1 \\ \quad v1_m \leftarrow v_m + v_{m-1} \cdot \dfrac{D1(v_{m-1}) \cdot \tau}{\Delta 1^2} - v_m \cdot \dfrac{2 \cdot D1(v_m) \cdot \tau}{\Delta 1^2} + v_{m+1} \cdot \dfrac{D1(v_{m-1}) \cdot \tau}{\Delta 1^2} \\ v1 \end{cases}$$

- Граничные условия

▼ Реализация разностной схемы

$$V1(t) := \begin{cases} u & \text{if } t = 0 \\ F1(V1(t-1)) & \text{otherwise} \end{cases}$$

- Решение второго уравнения Фика по рекуррентной схеме

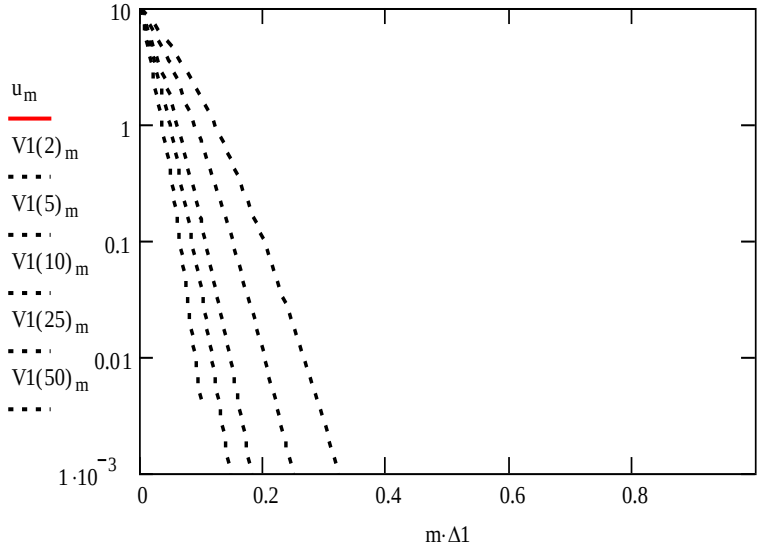


Рис. 132. Результаты расчета загонки – зависимость концентрации примеси от времени

По аналогичной схеме можно произвести расчет разгонки. Отличие заключается в том, что в качестве начального условия – сеточной функции используется функция распределения примеси, полученная при решении загонки $u_m=V1(5)_m$, и в блоке реализации разностной схемы производится вычисление концентрации на левой границе интервала:

РАЗГОНКА

$\tau := 0.0000$

$M2 := 20 \quad \Delta 2 := \frac{1}{M2}$ - **Условие устойчивости метода**, это значение должно быть меньше чем $\Delta 2$

$D2(u) := 10$

$\sqrt{2 \cdot D2(1) \cdot \tau} = 0.032$

$m := 0..M2$

$u_m := V1(5)_m$ - функция распределения примеси, полученная при решении загонки

$$F2(v) := \left| \begin{array}{l} v2_{M2} \leftarrow 0 \\ \text{for } m \in 1..M2-1 \\ \quad \left| \begin{array}{l} v2_0 \leftarrow \dfrac{\left(4 \cdot v_1 - v_2\right)}{3} \\ v2_m \leftarrow v_m + v_{m-1} \cdot \dfrac{D2\left(v_{m-1}\right) \cdot \tau}{\Delta 2^2} - v_m \cdot \dfrac{2 \cdot D2\left(v_m\right) \cdot \tau}{\Delta 2^2} + v_{m+1} \cdot \dfrac{D2\left(v_{m-1}\right) \cdot \tau}{\Delta 2^2} \end{array} \right. \\ v2 \end{array} \right.$$

- вычисление концентрации на левой границе интервала

$$V2(t) := \left| \begin{array}{ll} u & \text{if } t = 0 \\ F2(V2(t-1)) & \text{otherwise} \end{array} \right.$$

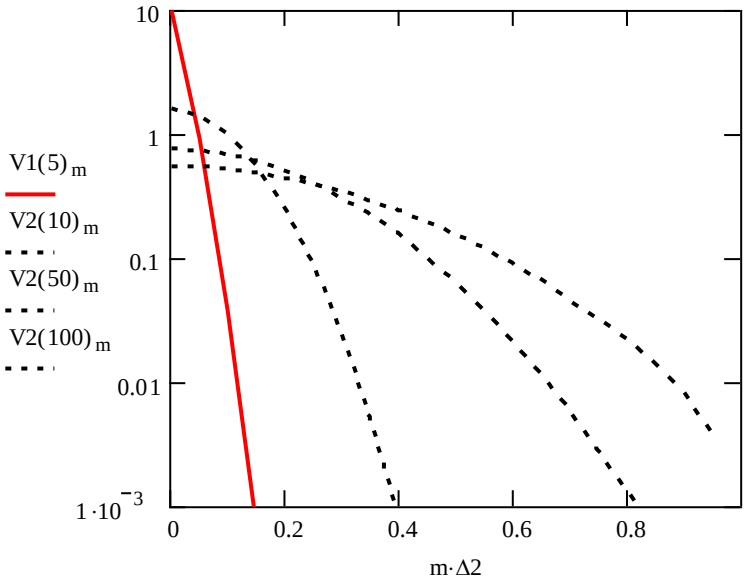


Рис. 133. Результаты расчета концентрации примеси от времени разгонки – зависимость кон-

3. ПАКЕТ MAPLE

3.1. Основные понятия и особенности. Определение переменных. Простейшие вычисления и вывод их результатов. Встроенные функции. Отображение информации в графическом виде.

Maple V Release 4 - это программный пакет для автоматизации символьных и численных вычислений. Способности данного пакета решать как простые, так и достаточно сложные задачи просто поразительны. Его функциональные возможности охватывают основные разделы математики, такие как линейная алгебра, дифференциальные вычисления, геометрия, статистика и многие, многие другие. По каждому разделу написано большое количество процедур и функций на встроенном языке Maple, что дает возможность просмотреть их содержание и, что немаловажно, добавлять свои, так называемые, процедуры пользователя.

Вычисления в пакете можно проводить двумя способами: в символьном (аналитическом) виде и численными методами. В первом случае достигается наибольшая точность, но, к сожалению, как показывает практика разработки курсовых проектов, многие классы задач просто невозможно решить таким образом. И здесь приходят на помощь численные методы, огромное количество которых находится во встроенных библиотеках.

Для написания программ на языке Maple не требуется глубоких знаний алгоритмических языков программирования. Все действия доступны любому пользователю, знакомому с Windows. Вы лишь концентрируете свое внимание на теоретической стороне решаемой задачи, на глобальном алгоритме, а все действия, например, по решению линейных и нелинейных алгебраических уравнений и систем, любого вида дифференциальных выражений, статистическому анализу, инте-

гральным преобразованиям, выполнит за вас Maple. Несмотря на все достоинства Maple V Release 4, решение некоторых задач, например, из теории автоматического управления, требует значительных ресурсов процессора, памяти, времени и терпения для достижения приемлемых результатов.

В рассматриваемую версию Maple включен редактор гипертекстовых документов. В одной среде пользователь решает свои задачи, оформляет и распечатывает документы высокого качества и сложности. Достаточно отметить, что встроенный редактор поддерживает механизм создания стилей, использует набор шрифтов операционной системы, обеспечивает создание иерархии документов, связывая их гипертекстовыми ссылками, и многое другое. Созданные в этой среде курсовые проекты, дипломы или отчеты являются живыми моделями, изменяющими свой вид при внесении в них корректив, наподобие того, как это происходит в электронных таблицах.

Рассмотрим подробнее механизм выполнения вычислений в среде Maple V Release 4. Все вычисления выполняются в рабочем документе (так называемом worksheet), в котором можно выделить строки ввода (команды), строки вывода (результаты), текст (комментарий), а также графику, трех- и двухмерную. При загрузке нового рабочего документа в его начале пользователь увидит знак ">. " – так называемое приглашение среды к вводу команды.

Команда - это строка, написанная на языке Maple и заканчивающаяся символом "**:**" или "**;**". В первом случае команда будет исполнена, но результат не будет выведен на экран, во втором случае ответ будет отображен на экране. Результаты вычислений обычно выводятся сразу после выполненных команд в той же секции.

В текст программы можно вставлять комментарии, которые являются обычным текстом, служащим для пояснения работы программы. Ниже покажем пример вычислений и объясним основные приемы работы в Maple V Release 4 (рис. 134).

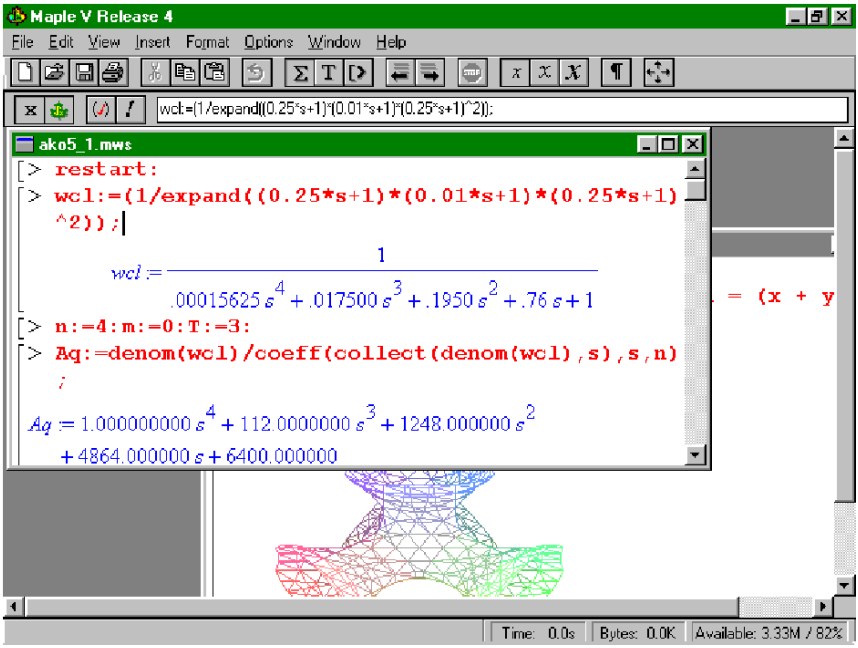


Рис. 134. Пример вычислений в Maple V Release 4.

Первое, что бросается в глаза - это вертикальные линии (квадратные скобки), расположенные слева от текста программы. Они разделяются на две категории: параграфы и секции. Секцией называется скобка, охватывающая одну или несколько команд, результаты вычислений и комментарии. Параграф – это скобка, охватывающая несколько секций. Параграф имеет кнопку, при нажатии на которую его можно свернуть или развернуть. Свернутые параграфы изображаются в виде кнопок с плюсом. Далее в книге примеры вычислений будут приведены без изображений скобок секций и параграфов.

Для эффективной работы в Maple необходимо знать некоторые тонкости языка. К ним относятся, например, команда restart и переменная Digits.

Команда restart - очищает память Maple-системы. Это означает, что все определенные до этого в программе переменные и другие объекты будут уничтожены. При этом текст программы останется неизменным. Рассмотрим пример:
Определим функцию и присвоим ей идентификатор f1:

```
> f1:=x^3+sin(cos(x^2));
```

$$f1 := x^3 + \sin(\cos(x^2))$$

```
> restart;
```

Проверим, сохранилось ли значение переменной f1 в памяти:

```
> f1;
```

$$f1$$

Как видно, Maple выдал в качестве результата имя самой переменной. Это означает, что данному идентификатору не присвоено никакого значения.

Переменная Digits - устанавливает количество значащих цифр (точность), которыми будет оперировать Maple при вычислениях. Обратите внимание на команду evalf(функция), которая выполняет вычисление функций, в данном случае числа «пи».

```
> Digits:=200
> evalf(Pi)
```

3.14159265358979323846264338327950288419716939
93751058209749445923078164062862089986280348 25

3421170679821480865132823066470938446095505822317253594081284811174502841027019385211055596446229489549303 82

В последнем примере система вычислила число π с точностью до 20 значащих цифр. По умолчанию переменная Digits равна 10.

Команда alias - переобозначение идентификаторов выражений. В следующем примере переобозначим имя мнимой единицы. После выполнения команды alias в качестве имени мнимой единицы можно использовать идентификатор j:

```
> alias(I=I, j=sqrt(-1))
                                  j
> (3*j)^2
                                 -9
```

Синтаксис Maple V Release 4 очень напоминает синтаксис таких языков программирования, как Паскаль и Фортран.

Символы и переменные. Как уже упоминалось раньше, команду необходимо заканчивать символом ":" или ";". При определении выражения используются стандартные символы: "+", "-", "*", "/", "^", "**", ":=", "=", "!". Например:

```
> y:=5!/((x^2-3*x-5)**(x+2));
```

$$y := \frac{120}{(x^2 - 3x - 5)^{(x+2)}}$$

Как видно из примера, возвести выражение в степень можно двумя способами: "^" и "**".
Для обозначения последовательности чисел используется символ "$"

```
> x!$x=1..4;
```

$$1, 2, 6, 24$$

Символ "@" - композиционный оператор. Например, чтобы вычислить вторую производную необходимо написать следующее:

```
> (D@@2)(ln);
```

$$a \to -\frac{1}{a^2}$$

Численный параметр после символа "@" может принимать и отрицательные значения:

```
> (sin@@(-1))(x);
```

$$\arcsin(x)$$

Интересную роль играет символ – " (кавычки). Одинарные кавычки ссылаются на результат предыдущей команды. Двойные - на результат, полученный две команды назад, тройные - на три команды назад. В качестве примера решим численным методом систему линейных уравнений, очистив предварительно память Maple и определив точность:

```
> restart: Digits:=2:
> 2*x+5*y-z=2;
```

$$2x + 5y - z = 2$$

> x+2*y=5;

$$x + 2\,y = 5$$

> 2*x-z=4!;

$$2\,x - z = 24$$

> fsolve({","","""},{x,y,z});

$$\{x = 14., z = 4.0, y = -4.5\}$$

Решение выводится в форме множества. Множество - это группа выражений, заключенных в фигурные скобки.

Существует также ряд команд для выполнения операций над множествами:

union - объединение множеств;

intersect - пересечение множеств;

minus - вычитание множеств.

Стандартные логические операции прекрасно дополняют возможности Maple V Release 4:

and - логическое "и";

or - логическое "или";

not - логическое отрицание.

Переменные в Maple. Переменные в Maple характеризуются именем и типом. В качестве имени переменной может использоваться любой набор символов латинского алфавита, не зарезервированных программой. Следует отметить, что система различает заглавное и строчное написание букв.

Константы в Maple бывают целочисленными, числами с плавающей запятой и обыкновенными дробями. Кроме этих типов констант существуют символьные константы - зарезервированные имена. Например: false, true, infinity, Pi, I и т.д. Следует помнить, что не рекомендуется использовать эти имена для описания своих собственных переменных:

> false:=1;

Error, attempting to assign to `false` which is protected

Но вот так уже можно:

> False:=1;

False:= 1

Функции в Maple. В Maple используется общепринятые среди математиков названия для основных математических функций, хотя есть некоторые исключения (см. табл. на рис 135).

ФУНКЦИЯ	ОПИСАНИЕ
abs	модуль
Re	действительная часть
Im	мнимая часть
factorial	факториал
log	обыкновенный логарифм
ln	натуральный логарифм
log10	десятичный логарифм
sqrt	квадратный корень
exp	экспонента
argument	аргумент комплексного числа
binomial	биномиальный коэффициент
round	округление
trunc	отсечение дробной части

Рис. 135. Названия для основных математических функций Maple V Release 4.

Тригонометрические функции записываются в форме, которая интуитивно понятна пользователю и не требуют детального описания:

sin, cos, tan, sec, csc, cot, sinh, cosh, tanh, sech, csch, coth, arcsin, arccos, arctan, arcsec, arccsc, arccot,arcsinh, arccosh, arctanh, arcsech, arccsch, arccoth.

В Maple запрограммированы некоторые математические функции, такие как гамма-функция, функция Лапласа, бета-функция, функция Бесселя, функция Дирака и Хэвисайда, функции Якоби и многие другие.

Операции с формулами. При работе с математическими выражениями практически всегда приходится выполнять множество таких операций, как раскрытие скобок, разложение на множители, приведение подобных членов, которые отнимают массу времени. Maple позволяет сосредоточится над основными преобразованиями, избегая рутинной работы. Ниже представлены основные команды этого класса (см. табл. на рис 136).

Начнем с самой распространенной операции - упрощения выражения. Переменной rex присвоим некоторую сумму из тригонометрических слагаемых.

```
> rex := cos(x)^3 + sin(x)^4 + 2*cos(x)^4 -2*sin(x)^4 -
cos(2*x):
```

Далее упростим полученный полином:

```
> simplify(rex);
```

$$\cos(x)^3 + \cos(x)^4$$

Из приведенного примера видно, что Maple преобразовал выражение rex, подтверждая известные тригонометрические правила. Ниже приведен еще один пример с использованием экспоненты и натурального логарифма.

КОМАНДА	ОПИСАНИЕ
collect(w, x)	Приведение подобных членов в выражении w относительно переменной x
denom(d)	Выделение знаменателя дроби d
expand(w)	Раскрытие скобок выражения w
factor(w)	Факторизация (разложение на множители) выражения w
lhs(ur)	Выделение левой части уравнения ur
normal(w)	Нормализация (сокращение) дроби w
numer(w)	Выделение числителя дроби d
op(i..j, e)	Выделение подвыражения из выражения e
rhs(ur)	Выделение правой части уравнения ur
simplify(w)	Упрощение выражения w
subs(x=t, w)	Подстановка в выражение w вместо выражения x выражение t
subsop(eq1,..,eqN, expr)	Замена некоторого операнда в выражении expr
trigsub(w)	Определение всех тригонометрических эквивалентов выражения w

Рис. 136. Названия для основных команд Maple V Release 4.

```
> w:=exp(a+ln(c*exp(c^a)));
> simplify(w);
```

$$w := \mathbf{e}^{\left(a + \ln\left(c\, \mathbf{e}^{(c^a)}\right)\right)}$$

$$c\, \mathbf{e}^{(c^a + a)}$$

Для раскрытия скобок используется команда expand. Далее в примере определим дробь w2 путем деления одного полинома на другой.

