Министерство науки и высшего образования
Федеральное государственное бюджетное образовательное
учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра промышленной электроники (ПрЭ)

Михальченко С. Г.

Введение в профессию

УЧЕБНОЕ ПОСОБИЕ

ТОМСК 2019

Михальченко Сергей Геннадьевич

Введение в профессию: Учебное пособие / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019. – 117 с. : ил., табл., прил. – Библиогр.: с. 110.

Настоящее руководство имеет целью ознакомление студентов направления *11.03.04 Электроника и наноэлектроника (профиль - Промышленная электроника)* и *09.03.01 Информатика и вычислительная техника (профиль - Микропроцессорные системы обработки информации и управления)* с полем их будущей профессиональной деятельности, погружение в предметную область и терминологию.

Вторичной целью данной дисциплины является формирование профессиональных компетенций в области применения математических пакетов для численных и аналитических расчетов в различных видах деятельности (инженерной, научно–исследовательской, управленческой, и др.).

Руководство может быть использовано *для проведения лекционных занятий*. Изучение данной дисциплины должно содержать проведение практических занятий и может сопровождаться одноименным *электронным курсом*.

Для освоения дисциплины *«Введение в профессию»* не требуется специальных знаний. Достаточно навыков, полученных обучающимся в школьных курсах математики, информатики и физики.

Настоящее учебное пособие может применяться так же для обучения студентов по дисциплинам *«Профессиональные математические пакеты», «Инженерные расчеты в MathCAD», «Линейная алгебра и аналитическая геометрия»* и *«Теоретические основы электротехники»*.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Вычислительная, или, как еще говорят, ***компьютерная математика*** либо ***компьютерная алгебра,*** – это большой раздел математического моделирования. В принципе, программы такого рода можно отнести к *инженерным программам автоматизированного проектирования*. Таким образом, в области инженерного проектирования выделяют три основных раздела:

- CAD — Computer Aided Design (проектирование);
- CAM — Computer Aided Manufacturing (производство);
- CAE — Computer Aided Engineering (инженерные расчеты).

Сегодня серьезное конструирование, градостроительство и архитектура, электротехника и масса смежных с ними отраслей, а также учебные заведения технической направленности уже не могут обойтись без *CAD\CAM\CAE* – систем (систем автоматизированного проектирования – *САПР*), производства и расчетов. А математические пакеты являются составной частью мира *CAE*-систем, но эта часть никак не может считаться второстепенной, поскольку некоторые задачи вообще невозможно решить без помощи компьютера. Более того, к системам вычислительной математики сегодня прибегают даже математики-теоретики, например для проверки своих гипотез.

В настоящее время в ***математических пакетах*** применяется ***принцип конструирования модели,*** а не традиционное искусство программирования. То есть пользователь лишь ставит задачу, а методы и алгоритмы решения система находит сама. Такие рутинные операции, как раскрывание скобок, преобразование выражений, нахождение корней уравнений, производных и неопределенных интегралов компьютер самостоятельно осуществляет в символьном виде, причем практически без вмешательства пользователя.

Современные математические пакеты можно использовать и как обычный калькулятор, и как средства для упрощения выражений при решении каких-либо прикладных задач.

В настоящее время практически все современные *CAE*-программы имеют встроенные функции *символьных вычислений*. Однако наиболее известными и приспособленными для математических символьных вычислений считаются *Mathematica* [1], *Maple* [2], *MatLab* [3], и *MathCAD* [4].

Что делают эти программы и как они упрощают математические выкладки? Основу курса математического анализа в высшей школе составляют такие понятия, как пределы, производные, первообразные функций, интегралы разных видов, ряды и дифференциальные уравнения. Тому, кто знаком с основами высшей математики, наверняка известны десятки правил нахождения пределов, взятия интегралов, нахождения производных и т.д. Если добавить к этому то, что для нахождения большинства интегралов нужно также помнить таблицу основных интегралов, то получается поистине огромный объем информации. И если какое-то время не тренироваться в решении подобных задач, то многое быстро забывается и для нахождения, например, интеграла посложнее придется уже заглядывать в справочники. Но ведь взятие интегралов и нахождение пределов в реальной работе не является главной целью вычислений, тем более – для разработчика микроэлектронных и наноэлектронных систем. Реальная цель заключается в решении каких-либо проблем, а вычисления – всего лишь промежуточный этап, инструмент на пути к этому решению. С помощью описываемого *программного обеспечения (ПО)* можно сэкономить массу времени и избежать многих ошибок при вычислениях.

Спектр задач, решаемых подобными системами, очень широк:

- проведение математических исследований, требующих вычислений и аналитических выкладок;
- разработка и анализ алгоритмов;
- математическое моделирование и компьютерный эксперимент;
- анализ и обработка данных;
- визуализация, научная и инженерная графика;
- разработка графических и расчетных приложений.

Кроме того, поскольку *CAE*-системы содержат операторы для базовых вычислений, то почти все алгоритмы, отсутствующие в стандартных функциях, можно реализовать посредством написания собственной программы, включающей подпрограммы математического пакета.

Несмотря на то, что в области компьютерной математики не наблюдается такого разнообразия, как, скажем, в среде компьютерной графики, за видимой ограниченностью рынка математических программ скрываются их поистине безграничные возможности. Как правило, CAE-системы охватывают практически все области математики и инженерных расчетов.

Когда-то системы символьной математики были ориентированы исключительно на узкий круг профессионалов и работали на больших компьютерах (мэйнфреймах). Но с ростом вычислительной мощности ПК эти системы были переработаны под них и доведены до уровня массовых серийных программных систем. Сейчас на рынке сосуществуют системы символьной математики самого разного калибра – от рассчитанной на широкий круг потребителей системы *MathCAD* до компьютерных монстров *Mathematica, MatLab и Maple,* имеющих тысячи встроенных и библиотечных функций, широкие возможности графической визуализации вычислений и развитые средства для подготовки документации.

Отметим, что практически все эти системы работают не только на персональных компьютерах, оснащенных популярными операционными системами *Windows*, но и под управлением операционных системы *Linux, UNIX, Mac OS,* а также на *КПК.* Они давно знакомы пользователям и широко распространены на всех платформах – от наладонника до суперкомпьютера.

1. ПРОГРАММНЫЙ КОМПЛЕКС MATHCAD

1.1. Обзор вычислительных математических пакетов

Система компьютерной алгебры Mathematica

Mathematica – система компьютерной алгебры компании *Wolfram Research* [http://www.wolfram.com/]. Содержит множество функций, как для аналитических преобразований, так и для численных расчётов. Кроме того, программа поддерживает работу с графикой и звуком, включая построение двух- и трёхмерных графиков функций, рисование произвольных геометрических фигур, импорт и экспорт изображений и звука *[1]*.

Мощная математическая система, претендующая на мировое лидерство, предназначена для автоматизации выполнения математических расчетов любой степени сложности. Несмотря на свою направленность на серьезные математические вычисления, системы класса *Mathematica* просты в освоении и могут использоваться довольно широкой категорией пользователей – студентами и преподавателями вузов, инженерами, аспирантами, научными работниками и даже учащимся математических классов общеобразовательных и специальных школ. Все они найдут в подобной системе многочисленные полезные возможности для применения. Можно даже сказать, что *Mathematica* обладает значительной функциональной избыточностью (там, в частности, есть даже возможность для синтеза звука).

Основное достоинство системы *Mathematica* состоит в том, что практически все вычисления, производимые этим пакетом, – *аналитические*, т.е. точные – не содержащие ошибки округления и погрешности численного метода *[7, 8]*.

На момент написания данного учебного пособия актуальной версией программного продукта являлось: *Mathematica* 9.0.0 от 28.11.1012 г. *[1]*.

Программа *Mathematica* из всех рассматриваемых систем наиболее полна и универсальна, однако у каждой программы есть как свои достоинства, так и недостатки. Заметим поэтому, что те исследователи, которые серьезно работают с системами компьютерной математики, должны пользоваться несколькими программами, ибо только это гарантирует высокий уровень надежности сложных вычислений.

Mathematica была задумана как система, максимально автоматизирующая труд научных работников и математиков-аналитиков, поэтому она заслуживает изучения даже в качестве типичного представителя элитных и высокоинтеллектуальных программных продуктов высшей степени сложности. Однако куда больший интерес она представляет как мощный и гибкий математический инструментарий, который может оказать неоценимую помощь большинству научных работников, преподавателей университетов и вузов, студентов, инженеров и даже школьников.

Центральное место в пакете *Mathematica* занимает машинно-независимое ***ядро математических операций***, которое позволяет переносить систему на различные компьютерные платформы. Ядро сделано достаточно компактным для того, чтобы можно было очень быстро вызвать из него любую функцию. Для расширения набора функций служат библиотека (*Library*) и набор пакетов расширения (*Add-on Packages*). Пакеты расширений готовятся на собственном языке программирования систем *Mathematica* и являются главным средством для развития возможностей системы и их адаптации к решению конкретных классов задач пользователя. Кроме того, системы

имеют встроенную электронную справочную систему – *Help*, которая содержит электронные книги с реальными примерами.

К основным функциональным возможностям пакета относится: вычисление значений функций, в том числе специальных, с произвольной точностью; решение уравнений и систем уравнений и неравенств; упрощение выражений; нахождение пределов; интегрирование и дифференцирование функций; нахождение конечных и бесконечных сумм и произведений; решение дифференциальных уравнений и уравнений в частных производных; преобразования Фурье и Лапласа, а также Z-преобразование; преобразование функции в ряд Тейлора, операции с рядами Тейлора: сложение, умножение, композиция, получение обратной функции и т.д.; вейвлет-анализ; интерполяция функции от произвольного числа аргументов по набору известных значений; разложение числа на простые множители, нахождение НОД и НОК; операции с матрицами: сложение, умножение, нахождение обратной матрицы, умножение на вектор, вычисление экспоненты, получение определителя; поиск собственных значений и собственных векторов; построение графиков функций, в том числе параметрических кривых и поверхностей; построение и манипулирование графами.

С точки зрения разработки программного обеспечения, пакет обладает следующим функционалом: автоматическое генерирование *Cu* кода и его компоновка; автоматическое преобразование компилируемых программ системы *Mathematica* в *Cu* код для автономного или интегрированного использования; использование *SymbolicC* для создания, обработки и оптимизации *Cu* кода; интеграция внешних динамических библиотек; поддержка CUDA и OpenGL; поддержка процедурного программирования с применением стандартных операторов и объектно-ориентированного подхода (на языке C++).

К недостаткам системы Mathematica следует отнести разве что весьма необычный язык программирования, обращение к которому, впрочем, облегчает подробная система помощи.

Пакет символьной математики Maple

Maple – программный пакет, система компьютерной алгебры [2]. Пакет Maple – совместная разработка Университета Ватерлоо (шт. Онтарио, Канада) и Высшей технической школы (ETHZ, Цюрих, Швейцария). Для его продажи была создана специальная компания – *Waterloo Maple Inc* [http://www.maplesoft.com]. Сейчас эта компания работает совместно с *MathSoft Inc.* – создательницей весьма популярных и массовых систем для численных расчетов *MathCAD*, ставших международным стандартом для технических вычислений.

Система *Maple* предназначена для символьных вычислений, хотя имеет ряд средств и для численного решения дифференциальных уравнений и нахождения интегралов. Обладает развитыми графическими средствами. Имеет собственный язык программирования, напоминающий Паскаль.

На момент написания данного учебного пособия актуальной версией программного продукта являлось: *Maple 17* от 13.03.1013 г.

В самом общем смысле *Maple* – это среда для выполнения математических расчетов на компьютере. В отличие от языков программирования высокого уровня, таких как *Cu* или *Паскаль*, *Maple* может решать большое количество математических задач путем введения команд, без всякого предварительного программирования. Кроме того, *Maple* может оперировать не только приближенными числами, но и точными целыми и рациональными числами. Это позволяет получить ответ с высокой, в идеале с бесконечной, точностью [9].

Но, что самое важное, решение задач может быть получено *аналитически*, то есть в виде формул, состоящих из математических символов. Вследствие этого *Maple* называют также *пакетом символьной математики [10]*.

Maple умеет выполнять сложные алгебраические преобразования и упрощения над полем комплексных чисел, находить конечные и бесконечные суммы, произведения, пределы и интегралы, решать в символьном виде и численно алгебраические (в том числе трансцендентные) системы уравнений и неравенств, находить все корни многочленов, решать аналитически и численно системы обыкновенных дифференциальных уравнений и некоторые классы уравнений в частных производных. *В Maple* включены пакеты подпрограмм для решения задач линейной и тензорной алгебры, Евклидовой и аналитической геометрии, теории чисел, теории вероятностей и математической статистики, комбинаторики, теории групп, интегральных преобразований, численной аппроксимации и линейной оптимизации (симплекс метод) а также задач финансовой математики и многих, многих других задач [*11, 12*].

Отметим, что ***символьный анализатор*** программы *Maple* является наиболее сильной частью этого ПО, поэтому именно он был позаимствован и включен в ряд других *CAE*-пакетов, таких как *MathCAD* и *MatLab*, а также в состав пакетов для подготовки научных публикаций *Scientific WorkPlace* и *Math Office for Word.*

Пакет *Maple* состоит из *ядра* (процедур, написанных на языке *Си* и хорошо оптимизированных), библиотеки, написанной на *Maple*-языке, и развитого внешнего *интерфейса*. Ядро выполняет большинство базовых операций, а библиотека содержит множество команд – процедур, выполняемых в режиме интерпретации.

Интерфейс *Maple* основан на концепции рабочего поля (*worksheet*) или документа, содержащего строки ввода-вывода и текст, а также графику. Работа с пакетом происходит в *режиме интерпретатора*. В строке ввода пользователь задает команду, нажимает клавишу <*Enter*> и получает результат – строку (или строки) вывода либо сообщение об ошибочно введенной команде. Тут же выдается приглашение вводить новую команду и т.д.

Система *Maple* позволяет вводить электронные таблицы, содержащие как числа, так и символы. Они совмещают в себе математические возможности системы *Maple* с уже знакомым форматом из строк и столбцов традиционных электронных таблиц.

Систему *Maple* можно использовать и на самом элементарном уровне ее возможностей – как очень мощный калькулятор для вычислений по заданным формулам, но главным ее достоинством является способность выполнять арифметические действия в *символьном виде*, то есть так, как это делает человек. При работе с дробями и корнями программа не приводит их в процессе вычислений к десятичному виду, а производит необходимые сокращения и преобразования в столбик, что позволяет избежать ошибок при округлении. Для работы с десятичными эквивалентами в системе *Maple* имеется специальная команда, аппроксимирующая значение выражения в формате чисел с плавающей запятой. Система *Maple* вычисляет конечные и бесконечные суммы и произведения, выполняет вычислительные операции с комплексными числами, легко приводит комплексное число к числу в полярных координатах, вычисляет числовые значения элементарных функций, а также знает много специальных функций и математических констант (таких, например, как π). *Maple* поддерживает сотни специальных функций и чисел, встречающихся во многих областях математики, науки и техники. Для технических применений в *Maple* включены справочники физических констант и единицы физических величин с автоматическим пересчетом формул.

Система *Maple* предлагает различные способы представления, сокращения и преобразования выражений, например такие операции, как упрощение и разложение на множители алгебраических выражений и приведение их к различному виду. *Maple*

также имеет множество мощных инструментальных средств для вычисления выражений с одной или несколькими переменными. Программу можно использовать для решения задач дифференциального и интегрального исчисления, вычисления пределов, разложений в ряды, суммирования рядов, умножения, интегральных преобразований (таких как преобразование Лапласа, Z-преобразование, преобразование Меллина или Фурье), а также для исследования непрерывных или кусочно-непрерывных функций.

Maple может вычислять пределы функций, как конечные, так и стремящиеся к бесконечности, а также распознает неопределенности в пределах. В этой системе можно решать множество обычных дифференциальных уравнений (*ODE*), а также дифференциальные уравнения в частных производных (*PDE*), в том числе задачи с начальными условиями (*IVP*) и задачи с граничными условиями (*BVP*).

Одним из наиболее часто используемых в системе Maple пакетов программ является *пакет линейной алгебры*, содержащий мощный набор команд для работы с векторами и матрицами. *Maple* может находить собственные значения и собственные векторы операторов, вычислять криволинейные координаты, находить матричные нормы и вычислять множество различных типов разложения матриц.

Графические средства Maple позволяют строить двумерные графики сразу нескольких функций, создавать графики конформных преобразований функций с комплексными числами и строить графики функций в логарифмической, двойной логарифмической, параметрической, фазовой, полярной и контурной форме. Можно графически представлять неравенства, неявно заданные функции, решения дифференциальных уравнений и корневые годографы. *Maple* может строить поверхности и кривые в трехмерном представлении, включая поверхности, заданные явной и параметрической функциями, а также решениями дифференциальных уравнений. При этом представлять можно не только в статическом виде, но и в виде двух- или трехмерной анимации. Эту особенность системы можно использовать для отображения процессов, протекающих в режиме реального времени.