Отметим, что и в числителе и в знаменателе присутствует общий множитель (x+3).

```
> expand((x+3)*(x-y)):expand((x+3)*(x+1)):
> w2:=('')/('');
```

$$w2 := \frac{x^2 + 4x + 3}{x^2 - xy + 3x - 3y}$$

А теперь произведем факторизацию выражения (операцию обратную expand), при этом дробь w2 должна упроститься, т.к. множительw2 сократится.

```
> factor(w2);
```

$$\frac{x+1}{x-y}$$

Примеры из курса математического анализа. Основу курса математического анализа составляют такие понятия как пределы, производные, первообразные функций, интегралы разных видов, ряды и дифференциальные уравнения.

Рассмотрим на примерах команду limit, которая позволяет находить пределы функций.

```
> restart;
> f(x):=(x^3-3*x^2+2*x-5)/(x^2+2);
```

$$f(x) := \frac{x^3 - 3x^2 + 2x - 5}{x^2 + 2}$$

```
> Limit (f(x), x=-1)= limit (f(x), x=-1)
```

$$\lim_{x \to (-1)} \frac{x^3 - 3x^2 + 2x - 5}{x^2 + 2} = \frac{-11}{3}$$

Можно также находить односторонние пределы. Для этого достаточно указать ключевое слово left или right.

```
> g(x):=1/x
```

$$g(x) := \frac{1}{x}$$

```
> Limit(g(x),x=0,left)=limit(g(x),x=0,left);
```

$$\lim_{x \to 0-} \frac{1}{x} = -\infty$$

Чтобы продифференцировать функцию, достаточно воспользоваться командой diff.

```
> restart:
> f(x):=ln(sqrt(exp(3*x)/(1+exp(3*x))));
```

$$f(x) := \ln\left(\sqrt{\frac{e^{(3x)}}{1 + e^{(3x)}}}\right)$$

```
> simplify(diff(f(x),x);
```

$$\frac{3}{2} \frac{1}{1 + e^{(3x)}}$$

Можно взять частные производные от функции многих переменных:

> diff(f(x,y),x,y);

$$\frac{\partial^2}{\partial y\,\partial x}\,\mathrm{f}(x,y)$$

При помощи оператора формирования последовательности ($) можно брать производные высоких порядков.

> Diff(sin(x),x$3)=diff(sin(x),x$3);

$$\frac{\partial^3}{\partial x^3}\sin(x) = -\cos(x)$$

Взять интеграл от какой-либо функции можно при помощи оператора int.

> restart:

Неопределенный интеграл:

> Int((3*x^2+8)/(x^3+4*x^2+4*x), x)=
int((3*x^2+8)/(x^3+4*x^2+4*x),x);

$$\int \frac{3x^2+8}{x^3+4x^2+4x}\,dx = 2\ln(x) + \frac{10}{x+2} + \ln(x+2)$$

Определенный интеграл:

> Int(sin(phi)^3*sqrt(cos(phi)),phi=0..Pi/2) =
int(sin(phi)^3*sqrt(cos(phi)),phi=0..Pi/2);

$$\int_0^{\frac{1}{2}\pi} \sin(\phi)^3 \sqrt{\cos(\phi)}\, d\phi = \frac{8}{21}$$

Несобственные интегралы (первого и второго рода):

```
> Int(1/(x-1)^2,x=0..2)=int(1/(x-1)^2,x=0..2);
```

$$\int_0^2 \frac{1}{(x-1)^2}\, dx = \infty$$

```
> Int(1/(x^2+2*x+2),x=-infinity..infinity)=
int(1/(x^2+2*x+2),x=-infinity..infinity);
```

$$\int_{-\infty}^{\infty} \frac{1}{x^2+2x+2}\, dx = \pi$$

В тех случаях, когда интеграл может быть вычислен в численной форме, поможет функция evalf.

```
> ww:=int( exp(-x^3), x = 0..1 );
```

$$ww := \int_0^1 e^{(-x^3)}\, dx$$

```
> evalf(ww,5);
```

.80751

В Maple можно при помощи команды sum находить предел сходимости ряда.

```
> Sum(7^(3*n)/(2*n-5)!,n=3..infinity)=
evalf(sum(7^(3*n)/(2*n-5)!,n=3..infinity));
```

$$\sum_{n=3}^{\infty} \frac{7^{(3n)}}{(2n-5)!} = .1203515449\ 10^{15}$$

Полезной может оказаться и функция вычисления произведения product:

```
> product( a[k], k=0..n );
```

$$\prod_{k=0}^{n} a_k$$

Интерфейс графической двухмерной системы. При выполнении любых видов графических построений на плоскости используется команда plot(*функция, начало диапазона, окончание диапазона*) (см. рис. 137) в которой указывается отображаемая функция с указанием диапазона отображения и перед пользователем появляется интерфейс двухмерной графической системы.

В данном случае графики строятся в отдельном окне, так как в пункте меню Options рабочего документа режим Plot Display установлен в значение Window. Можно также изображать графические построения непосредственно в рабочем документе. В этом случае необходимо установить режим Plot Display в значение Inline.

При этом строка команд содержит следующие дополнительные к основным для обычной программы пункты меню:

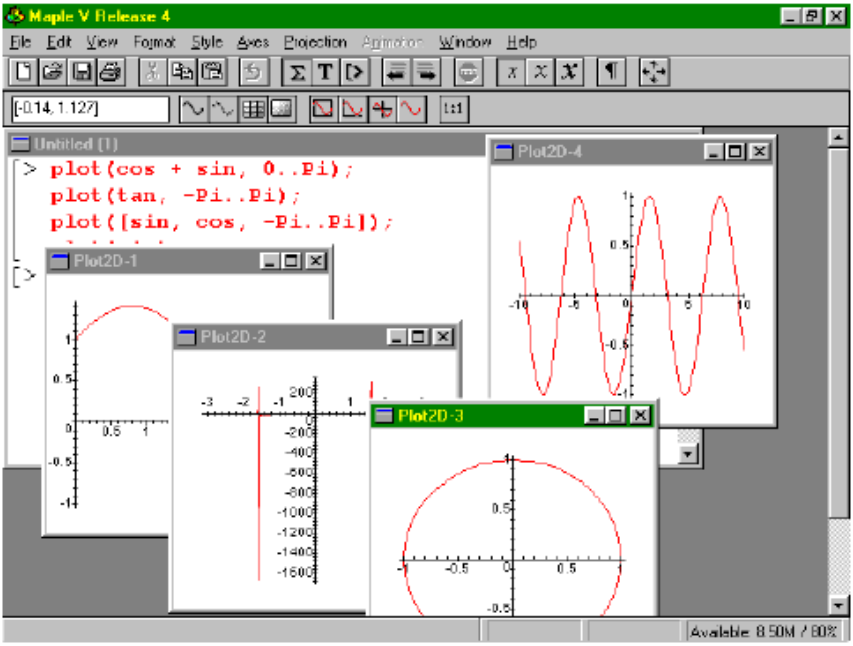


Рис. 137. Пример построения двумерных графиков в Maple V Release 4.

Style - определяет стиль построения (вид линий, сетка и т.п.);
Axes - управляет стилем координатных осей;
Projection - определяет масштаб изображения;
Animation - анимация графиков.

Интерфейс графической трехмерной системы. При любом виде трехмерного построения, выполняемого при помощи команды plot3d (см. рис. 138) перед пользователем возникает интерфейс графической трехмерной системы.

Перечислим пункты меню, которые не описаны в стандартном интерфейсе рабочего документа:
Style - определяет стиль построения;
Color - определяет цвет построения;

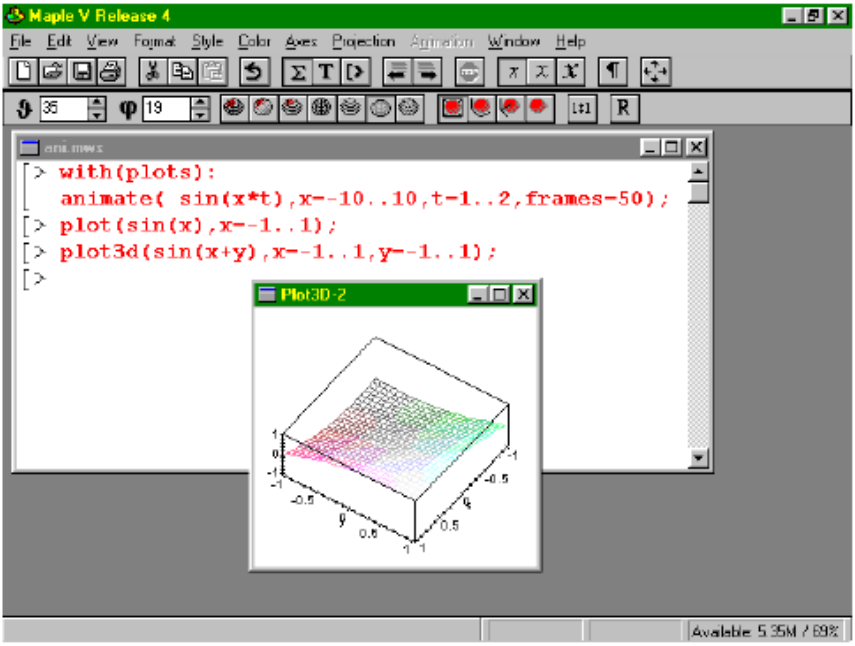


Рис. 138. Пример построения трехмерных графиков в Maple V Release 4.

Axes - вид осей (аналогичен соответствующему пункту из интерфейса графической системы);
Projection - определяет масштаб изображения;
Animation - анимация (полностью совпадает с управлением двумерной анимацией). /3/

4. ПАКЕТ MATHLAB

4.1. Основные понятия и особенности. Определение переменных. Простейшие вычисления и вывод их результатов. Встроенные функции. Отображение информации в графическом виде.

MATLAB – это высоко производительный язык для технических расчетов. Он включает в себя вычисления, визуализацию и программирование в удобной среде, где задачи и решения выражаются в форме, близкой к математической. типичное использование MATLAB – это:

- математические вычисления;
- создание алгоритмов;
- моделирование;
- анализ данных, исследования и визуализация;
- научная и инженерная графика;
- разработка приложений, включая создание графического интерфейса.

MATLAB – это интерактивная система, в которой основным элементом данных является массив. Это позволяет решать различные задачи, связанные с техническими вычислениями, особенно в которых используются матрицы и вектора, в несколько раз быстрее, чем при написании программ с использованием "скалярных" языков программирования, таких как Си или Фортран.

Слово MATLAB означает матричная лаборатория(matrix laboratory). MATLAB был специально написан для обеспечения легкого доступа к LINPACK и EISPACK, которые представляют собой современные программные средства для матричных вычислений.

MATLAB развивался в течении нескольких лет. ориентируясь на различных пользователей. В университетской среде он представлял собой стандартный инструмент для работы в различных областях математики, машиностроении и науки. В промышленности MATLAB – это инструмент для высоко продуктивных исследований, разработок и анализа данных.

В MATLAB важная роль отводится специализированным группам программ, называемым toolboxes. Они очень важны для большинства пользователей MATLAB, так как позволяют

изучать и применять специализированные методы. Toolboxes – это всесторонняя коллекция функций MATLAB (M-файлов), которые позволяют решать частные классы задач. Toolboxes применяются для обработки сигналов, систем контроля, нейронных сетей, нечеткой логики, вейвлетов, моделирования и т.д.

Система MATLAB состоит из пяти основных частей:

- Язык MATLAB. Это язык матриц и массивов высокого уровня с управлением потоками, функциями, структурами данных, вводом выводом и особенностями объектно-ориентированного программирования. Это позволяет как программировать в "небольшом масштабе" для быстрого создания черновых программ, так и "большом" для создания больших и сложных приложений.

- Среда MATLAB. Это набор инструментов и приспособлений, с которыми работает пользователь или программист MATLAB. Она включает в себя средства для управления переменными в рабочем пространстве MATLAB, вводом и выводом данных, а также создания и отладки M-файлов и приложений MATLAB.

- Управляемая графика. Это графическая система MATLAB, которая включает в себя команды высокого уровня для визуализации двух- и трехмерных данных, обработки изображений, анимации и иллюстрированной графики. Она также включает в себя команды низкого уровня, позволяющие полностью редактировать внешний вид графики, так же при создании Графического Пользовательского Интерфейса (GUI) для MATLAB приложений.

- Библиотека математических функций. Это обширная коллекция вычислительных алгоритмов от элементарных функций таких как сумма, синус, косинус, комплексная арифметика, до более сложных, таких как обращение матриц, нахождение собственных значений, функции Бесселя, быстрое преобразование Фурье.

- Программный интерфейс. Это библиотека, которая позволяет писать программы на Си и Фортране, которые взаимодействуют с MATLAB (динамическая связь), вызывая MATLAB как вычислительный инструмент и для чтения и записи MAT – файлов.

Как уже говорилось выше – MATLAB работает данными в виде с матриц. В MATLAB матрица – это прямоугольный массив чисел. Особое значение придается матрицам 1×1, которые являются скалярами, и матрицам, имеющим одну строку или один столбец – векторам. MATLAB использует различные способы для хранения численных или не численных данных, однако в начале лучше всего рассматривать все данные как матрицы. MATLAB организован так, чтобы все операции в нем были как можно более естественными. В то время как другие программные языки работают с числами как с элементами языка, MATLAB позволяет вам быстро т легко оперировать целыми матрицами.

Ввод матриц. Ввод матриц можно осуществлять несколькими способами:

- вводить полный список элементов;
- загружать матрицы из внешних файлов;
- генерировать матрицы, используя встроенные функции;
- создавать матрицы с помощью ваших собственных функций в М-файлов.

Рассмотрим ввод матрицы на примере так называемой матрицы Дюрера как списка элементов. Для этого нужно соблюдать следующие правила:

- отделять элементы строки пробелами или запятыми;
- использовать точку с запятой для обозначения окончания каждой строки;
- окружить все элементы квадратными скобками.

Наберите:

```
A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

MATLAB отобразит введенную матрицу:

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

Транспонирование матрицы осуществляется при помощи оператора транспонирования, который обозначается апострофом:

```
A'
ans =
  16     5     9     4
   3    10     6    15
   2    11     7    14
  13     8    12     1
```

Заметим, что когда выходная переменная не определена, то MATLAB использует временную переменную ans (от слова answer – ответ).

Сумму элементов в столбце можно определить при помощи функции sum:

```
sum(A)
ans =
  34    34    34    34
```

Для определения сумм элементов в строках надо транспонировать матрицу, определить сумму элементов в столбцах, и снова транспонировать результат:

```
sum(A')'
 ans =
   34
   34
   34
   34
```

Функция diag выбирает элементы на главной диагонали:

```
diag(A)
 ans =
   16
   10
    7
    1
```

Поэтому для суммирования элементов главной диагонали нужно применить функции diag и sum:

```
sum(diag(A))

ans =
  34
```

Функция fliplr зеркально отображает матрицу слева направо, поэтому вычисление суммы элементов антидиагонали (так называется другая диагональ) можно провести так:

```
sum(diag(fliplr(A)))

ans =
    34
```

Как видно, суммы элементов строк, столбцов, диагоналей в рассмотренной матрице Дюрера одинаковы и равны 34.

Элемент в строке i и столбце j матрицы A обозначается A(i,j). Поэтому можно выполнять арифметические действия с отдельными элементами матриц:

```
A(1,4) + A(2,4) + A(3,4) + A(4,4)

ans =
    34
```

Двоеточие, :, - это один из наиболее важных операторов MATLAB. Он проявляется в различных формах. Выражение

```
1:10
```

-это вектор-строка, содержащая целые числа от 1 до 10

```
 1     2     3     4     5     6     7     8     9    10
```

Для получения обратного интервала опишем приращение:

```
100:-7:50

 100    93    86    79    72    65    58    51
```

Можно задавать последовательности с нецелым шагом

```
0:pi/4:pi

0    0.7854    1.5708    2.3562    3.1416
```

Индексное выражение, включая двоеточие, относится к части

матрицы, например, A(1:k,j) это первые k элементов j-ого столбца матрицы A.

Как и большинство других языков программирования, MATLAB предоставляет возможность использования математических выражений, но в отличии от многих из них, это выражение MATLAB включает матрицы, основные составляющие выражения:

- переменные
- числа
- операторы
- функции

Переменные. В MATLAB нет необходимости в определении типа переменных или размерности. Когда MATLAB встречает новое имя переменной, он автоматически создает переменную и выделяет соответствующий объем памяти. Если переменная уже существует, MATLAB изменяет ее состав и, если это необходимо, выделяет дополнительную память.

Числа. MATLAB использует принятую десятичную систему счисления, с необязательной десятичной точкой и знаками плюс-минус для чисел. Научная система счисления использует букву **e** для определения множителя степени десяти. Мнимые числа используют i или j как суффикс. Вот некоторые примеры правильной записи чисел:

3	-99	0.0001
9.6397238	1.60210e-20	6.02252e23
1i	-3.14159j	3e5i

Все числа для хранения используют формат long, определенный стандартом плавающей точкой IEEE. Числа с плавающей точкой обладают ограниченной точностью – приблизительно 16 значащих цифр и ограничен диапазоном – приблизительно от 10^{-308} до 10^{308}.

Операторы. Выражения используют обычные арифметические операции и правила старшинства.

+	сложение
-	вычитание
*	умножение
/	деление
\	левое деление(описано в разделе Матрицы и Линейная Алгебра в книге "*Using MATLAB*")
^	степень
'	комплексно сопряженное транспонирование
()	определение порядка вычисления

Функции. MATLAB предоставляет большое количество элементарных математических функций, таких как abs, sqrt, exp, sin. MATLAB представляет и более сложные функции, включая Гамма функцию и функции Бесселя. Большинство из этих функций имеют комплексные аргументы. Чтобы вывести список всех элементарных математических функций, нужно ввести:

```
help elfun
```

Для вывода более сложных математических и матричных функций введите соответственно:

```
help specfun
help elmat
```

Некоторые функции, такие как sqrt, и sin – встроенные, другие, например, такие как gamma и sinh – реализованы в специальных M-файлах, поэтому можно получить доступ к коду вычисления и даже модифицировать его.

Некоторые специальные функции представляют значения часто используемых констант:

pi	3.14159265…
i	мнимая единица, $\sqrt{-1}$
j	то же самое, что и i
eps	относительная точность числа с плавающей точкой, 2^{-52}
realmin	наименьшее число с плавающей точкой, 2^{-1022}
realmax	наибольшее число с плавающей точкой, $(2-\varepsilon)2^{1023}$
Inf	бесконечность
NaN	не число

Бесконечность появляется при делении на ноль или при переполнении, то есть при превышении realmax. Не число (NaN) появляется в результате выполнения 0/0 или Inf-Inf (то есть неопределенность).

Имена функций не являются зарезервированными, поэтому можно изменять их значение.

Приведем примеры некоторых выражений и полученных результатов:

```
rho = (1+sqrt(5))/2

rho =
    1.6180


a = abs(3+4i)

a =
     5


z = sqrt(besselk(4/3,rho-i))

z =
   0.3730 + 0.3214i


huge = exp(log(realmax))

huge =
1.7977e+308
```

```
toobig = pi*huge

toobig =
 Inf
```

Генерирование матриц. MATLAB имеет четыре функции, которые создают основные матрицы:

zeros	все нули
ones	все единицы
rand	равномерное распределение случайных элементов
randn	нормальное распределение случайных элементов

Некоторые примеры:

```
Z = zeros(2,4)

Z =
     0     0     0     0
     0     0     0     0

F = 5*ones(3,3)

F =
     5     5     5
     5     5     5
     5     5     5

N = fix(10*rand(1,10))

N =
     9     2     6     4     8     7     4     0     8     4


R = randn(4,4)

R =
   -0.4326   -1.1465    0.3273   -0.5883
   -1.6656    1.1909    0.1746    2.1832
    0.1253    1.1892   -0.1867   -0.1364
    0.2877   -0.0376    0.7258    0.1139
```

Загрузка матриц. Команда load считывает двоичные файлы, содержащие матрицы, созданные в MATLAB ранее, или текстовые файлы, содержащие численные данные. Тек-

стовые файлы должны быть сформированы в виде прямоугольной таблицы чисел, отделенных пробелами, с равным количеством элементов в каждой строке. Например, создадим вне MATLAB в текстовом редакторе текстовый файл:

```
16.0    3.0    2.0    13.0
 5.0   10.0   11.0     8.0
 9.0    6.0    7.0    12.0
 4.0   15.0   14.0     1.0
```

И сохраним его под именем magik.dat. тогда команда

```
load magik.dat
```

прочитает этот файл и создаст переменную magik, содержащую данную матрицу.

М-файлы. Это текстовые файлы, содержащие как данные, так и фрагменты кода MATLAB. Такие файлы имеют расширение .m . Для создания М-файлов можно использовать текстовый редактор. Например, создадим файл из 5 строк:

```
A = [ ...
16.0    3.0    2.0    13.0
 5.0   10.0   11.0     8.0
 9.0    6.0    7.0    12.0
 4.0   15.0   14.0     1.0    ];
```

Сохраним его под именем magik.m . Тогда выражение

```
magik
```

прочитает файл и создаст переменную A, содержащую данную матрицу.

Объединение. Объединение – это процесс соединения нескольких матриц в одну. Пара квадратных скобок – это и есть оператор объединения. Первая матрица A была создана именно таким способом, когда были объединены 16 элементов матриц 1×1. Сформируем матрицу

```
B = [A A+32; A+48 A+16]
```

Результатом будет матрица 8×8, получаемая соединением че-

тырех подматриц A

```
B =
    16     2     3    13    48    34    35    45
     5    11    10     8    37    43    42    40
     9     7     6    12    41    39    38    44
     4    14    15     1    36    46    47    33
    64    50    51    61    32    18    19    29
    53    59    58    56    21    27    26    24
    57    55    54    60    25    23    22    28
    52    62    63    49    20    30    31    17
```

Удаление строк и столбцов. Можно удалять строки и столбцы использую пару квадратных скобок. Рассмотрим

```
X = A;
```

Теперь удалим второй столбец матрицы X

```
X(:,2) = []
```

Теперь матрица X выглядит так:

```
X =
    16     3    13
     5    10     8
     9     6    12
     4    15     1
```

Если удалить всего один элемент матрицы

```
X(1,2) = []
```

результат уже не будет матрицей и в результате вычислений будет выдано сообщение об ошибке. Однако использование одного индекса удаляет отдельный элемент или последовательность элементов и преобразует оставшиеся элементы в вектор-строку:

```
X(2:2:10) = []

X =
    16     9     3     6    13    12     1
```

Графика. MATLAB имеет широкие возможности для графического изображения векторов и матриц, а также для создания комментариев и печати графики.

Создание графика осуществляется при помощи функции *plot*,

использование которой приводит к различным результатам в зависимости от вида входных параметров. Например, *plot(y)* создает кусочно-линейный график зависимости элементов *y* от их индексов. Если задаются два вектора в качестве аргументов, *plot(x,y)* создаст графики зависимости *y* от *x*. Например, для построения графика значений функции *sin* от нуля до 2π необходимо сделать следующее (рис.139):

```
t = 0:pi/100:2*pi;
y = sin(t);
plot(t,y)
```

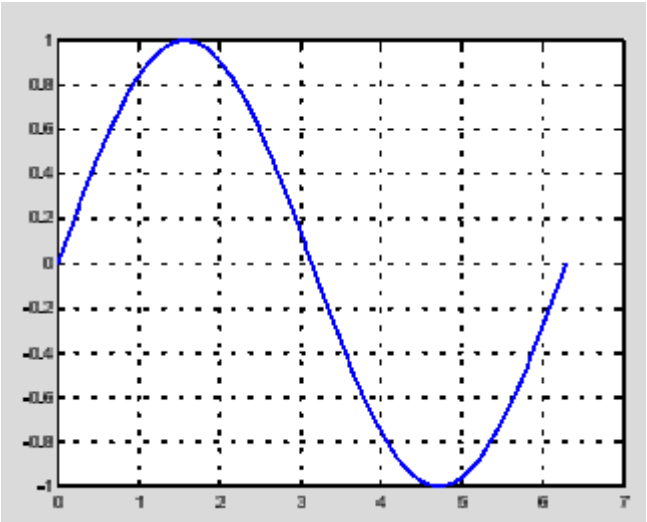


Рис. 139. Создание графика функции *sin*.

Если функция *plot(x,y)*используется с разными параметрами, MATLAB автоматически присваивает каждому графику свой цвет, исключая случаи, когда это сделано пользователем. Например, график трех функций будет выглядеть так (функция *y* определена в предыдущем примере):

```
y2 = sin(t-.25);
y3 = sin(t-.5);
plot( t, y, t, y2, t, y3)
```

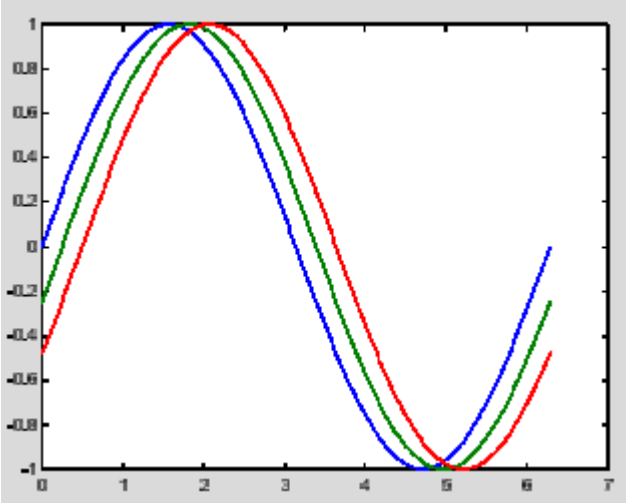


Рис. 140. График трех функций.

Установка атрибутов зависимости на графике, таких как цвет линий и тип маркера, осуществляется следующим образом:

plot(x,y 'цвет_стиль_маркер')

где: цвет обозначается буквами c, m, y, r, g, b, w, k - голубой, малиновый, желтый, красный, зеленый, синий, белый, черный соответственно;

стиль линий обозначается '–' сплошная линия, '– –' линия с разрывами, ':' пунктирная, '–.' штрихпунктирная, 'none' без линии;

маркеры обозначаются '+', 'o', 'x', '*'.

Например, выражение

```
plot(x,y,'y:+')
```

строит желтый пунктирный график и помещает маркеры '+' в каждую точку данных.

Окно изображений. Функция *plot* автоматически откры-

вает новое окно изображения, если до этого его не было на экране. Если окно уже существует, то *plot* использует его по умолчанию. Для открытия нового окна и выбора его по умолчанию, необходимо использовать функцию *figure(n)* где n – номер в заголовке окна. В этом случае результаты всех последующих команд будут выводиться в это окно.

Добавление кривых на график. Команда *hold on* не стирает уже выведенный в окно график, а добавляет в него новую кривую, изменяя, если требуется, масштаб осей. Например, в следующем примере сначала создаются контурные линии функцией *peaks*, а затем накладывает на нее псевдоцветной график той же функции:

```
[x,y,z] = peaks;
contour(x,y,z,20,'k')
hold on
pcolor(x,y,z)
shading interp
```

Команда *hold on* является причиной того, что график *pcolor* комбинируется с графиком *contour* в одном окне.

Графики в полярной системе координат. В полярной системе координат любая точка представляется как конец радиус-вектора, исходящего из начала системы координат, имеющего длину RHO и угол THETA. Для построения графика функции RHO(THETA) используются приведенные ниже команды. Угол THETA обычно меняется от 0 до 2*pi. Для построения графиков функций в полярной системе координат используются команды типа *polar(...)*:

- *polar(THETA, RHO)* — строит график в полярной системе координат, представляющий собой положение конца радиус-вектора с длиной RHO и углом THETA;
- *polar(THETA,RHO.S)* — аналогична предыдущей команде, но позволяет задавать стиль построения с помощью строковой константы S по аналогии с командой *plot*(см. выше).

Рис. 141 демонстрирует результат выполнения команд:

```
t=0:pi/50:2*pi;
```

```
polar(t.sin(5*t))
```

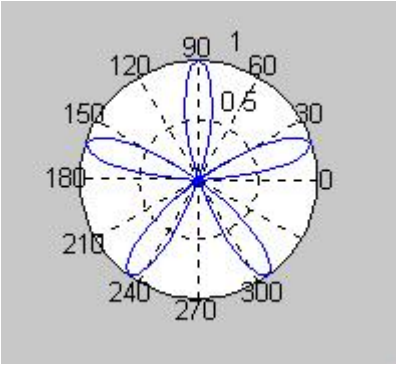


Рис. 141. График функции в полярной системе координат.

Контурные графики. Контурные графики служат для представления на плоскости функции двух переменных вида *z(x, y)* с помощью линий равного уровня. Они получаются, если трехмерная поверхность пересекается рядом плоскостей, расположенных параллельно друг другу. При этом контурный график представляет собой совокупность спроецированных на плоскость *(x, y)* линий пересечения поверхности *z(x, y)* плоскостями.

Для построения контурных графиков используются команды *contour*:

- *contour(Z)* — строит контурный график по данным матрицы Z с автоматическим заданием диапазонов изменения *x* и *y;*
- *contour(X,Y,Z)* — строит контурный график по данным матрицы Z с указанием спецификаций для X и Y;
- *contour(Z,N)* и *contour(X,Y,Z,N)* — дает построения, аналогичные ранее описанным командам, с заданием N линий равного уровня (по умолчанию N=10);
- *contour(Z,V)* и *contour(X,Y,Z,V)* — строят линии равного уровня для высот, указанных значениями элементов вектораV;

- *contour(Z,[v v])* или *contour(X,Y,Z,[v v])* — вычисляет одиночный контур для уровня v;
- *contour(... 'LINESPEC')* — позволяет использовать перечисленные выше команды с указанием вида линий (см. выше функцию *plot*), которыми идет построение.

Пример построения контурного графика поверхности, заданной функций peaks:

```
z=peaks(27);
contour(z,15)
```

Построенный в этом примере график показан на рис. 142. Заметим, что объект — функция peaks — задан в системе в готовом виде.

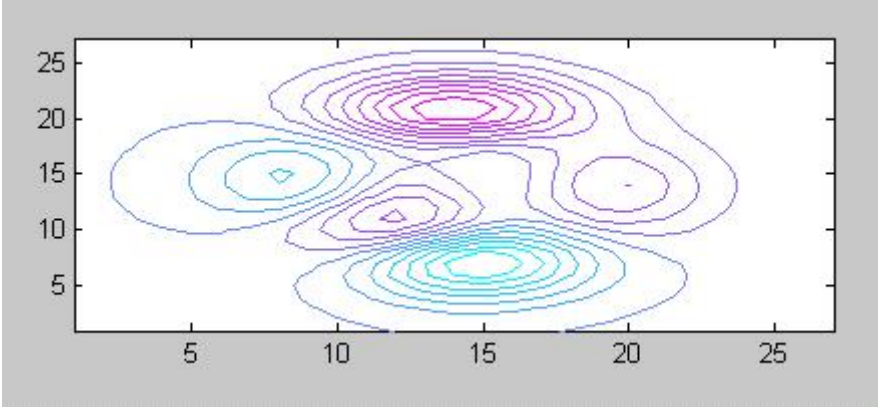


Рис. 142. Контурный график, построенный с помощью команды contour .

Графики поля градиентов (векторного поля) quiver . Для построения графиков полей градиента служат команды quiver:
- *quiver(X,Y,U,V)* — строит график поля градиентов в виде стрелок для каждой пары элементов массивов X и Y, причем элементы массивов U и V указывают направление и размер стрелок;

- *quiver(U,V)* — строит векторы скорости в равнорасположенных точках на плоскости *(x, y);*
- *quiver(U,V,S)* или *quiver(X,Y,U,V,S)* — автоматически масштабирует стрелки по сетке и затем вытягивает их по значению S. Используйте S=0, чтобы построить стрелки без автоматического масштабирования;
- *quiver(...,LINESPEC)* — использует для векторов указанный тип линии. Указанные в LINESPEC маркеры рисуются у оснований, а не на концах векторов. Для отмены любого вида маркера используйте спецификацию '.'. Спецификации линий, цветов и маркеров были подробно описаны в разделе, посвященном команде *plot*;
- quiver(.... 'filled') — дает график с закрашенными маркерами;

Ниже представлен пример применения команды quiver:

```
x = -2:.2:2: y = -1:.2:1;
[xx.yy] = meshgrid(x.y);
zz = xx.*exp(-xx. 2-yy. 2);
[px.py] = gradient(zz..2..2);
quiver(x.y.px.py,2);
```

Построенный в этом примере график показан на рис. 143.

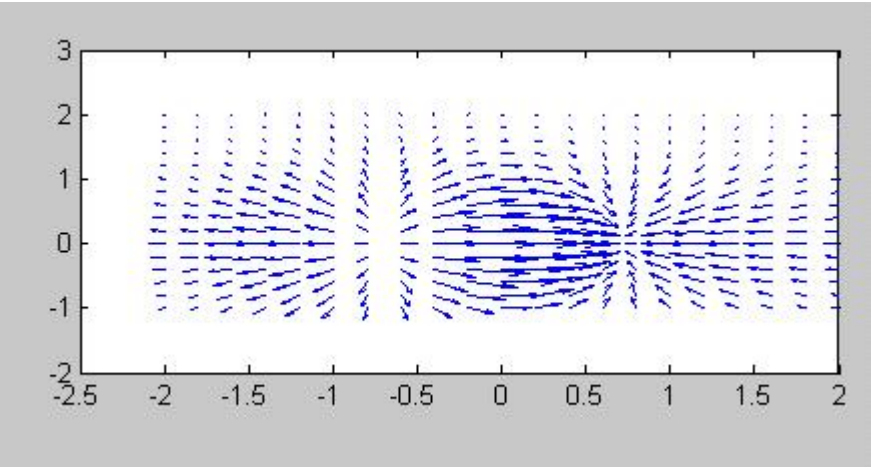


Рис. 143. Пример построения графика поля градиентов

Управление осями. Функция *axis* имеет несколько возможностей для настройки масштаба, ориентации и коэффициента сжатия.

Обычно MATLAB находит максимальное и минимальное значение и выбирает соответствующий масштаб и маркитирование осей. Функция *axis* заменяет значения по умолчанию значениями, вводимыми пользователем:

```
axis( [xmin xmax ymin ymax] )
```

Можно так же использовать ключевые слова для управления внешним видом осей. Например:

axis square - создает оси x и y равной длины,

axis equal - создает равные отметки приращений для осей,

axis auto – возвращает автоматический режим определения,

axis on – включает отображение осей и промежуточных делений,

axis off – отключает отображение осей.

Аналогично, функция *grid off* – отключает отображение сетки координат, *grid off* – включает отображение сетки.

Подписи к осям и заголовки. Функции *xlabel, ylabel, zlabel* добавляют подписи к соответствующим осям, функция *title* добавляет заголовок в верхнюю часть окна, а функция *text* добавляет текст, любые символы в любое место на графике (рис. 144).