Специализированные приложения. Обширный набор мощных инструментальных приложений ***Maple PowerTools*** и пакетов для таких областей, как анализ методом конечных элементов (*FEM*), нелинейная оптимизация и др., полностью удовлетворят пользователей с университетским математическим образованием. В *Maple* включены также пакеты подпрограмм для решения задач линейной и тензорной алгебры, евклидовой и аналитической геометрии, теории чисел, теории вероятностей и математической статистики, комбинаторики, теории групп, интегральных преобразований, численной аппроксимации и линейной оптимизации (симплекс-метод), а также задач финансовой математики и многих, многих других.

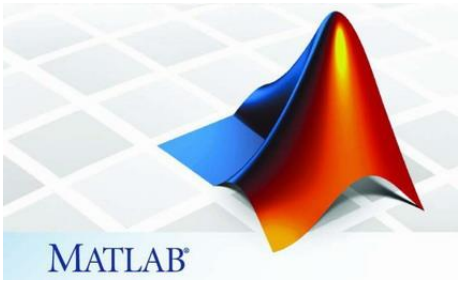
Программирование. Система *Maple* использует процедурный язык 4-го поколения (***4GL***). Этот язык специально предназначен для быстрой разработки математических подпрограмм и пользовательских приложений. *Maple* может генерировать код, совместимый с такими языками программирования, как *C++*, и с языком набора текста *LaTeX*. Используя систему *Maple*, можно разработать определенную математическую модель, а затем с ее помощью сгенерировать код на языке *Cи*, соответствующий этой модели.

К недостаткам системы Maple можно отнести лишь ее некоторую «задумчивость», причем не всегда обоснованную, а также очень высокую стоимость этой программы.

Пакет математических и инженерных вычислений MatLab

Система математических и инженерных вычислений *MatLab* [3], разработанная MathWorks Inc. [http://www.mathworks.com/] относится к среднему уровню продуктов, предназначенных для символьной математики, но рассчитана на широкое применение в сфере *CAE*. *MatLab* – одна из старейших, тщательно проработанных и проверенных

временем систем автоматизации математических расчетов, построенная на расширенном представлении и применении *матричных операций*. Это нашло отражение и в самом названии системы – *MATrix LABoratory*. Однако синтаксис языка программирования системы продуман настолько тщательно, что данная ориентация почти не ощущается теми пользователями, которых не интересуют непосредственно матричные вычисления.



Несмотря на то, что изначально MatLab предназначалась исключительно для *численных расчетов*, в дополнение к прекрасным вычислительным средствам, у фирмы *Waterloo Maple Inc.* по лицензии для *MatLab* было приобретено ядро символьных преобразований, а также появились библиотеки, которые обеспечивают в *MatLab* уникальные для математических пакетов функции. Например, широко известная ***библиотека Simulink***, реализуя принцип визуального программирования, позволяет построить логическую схему сложной системы управления из одних только стандартных блоков, не написав при этом ни строчки кода. После конструирования такой схемы можно детально проанализировать ее работу.

В системе *MatLab* также существуют широкие возможности для программирования. Ее библиотека *C Math* (компилятор *MatLab*) является объектной и содержит свыше 300 процедур обработки данных на языке *Cи*. Внутри пакета можно использовать как процедуры самой *MatLab*, так и стандартные процедуры языка *Cи*, что делает этот инструмент мощнейшим подспорьем при разработке приложений (используя компилятор *C Math*, можно встраивать любые процедуры *MatLab* в готовые приложения).

Библиотека *C Math* позволяет пользоваться следующими категориями функций: операции с матрицами; сравнение матриц; решение линейных уравнений; разложение операторов и поиск собственных значений; нахождение обратной матрицы; поиск определителя; вычисление матричного экспоненциала; элементарная математика; функции beta, gamma, erf и эллиптические функции; основы статистики и анализа данных; поиск корней полиномов; фильтрация, свертка; быстрое преобразование Фурье (FFT); интерполяция; операции со строками; операции ввода-вывода файлов и т.д.

При этом все библиотеки *MatLab* отличаются высокой скоростью численных вычислений. Однако матрицы широко применяются не только в таких математических расчетах, как решение задач линейной алгебры и математического моделирования, обсчета статических и динамических систем и объектов. Они являются основой автоматического составления и решения уравнений состояния динамических объектов и систем. Именно универсальность аппарата матричного исчисления значительно повышает интерес к системе *MatLab,* вобравшей в себя лучшие достижения в области быстрого решения матричных задач. Поэтому *MatLab* давно уже вышла за рамки специализированной матричной системы, превратившись в одну из наиболее мощных универсальных интегрированных систем компьютерной математики.

Для визуализации моделирования система *MatLab* имеет библиотеку *Image Processing Toolbox*, которая обеспечивает широкий спектр функций, поддерживающих визуализацию проводимых вычислений непосредственно из среды *MatLab*, увеличение и анализ, а также возможность построения алгоритмов обработки изображений. Усовершенствованные методы графической библиотеки в соединении с языком программирования *MatLab* обеспечивают открытую расширяемую систему, которая может быть использована для создания специальных приложений, пригодных для обработки графики.

Основные средства библиотеки *Image Processing Tollbox*: построение фильтров, фильтрация и восстановление изображений; увеличение изображений; анализ и

статистическая обработка изображений; выделение областей интересов, геометрические и морфологические операции; манипуляции с цветом; двумерные преобразования; запись/чтение графических файлов. Среди других библиотек системы *MatLab* можно также отметить System Identification Toolbox – набор инструментов для создания математических моделей динамических систем, основанных на наблюдаемых входных/выходных данных. Эта подсистема позволяет организовать итеративный процесс создания моделей для получения оценок и выделения наиболее значимых данных, выделения области возможных значений данных, удаления погрешностей, предотвращения ухода данных от характерного для них уровня и т.п.

Что касается математических вычислений, то *MatLab* предоставляет доступ к огромному количеству подпрограмм, содержащихся в библиотеке *NAG Foundation Library* компании *Numerical Algorithms Group Ltd* [http://www.nag.co.uk/] (инструментарий имеет сотни функций из различных областей математики, и многие из этих программ были разработаны широко известными в мире специалистами). Это уникальная коллекция реализаций современных численных методов компьютерной математики, созданных за последние четыре десятка лет. Таким образом, *MatLab* вобрала и опыт, и правила, и методы математических вычислений, накопленные за тысячи лет развития математики.

Из недостатков системы *MatLab* можно отметить невысокую интегрированность среды (очень много окон, с которыми лучше работать на двух мониторах), не очень внятную справочную систему (а между тем объем фирменной документации достигает почти 5 тыс. страниц, что делает ее трудно обозримой) и специфический редактор кода *MatLab*-программ.

Программный комплекс MathCAD

В отличие от мощных и ориентированных на высокоэффективные вычисления программных пакетов, описанных выше, программа *MathCAD* – это, скорее, простой, но продвинутый редактор математических текстов с широкими возможностями символьных вычислений и прекрасным интерфейсом.

MathCAD [4] был задуман и первоначально написан *Алленом Раздовом* из Массачусетского технологического института (*MIT*), соучредителем компании *Mathsoft* [http://www.mathsoft.com/], которая с 2006 г. является частью корпорации *PTC* [http://communities.ptc.com] (*Parametric Technology Corporation*).

MathCAD не имеет языка программирования как такового, а ядро символьных вычислений заимствовано из пакета *Maple*. Некоторые из математических возможностей *MathCAD* (версии до 13.1 включительно) основаны на подмножестве системы компьютерной алгебры *Maple* (*MKM, Maple Kernel Mathsoft).* Начиная с 14 версии — использует символьное ядро *MuPAD.*

Зато интерфейс программы *MathCAD* очень простой, а возможности визуализации богатые. Все вычисления здесь осуществляются на уровне визуальной записи выражений в общеупотребительной математической форме. Пакет имеет хорошие подсказки, подробную документацию, функцию обучения использованию, целый ряд дополнительных модулей и приличную техническую поддержку производителя. Однако пока математические возможности *MathCAD* в области компьютерной алгебры намного уступают системам *Maple, Mathematica, MatLab.* Однако по программе *MathCAD* выпущено много книг и обучающих курсов, в том числе у нас в России. Сегодня эта система стала буквально международным стандартом для прикладных технических вычислений.

1.2. Интерфейс математического процессора MathCAD

Программный комплекс ***MathCAD*** (*Mathematical Computer Aided Design* – математическое автоматизированное проектирование) предназначен для автоматизации решений широкого круга задач, связанных с математическими расчетами. Это многофункциональная вычислительная среда, снабженная дружественным, во многом интуитивно понятным графическим интерфейсом.

MathCAD имеет интуитивный и простой для использования интерфейс пользователя. Для ввода формул и данных можно использовать как клавиатуру, так и специальные панели инструментов.

При запуске *MathCAD* на экране появляется окно интерфейса (*Рис. 1*).

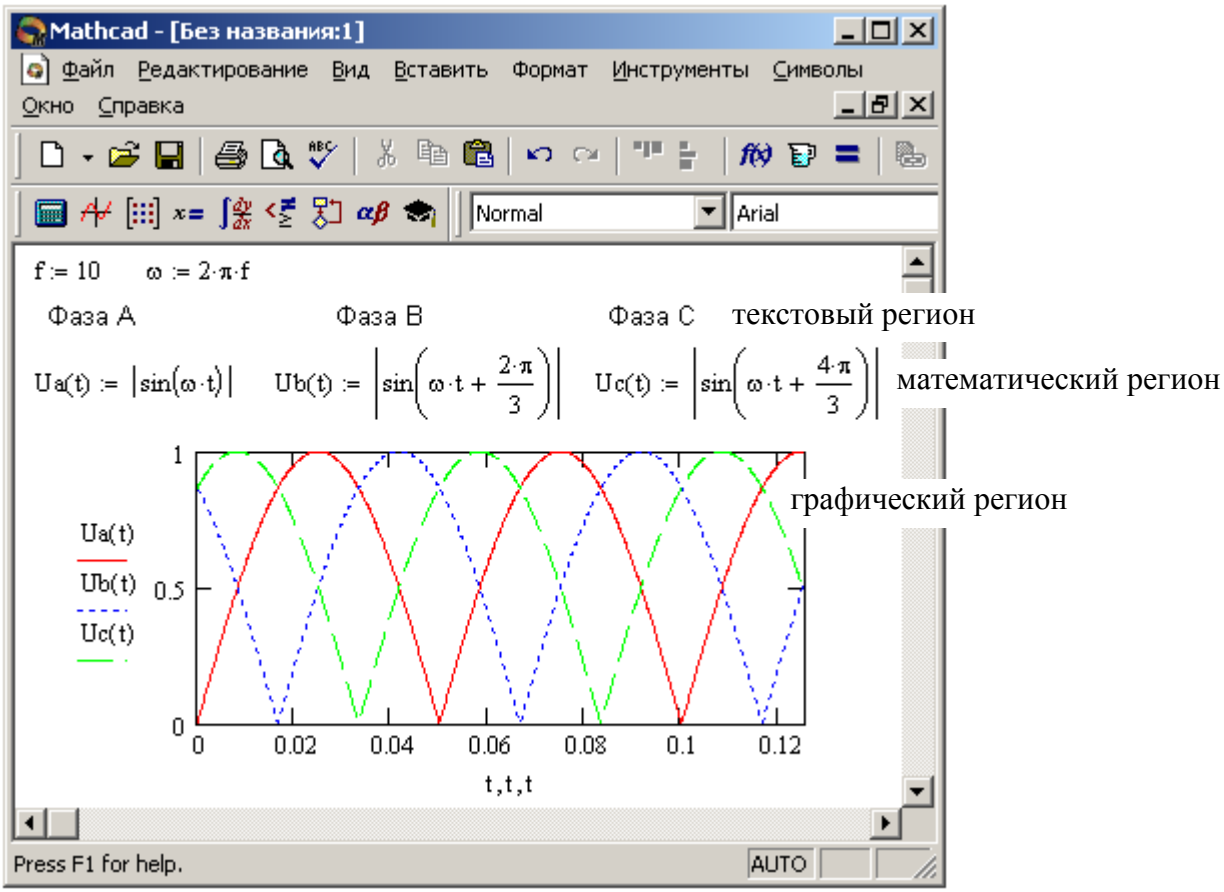


Рис. 1 – Окно интерфейса MathCAD с загруженным документом

Работа осуществляется в пределах рабочего листа, на котором уравнения и выражения отображаются графически, в противовес текстовой записи в языках программирования. При создании документов-приложений используется принцип *WYSIWYG* (*What You See Is What You Get* – что видишь, то и получаешь).

Рабочий лист и регионы

В поле основного окна отображается продолжаемый вправо и вниз бесконечный рабочий лист (*worksheet*). На рабочем листе в произвольном порядке располагаются ***регионы***. Существует три типа регионов:

- математические,
- графические,
- текстовые.

Математические регионы содержат формульные выражения, операторы назначения и индикации результатов. С математическими регионами связаны вычислительные модули – участки исполняемого кода. Преобразование входной информации, занесенной в математический регион, в реализуемый код выполняется

интерпретатором *MathCAD*. В отличие от компиляторов, выполняющих просмотр текста программы целиком, интерпретатор обрабатывает каждый регион в отдельности, что, собственно, и дает возможность использовать *MathCAD* как мощный калькулятор для разовых вычислений. Последовательность обработки математических регионов – *слева направо и сверху вниз* по рабочему листу. К моменту выполнения математического региона все объекты, участвующие в его выражениях, должны быть определены либо непосредственно в этом регионе, либо в предшествующих по времени выполнения регионах. Для удобства ввода информации в математические регионы в *MathCAD* широко используется система шаблонов.

Текстовые регионы содержат комментирующий текст и служат для оформления рабочего листа как удобочитаемого документа. Поскольку никакой исполнительный код с текстовыми регионами не связывается, их расположение на рабочем листе ничем не регламентировано. В текстовые регионы можно помещать изображения.

Графические регионы предназначены для вывода информации в виде графиков и изображений. Для программирования графического региона также применяются шаблоны.

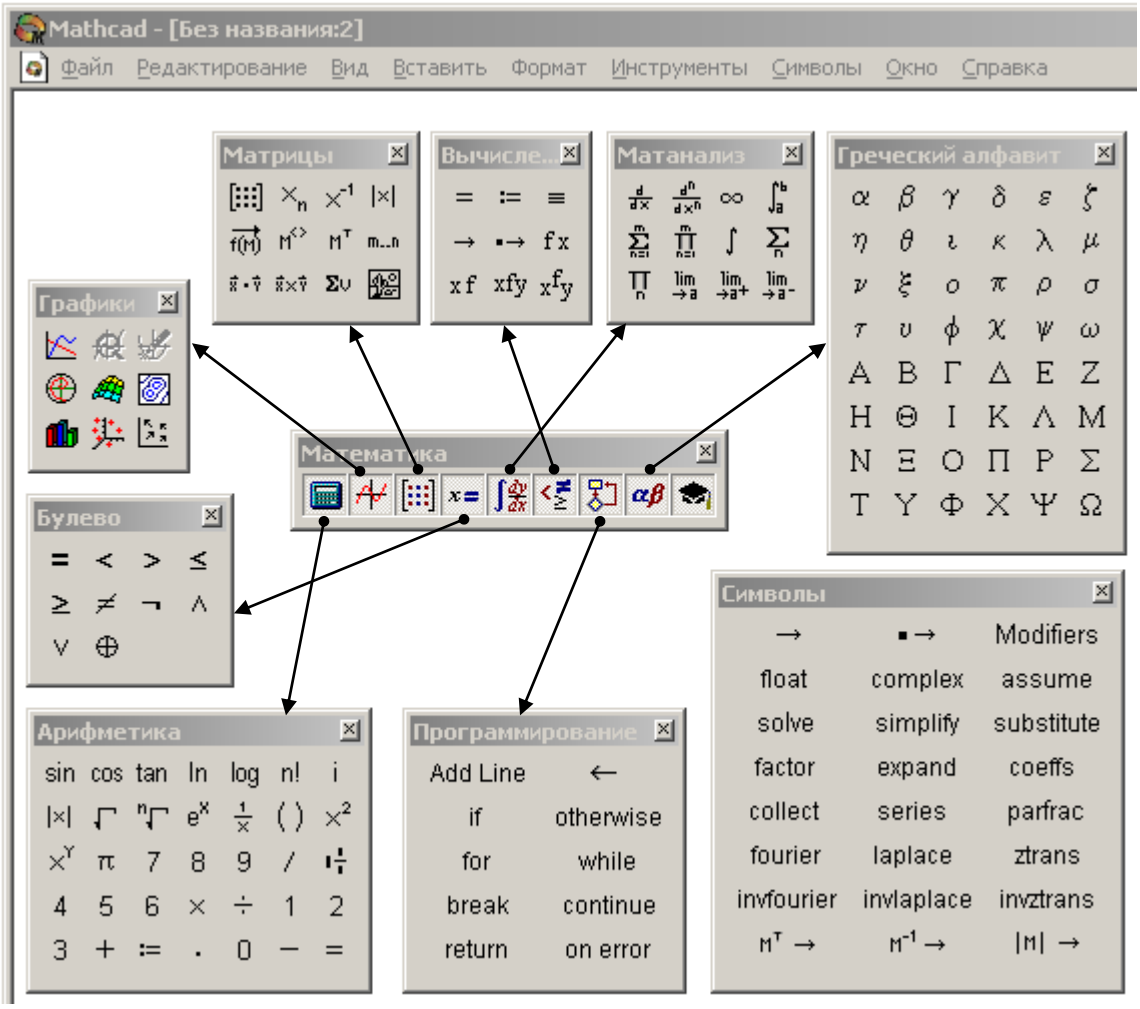


Рис. 2 – Панели инструментов и линейка их вызова

1.3. Инструментальные панели и шаблоны

Интерфейс *MathCAD* предоставляет пользователю представительный набор шаблонов для программирования математических и графических регионов. Для вызова шаблонов и занесения специфических символов операций используются инструментальные панели с виртуальными кнопками или «горячие» клавиши – акселераторы.

Панели инструментов объединены в 9 тематических групп (палитр). Каждая группа может быть вызвана на экран из *линейки вызова палитр*: – она же *панель «Математика»*.

Соответствие кнопок линейки вызываемым панелям показано на *Рис. 2*.

Всего на инструментальных панелях представлено 169 позиций, воздействие на которые вызывает вставку определенного шаблона или символа в точку расположения курсора на рабочем листе. Как видно из *Рис. 2*, некоторые позиции дублируются, присутствуя на разных панелях, некоторые предназначены для вставки обычных символов, имеющихся на клавиатуре, так что более удобно вводить непосредственным набором с клавиатуры, нежели через панель инструментов. Назначение ряда инструментов очевидно из соответствующих пиктограмм и не требует пояснений.

Здесь мы рассмотрим лишь те инструменты, применение которых необходимо для выполнения заданий лабораторных работ, информацию обо всех инструментах можно получить, воспользовавшись справочной системой *MathCAD*, вызываемой нажатием кнопки <*F1*> или из основного меню.

В составе пакета *MathCAD* имеется достаточно обширный набор встроенных функций. Вставка встроенных функций на рабочий лист может быть выполнена двумя путями:

- набором имени функции на клавиатуре – «горячих клавиш»;
- выбором функции из списка в окне диалога, открывающегося после воздействия на элемент управления с пиктограммой: 

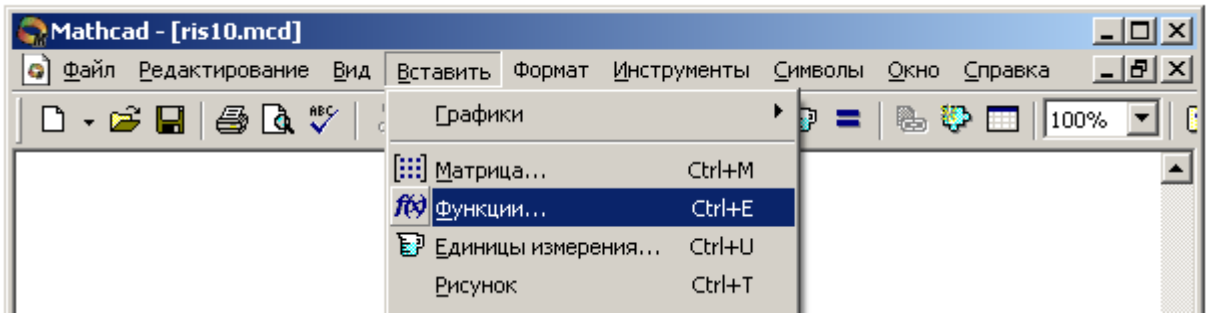


Рис. 3 – Панель вызова набора встроенных функций

Второй вариант представляется более удобным, так как на рабочий лист помещается шаблон функции с правильным названием и шаблоном списка параметров. Кроме того, в открывающемся окне диалога выбора функции имеется кнопка вызова контекстно-зависимой справки (<*Ctrl-F1*>).

Таблица 1 – Инструменты и клавиши быстрого вызова

Описание операции	*Пиктограмма*	*Клавиши*	*Панель*
Извлечение квадратного корня	$\sqrt{\square}$	\	Calculator
Возведение в степень	$\mathbf{x^Y}$	Shift+6	Calculator
Ввод матрицы или вектора	$\square\square$	Ctrl+M	Matrix
Ввод индекса (в элементах массива)	$\mathbf{x_n}$	[	Matrix
Определитель, модуль, абсолютное значение	$\mathbf{\lvert x \rvert}$	|	Matrix
Выделение столбца матрицы	$M^{\langle\rangle}$	Ctrl+6	Matrix
Создание ранжированной переменной	m..n	;	Matrix
Векторизация функции	$\overrightarrow{f(M)}$	Ctrl+ –	Matrix

Описание операции	*Пиктограмма*	*Клавиши*	*Панель*
Оператор назначения – присваивания	:=	:	Evaluation
Оператор глобального назначения	≡	~	Evaluation
Индикация значения в символьной форме	→	Ctrl+.	Evaluation
График в прямоугольных координатах		Shift+2	Graph

1.4. Операторы определения объектов и индикации значений

Операторы определения (*присваивания, назначения*) объектов размещаются в математических регионах и выполняют три функции:

- создают объект, связывая с ним его имя – идентификатор;
- определяют тип объекта;
- инициализируют (задают) объект значением.

В качестве определяемого объекта могут выступать *переменные и функции. Листинг 1* демонстрирует примеры применения операторов определения объектов различного типа: вещественного числа ***a***, векторов ***b*** и ***c***, матрицы ***A*** и функции ***z(x).***

Листинг 1. Операторы определения и индикации

$a := 17 \quad b := (1\ 2\ 3)$

$a = 17 \quad b = (1\ 2\ 3)$

$A := \begin{pmatrix} 1 & 2 & 3 \\ 6 & 5 & 4 \\ 7 & 8 & 9 \end{pmatrix} \quad A = \begin{pmatrix} 1 & 2 & 3 \\ 6 & 5 & 4 \\ 7 & 8 & 9 \end{pmatrix}$

$c := \begin{pmatrix} 1.1 \\ 0.8 \\ 2.1 \end{pmatrix} \quad c = \begin{pmatrix} 1.1 \\ 0.8 \\ 2.1 \end{pmatrix}$

$z(x) := Re(x) + i \cdot Im(x)$

$z(x) \rightarrow x$

$z(3 - i) = 3 - i$

Тип объекта, соответствующего тому или иному имени-идентификатору, определяется типом правой части. Далее любые операции с объектом интерпретируются в соответствии с его определенным типом.

В примере (*Листинг 1*) определена функция ***z(x)***, которая получает в качестве аргумента комплексное число ***x*** и возвращает в качестве значения комплексно-сопряженное число. Функция, задаваемая подобным образом, называется *функцией, определяемой пользователем*. В скобках после имени функции находится список *формальных параметров*. В рассмотренном примере этот список состоит из единственной переменной ***x***. При вызове – применении этой функции формальные параметры заменяются фактическими. До вызова функции, переменные, перечисленные в списке формальных параметров, будут оставаться неопределенными.

Тип объекта, возвращаемого функцией, т.е. связанного с ее именем, определяется по контексту определения точно так же, как это происходит при определении переменной. Например, ***sin(x)*** возвращает вещественное число, а ***|x|*** – целое число.

В составе определения функции могут присутствовать переменные, не передаваемые через список формальных параметров. В этом случае такие переменные должны быть определены до оператора назначения функции. Необходимо отметить, что передача параметров через глобальные переменные в общем случае нежелательна, т.к. нарушает изолированность функции и может служить источником трудно обнаруживаемых ошибок.

Виды операторов определения MathCAD

Операторы назначения в *MathCAD* имеют три разновидности:

- **:=** оператор назначения (стандартный);
- **≡** оператор глобального назначения;

- ← оператор локального назначения в блоке.

Все три разновидности операторов, помимо операции присваивания значения правого операнда левому, выполняют *передачу типа* левому операнду.

Оператор **:=** связывает значение и тип объекта с его идентификатором во всех регионах, выполняемых после региона, содержащего этот оператор.

Оператор **≡** действует на все регионы рабочего листа независимо от их расположения относительно региона, содержащего этот оператор.

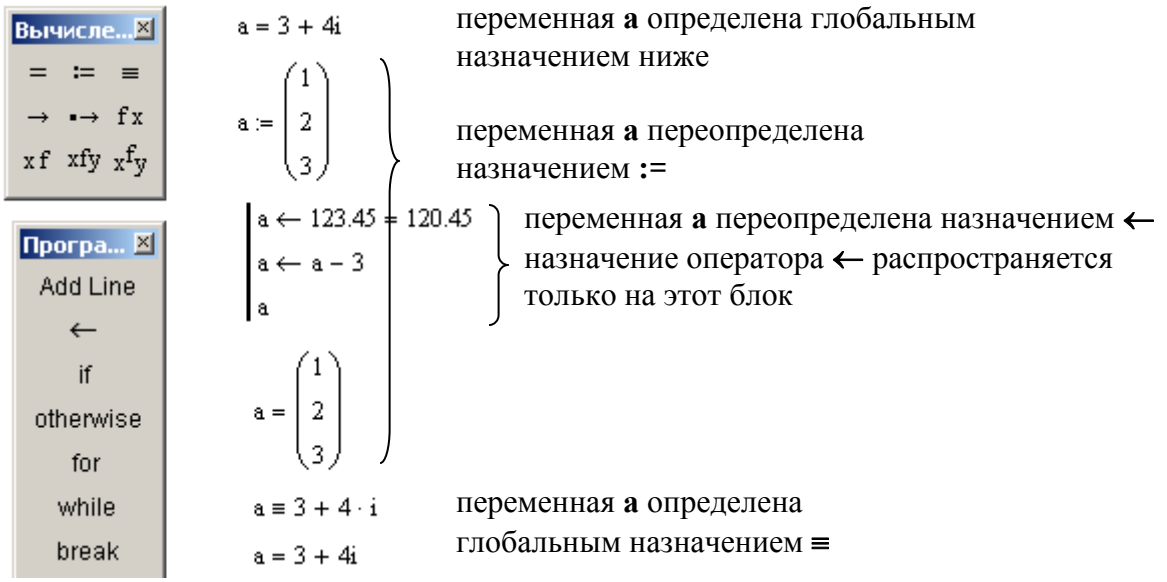


Рис. 4 – Области действия (видимости) операторов назначения

Оператор ← предназначен для использования в изолированных блоках, формируемых при программировании в пределах одного региона. Действие этого оператора распространяется только на тот регион, в котором он расположен. На *Рис. 4* показан пример переопределения типа и значения объекта ***a***.

Действие оператора глобального назначения ≡ распространяется на все пространство рабочего листа. Появление в регионе оператора назначения **:=** вызывает переопределение типа объекта, сохраняющееся до появления следующего оператора назначения, которым может быть либо еще один оператор :=, либо (как в примере) оператор глобального назначения **≡**.

Оператор локального назначения ← не оказывает никакого действия на тип и значение объекта **a** за пределами блока, выделенного слева жирной вертикальной чертой. Такие блоки создаются при использовании инструментальной *панели «Программирование»* и занимают один регион.

Определенные однажды объекты могут быть переопределены в последующих регионах. Таким образом, при интерпретации действий с объектом актуально последнее, ближайшее к точке выполнения определение его типа. Если на рабочем листе есть несколько операторов глобального назначения **≡**, то свойство «глобальности» сохраняет только последний из них.

MathCAD содержит несколько сотен встроенных стандартных функций, которые можно использовать в расчетах. Для вставки функции нужно нажать пиктограмму или в меню выбрать опцию «*Вставить Функцию*» – откроется окно выбора (*Рис. 5*), где все функции размещены по категориям.

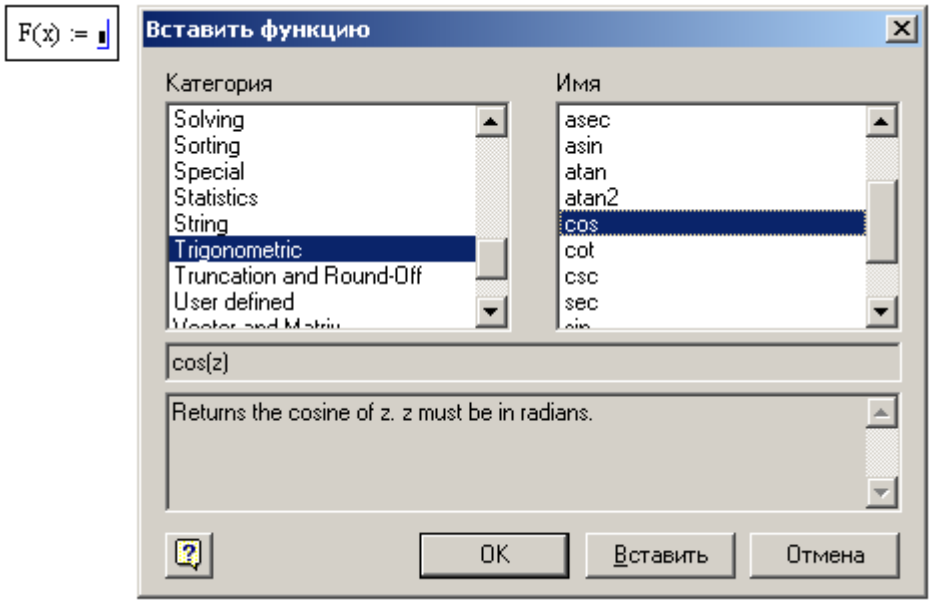


Рис. 5 – Диалоговое окно выбора стандартной функции из библиотеки

Операторы индикации значений

Весьма часто при выполнении вычислений требуется выводить на экран значения того или иного объекта, либо результата выполнения выражения. Эта задача решается *операторами индикации значений*. В *MathCAD* существуют два вида операторов индикации:

- = оператор индикации числовых значений;
- → оператор индикации символьных значений.

$x := \frac{1}{3}$ $x \rightarrow \frac{1}{3}$

$x = 3.3333333333 \times 10^{-1}$

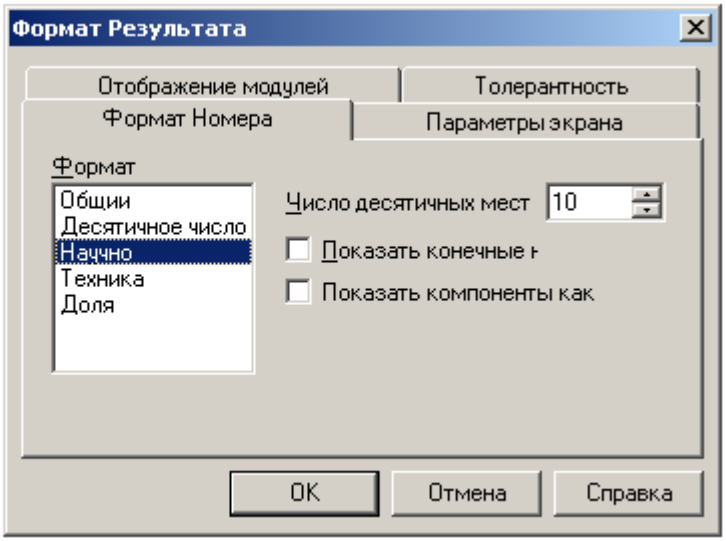


Рис. 6 – Диалоговое окно задания формата вывода чисел

Оператор индикации символьных значений → применяется для вывода результатов символьных преобразований, значений объектов, в составе которых имеются компоненты с неопределенными числовыми значениями. Такими объектами являются, например, функции с формальными параметрами, формульные выражения при выполнении над ними операций *символьных преобразований* (*Рис. 6*).

1.5. Вывод графиков в MathCAD

Вывод графической информации производится в графических регионах с использованием *шаблонов*, создаваемых инструментами панели *«Графики»*. На рисунке показан начальный вид шаблона в прямоугольной (декартовой) системе координат (значок  или комбинация клавиш *Shift+2*). Здесь изображена прямоугольная область «экрана», на которой будет строиться график, а также имеются поля ввода выражения для переменной, значения которой отображаются по вертикальной оси (*Поле Y – ордината*) и переменной, отображаемой по горизонтальной оси (*Поле X – абсцисса*).