```
t  = -pi:pi/100:pi;
y = sin(t);
plot(t,y)
axis([-pi pi -1 1])
xlabel( ' -\pi \leq \itt \leq \pi ' )
ylabel( ' sin(t) ' )
title( ' График функции sin ' )
text(-1, -1/3, ' \it{Отметьте нечетную симметрию} ' )
```

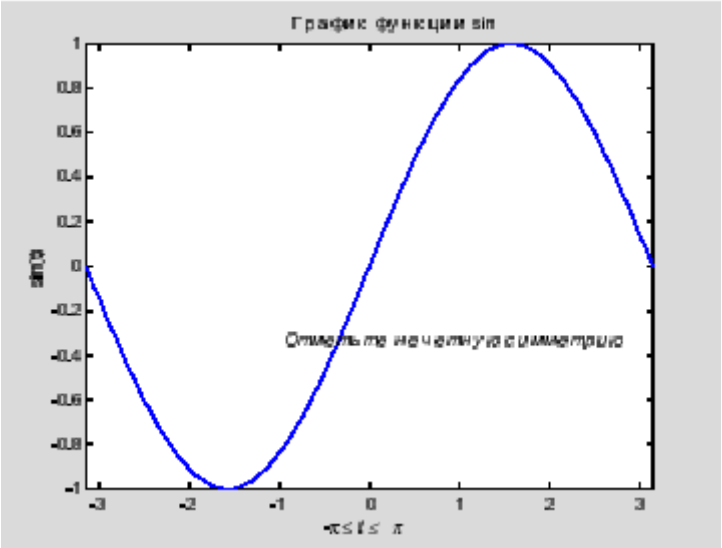


Рис. 144. Добавление подписей к рисункам.

Трехмерная графика. MATLAB определяет поверхность z как координаты точек над координатной сеткой плоскости x-y, используя прямые линии для соединения соседних точек. Функции *mesh* и *surf* отображают поверхность в трех измерениях. При этом *mesh* создает поверхность каркасного типа – без закрашивания промежутков, а *surf* – вместе с линиями отображает в цвете и саму поверхность.

Для отображения функции двух переменных z=f(x,y) требуется создание перед использованием функции *mesh* и *surface* матриц X и Y, состоящих из повторяющихся строк или столбцов, соответственно. Так как набирать повторяющиеся строки и столбцы для создания матриц X и Y утомительно, используется функция *meshgrid* для создания из векторов x и y матриц X и Y. Все строки матрицы X дублируют вектор x, а столбцы Y - дублируют вектор y.

На рис. 145 показан график поверхности, созданной ко-

мандой *mesh(X,Y,Z)*, на рис. 146 показан график поверхности, созданной командой *surf(X,Y,Z)*

```
[X,Y]=meshgrid([-3:0.15:3]);
Z=X.^2+Y.^2;
mesh(X,Y,Z)
```

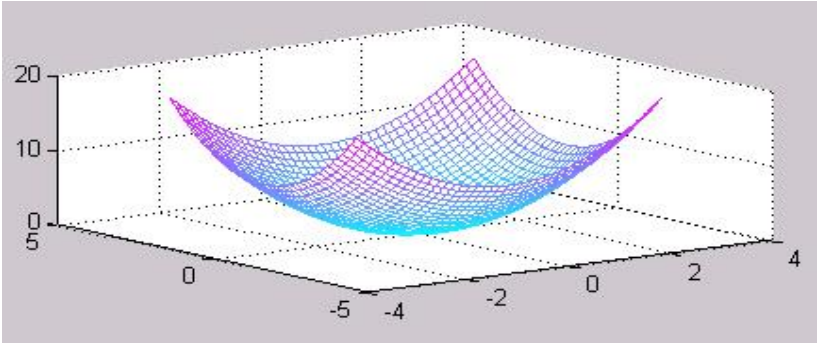


Рис. 145. График поверхности, созданный командой *mesh(X,Y,Z)*.

```
[X,Y]=meshgrid([-3:0.15:3]);
Z=X.^2+Y.^2;
surf(X,Y,Z)
```

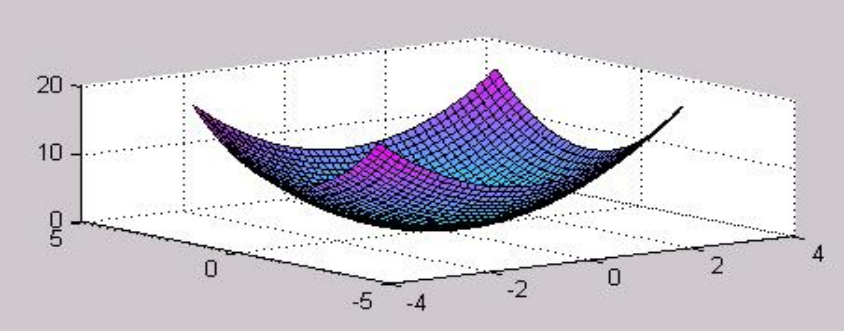


Рис. 146. График поверхности, созданный командой *surf(X,Y,Z)*.

Первая строка создает координатную сетку на плоскости X-Y от минус 3 до 3 с шагом 0.15. Нетрудно заметить, что функциональная окраска линий поверхности заметно усиливает наглядность ее представления. Можно добавить, что существует огромное количество функций для представления графиков во всех мыслимых видах. Тем не менее параметры для каждой задаются по общей одинаковой схеме.

Печать графики. Опция **Print** в меню **File** или команда *print* печатает графику в MATLAB. Опция **Print** вызывает диалоговое окно, которое позволяет выбрать стандартные варианты печати. Команда *print* позволяет более гибко настроить процедуру печати. Однако для правильного её использования необходимо знать возможности конкретного используемого принтера, например, поддерживает ли он печать PostScript уровня 1 или 2. Более подробно об этой функции можно узнать в справочной системе.

Справочная система. Для вызова справочной системы используется команда *help* которая ищет информацию по заданному после данной функции слову (например, имени функции, по которой необходимо получит информацию) и отображает его прямо в командном окне. Например:

```
   help magic
```

выдаст

```
 MAGIC  Magic square.
    MAGIC(N) is an N-by-N matrix constructed from the integers
    1 through N^2 with equal row, column, and diagonal sums.
    Produces valid magic squares for N = 1,3,4,5,...
```

или

```
help matfun

Matrix functions - numerical linear algebra.

Matrix analysis.
  norm         - Matrix or vector norm.
  normest      - Estimate the matrix 2-norm.
...
```

Внимание – данная команда чувствительна к регистру вводимых символов, поэтому следует помнить, что имена всех функций в MATLAB начинаются со строчных букв.

Команда *lookfor* позволяет искать имена функций по ключевому слову, которое сопоставлено авторами MATLAB с определенным набором функций. Например,

```
lookfor inverse
```

в зависимости от состава установленных компонентов выдаст:

```
INVHILB Inverse Hilbert matrix.
ACOS   Inverse cosine.
ACOSH  Inverse hyperbolic cosine.
ACOT   Inverse cotangent.
ACOTH  Inverse hyperbolic cotangent.
ACSC   Inverse cosecant.
ACSCH  Inverse hyperbolic cosecant.
ASEC   Inverse secant.
ASECH  Inverse hyperbolic secant.
ASIN   Inverse sine.
ASINH  Inverse hyperbolic sine.
ATAN   Inverse tangent.
...
```

то есть все имена функций, в описании которых встречается ключевое слово inverse.

Спектральный анализ и фильтрация. Рассмотрим некоторые применения MATLAB для цифровой обработки сигналов, в качестве которых могут выступать, например, выходные данные эксперимента.

Спектральный анализ. Разработка преобразований

Фурье сыграла огромную роль в появлении и развитии ряда новых областей науки и техники. Достаточно отметить, что электротехника переменного тока, электрическая связь и радиосвязь базируются на спектральном представлении сигналов. Ряды Фурье также можно рассматривать как приближение произвольных функций (определенные ограничения в этом известны) тригонометрическими рядами бесконечной длины. При конечной длине рядов получаются наилучшие среднеквадратические приближения. MATLAB содержит функции для выполнения быстрого одномерного и двумерного быстрого дискретного преобразования Фурье.

Прямое преобразование Фурье переводит описание сигнала (функции времени) из временной области в частотную, а обратное преобразование Фурье переводит описание сигнала из частотной области во временную. На этом основаны многочисленные методы фильтрации сигналов.

Для простоты рассмотрим одномерное преобразование Фурье. MATLAB может проводить и многомерное преобразование.

В описанных ниже функциях реализован особый метод *быстрого преобразования Фурье* — Fast Fourier Transform (FFT, или БПФ), позволяющий резко уменьшить число арифметических операций в ходе приведенных выше преобразований. Он особенно эффективен, если число обрабатываемых элементов (отсчетов) составляет 2^m, где m — целое положительное число. Используется следующая функция:

- *fft(X)* — возвращает для вектора X дискретное преобразование Фурье, по возможности используя алгоритм быстрого преобразования Фурье. Если X — матрица, функция *fft* возвращает преобразование Фурье для каждого столбца матрицы;
- *fft(X,n)* — возвращает n-точечное преобразование Фурье. Если длина вектора X меньше n, то недостающие элементы заполняются нулями. Если длина X больше n, то лишние эле-

менты удаляются. Когда X — матрица, длина столбцов корректируется аналогично.

Для иллюстрации применения преобразования Фурье создадим трехчастотный сигнал на фоне сильного шума, создаваемого генератором случайных чисел:

```
t=0:0.0005:1:
x=sin(2*pi*200*t)+0.4*sin(2*pi*150*t)
    +0.4*sin(2*pi*250*t):
y=x+2*randn(size(t));
plot(y(l:100),'b')

Y=fft(y,1024):
Pyy=Y.*conj(Y)/1024;
f=2000*(0:150)/1024;
plot(f,Pyy(l:151)),grid
```

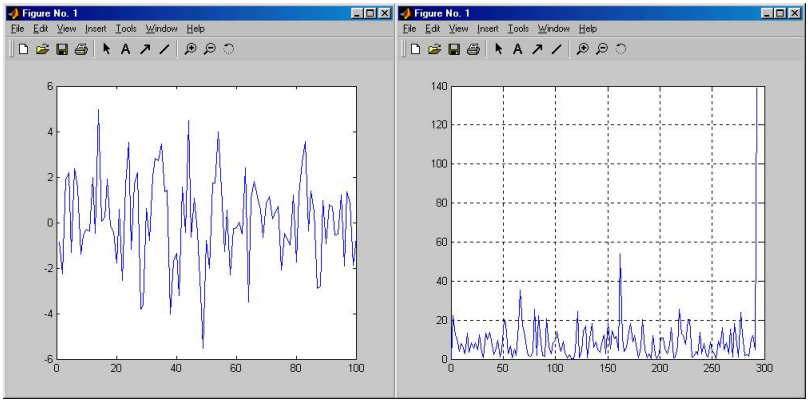


Рис. 147. Форма зашумленного сигнала (слева) и график спектральной плотности этого сигнала

Даже беглого взгляда на рисунок достаточно, чтобы убедиться в том, что спектрограмма сигнала имеет явный пик на средней частоте амплитудно-модулированного сигнала и два боковых пика. Все эти три частотные составляющие сигнала

явно выделяются на общем шумовом фоне. Таким образом, данный пример наглядно иллюстрирует технику обнаружения слабых сигналов на фоне шумов, лежащую в основе работы радиоприемных устройств.

Фильтрация. MATLAB может использоваться для моделирования работы цифровых фильтров. Для обеспечения дискретной одномерной фильтрации используется функция *filter* в следующей форме записи:

- *filter(B,A,X)* — фильтрует одномерный массив данных X, используя дискретный фильтр, описываемый следующим конечноразностным уравнением:

```
a(1)*y(n) = b(1)*x(n) + b(2)*x(n-1) + ... +
b(nb+1)*x(n-nb) -- a(2)*y(n-1) - ... - a(na+1)*y(n-
na)
```

Если a(1) не равно 1, то коэффициенты уравнения нормализуются относительно a (1). Когда X — матрица, функция *filter* оперирует столбцами X. Возможна фильтрация многомерного (размерности *N)* массива. Для этого используется функция *filtern* – где n – размерность фильтрации. Например:

- *filter2(B,X)* — фильтрует данные в двумерном массиве X, используя дискретный фильтр, описанный матрицей B. Результат Y имеет те же размеры, что и X. /4/

5. ПАКЕТ STATISTICA

5.1. Первичный анализ данных. Модульная структура пакета. Вычисление описательных статистик. Вычисление корреляций. Критерий Стьюдента. Построение простейших статистических графиков. Регрессионный и дискриминантный анализ.

Пакет STATISTICA был разработан специально для проведения статистического анализа данных. Первоначально статистика как наука зародилась во Франции в конце 17 века из того, что потом стало называться "теория игр" – это оценка вероятностей выпадения тех или иных комбинаций на игральных костях, рулетке, комбинаций в карточных играх.

Первая версия пакета STATISTICA (для DOS) была разработана фирмой StatSoft, Inc. (США) и сразу заняла лидирующее положение на совсем новом рынке статистического программного обеспечения. В ней впервые реализован так называемый графически-ориентированный подход к анализу данных, который состоит в том, чтобы получать всестороннее визуальное представление данных на всех этапах статистической обработки и на основе этого выбирать следующий шаг обработки.

Статистическая обработка состоит из следующих основных этапов:

- первичный или разведочный анализ данных;
- оценивание зависимостей в данных – регрессия;
- классификация – отнесение объекта к определенной группе – дискриминантный анализ.

Модульная структура пакета. STATISTICA организована по модульному принципу. Это означает, что все методы статистической обработки, реализованные в системе, разбиты на несколько групп – модулей – в соответствии с разделами статистического анализа.

Например, модуль **Basic Statistica and Tables** (Основные статистики и таблицы) содержит основные описательные статистики, методы статистического анализа различных таблиц, разносторонний инструментарий для проведения разведочного анализа данных. Полный список модулей зависит от комплекта установки, по мере надобности можно удалить или добавить модули. Некоторые модули или обновления можно получить из сети Интернет.

Каждый модуль является отдельным Windows-приложением и может работать независимо от остальных модулей системы. /5, 6/

Для вызова модуля используется Переключатель модулей STATISTICA (**STATISTICA Module Switcher**). Его вид можно настроить, однако в современных версиях он практически не нужен, так как по мере надобности различные модули подключаются одним кликом мыши и совершенно незаметно, пользователь может открывать десятки модулей, данные будут автоматически передаваться в нужный модуль, обрабатываться и отображаться.

Ввод данных. Рассмотрим создание файла с данными.

Для создания файла с новыми данными необходимо выбрать меню пункт **File** (Файл) и выбрать в выпавшем окне пункт **New Data** (Новые данные) с изображением чистого белого листа. После этого появится запрос на ввод количества строк – наблюдений (так они называются в системе STATISTICA) и столбцов – переменных. Чтобы присвоить уникальные имена каждому наблюдению или переменной - достаточно сделать двойной щелчок на соответствующем заголовке, тогда появится окно, где можно выбрать несколько параметров, в том числе и название (рис. 148). Там же можно определить формат числовых значений, и т.п. Здесь же можно ввести формулу (нижнее окно рис. 148), по которой будет автоматически заполняться весь столбец. Ввод данных осуществляется обычным путем при помощи клавиатуры.

Описательные статистики. Эти статистики служат для описания самых общих свойств наблюдаемых величин. В системе STATISTICA вычисление описательных статистик осуществляется несколькими щечками мыши.

К числу основных описательных статистик относятся: среднее, выборочная дисперсия, стандартное отклонение, медиана, мода, максимальное или минимальное значение,

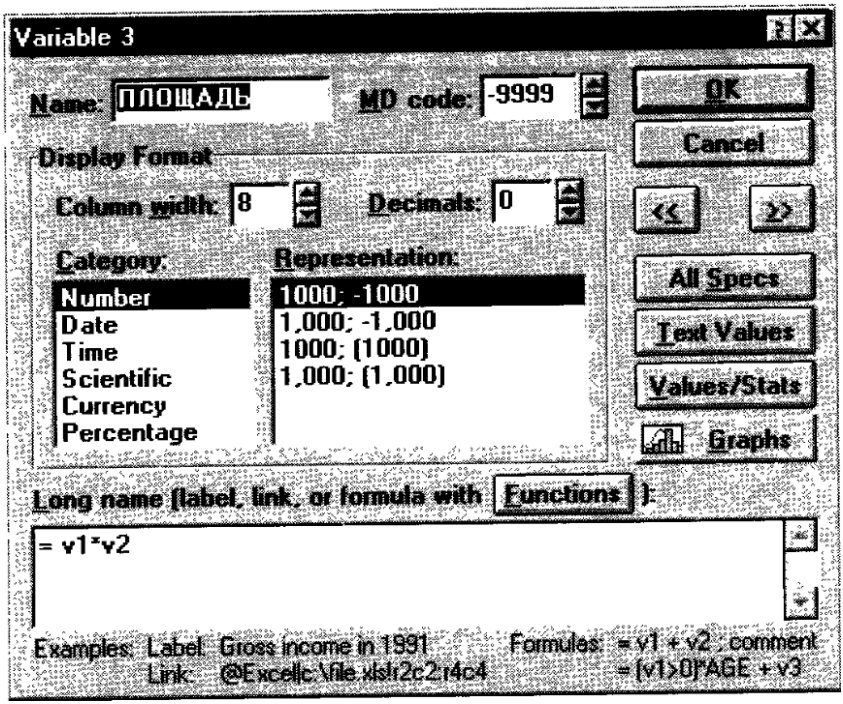


Рис. 148. Окно свойств переменной с формулой для автоматического заполнения значениями.

размах.

Среднее M – или как говорят точнее, оценка среднего, вычисляется как среднее арифметическое наблюдений $M=(X(1)+\ldots +X(N))/N$.

Выборочное среднее является той точкой, сумма отклонений от которой всех наблюдений равна 0. Формально это записывается так: $(M-X(1))+(M-X(2))+\ldots+(M-(X(N))=0$.

Выборочная дисперсия есть состоятельная оценка дисперсии распределения $((X(1)-M(N))^2+\ldots+(X(N)-M(N))^2)/(N-1)$ Дисперсия является мерой изменчивости, вариации признака и представляет собой средний квадрат отклонений случаев от

среднего значения признака. В отличии от других показателей вариации дисперсия может быть разложена на составные части, что позволяет тем самым оценить влияние различных факторов на вариацию признака. Дисперсия - один из существеннейших показателей, характеризующих явление или процесс, один из основных критериев возможности создания достаточно точных моделей.

Стандартное отклонение равно корню квадратному отклонений из суммы квадратов отклонений наблюдаемых значений от среднего, деленной на N-1, где N – число наблюдений.

Мода – это наиболее часто встречающееся значение распределения.

Медиана – это серединное наблюдение в выборке. Если упорядочить значения по возрастанию, то медиана есть значение этого ряда с номером (N-1)/2 – при нечетном N, при четном N - полусумма значений с номерами N/2 и N/2+1

X(1) – минимальное значение в выборке, X(N) - максимальное значение в выборке; разность X(N)- X(1) называется размах.

Вычисление описательных статистик осуществляется нажатием одной кнопкой. Для этого, например, в пакете STATISTICA 6.0 нужно выделить нужный столбец (переменную) и выбрать в главном меню пункт **Statistica** (Статистика) пункт **Quick Basic Stats** (Основные статистики/Таблицы). Появится окно Описательная статистика, где можно выбрать как вкладку **Quick** (Быстрое) - для быстрого анализа заранее выбраныхпуктов, так и **Advanced** (Расширенное) – где можно выбрать нужные пункты, значение которых надо рассчитать. Перечислим некоторые из этих пунктов (количество и название может отличаться в зависимости от версии пакета):

- **Valid N** – истинное число случаев переменной (число случаев без пропусков);
- **Mean** – выборочное среднее;

- **Confid – 95%** - нижняя граница 95% доверительного интервала для среднего (Интервал, в который с вероятностью 0,95 попадает среднее значение признака генеральной совокупности);
- **Confid + 95%** - верхняя граница 95% доверительного интервала для среднего;
- **Sum** – сумма значений переменной;
- **Minimum** – минимальное значение переменной;
- **Maximum** – максимальное значение переменной;
- **Range** – размах (разница между максимумом и минимумом);
- **Variance** – выборочная дисперсия;
- **Std.Dev.** – стандартное отклонение;
- **Std.Err.** – стандартная ошибка;
- **Skewness** – выборочный коэффициент ассиметрии. Асимметрия характеризует степень смещения вариационного ряда относительно среднего значения по величине и направлению. В симметричной кривой коэффициент асимметрии равен нулю. Если правая ветвь кривой, начиная от вершины) больше левой (правосторонняя асимметрия), то коэффициент асимметрии больше нуля. Если левая ветвь кривой больше правой (левосторонняя асимметрия), то коэффициент асимметрии меньше нуля. Асимметрия менее 0,5 считается малой;
- **Std.Err. Skewness** – стандартная ошибка коэффициента ассиметрии;
- **Kurtosis** – выборочный коэффициент эксцесса. Эксцесс характеризует степень концентрации случаев вокруг среднего значения и является своеобразной мерой крутости кривой. В кривой нормального распределения эксцесс равен нулю. Если эксцесс больше нуля, то кривая распределения характеризуется островершинностью, т.е. является более крутой по сравнению с нормальной, а случаи более густо группируются вокруг среднего. При отрицательном эксцессе кривая является более плосковершинной, т.е. более пологой по сравнению с нормальным распределением. Отрицательным пределом вели-

чины эксцесса является число -2, положительного предела – нет;

- **Std.Err. Kurtosis** – стандартная ошибка эксцесса.

Рассмотрим таблицу с исходными данными рис. 149 и таблицу с полученными описательными статистиками рис. 150.

TEXT VALU	ОЛИМПИЙСКИЕ ЧЕМПИОНЫ В БЕГЕ НА 100 М			
	1 ГОД	2 ЧЕМПИОН	3 СТРАНА	4 ВРЕМЯ
1	1896	Бэрк	США	12.0
2	1900	Джервис	США	10.8
3	1904	Хан	США	11.0
4	1906	Хан	США	11.2
5	1908	Уолкер	ЮАР	10.8
6	1912	Крейг	США	10.8
7	1920	Пэддок	США	10.8
8	1924	Абрахамс	Англия	10.6
9	1928	Уияльямс	Канада	10.8
10	1932	Тоулэн	США	10.3
11	1936	Оуэнс	США	10.3
12	1948	Диллард	США	10.3
13	1952	Реминджи	США	10.4
14	1956	Морроу	США	10.5
15	1960	Хари	ФРГ	10.2
16	1964	Хейес	США	10.0
17	1968	Хайнс	США	9.9
18	1972	Борзов	СССР	10.1
19	1976	Кроуфорд	Тринид	10.6

Рис. 149. Исходные данные результатов чемпионов Олимпийских игр в беге на 100 м для последующего анализа.

На самом деле таблица на рис. 150 выглядит в виде одной длинной строки, но для удобства отображения на рисунке она приведена по частям вся.

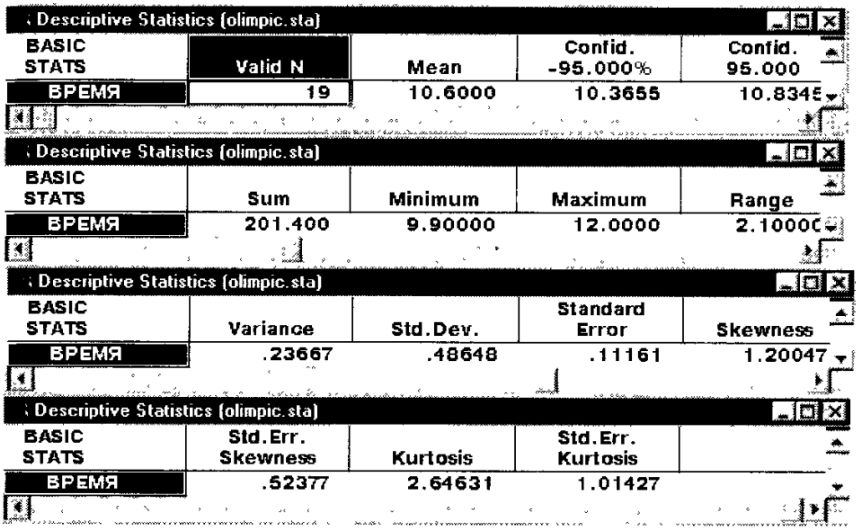


Рис. 150. Таблица с описательными статистиками результатов Олимпийских игр в беге на 100 м

Вычисление корреляций. Корреляция, или, точнее, коэффициент корреляции, является мерой зависимости двух величин. Коэффициент корреляции – это безразмерная величина.

Значение коэффициента корреляции лежит между минус единицей и плюс единицей. Если наблюдается тенденция возрастания одной величины при росте другой, то говорят о положительной коррелированности величин, если наблюдается тенденция увеличения одной величины при уменьшении другой, то говорят о отрицательной коррелированности величин.

Рассмотрим пример вычисления корреляции в системе STATISTICA (рис. 151). Вычислим корреляцию между переменными ДОЛЛАР и МАРКА. Для этого необходимо щелкнуть левой кнопкой мыши на первом случае переменной ДОЛЛАР. Удерживая нажатой левую кнопку мыши выделите

блок данных ДОЛЛАР, МАРКА как показано на рис. 151.

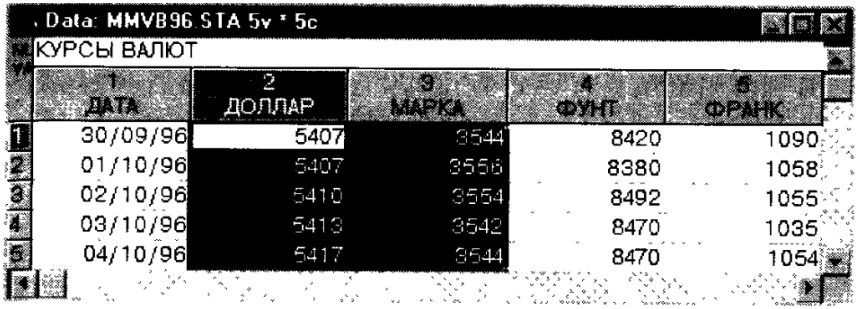


Рис. 151. Файл с выделенным блоком данных ДОЛЛАР – МАРКА.

Затем необходимо выбрать в главном меню пункт **Statistica** (Статистика) пункт **Quick Basic Stats** (Основные статистики/Таблицы). Появится окно, где необходимо выбрать пункт **Correlation matrices** (Корреляционные матрицы). Появится таблица с коэффициентами корреляции (см. рис. 152).

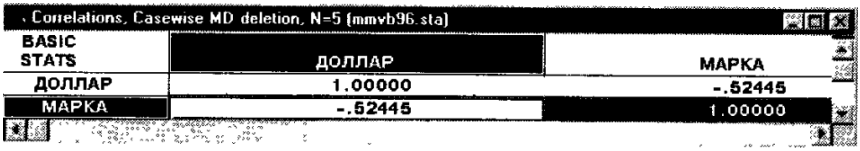