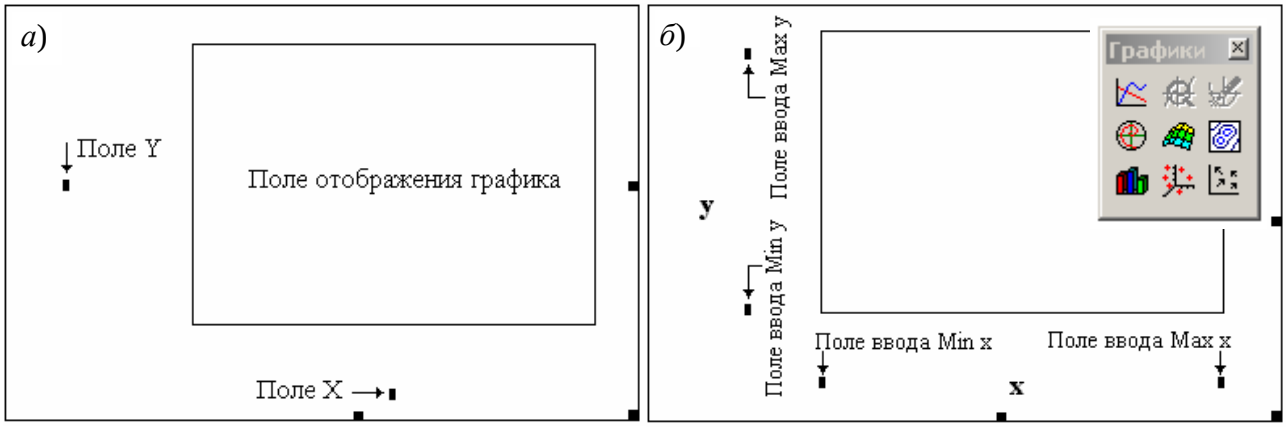


Рис. 7 – Шаблон вывода графика

После занесения выражений отображаемых переменных на шаблоне появляются поля ввода *границ диапазонов* этих переменных (*Min X, Max X, Min Y, Max Y*), в пределах которых происходит построение графика (Рис. 7, *б*). Если выражение для $y(x)$ определено к моменту заполнения шаблона, то график выдается сразу после его занесения в *Поле Y*, при этом границы изменения x устанавливаются по умолчанию, а границы изменения y определяются автоматически.

В *MathCAD* при построении графиков непрерывных зависимостей $y(x)$ производится автоматический выбор шага независимой переменной x. Если требуется, выбора шага построения графика можно *задать принудительно*. Для этого надо перед графическим регионом ранжировать массив x переменных с требуемым шагом изменения, например, $x:= 0, 0.1 .. 1$. В этом случае при построении графика значения $y(x)$ будут вычисляться только в точках $x_1=0$; $x_2=0.1$; $x_3=0.2$ и т.д. до $x_{11}=1$. Массивы значений $(x_i, y(x_i))$ будут образовывать узловые точки графика, по умолчанию узловые точки графика будут соединены отрезками прямой.

Графические регионы позволяют строить в одних осях несколько графиков одновременно. Для этого необходимо в *Поле Y* поместить *список выводимых переменных, разделенных запятыми*. Запятая при этом не отображается, а каждое имя вводимой переменной помещается на новую строку. На рисунке (*Рис. 8, а*) приведен пример графика, отображающего две различные зависимости (от одной переменной x).

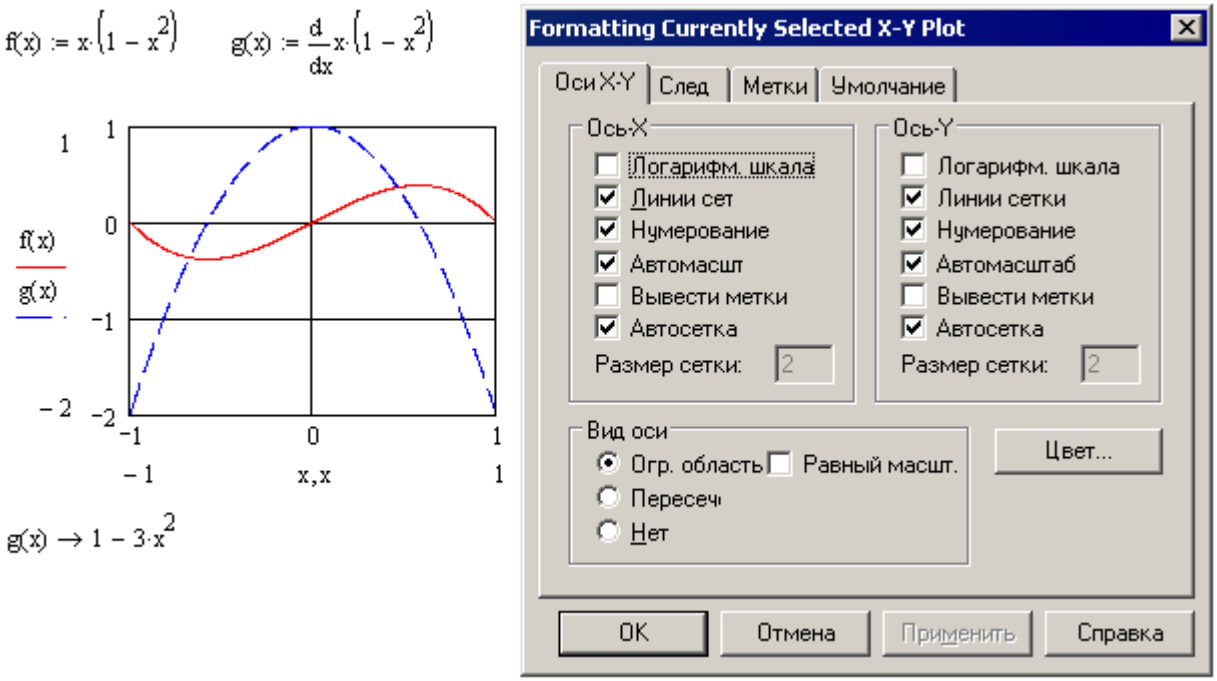


Рис. 8 – Вывод нескольких графиков в одних осях. Панель настройки графика

Абсциссы графиков так же могут быть представлены разнотипными переменными – как дискретными, так и непрерывными, они так же должны перечисляться через запятую.

График в *MathCAD* допускает гибкую настройку. Двойной клик в поле шаблона вызывает на экран диалоговое окно настройки с четырьмя вкладками (Рис. 8, б). Назначение большинства элементов управления, размещенных на вкладках окна, понятно из соответствующих подписей.

На вкладке *X-Y Axes* устанавливаются параметры осей графика и размерной сетки в его поле. Вкладка *Traces* позволяет настроить параметры линий, отображающих графики. Здесь можно выбрать тип графика (*Type*), цвет отображения графика (*Color*), вид линии (*Line*), вид символа, помечающего положение узловых точек (*Symbol*).

1.6. Исследование функции при помощи пакета MathCAD

Общая схема исследования функции известна еще из курса средней школы:

1. Найти область определения функции. Выделить особые точки (точки разрыва).
2. Проверить наличие вертикальных асимптот в точках разрыва и на границах области определения.
3. Найти точки пересечения с осями координат.
4. Установить, является ли функция чётной или нечётной.
5. Определить, является ли функция периодической или нет;
6. Найти точки экстремума и интервалы монотонности (возрастания и убывания) функции.
7. Найти точки перегиба и интервалы выпуклости-вогнутости.
8. Найти наклонные асимптоты функции.
9. Построить график функции.

Рассмотрим, насколько облегчается этот процесс при использовании MathCAD. Рассмотрим сначала последний (целевой) пункт списка – построение графика. Средства MathCAD позволяют начать(!) исследование функции с построения графика. Согласитесь, это значительно проще и нагляднее. Однако, имейте ввиду, что построение графика – это еще не конец исследований, это начало.

Исследование области определения

Это очень важный шаг исследования функции, так как все дальнейшие действия будут проводиться на области определения.

В приложении к элементарным функциям и их суперпозициям, учитываются следующие знания:

- для дробных функций $f(x)/g(x)$ необходимо найти нули знаменателя и исключить их из области определения $g(x) \neq 0$;
- для логарифмической функции $\log_a(g(x))$ – область определения задается неравенством $g(x) > 0$;
- для корня $\sqrt[n]{g(x)}$ четной степени n – область определения находится из неравенства $g(x) \geq 0$.

Если имеется возможность сразу же оценить область значений функции, то это необходимо сделать.

Вычисление пределов функции в MathCAD

Самое главное при исследовании области определения – осмыслить поведение функции на ее границах – в особых точках и на бесконечности (при **x**→±∞). Для этого используется такой математический аппарат, как *предел (limit) функции*, при этом используется шаблон ***предел функции*** и ***предел функции справа (слева)***: из вкладки ***calculus (мат.анализ).***

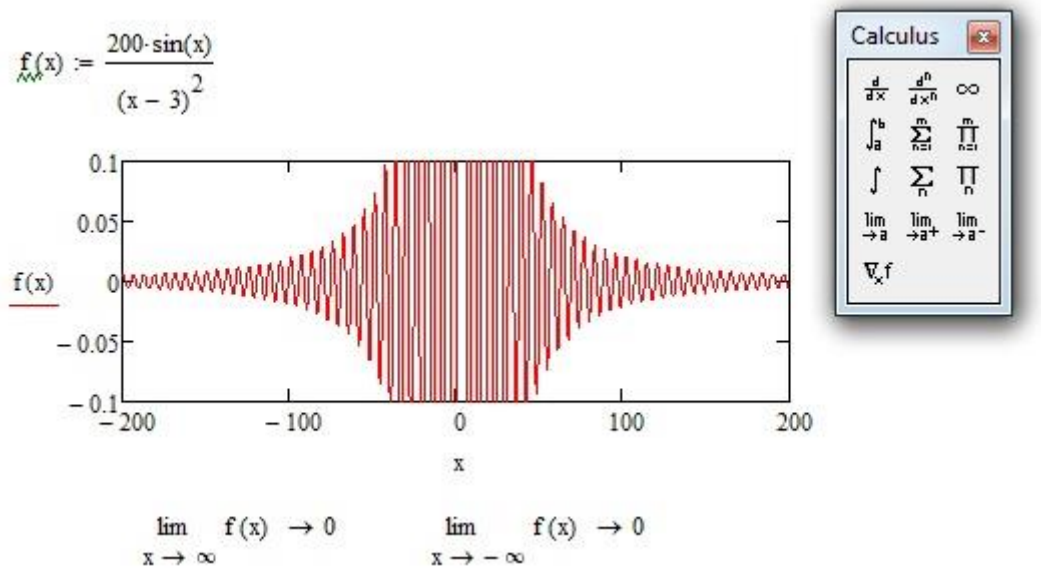


Рис. 9 - Вычисление пределов функции на +∞ и на -∞

Из графика не всегда ясно, как ведет себя функция в тех или иных точках. Например, на *Рис. 9*, четко видно, что при **x**→∞ и при **x**→**-**∞ функция **f(x)** бесконечно убывает, что и подтверждается вычисленными пределами. Но поведение **f(x)** вблизи особой точки **x=3** из графика не ясно. Изменив масштаб графика можно более подробно рассмотреть особую точку (см. *Рис. 10* - это та же самая функция, что и на *Рис. 9*). Рассмотрим пределы справа и слева в этой точке **x=3.**

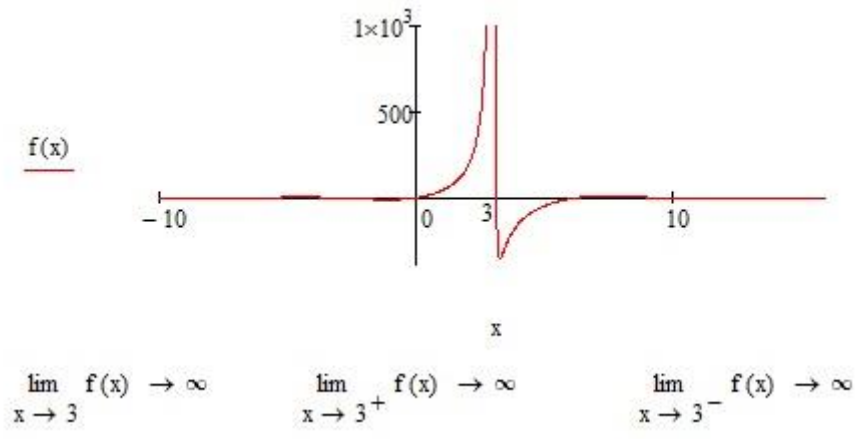


Рис. 10 - Вычисление пределов функции справа и слева в точке

Пределы могут быть вычислены не всегда. Рассмотрим *Рис. 11*: заданная здесь функция **g(x)** так же имеет особую точку **x=3**, однако пределы при приближении слева и справа к этой точке различные, а потому общий предел не существует (*undefintd*).

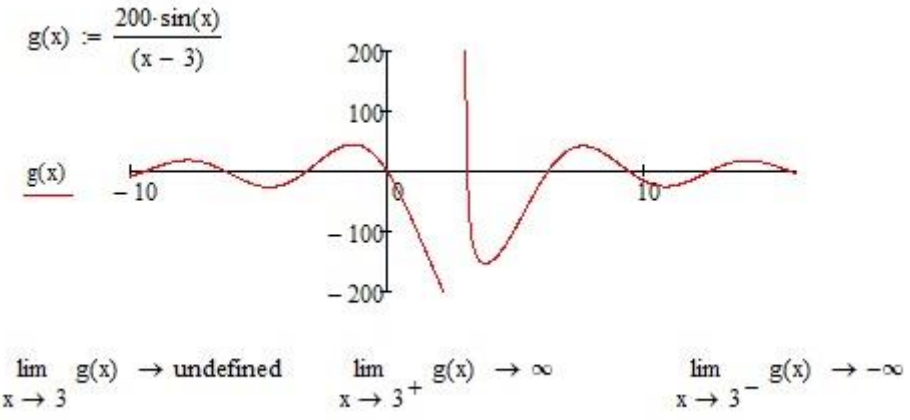
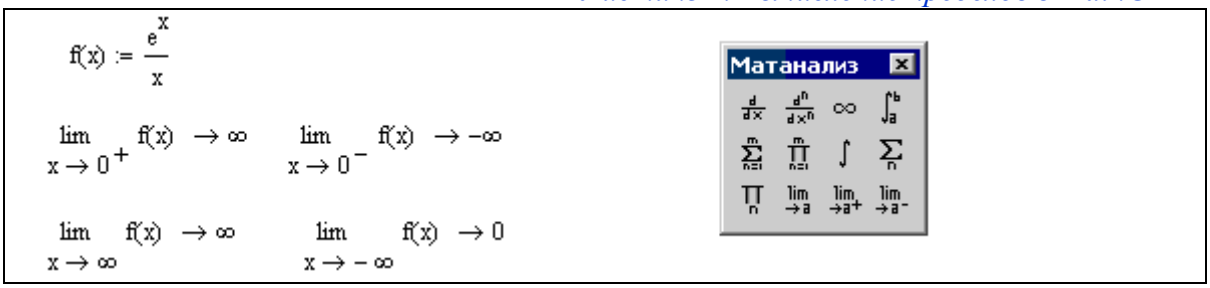


Рис. 11 - Вычисление пределов функции справа и слева в точке

Листинг 2. Вычисление пределов в MathCAD



Область определения приведенной на листинге функции: $x \neq 0$, то есть $(-\infty, 0) \cup (0, +\infty)$. С учетом вычисленных пределов, можно заключить, что прямая $x = 0$ – вертикальная асимптота, при $x \to +\infty$ функция стремится вверх со скоростью экспоненты, а при $x \to -\infty$ монотонно возрастает бесконечно, приближаясь к нулю снизу.

Поиск корней функции

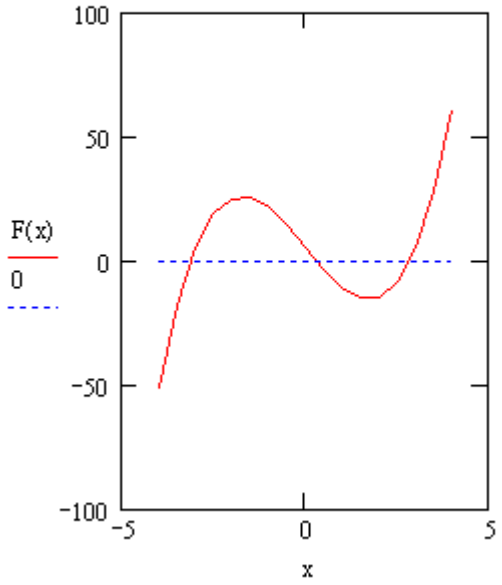
Задача решения нелинейных (трансцендентных) уравнений состоит в поиске всех значений переменных, удовлетворяющих данному уравнению. На листинге ниже (*Листинг 3*) приводится пример поиска корней кубической параболы $F(x)$, на интервале $x \in [-5, 5]$. Этот способ визуальный и очень грубый.

Корни могут быть уточнены методом табуляции – вычисления значений функции в точках x, выбираемых из интервала $[-4, 4]$ с шагом 0.5. Точность вычисления корней при таком методе не превосходит ε=0.5. Для выделения корня с заданной точностью необходимо повторить вычислительный эксперимент требуемое число раз, уменьшая каждый раз границы интервала табуляции, шаг табуляции и точность вывода полученных значений на экран (*Рис. 6*).

Листинг 3. Табуляция функции и построение графика

$x := -4, -3.5 .. 4 \quad a := 2 \quad b := -18 \quad c := 5 \quad F(x) := a \cdot x^3 + b \cdot x + c$

x =	F(x) =
-4	-51
-3.5	-17.75
-3	5
-2.5	18.75
-2	25
-1.5	25.25
-1	21
-0.5	13.75
0	5
0.5	-3.75
1	-11
1.5	-15.25
2	-15
2.5	-8.75
3	5
3.5	27.75



В пакете *MathCAD* существует встроенная функция *root*(*f*(*x*),*x*), выполняющая такие действия автоматически. Нужно только *явно* задать функцию *f*(*x*) и начальное значение x_0, от которого начинается *итерационное* численное уточнение корня.

В примере (*Листинг 4*) приведены примеры уточнения корня трансцендентной функции $f(x) = x^2 \cdot \sin(x) - e^{-x}$, для различных начальных условий.

Можно видеть, что значения отличаются друг от друга, начиная с четвертого знака после запятой. Это связано с тем, что точность расчетов по умолчанию имеет значение $TOL = 10^{-3}$.

Численные расчеты всегда выполняются с определенной точностью. Эта точность задается константой *TOL*.

Меняя принудительно значение *TOL*, как в примере (*Листинг 4*), можно управлять точностью вычислений.

Листинг 4. Поиск корней функции. Точность численного метода

$f(x) := x^2 \cdot \sin(x) - e^{-x}$

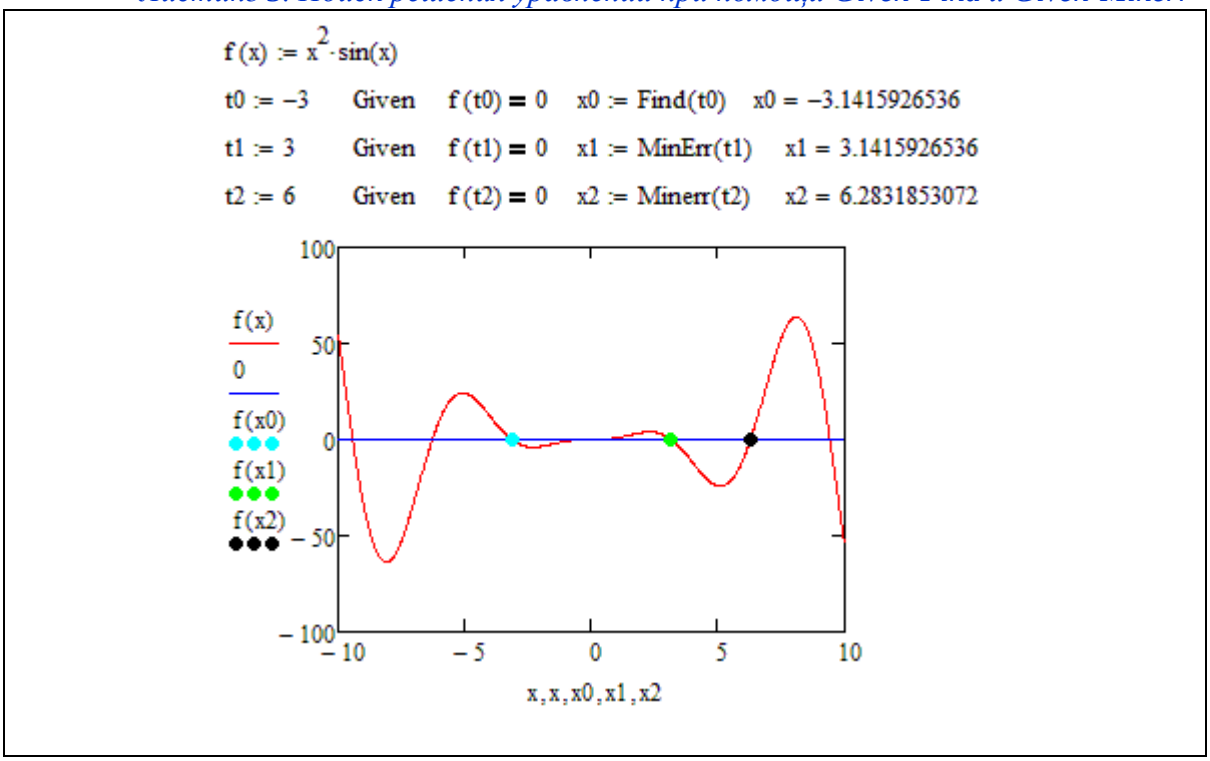
$TOL := 10^{-6}$

z := 0.7951	root(f(z),z) = 0.79519783	z := 0.7951	root(f(z),z) = 0.79519788
z := 0.1	root(f(z),z) = 0.7951641	z := 0.1	root(f(z),z) = 0.79519783
z := 0.9	root(f(z),z) = 0.79563728	z := 0.9	root(f(z),z) = 0.79519788

Аналитические функции поиска корней нелинейного уравнения

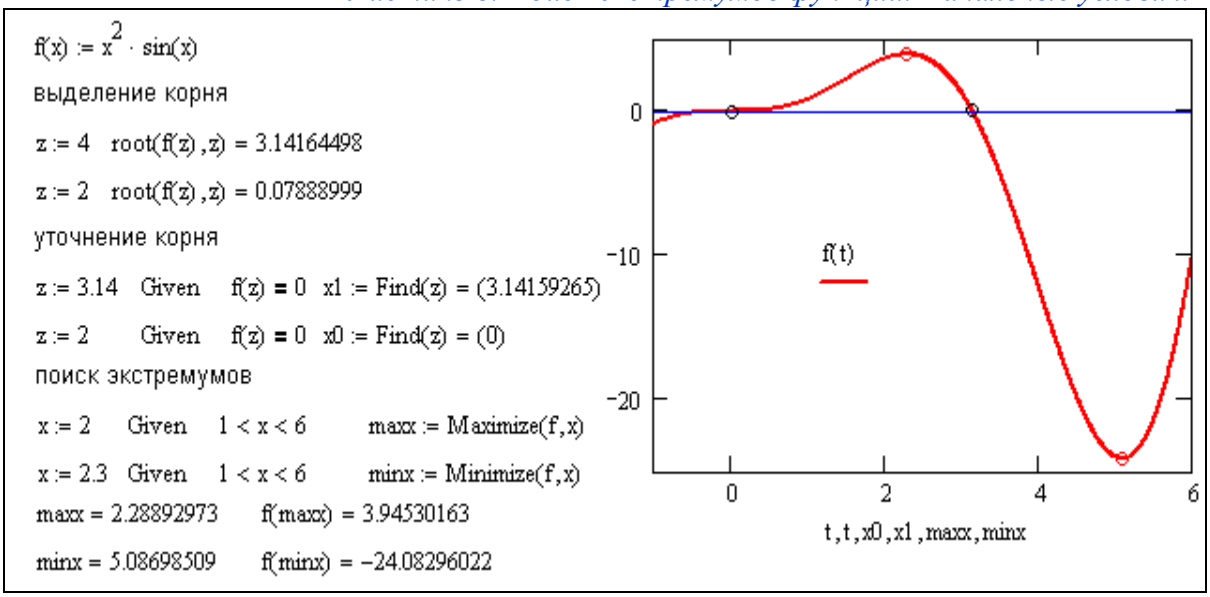
Пара операторов *Given-Find* и *Given-Minerr* позволяет находить корни уравнения *аналитически* (см. *Листинг 5*). Здесь точность вычисления корня не зависит от *TOL*, а ограничивается точностью округления вещественных чисел.

Листинг 5. Поиск решения уравнений при помощи Given-Find и Given-Minerr



Для автоматизированного поиска экстремумов функции используются операторы *Given-Minimize* и *Given-Maximize* (*Листинг 6*). Для выполнения поиска экстремумов необходимо задать *начальное приближение x*, так как максимумов и минимумов у функции может быть несколько, и поиск производится в окрестности начального приближения.

Листинг 6. Поиск экстремумов функции. Начальные условия.



Кроме того, диапазон поиска экстремума можно ограничить при помощи неравенства-ограничения, накладываемого на начальное приближение *x*. Список неравенств-ограничений (в предыдущем примере он такой: $1 < x < 6$) располагается после оператора *Given*, но перед *Minimize* или *Maximize*.

Построение точек экстремума и точек перегиба

Из определения и геометрических свойств производной функции в точке следует, что изменение знака первой производной определяет участки монотонности функции и точки экстремума, а знак второй производной – участки выпуклости (вогнутости) и точки перегиба.

В MathCAD существует встроенный инструмент, позволяющий *аналитически находить производную любой функции*, составленной из произвольной комбинации элементарных функций. Если же этого не достаточно, то существует возможность *численно находить производную любой функции* в точке (из области определения).

Как видно из примера (*Листинг 7*), корни полинома $p(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n$ можно найти используя встроенную функцию ***polyroots***(***v***). В качестве аргумента функции *polyroots* должен вводиться вектор коэффициентов полинома $\boldsymbol{v} = (a_0, a_1, a_2, ..., a_n)$. А результат действия функции *polyroots* – это вектор корней полинома $p(x)$, причем, кратные корни встречаются положенное количество раз.

В примере (*Листинг 7*) опущен процесс поиска корней производной функции $dp(x)$ – точек экстремума и корней второй производной $dp(x)$ – точек перегиба: этот поиск производится аналогично при помощи функции *polyroots*.

Исследование полинома – самый простой из примеров исследования функции. Полином 5 степени, очевидно, имеет 5 корней – $(x_1 = -5,\ x_2 = -1,\ x_3 = -1,\ x_4 = 2,\ x_5 = 3)$, и (так как первая производная имеет 4-ю степень) – 4 экстремума: $(x'_1 = -3.981,\ x'_2 = -1,\ x'_3 = 0.797,\ x'_4 = 2.584)$. Кроме того, он имеет 3 точки перегиба $(x''_1 = -2.943,\ x''_2 = -0.144,\ x''_3 = 1.887)$, так как вторая производная имеет 3-ю степень.

Листинг 7. Исследование экстремумов и перегибов

$p(x) := x^5 + 2 \cdot x^4 - 18 \cdot x^3 - 8 \cdot x^2 + 41 \cdot x + 30$

v_coef_p := p(x) coeffs,x

x_res := polyroots(v_coef_p)

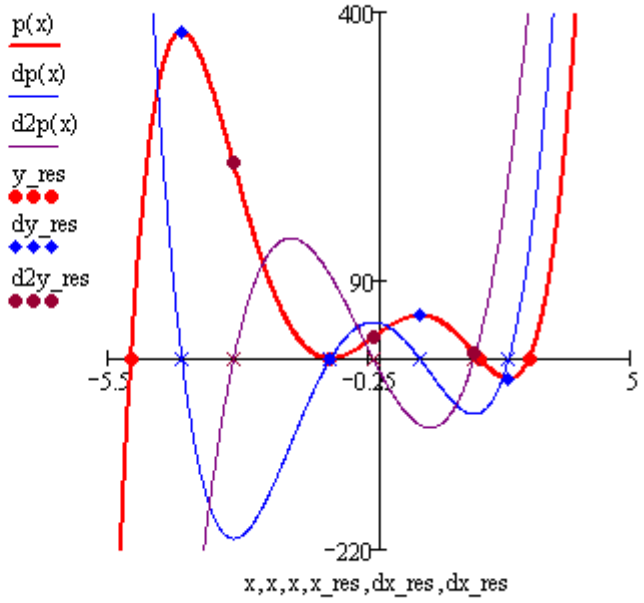
$dp(x) := \frac{d}{dx} p(x)$

$dp(x) \rightarrow 5 \cdot x^4 + 8 \cdot x^3 - 54 \cdot x^2 - 16 \cdot x + 41$

$d2p(x) := \frac{d}{dx} dp(x)$

$d2p(x) \rightarrow 20 \cdot x^3 + 24 \cdot x^2 - 108 \cdot x - 16$

$x\_res = \begin{pmatrix} -5 \\ -1 \\ -1 \\ 2 \\ 3 \end{pmatrix}$ v_coef_p $\rightarrow \begin{pmatrix} 30 \\ 41 \\ -8 \\ -18 \\ 2 \\ 1 \end{pmatrix}$



Возможность автоматически строить производные практически любой функции в комплексе с богатым набором средств *отыскания корней* функции делают задачу отыскания точек экстремума и точек перегиба чрезвычайно легкой. Для этого необходимо:

• вычислить $dp(x) = df(x)/dx$ – первую производную функции $f(x)$ и найти ее корни аналитически или численно;
• найденные точки – *точки экстремума*, а интервалы между ними и границами области определения – *участки монотонности*;
• вычислить $ddp(x) = d^2 f(x)/dx^2 = dp(x)/dx$ – вторую производную функции $f(x)$ (или первую производную от $dp(x)$) и найти ее корни;
• найденные точки – *точки перегиба*, а участки, расположенные между *точками перегиба* и границами области определения – участки вогнутости или выпуклости.

Листинг 8. Исследование экстремумов и перегибов

$y(x) := \ln\frac{x+1}{x+2}$; область определения $D(y) = (-\infty;-2)\cup(-1;+\infty)$.

Исследуем функцию на экстремумы и монотонность. Вычислим первую производную:

$$y(x) := \ln\left(\frac{x+1}{x+2}\right) \quad dy(x) := \frac{d}{dx}y(x)\ \text{simplify} \rightarrow \frac{1}{(x+1)\cdot(x+2)} \quad dy(x) \rightarrow \frac{1}{(x+1)\cdot(x+2)}$$

Находим особые точки $x_1 = -1$, $x_2 = -2$. Определим знак производной на интервалах, на которые особые точки делят область определения функции:

$dy(5) = 0.024 \quad dy(-5) = 0.083$

$y(x)$	*возрастает*		*возрастает*	
$y'(x)$	+	-2	-1	+

x

Функция возрастает на интервалах $(-\infty;-2)$, $(-1;+\infty)$. Экстремумов нет.

Исследуем функцию на выпуклость и точки перегиба. Найдем вторую производную:

$$ddy(x) := \frac{d^2}{dx^2}y(x)\ \text{simplify} \rightarrow \frac{-(2\cdot x+3)}{(x+2)^2\cdot(x+1)^2} \quad ddy(x) \rightarrow \frac{(-2)\cdot x-3}{(x+2)^2\cdot(x+1)^2}$$

Особые точки $x_1 = -1$, $x_2 = -2$, $x_3 = -1.5$. Исследуем знак второй производной на интервалах, на которые особые точки делят область определения функции:

$ddy(5) = -7.37\times10^{-3} \quad ddy(-5) = 0.049$

$y(x)$	*выпукла вниз*		*выпукла вверх*	
$y''(x)$	+	-2	-1	–

x

Функция выпукла вниз на интервале $(-\infty;-2)$ и выпукла вверх на $(-1;+\infty)$. Перегибов нет.

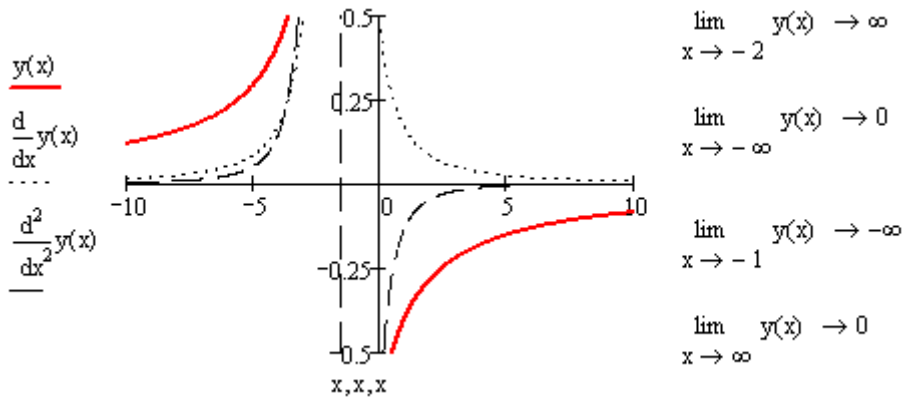


Рис. 12 – Исследование экстремумов и перегибов (2)

Листинг 9. Исследование экстремумов и перегибов

$y(x) := \frac{e^x}{x}$; область определения $D(y) = (-\infty; 0) \cup (0; +\infty)$.