Рис. 152. Таблица с коэффициентами корреляции переменных ДОЛЛАР –МАРКА.

На пересечении строки МАРКА и столбца ДОЛЛАР вы видите значение соответствующего коэффициента корреляции. Корреляция в данном случае отрицательная, так как курс доллара увеличивается, а курс марки уменьшается три дня подряд.

Таким же самым способом можно вычислять и более

объемные матрицы корреляционных коэффициентов, выделяя большие блоки данных, как показано на рис. 153.

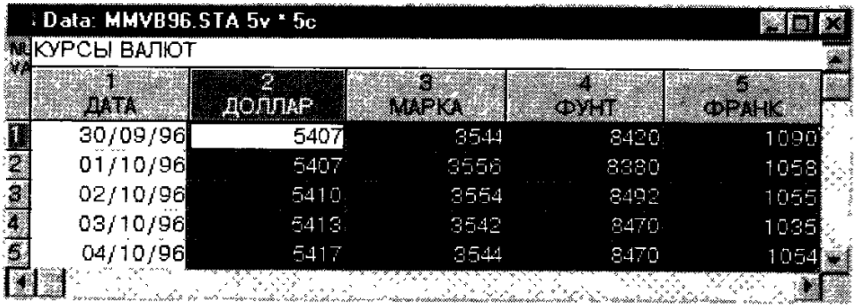


Рис. 153. Файл с выделенным блоком данных ДОЛЛАР – МАРКА – ФУНТ – ФРАНК.

Последовательность действий аналогичная, и полученная матрица корреляционных коэффициентов будет выглядеть следующим образом (см. рис. 154):

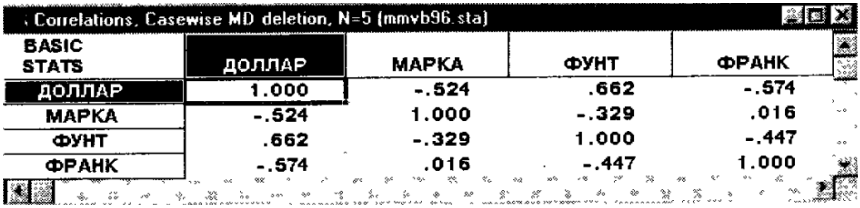


Рис. 154. Таблица с коэффициентами корреляции переменных ДОЛЛАР –МАРКА – ФУНТ – ФРАНК.

На пересечении столбцов и строк стоят значения коэффициентов корреляций между переменными. Заметьте, матрица симметричная, это отражает тот факт, что коэффициент корреляции между переменными, например, ДОЛЛАР и МАРКА, естественно, равен коэффициент корреляции переменных МАРКА и ДОЛЛАР.

Построение простейших статистических графиков. Покажем, как строятся простейшие статистические графики: диаграмма рассеяния и гистограмма.

Рассмотрим построение диаграммы рассеяния на примере таблицы с данными результатов забегов Олимпийских чемпионов на 100 м на рис. 149 по шагам.

Шаг 1. Выделяем при помощи щелчка левой кнопкой мыши какое-нибудь значением переменной ГОД в таблице.

Шаг 2. Нажмите кнопку **Quick Stat Graphs** (Быстрые Стат-графики) (в системе STATISTICA 6.0 нужно выбрать пункт главного меню Графы) для открытия для данной переменной меню стандартных, наиболее часто используемых видов статистических графиков.

В выпадающем меню среди большого многообразия предложенных графиков необходимо выбрать подпункт **Scatterplot by** (Диаграмма рассеяния) (Точечные вычерчивания в системе STATISTICA 6.0) и далее пункт Regular (Регулярный). На экране появится диалоговое окно: **Select one variable** (Выбрать одну переменную) (рис. 155):

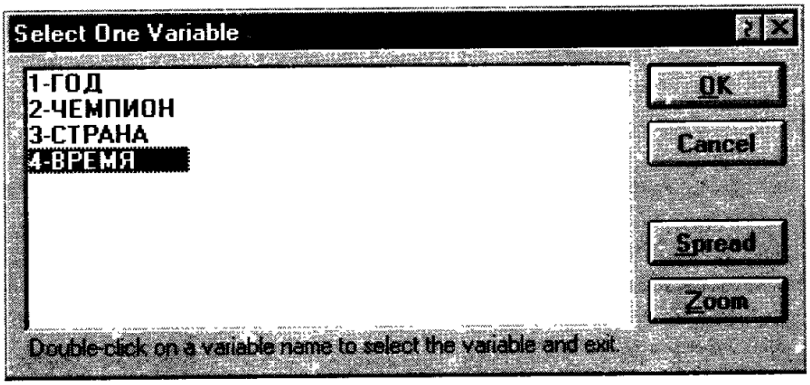


Рис. 155. Окно выбора переменной для диаграммы рассеяния.

Шаг 3. Выделите мышью имя переменной ВРЕМЯ и

нажмите кнопку ОК. график появится на экране (рис. 156)

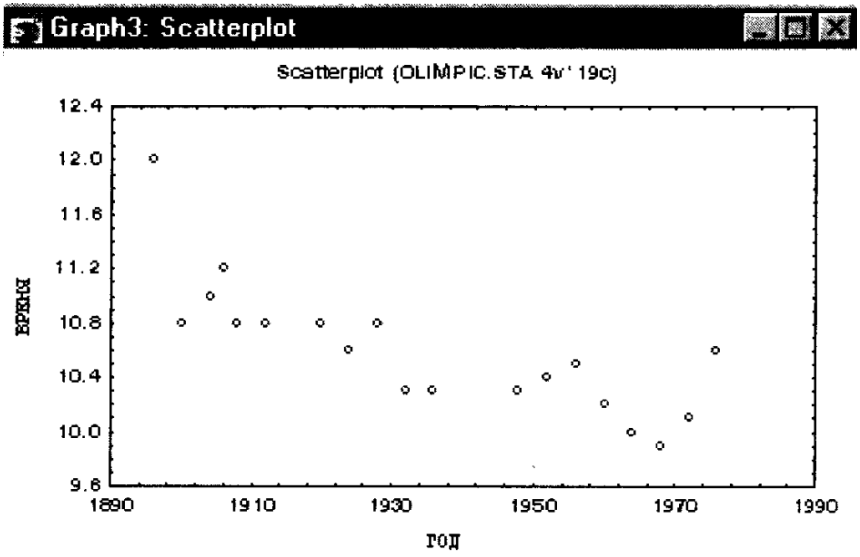


Рис. 156. Диаграмма рассеяния результатов олимпийских чемпионов в беге на 100 м.

Как видно из данной диаграммы, имеется четко прослеживаемая тенденция улучшения результатов. Чтобы сохранить данный график, необходимо нажать **CTRL+S** и ввести в появившемся окне имя файла с графиком.

Гистограмма является чрезвычайно популярным статистическим графиком, она позволяет в удобном виде представить частоту попадания величин в определенные интервалы. Особенно полезны гистограммы при большом количестве наблюдений (больше 100). Рассмотрим процедуру построения гистограммы.

Шаг 1. В таблице предыдущего примера выделите какое-нибудь значение переменной ВРЕМЯ.

Шаг 2. Нажмите кнопку **Quick Stat Graphs** (Быстрые Стат-графики) (в системе STATISTICA 6.0 нужно выбрать

пункт главного меню Графы) для открытия для данной переменной меню стандартных, наиболее часто используемых видов статистических графиков.

В выпадающем меню среди большого многообразия предложенных графиков необходимо выбрать подпункт **Histogram of ВРЕМЯ** (Гистограмма переменной ВРЕМЯ) (Гистограммы в системе STATISTICA 6.0) и далее пункт **Regular** (Регулярный). На экране появится гистограмма:

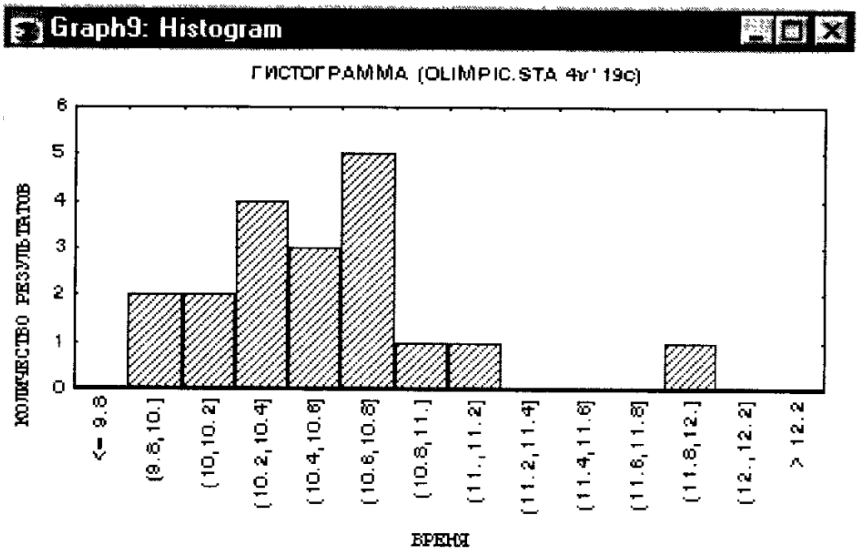


Рис. 157. Гистограмма значений переменной ВРЕМЯ чемпионов в беге на 100 м.

Чтобы сохранить данную гистограмму, необходимо нажать **CTRL+S** и ввести в появившемся окне имя файла с графиком.

Данная гистограмма построена следующим образом: диапазон изменения переменной ВРЕМЯ разбит на равные полуинтервалы длиной 0,2 секунды (левый конец интервала исключается, правый включается). По оси Y отложено число

значений переменной ВРЕМЯ, попавших в определенные интервалы. Например, в интервал (10.0, 10.2] попало 2 значения, в интервал (11.4, 11.6] не попало ни одного значения. Наибольшее число значений переменной ВРЕМЯ лежит в диапазоне (10.6, 10.8] – 5 значений.

Критерий Стьюдента. Процедура **t-test for independent samples** (t-критерий для независимых выборок) используется для установления достоверной статистической разницы между средними значениями выборок на основе t-критерия Стьюдента.

Рассмотрим следующий пример (взят из /7/): имеются результаты определения водопроницаемости почвы на площадках с различным характером напочвенного покрова. Создадим файл с данными с четырьмя переменными (рис. 158):

Data: VODA.STA 4v * 10c

NUM VAL	Водопроницаемость почв (мм/ми) в зависимости от х			
	1 VAR1	2 VAR2	3 VAR3	4 VAR4
1	303,0	78,7	53,5	67,9
2	238,0	82,0	68,0	105,3
3	303,0	58,1	38,8	149,3
4	238,0	97,1	49,5	138,9
5	303,0	73,0	70,4	45,5
6	200,0	142,9	40,5	98,0
7	400,0	55,6	25,1	61,3
8	238,0	108,7	12,2	75,8
9	263,0	69,9	33,6	71,4
10	303,0	120,5	28,3	35,7

Рис. 158. Окно с файлом данных "Водопроницаемость почвы (мм/мин) в зависимости от характера напочвенного покрова"

В данном случае: **VAR1** - Водопроницаемость на площадке 1 (Мертвый покров, лесная подстилка 2.5см), **VAR2** - Водопроницаемость на площадке 2 (Травяной покров, проективное покрытие 40-50%, задернение 10%), **VAR3** - Водопроницаемость на площадке 3 (Травяной покров, проективное покрытие 100%, задернение 70%), **VAR4** - Водопроницаемость на площадке 4 (Травяной покров, проективное покрытие 30-40%, задернения нет).

Влияет ли характер напочвенного покрова на водопроницаемость почвы с ее поверхности? Воспользуемся процедурой **t-test for independent samples** для расчета средних величин водопроницаемости по вариантам опыта и одновременно проверим достоверность различий между средними значениями.

В окне "**T-Test for independent samples (Groups)**" (рис. 159) в опции **Input file** следует указать тип файла с данными:

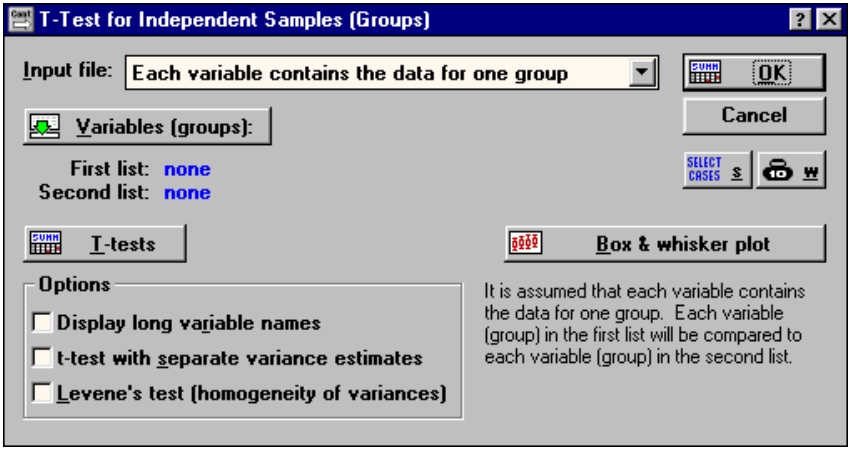


Рис. 159. Окно "**T-Test for independent samples (Groups)**"

- **One record percase (use a grouping variable)** - одна запись на случай (используя группирующую переменную);
- **Each variable contains the data for one group** - каждая переменная содержит данные одной группы.

Использующийся нами файл данных (рис. 155) относится ко второму типу (**Each variable contains the data for one group**).

При помощи кнопки **Variables** выбираются переменные для по парного сравнения. При этом должны быть выбраны переменные в обоих списках. Чтобы сравнить попарно сразу все варианты опыта друг с другом, следует выбрать переменные так, как показано на рис. 160.

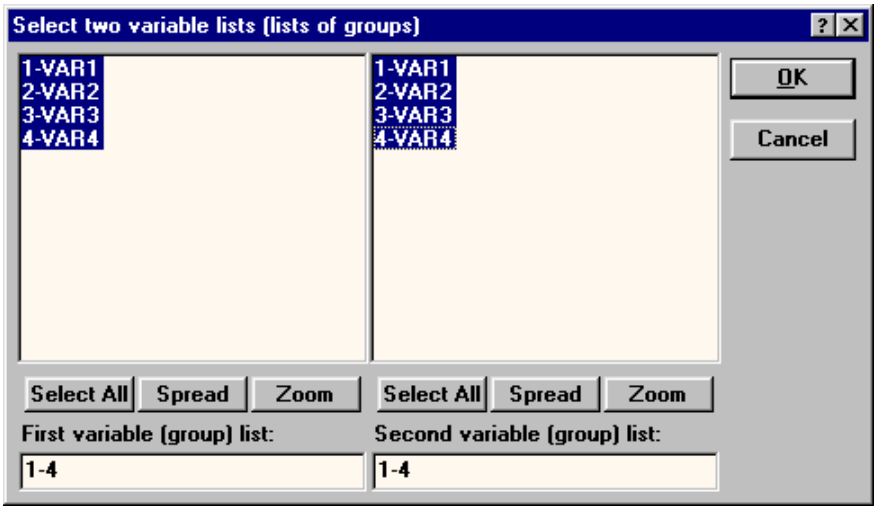
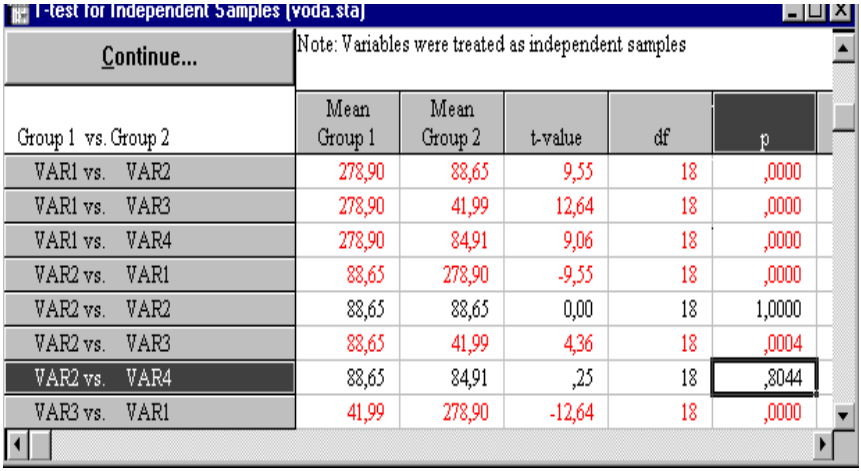


Рис. 160. Выбор переменных для попарного сравнения.

После нажатия на кнопку **OK** или **T-test** на экране появляется таблица с результатами сравнения по t-критерию. Фрагмент окна с результатами проведения процедуры приводится на рис. 161.


T-test for Independent Samples (voda.sta)

Continue...

Note: Variables were treated as independent samples

Group 1 vs. Group 2	Mean Group 1	Mean Group 2	t-value	df	p
VAR1 vs. VAR2	278,90	88,65	9,55	18	,0000
VAR1 vs. VAR3	278,90	41,99	12,64	18	,0000
VAR1 vs. VAR4	278,90	84,91	9,06	18	,0000
VAR2 vs. VAR1	88,65	278,90	-9,55	18	,0000
VAR2 vs. VAR2	88,65	88,65	0,00	18	1,0000
VAR2 vs. VAR3	88,65	41,99	4,36	18	,0004
VAR2 vs. VAR4	88,65	84,91	,25	18	,8044
VAR3 vs. VAR1	41,99	278,90	-12,64	18	,0000

Рис. 161. Результаты проведения процедуры "**t-test for independent samples**".

Согласно нулевой гипотезы между средними значениями водопроницаемости достоверного различия нет, т.е. две выборки однородны и представляют одну генеральную совокупность. Если вероятность нулевой гипотезы (p) меньше 5% (т.е. $p < 0,05$), то с вероятность 0,95 нулевую гипотезу можно отбросить. По парное сравнение средних величин водопроницаемости показало достоверное различие между всеми вариантами опыта, кроме вариантов 2 и 4. Нулевую гипотезу в последнем случае отбросить нельзя, так как ее вероятность чересчур высока (p=0,804). /7/

Это означает, что несмотря на формальную разницу исходных условий экспериментов 2 и 4, результаты, полученные в ходе экспериментов, в общем сходны. Следовательно, или различия в исходных условиях незначительны (как в данном случае), либо исходные факторы имеют сходный механизм влияния на конечный результат. Такой анализ часто проводится с целью заменить в технологических процессах дорогостоящее либо опасное вещество или действующий фактор бо-

лее дешевым или безопасным со сходным механизмом действия на результат.

Регрессионный анализ. Типичной практической задачей является определения зависимостей в системе данных. Пусть X – независимая переменная (предиктор), переменная Y - зависимая переменная (отклик). Данная терминология связана с тем, что мы хотим определить именно зависимость Y от X или предсказать, какими будут значения Y при данных значениях X.

Значение переменной X в i-ом опыте будем обозначать X(i), соответствующее значение величины Y обозначим через Y(i),.

Итак, наблюдая значения независимой переменной X(i) и соответствующее им значениея зависимой Y(i) $0<i\leq n$, хотим оценить зависимость Y от X. В статистике такие задачи решаются в рамках регрессионной модели. Рассмотрим самую простую регрессионную модель – линейную. Однако в рамках такой модели могут быть решены самые разнообразные практические задачи. Задачи нелинейной регрессии решаются по аналогичному алгоритму.

Регрессионный анализ в системе STATISTICA проводится в модуле Множественная регрессия (Multiple regression).

Всё начинается с описание регрессионной модели, в рамках которой будет исследоваться зависимость Y от X.

Примем, что наблюдаемые величины связаны между собой регрессионной зависимость вида:

$$Y(i) = B1\times X(i) + B0 + e(i),\ 0<i\leq n$$

где: B1, B0 – неизвестные константы,
e(i) – ненаблюдаемые случайные величины со средним 0 (являющиеся несмещенными) и неизвестной дисперсией, не меняющейся от опыта к опыту.

Иногда случайные величины e(i) называют ошибками наблю-

дения. Относительно e(i) предполагается, что они не коррелированны в различных опытах. Кроме того, часто предполагается, что ошибки имеют нормальное распределение, в этом случае некоррелированность влечет независимость.

Можно решать и более общие линейные модели, с несколькими независимыми переменными.

Затем производится постановка задачи. Общая задача состоит в том, чтобы по наблюдениям (X(1), Y(1)), …(X(n), Y(n)):

- оценить параметры модели B1, B0 наилучшим образом;
- построить доверительные интервалы для B1, B0;
- проверить гипотезу о значимости регрессии;
- оценить степень адекватности модели и т.д.