Рассмотрим поведение функции в особой точке:

$\lim_{x \to 0^+} y(x) \to \infty$ $\lim_{x \to 0^-} y(x) \to -\infty$. $y(x) \equiv 0$ – вертикальная асимптота.

Исследуем функцию на экстремумы и монотонность. Вычислим первую производную:

$y(x) := \frac{e^x}{x}$ $dy(x) := \frac{d}{dx} y(x)$ simplify $\to e^x \cdot \frac{x-1}{x^2}$ $dy(x) \to e^x \cdot \frac{x-1}{x^2}$

Находим точки экстремума:

Given $dy(t) = 0$ $x0 := Find(t)$ $x0 = 1$ $y(x0) \to e$ $y0 := y(x0)$ $y0 = 2.718$

Первая производная пересекает ось OX в точке $(1, e)$ – экстремум. Определим знак производной на интервалах, на которые особые точки делят область определения функции:

$y(x)$	*убывает*		*убывает*		*возрастает*	
$y'(x)$	–	0	–	1	+	x

Функция убывает на интервалах $(-\infty; 0)$, $(0; 1]$ и возрастает на интервале $[1; +\infty)$. Точка $(1, e)$ – точка минимума.

Исследуем функцию на выпуклость и точки перегиба. Найдем вторую производную:

$ddy(x) := \frac{d^2}{dx^2} y(x)$ simplify $\to e^x \cdot \frac{x^2 - 2 \cdot x + 2}{x^3}$ $ddy(x) \to e^x \cdot \frac{x^2 - 2 \cdot x + 2}{x^3}$

Особая точка $x = 0$, действительных корней нет. Исследуем знак второй производной на интервалах, на которые особые точки делят область определения функции:

$ddy(-3) = -0.031$ $ddy(3) = 3.72$

$y(x)$	*выпукла вверх*		*выпукла вниз*	
$y''(x)$	–	0	+	x

Функция выпукла вверх на интервале $(-\infty; 0)$ и выпукла вниз на $(0; +\infty)$. Перегибов нет.

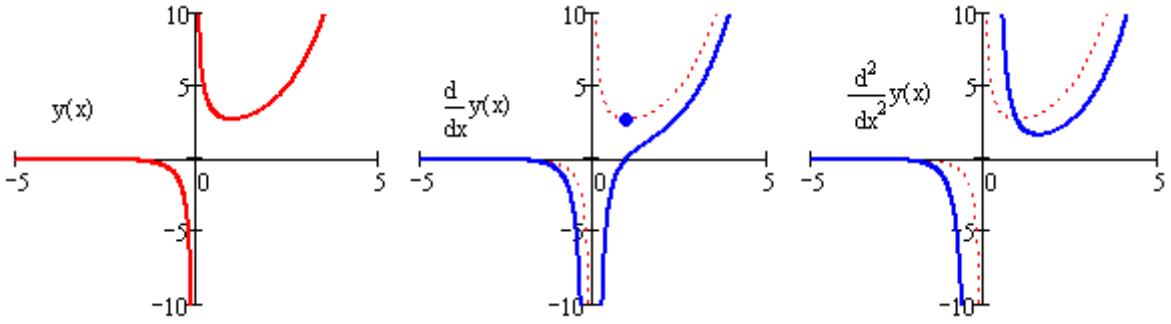


Рис. 13 – Исследование экстремумов и перегибов (3)

Листинг 10. Исследование экстремумов и перегибов

$y(x) := \frac{x^3}{x^2 - 1}$; область определения $D(y) = (-\infty; -1) \cup (-1; 1) \cup (1; +\infty)$.

Рассмотрим поведение функции в особых точках:

$\lim_{x \to -1^-} y(x) \to -\infty$ $\lim_{x \to -1^+} y(x) \to \infty$ $\lim_{x \to 1^-} y(x) \to -\infty$ $\lim_{x \to 1^+} y(x) \to \infty$

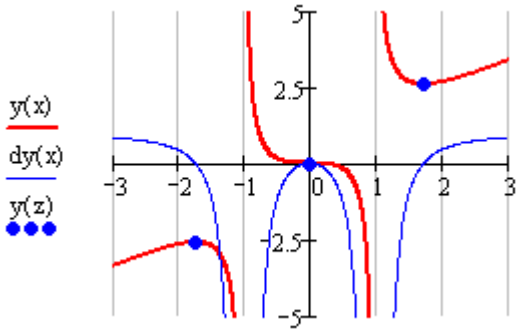
$y(x) \equiv -1$ и $y(x) \equiv 1$ – вертикальные асимптоты.

Исследуем функцию на экстремумы и монотонность. Вычислим первую производную:

$y(x) := \frac{x^3}{x^2 - 1}$ $dy(x) := \frac{d}{dx} y(x)$ simplify $\to x^2 \cdot \frac{x^2 - 3}{(x^2 - 1)^2}$

Находим точки экстремума:

$t := 2$ Given $dy(t) = 0$ $x0 := \text{Find}(t)$ $x0 \to \begin{pmatrix} & & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 3 & -3 \end{pmatrix}$ $z := \begin{pmatrix} x0 \\ 0 \\ -x0 \end{pmatrix}$ $z = \begin{pmatrix} 1.732 \\ 0 \\ -1.732 \end{pmatrix}$



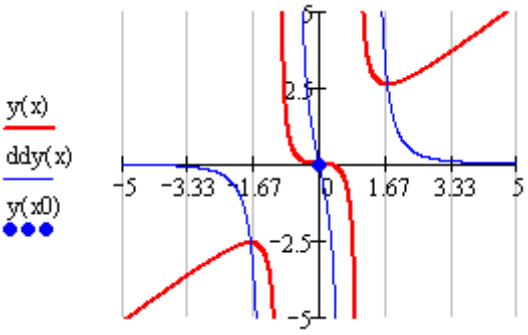
Первая производная пересекает ось *OX* в трех точках $\left(-\sqrt{3}, -1.5\sqrt{3}\right)$, $(0,0)$ и $\left(\sqrt{3}, 1.5\sqrt{3}\right)$ – экстремумы. Определим знак производной на интервалах, на которые особые точки делят область определения функции.

Функция возрастает на интервалах $\left(-\infty; -\sqrt{3}\right]$ и $\left[\sqrt{3}; \infty\right)$ и убывает на интервалах $\left[-\sqrt{3}; -1\right)$, $(-1, 0]$, $[0; 1)$ и $\left(1, \sqrt{3}\right]$.

Точка $\left(-\sqrt{3}, -1.5\sqrt{3}\right)$ - максимум, а $\left(\sqrt{3}, 1.5\sqrt{3}\right)$ – минимум.

Исследуем функцию на выпуклость и точки перегиба. Найдем вторую производную:

$ddy(x) := \frac{d^2}{dx^2} y(x)$ simplify $\to 2 \cdot x \cdot \frac{x^2 + 3}{(x^2 - 1)^3}$ $tt := 0.5$ Given $ddy(tt) = 0$ $x0 := \text{Find}(tt)$ $x0 = 0$



Точка перегиба $x = 0$. Исследуем знак второй производной на интервалах, на которые особые точки делят область определения функции:

$y(x)$	*вверх*		*вниз*		*вверх*		*вниз*	
$y''(x)$	–	-1	+	0	–	1	+	x

Функция выпукла вверх на интервалах $(-\infty; -1)$ и $(-1; 0]$ и выпукла вниз на участках $[0; 1)$ и $(1; +\infty)$. Точка перегиба $x = 0$.

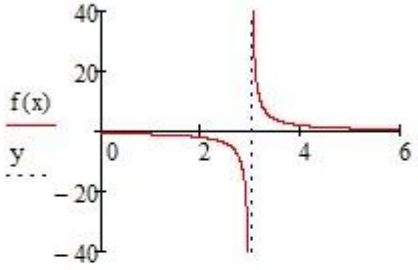
Построение асимптот

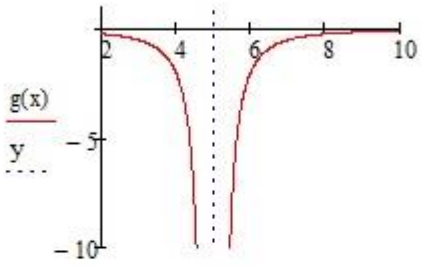
Асимптоты – это прямые линии, к которым приближается график функции $f(x)$ при $x \to \pm\infty$ или при $x \to x^* \pm 0$ – на бесконечности или на границах x^* области определения.

Вертикальные асимптоты – прямые, параллельные оси *OY*, задаваемые уравнением $x = x^*$, причем $\lim_{x \to x^*+0} f(x) = \pm\infty$ или $\lim_{x \to x^*-0} f(x) = \pm\infty$. При приближении к точке $x \to x^* \pm 0$ справа или слева функция $f(x)$ стремится к бесконечности.

Листинг 11. Построение асимптот: вертикальные асимптоты

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





$\lim_{x \to 3} f(x) \to \text{undefined}$ $\quad \lim_{x \to 5} g(x) \to -\infty$

$\lim_{x \to 3^+} f(x) \to \infty$ $\quad \lim_{x \to 5^+} g(x) \to -\infty$

$\lim_{x \to 3^-} f(x) \to -\infty$ $\quad \lim_{x \to 5^-} g(x) \to -\infty$

Горизонтальные или наклонные асимптоты ищутся в виде прямых $y = kx + b$, где $k = \lim_{x \to \pm\infty} \frac{f(x)}{x}$ – тангенс угла наклона асимптоты, и $b = \lim_{x \to \pm\infty} \left(f(x) - kx \right)$ - смещение по вертикальной оси.

Горизонтальные или наклонные асимптоты следует искать лишь тогда, когда функция определена на бесконечности.

Если k=0 и $b \neq \infty$, то наклонная асимптота станет горизонтальной.

Если горизонтальных или наклонных асимптот нет, но функция определена на всей числовой оси, то следует вычислить предел функции на плюс бесконечности и (или) минус бесконечности, чтобы иметь представление о поведении графика функции в пределе.

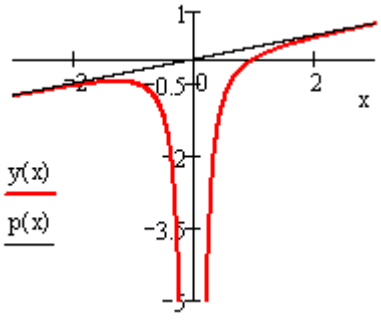
Рассмотрим примеры:

Листинг 12. Построение асимптот: наклонная асимптота

$y(x) := \frac{x^3 - 1}{4x^2}$ $\quad k := \lim_{x \to \infty} \frac{y(x)}{x} \to \frac{1}{4}$

$b := \lim_{x \to \infty} (y(x) - k \cdot x) \to 0$

$p(x) := k \cdot x + b$ $\quad p(x) \to \frac{1}{4} \cdot x$

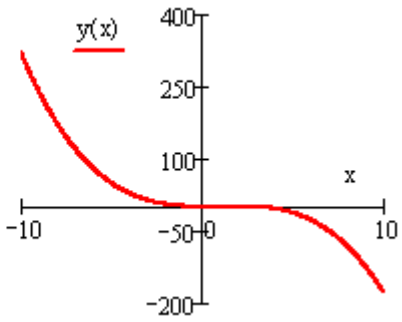


Наклонная асимптота $p(x) = \frac{1}{4}x$, вертикальная асимптота $x \equiv 0$.

Листинг 13. Построение асимптот: асимптот нет

$$y(x) := \frac{-1}{4}\cdot\left(x^3 - 3x^2 + 4\right) \qquad \lim_{x\to\infty}\frac{y(x)}{x} \to -\infty$$

асимптот нет



Листинг 14. Построение асимптот

$$y(x) := \frac{x^3}{x^2 - 1} \qquad \lim_{x\to\infty}\frac{y(x)}{x} \to 1$$

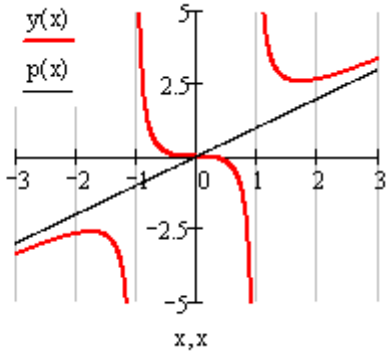
$$k := \lim_{x\to-\infty}\frac{y(x)}{x} \to 1$$

$$b := \lim_{x\to-\infty}(y(x) - k\cdot x) \to 0$$

$$p(x) := k\cdot x + b \quad p(x) \to x$$

$$\lim_{x\to -1^-} y(x) \to -\infty \qquad \lim_{x\to -1^+} y(x) \to \infty$$

$$\lim_{x\to 1^-} y(x) \to -\infty \qquad \lim_{x\to 1^+} y(x) \to \infty$$



Наклонная асимптота $p(x) = x$, две вертикальных асимптоты $x \equiv -1$ и $x \equiv 1$.

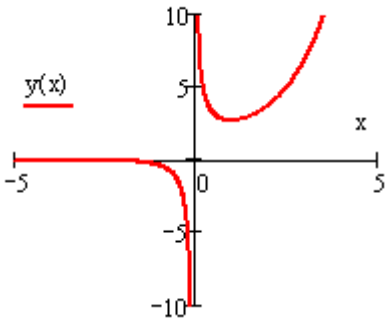
Листинг 15. Построение асимптот

$$y(x) := \frac{e^x}{x} \qquad \lim_{x\to\infty}\frac{y(x)}{x} \to \infty$$

$$k := \lim_{x\to-\infty}\frac{y(x)}{x} \to 0$$

$$b := \lim_{x\to-\infty}(y(x) - k\cdot x) \to 0$$

$$p(x) := k\cdot x + b \quad p(x) \to 0$$



Горизонтальная асимптота $y \equiv 0$, вертикальная асимптота $x \equiv 0$ при $x \to -\infty$.

Данный параграф иллюстрирует, насколько облегчается задача исследования функции с использованием такого мощного инструмента как MathCAD. Выполняемая практическая работа служит для закрепления навыков исследования функций.

Интегралы

В электронике интегралы используются для решения дифференциальных уравнений, описывающих поведение токов и напряжений в электрических цепях, а также вычисления интегральных показателей, таких как работа и мощность.
Система MathCAD позволяет производить значительное количество расчетов с таким математическим объектом, как интеграл. Это, в частности, вычисление неопределенного интеграла (первообразной функции):

Листинг 16. Вычисление первообразной функции с проверкой

$$f(x) := \frac{2}{x-3} \qquad \int f(x)\,dx \to 2\cdot\ln(x-3) \qquad \frac{d}{dx}(2\cdot\ln(x-3)) \to \frac{2}{x-3}$$

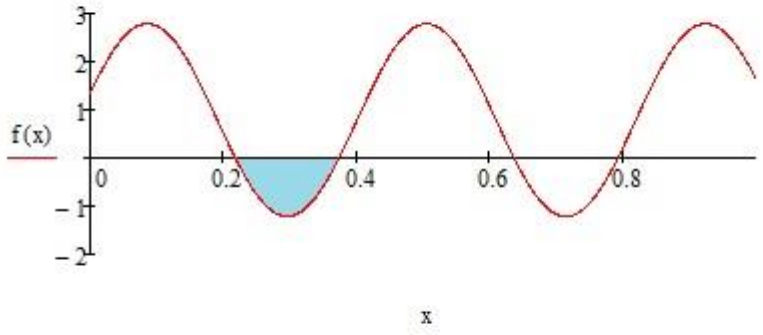
$$g(x) := \frac{-2}{(x-5)^2} \qquad \int g(x)\,dx \to \frac{2}{x-5} \qquad \frac{d}{dx}\frac{2}{x-5} \to -\frac{2}{(x-5)^2}$$

Имеется возможность находить определенный интеграл с заданием пределов интегрирования и даже несобственные интегралы. В примере ниже требуется вычислить площадь криволинейной фигуры (отмеченной голубым цветом), ограниченной графиком заданной функции - для этого используется определенный интеграл.

Сначала (см.) необходимо вычислить пределы интегрирования а это, очевидно, корни функции, близкие к $x_1=0.21$ и $x_2=0.35$.

Листинг 17. Вычисление площади фигуры с использованием определенного интеграла

$$TOL := 10^{-12} \qquad f(x) := 2\sin(15x-6)+\frac{\pi}{4}$$



$$x1 := 0.21 \qquad root(f(x1),x1) = 0.2174647969$$

$$x2 := 0.35 \qquad root(f(x2),x2) = 0.3730956929$$

$$\int_{0.2174647969}^{0.3730956929} f(x)\,dx = -0.1230123193$$

Вычислив корни функции ***f(x)*** при помощи функции **root(),** используем шаблон определенный интеграл, в который необходимо вписать интегрируемую функцию, пределы интегрирования и переменную интегрирования.

Следует отметить, что в зависимости от расположения кривой и оси *OX*, интеграл может получаться положительным и отрицательным, но площадь фигуры отрицательной быть не может.

Контрольные вопросы:

1. Что представляет из себя проект (программа) на MathCAD?
2. Какие виды оператора определения (присваивания, назначения) вам известны?
3. Как связаны нули функции и экстремумы производной?
4. Какие пакеты профессиональных математических вычислений вам известны?
5. Опишите окно интерфейса MathCAD. Рабочий лист и регионы.
6. Как установить при выводе на экран количество знаков после запятой?
7. Как задается точность численных расчетов в MathCAD?
8. Что такое участок возрастания и убывания функции? Как эти участки связаны с поведением производной?

9. Опишите инструментальные панели и шаблоны и методы работы с ними.
10. Как задается диапазон и шаг изменения переменной в MathCAD?
11. Опишите, как задать функцию и вычислить значение этой функции в точке?
12. Что такое корень уравнения?
13. Опишите способ вычисления корней при помощи стандартной процедуры root(f(x), x), как можно задать точность вычислений?
14. Что такое нули функции и как их вычислять?
15. Как построить график функции одного параметра, задать диапазон изменения параметров, цвет и ширину линии? Как поставить точку на графике?
16. Что такое участок возрастания и убывания функции? Как они выглядят на графике, как их можно вычислить? Как эти участки связаны с поведением производной?
17. Как в MathCAD построить график прямой, параллельной оси Ox или оси Oy?
18. Как определить экстремумы функции по поведению производных?
19. Что такое экстремум функции? Как его находить?
20. Как установить при выводе на экран количество знаков после запятой?
21. Как задается точность численных расчетов в MathCAD?
22. Опишите, как задать функцию и вычислить значение этой функции в точке?
23. Что такое корень уравнения?
24. Что такое непрерывная функция? Дайте строгое определение через пределы.
25. Что такое нули функции и как их вычислять?
26. Как построить в одном окне графики нескольких функции одного параметра? Как задать для них разные параметры?
27. Как найти горизонтальные и наклонные асимптоты функции?
28. Как построить график функции одного параметра? Как поставить точку на графике?
29. Как построить график прямой, параллельной оси Ox или оси Oy?
30. Как может вести себя функция на бесконечности? Перечислить все случаи.
31. Что такое область определения функции? Что такое особые точки?
32. Описать поведение функции на границах области определения и на бесконечности.
33. Как установить, является ли функция чётной или нечётной. Дайте строгое определение.
34. Как определить, является ли функция периодической или нет. Определение.
35. Как найти точки перегиба и интервалы выпуклости-вогнутости.
36. Как найти вертикальные асимптоты?
37. Опишите понятие предела функции в точке? На бесконечности?

2. АЛГОРИТМИЧЕСКИЕ ВОЗМОЖНОСТИ MATHCAD

2.1. Панель программирования и шаблоны операторов

Система *MathCAD* обладает широкими возможностями для написания сложных алгоритмических конструкций. Алгоритмические команды (*операторы*) вызываются из окна *Programming* (*Программирование*) как на

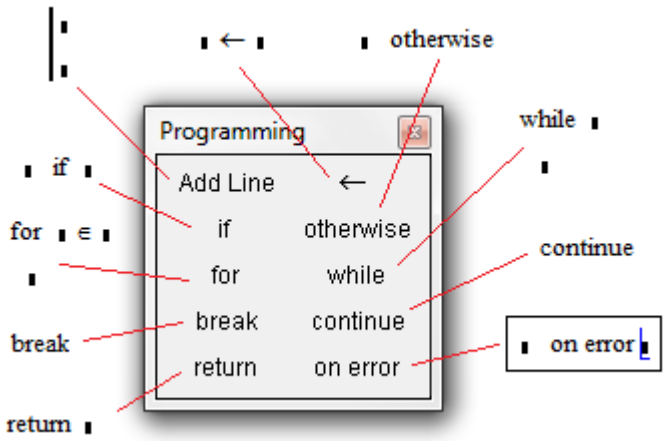


Рис. 14 – Окно вызова алгоритмических команд (операторов)

Операторный блок Add Line. Вертикальная линия (оператор **Add Line**) объединяет отдельные операторы в *операторный блок* с одним входом и одним выходом, который выполняется как единый оператор. Увеличить количество операторов внутри *операторного блока* можно многократным нажатием на кнопку «*Add Line*». Внутри операторного блока операторы выполняются последовательно сверху вниз.

Оператор локального присваивания. Кнопка «←» – это оператор присвоения значения локальной переменной. Область видимости такой переменной ограничена операторным блоком, задаваемым при помощи **Add Line**. Локальность переменной подразумевает ее невидимость вне операторного блока (программы), что не допускает путаницы переменных из разных программ.

Оператор выбора. Шаблон **if** (*если*) позволяет вводить в программу альтернативу с одной ветвью, иначе говоря, *оператор условного перехода*.

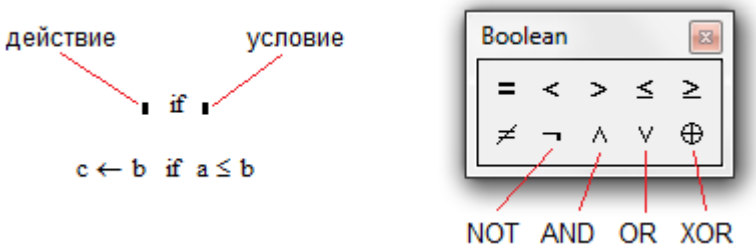


Рис. 15 – Структура оператора **if**. *Блок ввода логических команд*

Суть условного перехода в проверке *условия*, которое записывается в позиции справа от оператора **if.** Если *условие истинно*, то однократно выполняется *действие*, записанное в позиции слева от оператора **if,** после чего выполнение оператора заканчивается. Если *условие ложно*, то выполнение оператора **if** заканчивается немедленно, без выполнения *действия*.

Условие может состоять из довольно сложной ***логической конструкции***, построенной при помощи команд сравнения (=, ≠, <, ≤, >, ≥) и логических команд (NOT, OR, AND, XOR) – как на *Рис. 15*.

Возможности оператора **if** можно существенно расширить при помощи оператора **otherwise** (*иначе, в противном случае*). Он позволяет реализовать конструкции с двумя исходами (см. *Листинг 18, а*): в случае *истинности условия* выполняется одно *действие*, а в случае его *ложности* – другое (расположенное перед оператором **otherwise**).

Листинг 18. Оператор **if-otherwise** *и функция* **if( )**

а)
$$\text{maximum}(a,b) := \begin{cases} a & \text{if } a \geq b \\ b & \text{otherwise} \end{cases}$$

$\text{maximum}(3,2) = 3 \qquad \text{maximum}(3,4) = 4$

б)
$$\max \leftarrow \text{if}(A \geq B, A, B)$$

$$\text{if}(A \geq B, \max \leftarrow A, \max \leftarrow B)$$

Можно также воспользоваться вместо оператора **if** функцией **if**( ) – как на *Листинг 18, б.* У встроенной функции **if**( ) первый аргумент – *условие*, второй – *действие* при истинном условии, а третий аргумент – *действие* при невыполнении условия (otherwise).

Операторы цикла. В *MathCAD* имеется два оператора циклического повторения – оператор **while** и оператор **for**.

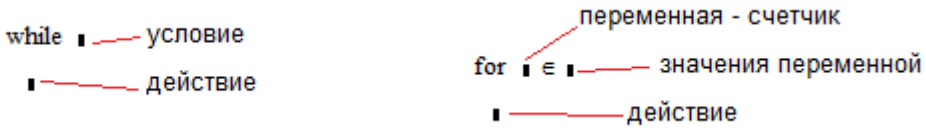


Рис. 16 – Структура операторов while *и* for

При выполнении оператора **while** (с предусловием) в первую очередь производится проверка *условия*, если оно *истинно*, то выполняется *действие*, после чего вновь проверяется *условие* и т.д. до тех пор, пока *условие* не станет *ложным*. Программист должен предусмотреть возможность выхода из цикла.

Оператор цикла **for** последовательно перебирает значения *переменной-счетчика* (слева от значка ∈) из *множества допустимых значений* (справа от символа ∈). Пока *переменная-счетчик* принадлежит *множеству допустимых значений*, *действие* повторяется, как только значение *переменной* выходит за границы *множества*, действие цикла заканчивается.

Программные блоки, описанные ниже (*Листинг 19*), вычисляют сумму *N* первых натуральных чисел. На примерах видно, что в операторе **for** изменение переменной-счетчика *i* происходит автоматически (неявно) в диапазоне 1..*N* , в то время как для оператора **while** изменение переменной *i* приходится явно прописывать.

Листинг 19. Операторы цикла **while** *и* **for**

```
SumA(N) := | S ← 0
           | i ← 1
           | while i ≤ N
           |     | S ← S + i
           |     | i ← i + 1
           | S
```

$\text{SumA}(3) = 6 \qquad \text{SumA}(4) = 10 \qquad \text{SumA}(5) = 15$

```
SumB(N) := | S ← 0
           | for i ∈ 1..N
           |   S ← S + i
           | S
```

$\text{SumB}(3) = 6 \qquad \text{SumB}(4) = 10 \qquad \text{SumB}(5) = 15$

На примерах ниже (*Листинг 20*) иллюстрируются некоторые свойства оператора **for**. Оператор **for** автоматически определяет, в какую сторону необходимо изменять (уменьшать или увеличивать) *переменную-счетчик.*

Оператор **for** может перебирать значения *переменной-счетчика* из *множества допустимых значений* довольно разнородной структуры – числовой, символьной, матричной.

Листинг 20. Некоторые свойства оператора for

```
SumB(N) := | S ← 0
           | for i ∈ 1..N
           |   S ← S + i
           | S

SumB(4) = 10

SumC(N) := | S ← 0
           | for i ∈ N..1
           |   S ← S + i
           | S

SumC(4) = 10

SumX := | S ← 0
        | for i ∈ 1,3,4,7,21,13,101
        |   S ← S + i
        | S

SumX = 150

V := | i ← 0
     | for VX ∈ [ 12.3   "abcd" ]
     |          [ 2+3·i    1   ]
     |   | V_i ← VX
     |   | i ← i + 1
     | V

V = [ 12.3   ]
    [ 2 + 3i ]
    [ "abcd" ]
    [ 1      ]
```

Операторы прерывания. В системе *MathCAD* операторы прерывания представлены тремя командами: **break, continue** и **return.**

Операторы **break** и **continue** используются для прерывания работы циклов **for** и **while**, а оператор **return** – для прерывания всего блока операторов.

Если при выполнении цикла встречается оператор **continue** (*перезапуск цикла*), то вычисления в теле цикла останавливаются и цикл *запускается снова*. Если в цикле встречается оператор **break** (*прерывание цикла*), то выполнение тела цикла прекращается и происходит *выход из цикла*.

В примере, приведенном ниже (*Листинг 21*), создается функция *PositivSubvector*, выбирающая положительные координаты вектора *x*. Оператор **continue** перезапускает цикл каждый раз, когда встречается отрицательный компонент вектора *x*. В этом же примере строится функция *FirstPositiv*, выбирающая первую положительную координату вектора *x*. Важно, что если такая координата найдена, то оператор **return** останавливает выполнение функции (дальнейший перебор элементов вектора *x* не производится).

Листинг 21. Операторы continue и return

```
x := [ 2  ]
     [ -1 ]
     [ 12 ]
     [ -8 ]
     [ 7  ]
     [ -0.8 ]

PositivSubvector(v) := | j ← -1
                       | for i ∈ 0..last(v)
                       |   | continue if v_i ≤ 0
                       |   | j ← j + 1
                       |   | w_j ← v_i
                       | w

PositivSubvector(x) = [ 2  ]
                      [ 12 ]
                      [ 7  ]

FirstPositiv(v) := for i ∈ 0..last(v)
                     return v_i if v_i > 0

FirstPositiv(x) = 2
```

Оператор **on error.** Этот оператор является обработчиком тех или иных ошибок, возникающих при вычислении. Он записывается так:

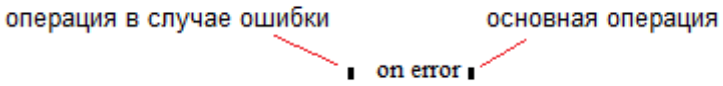


Рис. 17 – Структура оператора **on error**

Оператор **on error** выполняется следующим образом. Если при выполнении *основной операции* (расположенной справа от оператора **on error**) происходит *ошибка*, то вычисление этой операции прекращается и выполняется «*операция в случае ошибки*» (расположенная слева от *on error*):

Листинг 22. Обработчик ошибок **on error**

$$angl1(x,y) := \frac{x}{y} \qquad angl1(2,0) = \blacksquare$$

Divide by zero.

$$angl2(x,y) := 10^{10} \text{ on error } \frac{x}{y} \qquad angl2(2,0) = 1 \times 10^{10} \qquad angl2(4,2) = 2$$

2.2. Основные функции электроники и их представление в MathCAD

Рассмотрим функции, наиболее часто встречающиеся в электротехнике, электронике и энергетике, моделирование которых мы научимся производить в среде *MathCAD*.

Гармоническая функция

Гармонические функции (см. *Рис. 18*) наиболее часто встречаются в электротехнике и других областях естественных наук в связи с тем, что они предназначены для описания периодических вращающихся движений (процессов).

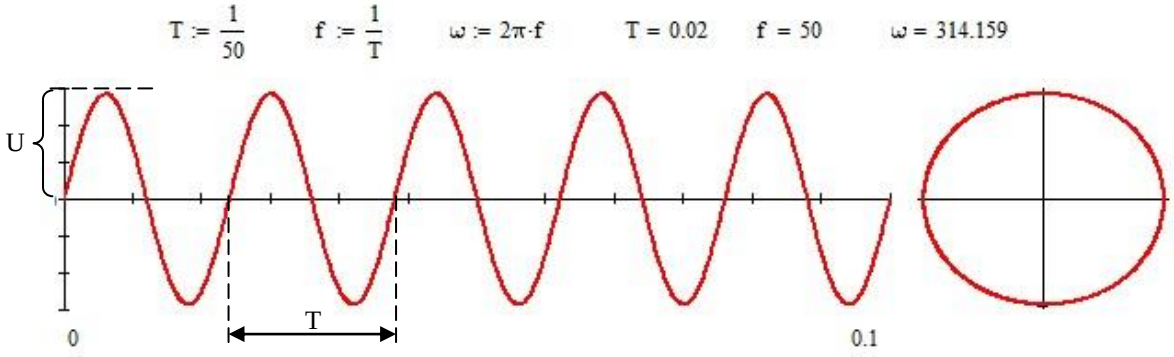


Рис. 18 – Основные параметры гармонической функции

Основными параметрами гармонических функций являются следующие:

1) *период колебаний* ***T*** – промежуток времени, за который система совершает одно полное колебание (то есть возвращается в то же состояние, в котором она находилась в первоначальный момент, выбранный произвольно). Измеряется в секундах [с].
2) обратное к нему понятие – *частота колебаний* ***f*** (*линейная частота*) определяется как число повторений события за единицу времени. Рассчитывается, как отношение количества возникновения событий к промежутку времени, за которое они совершены. Измеряется в герцах [Гц] = [$с^{-1}$].
3) *круговая частота* ***ω*** *(радиальная, циклическая, угловая частота)* равна модулю вектора угловой скорости (1), выражается в [*радианах в секунду*]. Круговая частота связана с линейной частотой *f* следующим образом: $\omega = 2\pi \cdot f$. При использовании в качестве единицы угловой частоты [*градусов в секунду*] связь с обычной частотой будет следующей: $\omega = 360° \cdot f$.

4) *амплитуда* **U** *(размах колебаний)* – это максимальная величина размаха (отклонения) периодического движения (1), см. *Рис. 18*. Для амплитуды единица измерения соответствует тому, какой циклический процесс анализируется: в электронике это может быть амплитуда тока [А], напряжения [В], мощности [Вт] и т.п.

$$U \cdot \sin(\overbrace{\omega \cdot t + \varphi}^{\text{полная фаза}}) \qquad (1)$$

амплитуда — круговая частота — начальная фаза

Гармоническое движение математически описывается, как правило, синусоидальной функцией, что и понятно: ведь при движении объекта (*рабочей точки, переменной состояния)* по окружности на оси *OY* вырисовывается *синусоидальная кривая*, а по оси *OX* – *косинусоидальная*. Если записать параметры гармонического сигнала как в формуле (1), то видно, что множитель перед sin( ) – это *амплитуда*, а множитель перед ***t*** (*время*) – это *круговая частота* (или *угловая скорость*).