Рассмотрим основной алгоритм решения задачи на примере из финансовой сферы. Пусть имеется заранее созданная таблица с данными по курсам продажи и покупки акций неких компаний (рис. 162):

ПОКУПКА-ПРОДАЖА АКЦИЙ

	1 ДАТА	2 ИРКУТ1	3 ИРКУТ2	4 КРСНЯРС1	5 КРАСНЯР2
1	22/10/96	.0966	.1103	.1345	.1758
2	23/10/96	.0980	.1110	.1362	.1792
3	24/10/96	.0986	.1117	.1362	.1808
4	25/10/96	.0992	.1121	1362	.1808
5	28/10/96	.0988	.1115	.1362	1808
6	29/10/96	.0992	.1114	.1372	.1858
7	30/10/96	.0979	.1121	.1372	.1858
8	31/10/96	.0981	.1149	.1378	.1867
9	01/11/96	.0972	.1118	.1409	.1800
10	04/11/96	.0974	.1118	.1399	.1800
11	05/11/96	.0981	.1119	.1350	.1800
12	06/11/96	.0990	.1127	.1322	.1800
13	10/11/96	.0986	.1127	.1280	.1825
14	11/11/96	.0987	.1129	.1273	.1860
15	12/11/96	.0996	.1137	.1236	.1717
16	13/11/96	.1000	.1145	.1239	.1808
17	14/11/96	.1003	.1156	.1258	.1875
18	15/11/96	.1016	.1163	.1288	.1856
19	18/11/96	.1018	.1161	.1347	.1983
20	19/11/96	.1039	1176	.1386	.2206
21	20/11/96	.1068	.1236	.1532	.2228

Рис. 162. Таблица с данными по акциям.

ДАТА – дата сделки;
ИРКУТ1 – цена покупки акции ИРКУТСКЭНЕРГО
ИРКУТ2 – цена продажи акции ИРКУТСКЭНЕРГО
КРАСНОЯР1 - цена покупки акции КРАСНОЯРСКЭНЕРГО
КРАСНОЯР2 - цена продажи акции КРАСНОЯРСКЭНЕРГО
Проведем анализ в модуле Множественная регрессия. Рассмотрим акции ИРКУТСКЭНЕРГО. Установим, как цена покупок связана с ценой продаж.

Шаг 1. Из переключателей модулей откройте модуль Множественная регрессия - **Multiple regression.**

Шаг 2. На экране появится стартовая панель модуля (рис. 163).

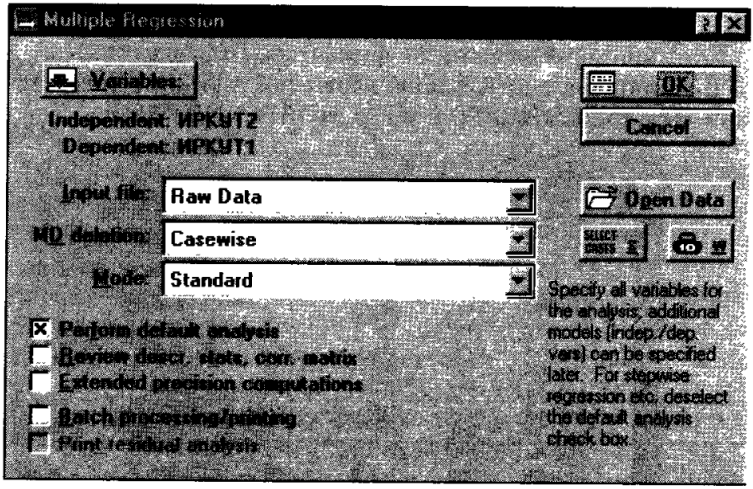


Рис. 163. Стартовая панель модуля Множественная регрессия.

Нажмите кнопку открыть данные (**Open data**) и откройте имеющийся файл с данными по акциям. Далее выберите переменные для анализа с помощью кнопки Переменные (**Variables**), находящейся в левом углу панели. После того, как

кнопка будет нажата, появится диалоговое окно Выбрать списки зависимых и независимых переменных – **Select dependent and independent variable list** – появится на экране (рис. 164).

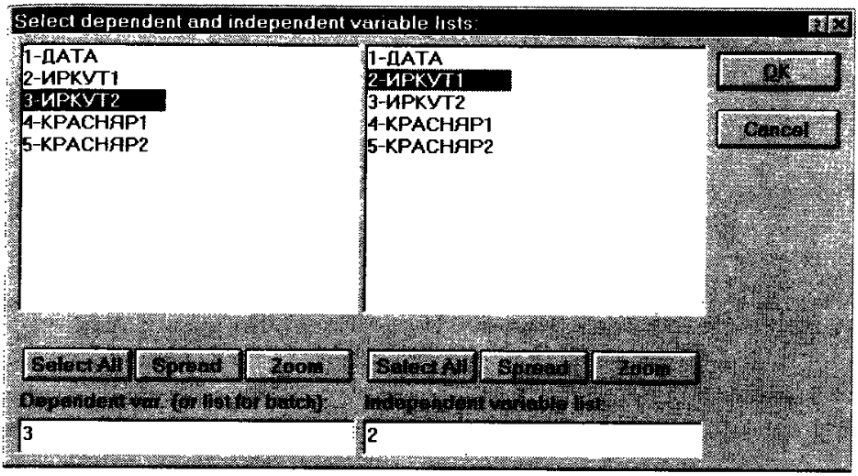


Рис. 164. Окно выбора переменных для анализа.

Выделив имя переменной в левой части окна, выберите зависимую переменную. В правой части окна выберите независимую переменную.

В данном примере независимой переменной является ИРКУТ1, зависимой – ИРКУТ2. Выделите их и нажмите кнопку ОК. Переменные для анализа выбраны и вы вернетесь в стартовое окно анализа (рис. 163). В данном случае никаких других установок делать не нужно. Нажмите ОК в вернем правом углу панели.

Шаг 3. На экране появится диалоговое окно Построение модели – **Model Definition** (рис. 165). В данном случае выберите стандартный метод в опции Метод (**Method**): Стандартный (**Standart**). Нажмите ОК. Программа произведет оценивание параметров модели стандартным методом, и на экране

появится окно результатов (рис. 166).

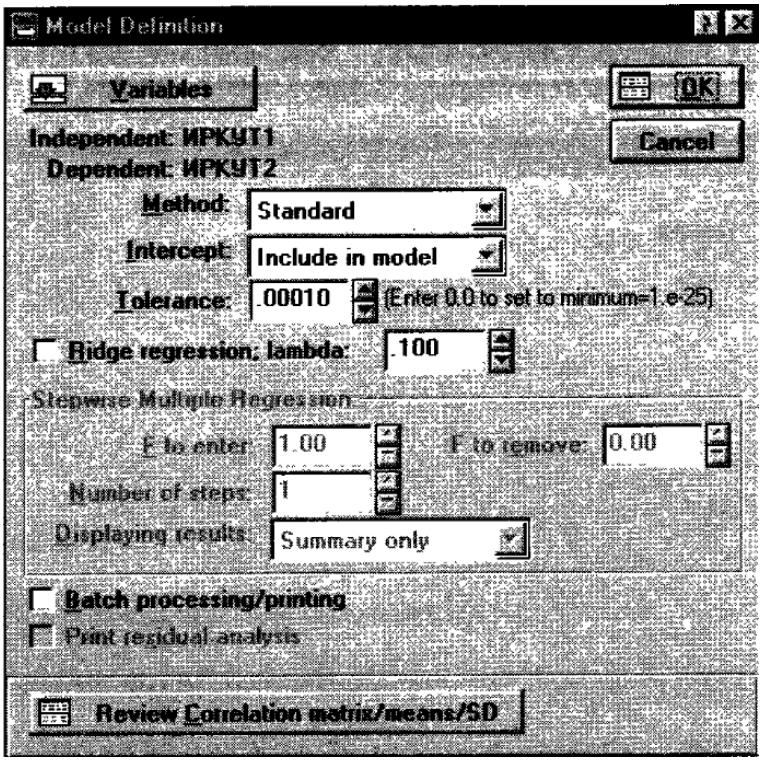


Рис. 165. Окно построения модели в модуле Множественная регрессия.

Шаг 4. В диалоговом окне Результаты множественной регрессии – **Multiple Regression Result** посмотрите результаты оценивания (рис. 166). Результаты можно посмотреть в численном и графическом виде. Окно имеет следующую структуру: верх окна – информационный. Он состоит из двух частей: в первой части содержится основная информация о результатах оценивания, во второй высвечиваются значимые регрессионные коэффициенты. Внизу окна находятся функ-

циональные кнопки, позволяющие всесторонне изучить результаты анализа.

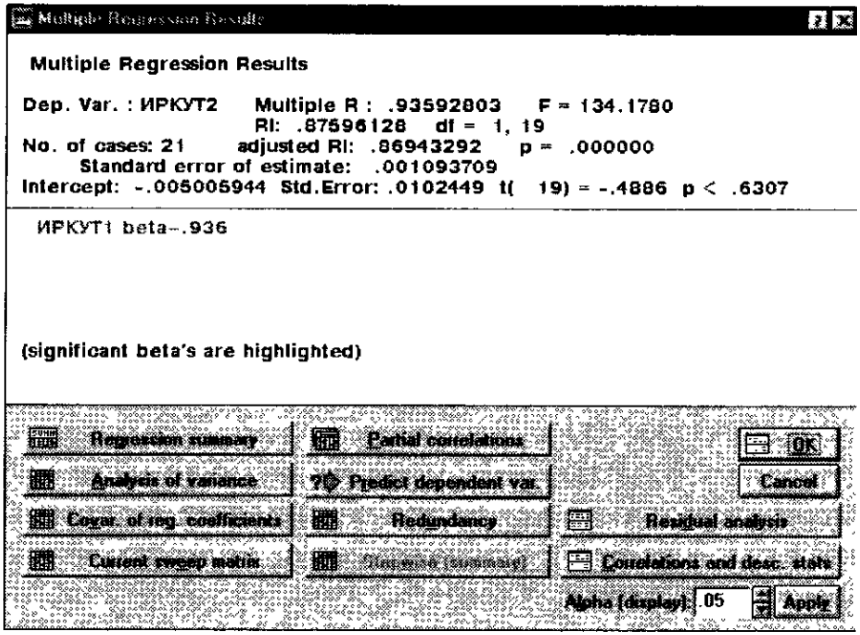


Рис. 166. Окно оценивания параметров в примере с продажей акций.

Рассмотрим подробно информационную часть окна. В ней содержатся сведения о результатах анализа. А именно:
- **Dep. Var.** (имя зависимой переменной). В данном случае – ИРКУТ2.
- **No. Of Cases** (Число случаев, по которым построена регрессия). В примере это число равно 21.
- **Multiple R** (Коэффициент множественной корреляции).
- **R-square – R1** (Квадрат коэффициента множественной корреляции), обычно называемы коэффициентом детерминации. Он является чрезвычайно важной характеристикой, он пока-

зывает долю общего разброса (относительно выборочного среднего зависимой переменной), которая объясняется построенной регрессией.
- **Adjusted R-square: adjusted R1** (Скорректированный коэффициент детерминации),и определяемый как:

Adjusted R-square = 1 – (1 - R-square)×(n/(n – p)),

где: n – число наблюдений в модели;
p – число параметров модели (число независимых переменных плюс 1, так как включен свободный член).

- **Std. Error of estimate** (стандартная ошибка оценки). Эта статистика является мерой рассеяния наблюдаемых значений относительно регрессионной прямой.
- **Intercept** (Оценка свободного члена регрессии). Значение коэффициента B0 в уравнении регрессии.
- **Std. Error** (Стандартная ошибка оценки свободного члена). Стандартная ошибка коэффициента B0 в уравнении регрессии.
- t**(dt) and p-value** (Значение t-критерия и уровень p). T- критерий используется для проверки гипотезы о равенстве 0 свободного члена регрессии.
- **F** – значение F-критерия.
- **df** – число степеней свободы F-критерия.
- **p** – уровень значимости.

В информационной части следует обратить внимание на значение коэффициента детерминации, его значение лежит в диапазоне от 0 до 1. В нашем примере R1 = 0,86. Это означает, что построенная регрессия объясняет более 86% разброса значений переменной ИРКУТ2 относительно среднего.

Далее посмотрим на значение F-критерия и уровень значимости p. F-критерий используется для проверки гипотезы о значимости регрессии. В данном случае, что между зависимой

переменной ИРКУТ2 и независимой переменной ИРКУТ1 нет линейной зависимости, то есть B1=0, против альтернативы B1≠0. В данном случае мы имеем большое значение F-критерия – 134,1780 и даваемый уровень значимости p=0,0000, показывающие, что построенная регрессия высоко значима.

Рассмотрим вторую часть информационного окна. В этой части система сама говорит о значимых регрессионных коэффициентах, высвечивая строку ИРКУТ1 beta = 0.936 и на пояснение "значимые beta высвечены – **significant beta's are highlignted**". Отметим, что в данном случае **beta** есть коэффициент B1 при независимой переменной ИРКУТ1.

Перейдем в функциональную часть окна результатов. Прежде всего нажмите кнопку "Итоговый результат регрессии – **regression summary**". На экране появится таблица вывода – **spreadsheet**, в которой представлены итоговые результаты оценивания регрессионной модели.

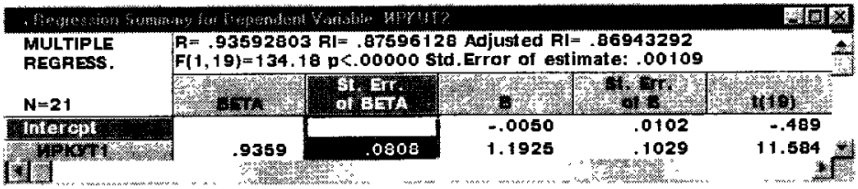


Рис. 167. Итоговая таблица регрессии.

Эта стандартная таблица вывода регрессионного анализа. В первом столбце даны значения коэффициентов **beta** - стандартизованные коэффициенты регрессионного уравнения, во втором – стандартные ошибки **beta**, в третьем – точечные оценки параметров модели:

Свободный член B0= - 0,0050.

Коэффициент B1 (при независимой переменной ИРКУТ1) = 1,1925.

Далее, стандартные ошибки для B0 и B1, значение t-критерия.

Из данной таблицы видно, что оценочная модель имеет вид:

ИРКУТ2 = 1,1925*ИРКУТ1 – 0,0050

На графике данные с подогнанной прямой имеют вид:

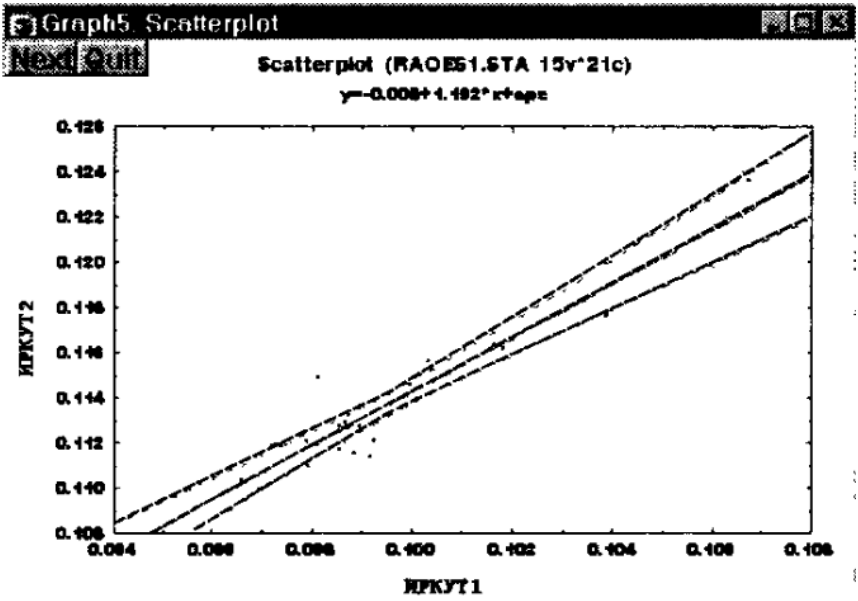


Рис. 168. Линейная регрессия для данных: ИРКУТ1 и ИРКУТ2.

Шаг 5. Оценка адекватности модели. Важным элементом анализа является оценка адекватности модели.

После того, как доказана адекватность модели, полученные результаты можно уверенно использовать при дальнейших расчетах.

Анализ адекватности основывается на анализе остатков. Остатки представляют собой разности между наблюдаемыми значениями и модельными, то есть значениями, подсчитанными по модели с оценочными параметрами.

В STATISTICA в модуле Множественная регрессия имеется специальное диалоговое окно, в котором проводится всесторонний анализ остатков.

При нажатии кнопки Анализ остатков – **Residual Analysis** появится следующее диалоговое окно:

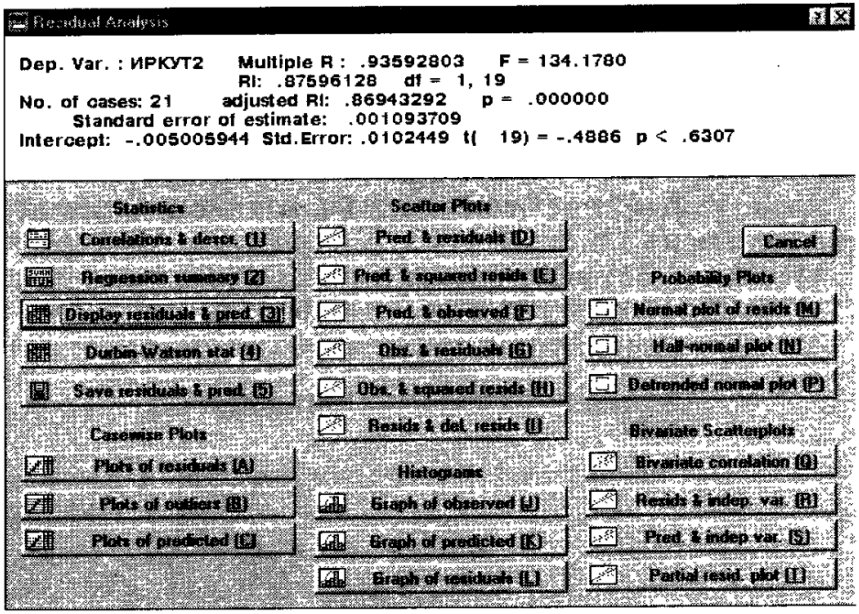


Рис. 169. Диалоговое окно Анализ остатков в модуле Множественная регрессия.

При нажатии, например, кнопки **Obs&residuals**, на экране появится график (рис. 168), который говорит о достаточной адекватности модели.

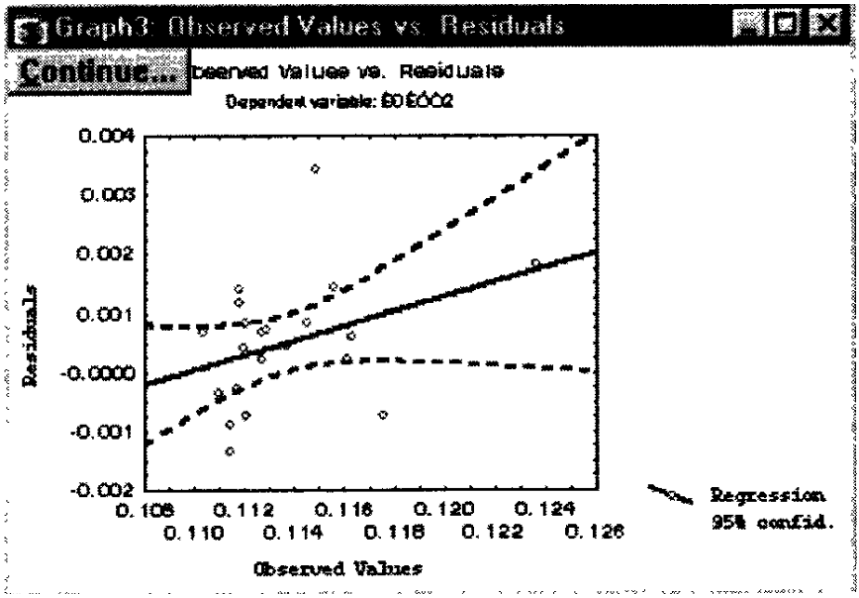


Рис. 170. График "Наблюдаемые переменные – остатки".

Иногда, когда предполагаемые зависимости заведомо не являются линейными в исходном виде, проводят преобразования, приводящие зависимости к линейному виду. /6/ Например, хорошо известны из курсов физики примеры, когда для определения энергии активации, например, примеси, строят зависимость ln(N)=f(1000/T) – где N – концентрация свободных носителей и T – температура образца. В таких координатах зависимость будет иметь вид, близкий к линейному и определить энергию активации как угол наклона можно геометрическим путем.

Дискриминантный анализ. Данный анализ является одним из видов многомерного статистического анализа. Цель его состоит в том, чтобы на основе измерения различных характеристик (признаков, параметров) объекта классифицировать его, то есть отнести к одной из нескольких групп (клас-

сов) оптимальным образом. Под оптимальным образом понимается снижение до минимума вероятности ложной классификации. Типичные области применения дискриминантного анализа – медицина, управление производством, экономика, геология, контроль качества.

Близкими к методам дискриминантного анализа являются методы дисперсионного анализа, кластерного и факторного анализов, а так же методов множественной регрессии.

Дискриминантный анализ работает при выполнении ряда следующих предположений:

Предположение о том, что измеряемые характеристики объекта имеют нормальное распределение. Это следует проверять при помощи специальных опций (проверок на нормальность), встроенных в модуль. Умеренные отклонения от этого предположения не являются фатальными.

Предположение об однородности дисперсий и ковариаций наблюдаемых переменных в разных классах. Если это условие не выполняется – то и проводить анализ не нужно – и так всё ясно, объекты очевидно разные. Умеренные отклонения допустимы.

Самая важная задача – это интерпретация результатов анализа. Рекомендуется проверять правильность результатов анализа на практике.

Рассмотрим с помощью STATISTICA широко известный пример Фишера об анализе цветков растения ириса.

Задача состоит в том, чтобы по результатам измерения длины и ширины чашелистиков и лепестков ириса отнести ирис к одному из трех сортов: SETOSA, VERSICOL, VIRGINIC.

Пусть имеется файл с набором измерений 150 цветков ириса, по 50 каждого сорта.

Шаг 1. Из переключателей модулей STATISTICA откроем стартовую панель модуля **Discriminant function analysis** (Дискриминантный функциональный анализ).

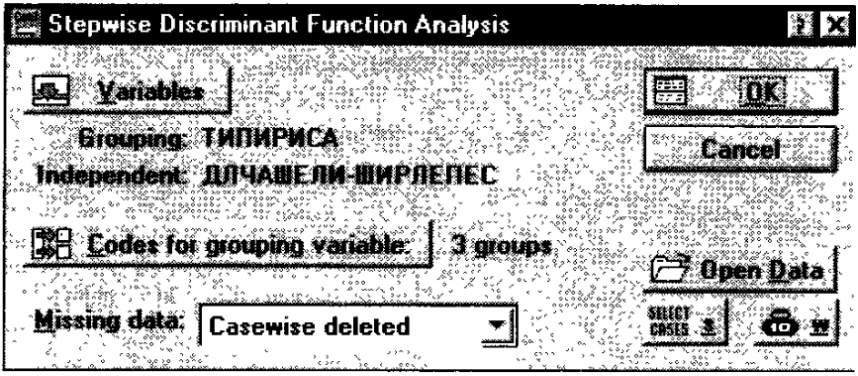


Рис. 171. Стартовая панель модуля Дискриминантный анализ.

Шаг 2. Нажмите кнопку **Open Data** (Открыть данные) и выберите заранее набранный файл с данными.

Data: IRISDAT.STA 5v * 150c

Fisher (1936) iris data: length & width of sepals and petals, 3 types of Iris

TEXT VALUES	1 ДЛЧАШЕЛИ	2 ШИРЧАШЕЛ	3 ДЛЛЕПЕСТ	4 ШИРЛЕПЕС	5 ТИПИРИСА
1	5.0	3 3	1.4	2	SETOSA
2	6.4	2.8	5 6	2.2	VIRGINIC
3	6.5	2.8	4.6	1.5	VERSICOL
4	6.7	3.1	5.6	2.4	VIRGINIC
5	6.3	2.8	5.1	1.5	VIRGINIC
6	4 6	3 4	1.4	.3	SETOSA
7	6.9	3.1	5 1	2.3	VIRGINIC
8	6.2	2.2	4.5	1.5	VERSICOL
9	5.9	3.2	4.8	1.8	VERSICOL
10	4.6	3.6	1.0	.2	SETOSA

Рис. 172. Файл с данными по ирисам.

Шаг 3. Нажмите кнопку **Variables** (Переменные) и выбе-

рите переменные для анализа. В качестве Группирующей переменной – **Grouping variable** выберите переменную IRIS-TYPE – ТИПИРИСА.

В качестве Независимых переменных – **Independent variables** выберите переменные ДЛЧАШЕЛИ, ШИРЧАШЕЛ, ДЛЛЕПЕСТ, ШИРЛЕПЕС.

Шаг 4. Нажмите кнопку ОК и откройте диалоговое окно **Model Definition** (Определение модели).

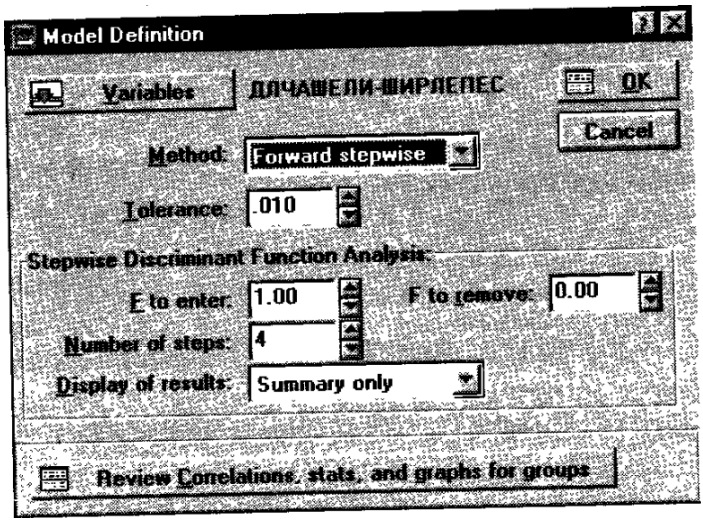


Рис. 173. Окно определения модели дискриминантного анализа.

В данном окне можно выбрать метод анализа, задать начальные установки в этих методах, а так же посмотреть описательные статистики для проверки предположений относительно переменных. Можно выбрать следующие методы:

Standart – стандартный;

Forward stepwise – пошаговый включения;

Backward stepwise – пошаговый исключения.

В опции **Tolerance** (Толерантность) задается нижняя граница

толерантности, переменные с меньшей толерантностью в модель не включаются.

Для пошаговых методов имеется специальная группа опций: **Stepwise Discriminat Function Analysis** (Пошаговый дискриминантный анализ), где задаются:

F to enter - значение статистики F для включения переменной в модель;

F to remove - значение статистики F для исключения переменной из модели;

Number of steps – число шагов;

Display of results отображение результатов (**Summary only** – только итоговые, **At each step** – на каждом шаге).

Расположенная ниже кнопка открывает дополнительное окно, в котором можно посмотреть описательные статистики для переменных.

Установите все поля так, как показано на рисунке и нажмите ОК

Шаг 5. Рассмотрим итоги в окне "Результаты дискриминантного анализа" (рис. 174).

Информационная часть окна сообщает, что использован

Stepwise analysis – пошаговый анализ, **Step 4** (**Final step**) – Шаг 4 (Заключительный шаг);

Number of variables in the model – Число переменных в модели 4;

Wilks lambda – значение лямбды Уилкса: 0,0234;

Approx F(4.292) = 199.1454 – приближенные значение F-статистики, связанной с лямбдой Уилкса;

P – уровень значимости F-критерия для значение 199.1454.

Значение статистики Уилкса показывает качество дискриминации и лежит в диапазоне [0,1]. Чем ближе к 0, тем выше качество дискриминации - вероятность того, что дискриминация проведена правильно.

Нажмите кнопку **Variables in model** (Переменные, включенные в модель).

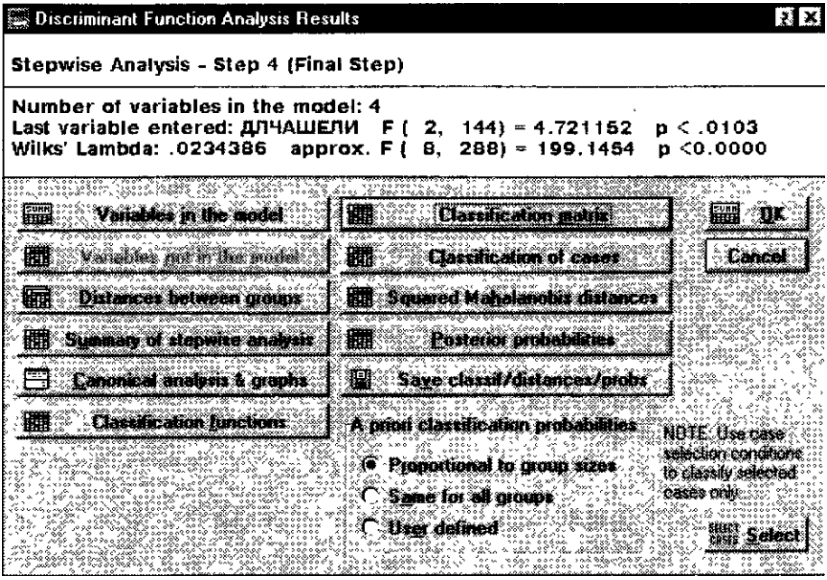
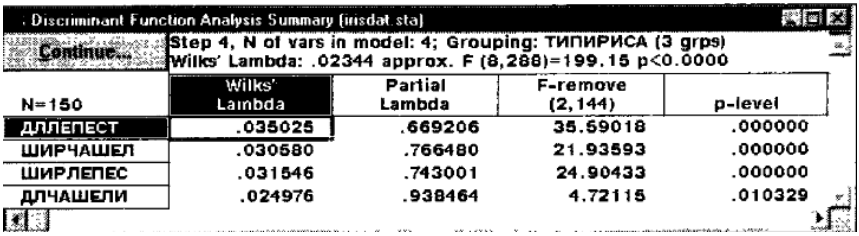


Рис. 174. Окно результатов дискриминантного анализа.

В появившемся окне отображаются полученные результаты по всем переменным модели:



N=150	Wilks' Lambda	Partial Lambda	F-remove (2,144)	p-level
ДЛЛЕПЕСТ	.035025	.669206	35.59018	.000000
ШИРЧАШЕЛ	.030580	.766480	21.93593	.000000
ШИРЛЕПЕС	.031546	.743001	24.90433	.000000
ДЛЧАШЕЛИ	.024976	.938464	4.72115	.010329

Рис. 175. Результаты анализа по всем переменным модели.

Посмотрим разделение групп на графике. Для этого нажмите кнопку **Canonical analysis & graphs** (Канонический анализ и

графики). В появившемся окне нажмите кнопку **Scatterplot of canonical scores** (Диаграмма рассеяния канонических значений). На экране появится следующий график:

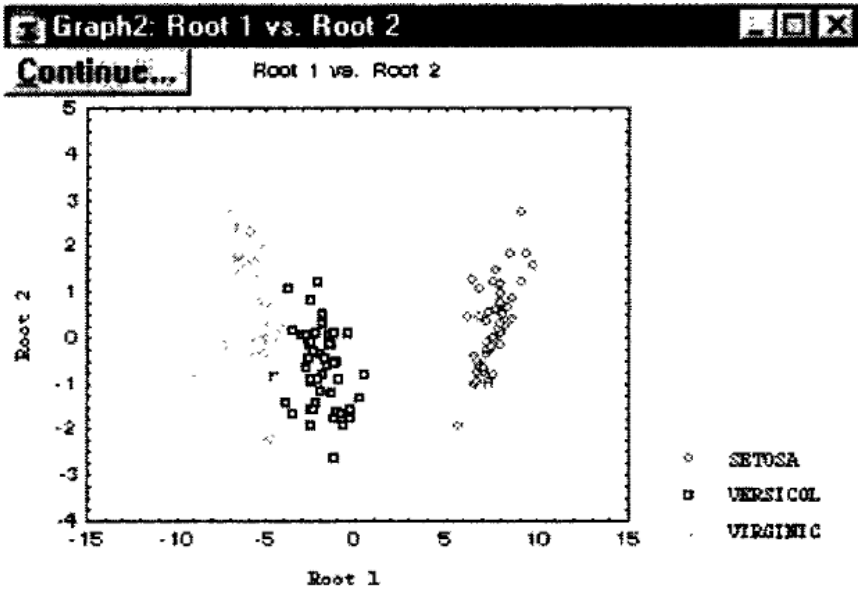


Рис. 176. Диаграмма разделения трех сортов ириса.

Просмотр результатов классификации осуществляется в окне Результаты дискриминантного анализа при помощи кнопки **Classification functions** (Функции классификации).

Classification Functions; grouping: ТИПИРИСА (irisdat sta)

Continue...	SETOSA p=.33333	VERSICOL p=.33333	VIRGIN p=.333
ДЛЛЕПЕСТ	-16.43	5.21	12
ШИРЧАШЕЛ	23.59	7.07	2
ШИРЛЕПЕС	-17.40	6.43	21
ДЛЧАШЕЛИ	23.54	15.70	1:
Constant	-86.31	-72.85	-10

Рис. 177. Функции классификации, построенные пошаговым методом.

С помощью этих функций можно вычислить классификационные значения (метки) для вновь наблюдаемых цветков по формулам:

SETOSA = -1643*ДЛЧАШЕЛИ + 23,69*ШИРЧАШЕЛ –
-17,4*ДЛЛЕПЕСТ + 23,54*ШИРЛЕПЕС – 86,31;
VERSICOL= 5,21*ДЛЧАШЕЛИ + 7,07*ШИРЧАШЕЛ –
- 6,43*ДЛЛЕПЕСТ + 15,70*ШИРЛЕПЕС – 72,85;
VIRGINIC= 12,76*ДЛЧАШЕЛИ + 3,69*ШИРЧАШЕЛ –
- 21,08*ДЛЛЕПЕСТ + 12,5*ШИРЛЕПЕС – 104,37.

Если появятся данные по новому образцу, то подставив их в эту формулы его можно будет отнести к тому сорту ириса (классу), для которого классификационное число будет больше.
Существуют и другие способы классификации, например, при помощи вычисления квадрата расстояния Махаланобиса от точек (случаев) до центров групп, для этого нужно нажать кнопку **Squared Mahalanobis distance** (Квадрат расстояния Махаланобиса).

Squared Mahalanobis Distances from Group Centroids

Continue...

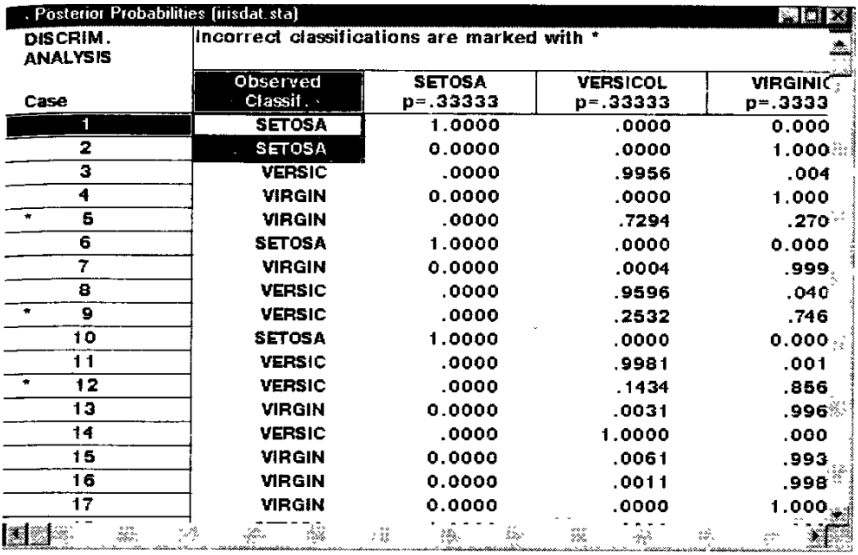
Incorrect classifications are marked with *

Case	Observed Classif.	SETOSA p=.33333	VERSICOL p=.33333	VIRGINIC p=.33333
1	SETOSA	.24	90.66	181.56
2	VIRGIN	208.57	27.32	1.89
3	VERSIC	105.27	2.23	13.07
4	VIRGIN	207.92	31.75	4.45
* 5	VIRGIN	133.07	5.25	7.24
6	SETOSA	1.33	84.01	170.06
7	VIRGIN	173.18	26.56	11.05
8	VERSIC	131.66	8.43	14.76
* 9	VERSIC	130.86	8.67	6.51
10	SETOSA	2.29	113.65	210.02
11	VERSIC	99.23	1.30	13.82
* 12	VERSIC	149.03	8.44	4.86

Рис. 178. Расстояния Махаланобиса для каждого случая.

Случай относится к группе, до которой расстояние Махаланобиса минимально.

Можно проводить анализ при помощи вычисления постериорных вероятностей. До проведения анализа необходимо задать вероятность, с которой каждый цветок относится к тому или иному сорту. Затем в окне "Результаты дискриминантного анализа" необходимо выбрать внизу кнопку **Posterior probabilities** (Апостериорные вероятности). Появится таблица апостериорных вероятностей:



Posterior Probabilities (irisdat.sta)

DISCRIM. ANALYSIS — Incorrect classifications are marked with *

Case	Observed Classif.	SETOSA p=.33333	VERSICOL p=.33333	VIRGINIC p=.3333
1	SETOSA	1.0000	.0000	0.000
2	SETOSA	0.0000	.0000	1.000
3	VERSIC	.0000	.9956	.004
4	VIRGIN	0.0000	.0000	1.000
* 5	VIRGIN	.0000	.7294	.270
6	SETOSA	1.0000	.0000	0.000
7	VIRGIN	0.0000	.0004	.999
8	VERSIC	.0000	.9596	.040
* 9	VERSIC	.0000	.2532	.746
10	SETOSA	1.0000	.0000	0.000
11	VERSIC	.0000	.9981	.001
* 12	VERSIC	.0000	.1434	.856
13	VIRGIN	0.0000	.0031	.996
14	VERSIC	.0000	1.0000	.000
15	VIRGIN	0.0000	.0061	.993
16	VIRGIN	0.0000	.0011	.998
17	VIRGIN	0.0000	.0000	1.000

Рис. 179. Таблица апостериорных вероятностей.

Интерпретация данной таблицы очень проста. В первом столбце указан сорт ириса для каждого случая, в остальных – апостериорные вероятности что данный случай принадлежит соответствующему сорту, который указан в заголовке столбца. Цветок относится к группе с максимальной апостериорной вероятностью.

При добавлении новых данных в исходную таблицу и нажатии соответствующих кнопок будет проведен анализ с включением новых данных.

Всегда имеет смысл провести анализ данных другим методом. Для этого необходимо закрыть окно "Результаты дискриминационного анализа" и вернуться в окно "Определение модели". В строке **Method** (Метод) выберите стандартный метод и нажмите кнопку ОК. В окне результатов с помощи **Classification function** (Функции классификации) посмотрите классификационные функции:

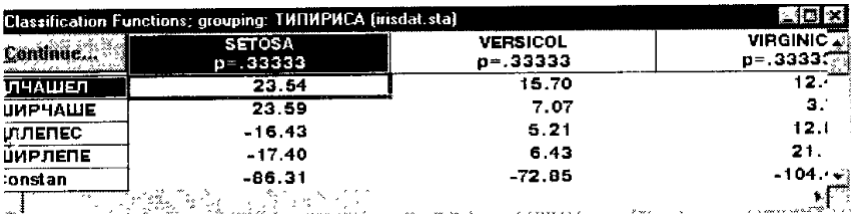


Рис. 180. Функции классификации, построенные стандартным методом.

Как видно из сравнения с рис. 174, функции классификации, полученные двумя разными методами, совпадают.

ЗАКЛЮЧЕНИЕ

В данном учебном пособии описаны приемы решения наиболее часто встречающихся задач при помощи математических пакетов MathCAD и Maple, работа с данными в матричном виде при помощи MATHLAB и рассмотрен пример статистической обработки экспериментальных данных в среде STATISTICA. Приведенные примеры свидетельствуют, что данные программы достигли весьма высокого уровня развития. Значительные математические возможности в них сочетаются с естественным и удобным пользовательским интерфейсом и богатыми возможностями по отображению графики. Для решения вычислительных задач предложены достаточно удачные реализации самых передовых математических методов и алгоритмов, реализован как автоматический, так и пользовательский выбор наиболее оптимального метода. Имеется возможность для программирования пользовательских модулей для проведения расчетов.

Из представленных пакетов наиболее универсальными являются MathCAD и Maple. MATHLAB и STATISTICA более узко специализированы и предназначены для решения конкретных задач.

Учебное пособие облегчит понимание и восприятие материала лекций по дисциплинам «Пакеты прикладных математических программ» и «Основы научных исследований и техника эксперимента», читаемых студентам второго курса специальности 210104 “Микроэлектроника и твердотельная электроника” (направление 140400 «Техническая физика») очной и заочной формы обучения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Давыдов А.В. Учебник по MathCAD/ А.В. Давыдов WEB:http://reactivmen.narod.ru/mathcad.htm;http://dspdav.nm.ru/mathbook.htm.

2. Технология СБИС: в 2-х кн.: пер. с англ./ под ред. С. Зи. – М.: Мир, 1986. – 352 с.

3. Математический пакет Maple V Release 4. Руководство пользователя/ Г.В. Прохоров, В.В. Колбеев, К.И. Желнов, М.А. Леденев. 1998. – 200 с.

4. MATLAB. Язык технических вычислений. Руководство пользователя/ пер. с англ. В.В. Конюшенко – 70 с.

5. StatSoft, Inc. Электронный учебник по статистике. Москва, StatSoft, 2001. WEB: http://www.statsoft.ru/home/textbook/ default.htm.

6. Боровиков В.П. Популярное введение в программу STATISTICA/ В.П. Боровиков - М.: Компьютер Пресс, 1998. – 267 с.

7. Использование пакета Statistica 5.0 для статистической обработки опытных данных: методические указания для дипломного проектирования для студентов лесного факультета специальностей 260400 "Лесное хозяйство" и 260500 "Садово-парковое и ландшафтное строительство"/ Сарат. гос. агр. ун-т.; сост.: С.В. Кабанов. Саратов, 2000.

ОГЛАВЛЕНИЕ

Учебное издание

Буслов Вадим Александрович

ПАКЕТЫ ПРИКЛАДНЫХ МАТЕМАТИЧЕСКИХ ПРОГРАММ

В авторской редакции

Компьютерная верстка В.А. Буслов

Подписано к изданию 05.12.2006
Уч.-изд. л. 11,3.

ГОУВПО «Воронежский государственный технический университет» 394026 Воронеж, Московский просп., 14

245