К параметрам гармонического сигнала относятся так же:

5) *начальная фаза* φ – начальный угол (состояние) рабочей точки в момент начала движения по окружности (*Рис. 19*), или, говоря иначе, сдвиг графика sin( ) от начала координат по оси *OX*. Обычно выражается в [*радианах*] или [*градусах*], поскольку это по сути угол.

6) *полная фаза* $\omega \cdot t + \varphi$ – это текущее положение рабочей точки на окружности (или на синусоиде) в заданный момент времени t при заданной скорости вращения ω и заданной начальной фазе φ см. формулу (1).

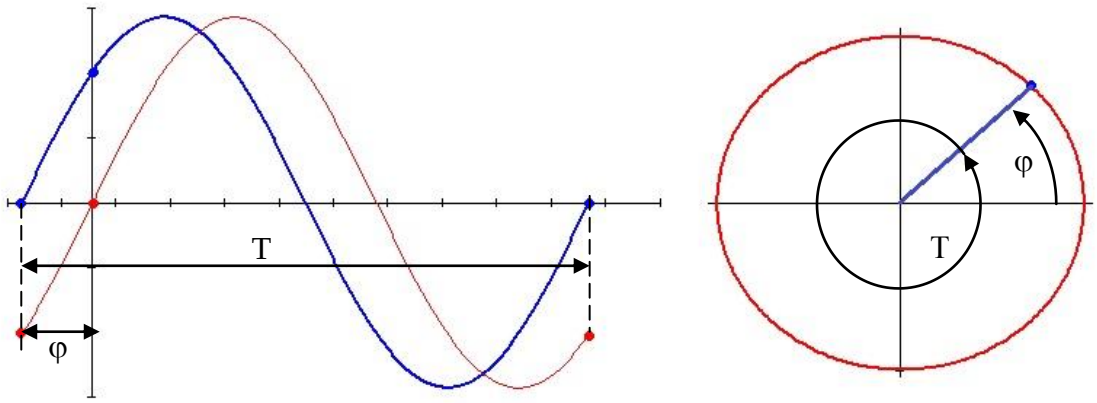


Рис. 19 – Основные параметры гармонической функции

Линейная функция

Линейная функция описывает *прямую линию*, основные её параметры – наклон k и смещение b, график прямой линии:

$$y = kx + b\,, \qquad (2)$$

где k – тангенс угла наклона, а b – смещение по вертикальной оси. Если $k = 0$, то прямая параллельна оси OX (нет наклона). Если $b = 0$, то прямая проходит через начало координат (нет смещения).

Линейно-нарастающая и линейно-спадающая функция

На основе графика прямой линии строится *линейно-нарастающая (ЛНФ)* и *линейно-спадающая (ЛСФ) функции*, они представляют собой периодически повторяющиеся отрезки прямой (см. *Рис. 20*).

По аналогии с гармонической функцией у ЛНФ также определяют *амплитуду, период, частоту* и *начальную фазу*. Действительно, если рассмотреть графики на

Рис. 20, то совершенно четко видно, что амплитуда сигнала составляет 12 единиц, период $\tau = 0.001$ [с], а следовательно, частота $f = 1000$ [Гц]. Поскольку график функции «выходит из нуля», то начальная фаза $\varphi = 0$ [рад].

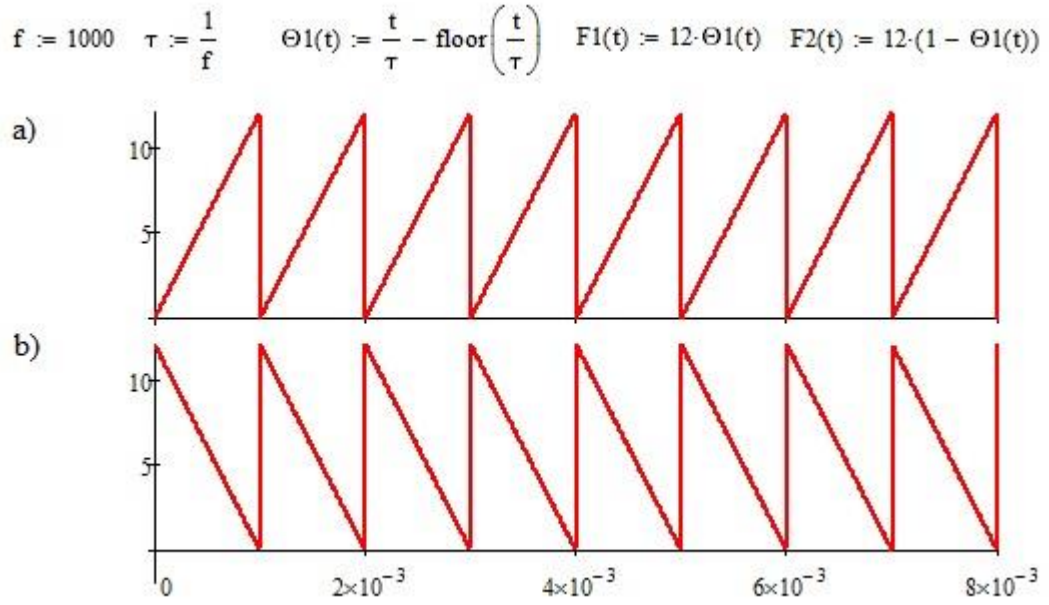


Рис. 20 – График (a) линейно-нарастающей и (b) линейно-спадающей периодических функций

Рассмотрим на примере процесс построения ЛНФ (и ЛСФ) в среде *MathCAD*. Сначала необходимо определиться с наклоном отрезка будущей функции, он, очевидно, зависит от частоты *f*. Зададимся частотой $f = 1000$ [Гц], следовательно период τ будет равен 0.001 [с]. Наклон *k* прямой линии (2) будет равен частоте, см *Рис. 21*.

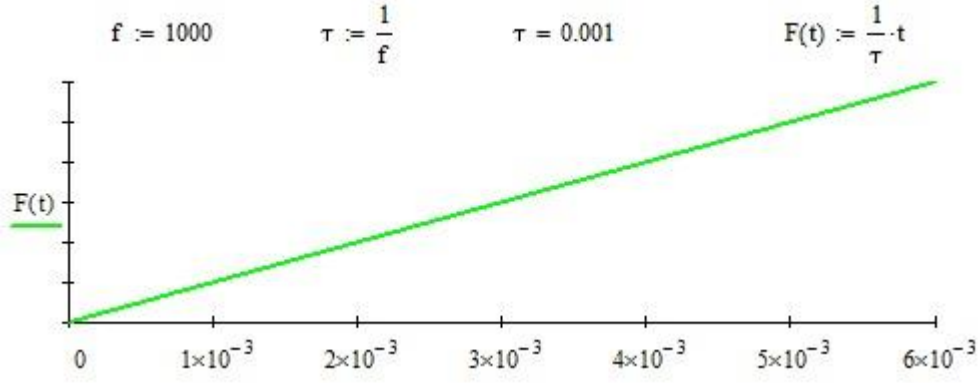


Рис. 21 – График линейной функции с наклоном 1/τ

Теперь воспользуемся встроенной функцией *MathCAD*, предназначенной для вычисления *целой части действительного числа* – **floor**( ). Применив функцию целой части к построенной нами линейной функции, получим ступенчатую функцию E1(t, τ), как на *Рис. 22*, которая дробит ось *OX* на интервалы длиной τ и ставит в соответствие любому моменту времени t из j-того интервала [(j-1)τ, jτ] его номер j.

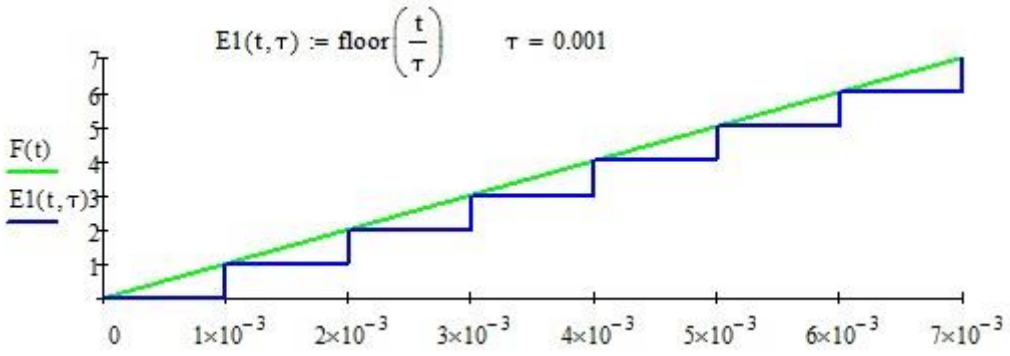


Рис. 22 – График ступенчатой функции с интервалами τ

Разность исходной линейной функции F(t) и ступенчатой функцией «*целая часть*» E1(*t*, *τ*) и есть искомая ЛНФ «дробная часть», традиционно обозначаемая Θ(*t*, *τ*). Линейно-спадающая функция строится как 1 - Θ(*t*, *τ*), см. *Рис. 20*.

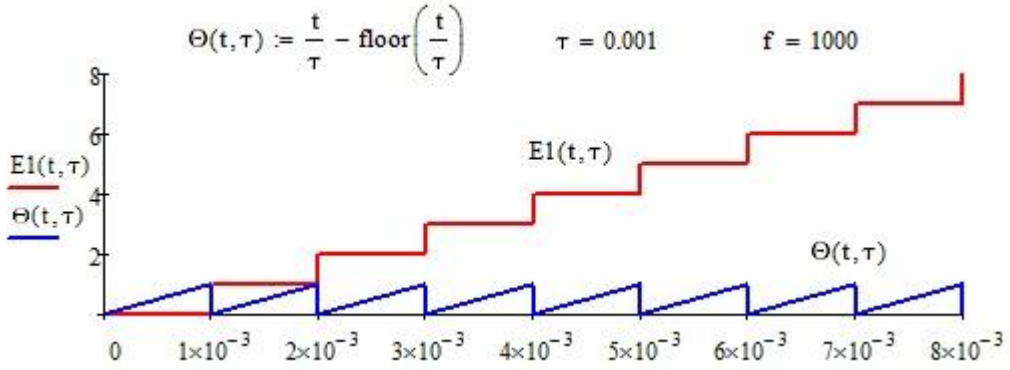


Рис. 23 – График линейно-нарастающей функции Θ(t, τ)

Таким образом, линейно-нарастающая и линейно-спадающая функции имеют вид (3) и (4) соответственно:

$$\frac{t}{\tau} - \text{floor}\left(\frac{t}{\tau}\right) \tag{3}$$

$$1 - \frac{t}{\tau} + \text{floor}\left(\frac{t}{\tau}\right) \tag{4}$$

Применение линейно-нарастающей Θ(*t*, *τ*) и линейно-спадающей функции 1 - Θ(*t*, *τ*) в моделировании систем управления электронных систем весьма велико, они применяются также при математическом описании различных процессов модуляции.

Сигнум функция

Довольно часто в математическом описании переключающихся сигналов используется *сигнум функция* **sign**( ) и её модификации, см. *Рис. 24*. Эта функция используется для определения *знака числа* – она равна 1 для любых положительных аргументов, -1 – для отрицательных и 0 – для нулевых.

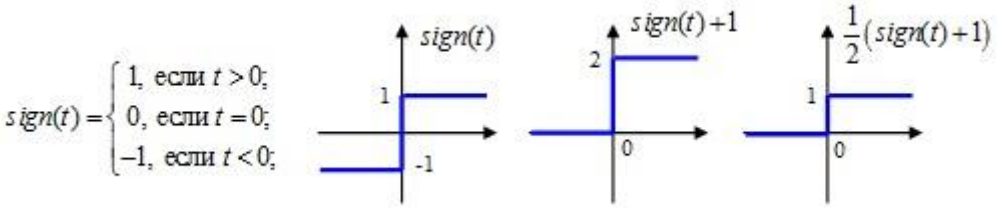


Рис. 24 – Сигнум функция и её модификации

Широтно-импульсная модуляция

Важным понятием в электротехнике, теории электропривода, теории автоматического управления и электронике является широтно-импульсная модуляция (ШИМ). Суть этого вида модуляции (кодирования) состоит в том, чтобы генерировать последовательность импульсов заданной амплитуды и частоты так, чтобы *ширина импульса* находилась в зависимости от уровня *задающего сигнала* (скважности) **γ**.

Как видно из *Рис. 25*, ширина импульса изменяется от 0 до τ. Когда скважность γ находится на середине амплитуды импульса (*Рис. 25,а*) ширина импульсов функции F(t, τ) равна 0.5. Если уровень γ повышается, то пропорционально растет и ширина импульса – на *Рис. 25,б* он равна 0.8, а если γ уменьшается, то импульсы сужаются, как на *Рис. 25,в*. , где ширина импульса равна 0.1.

При этом амплитуда, частота, период и фаза сигнала не меняются.

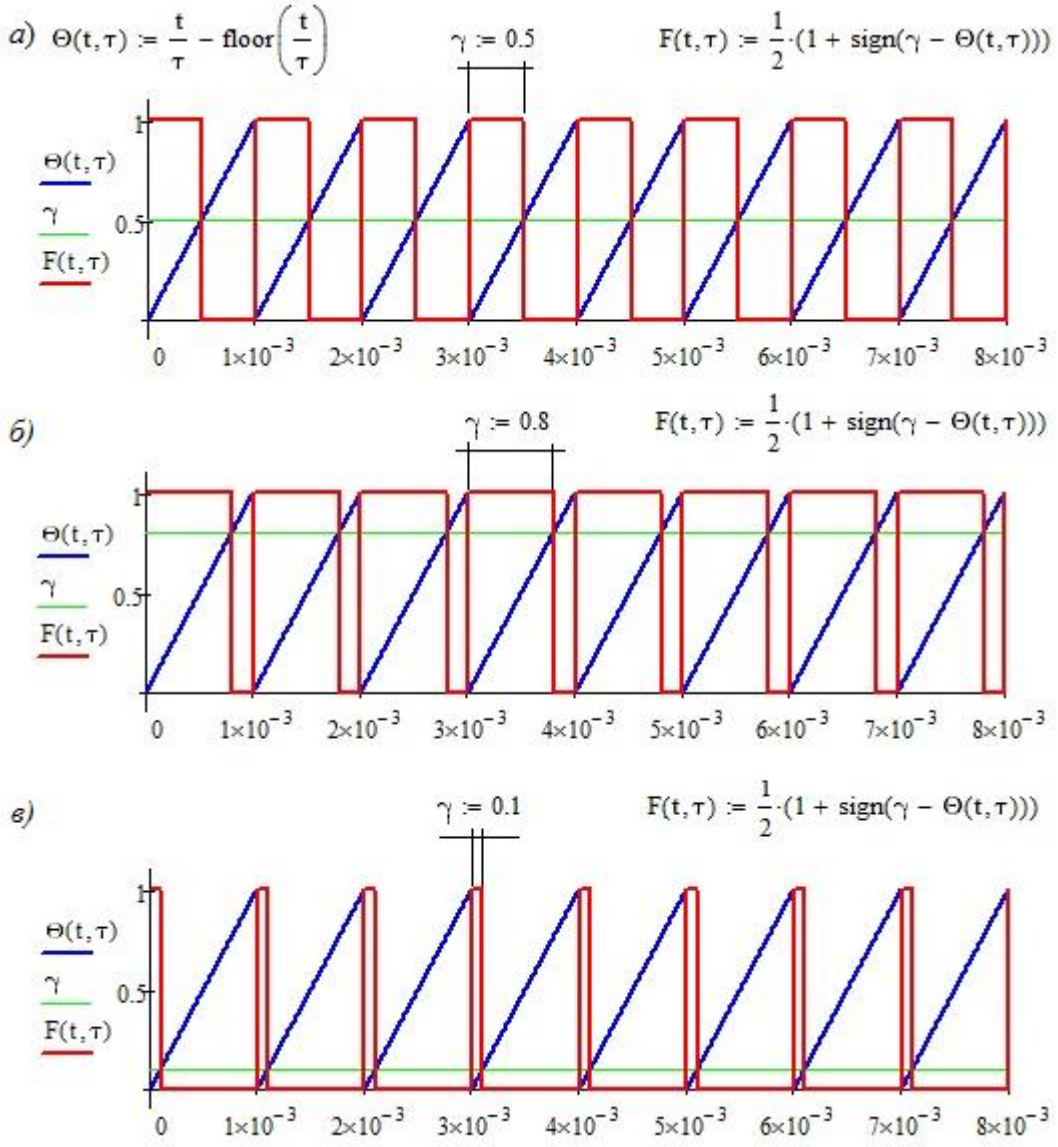


Рис. 25 – Широтно-импульсная модуляция

Как видно из *Рис. 25*, ширина импульса изменяется от **0** до $\boldsymbol{\tau}$. Когда скважность $\boldsymbol{\gamma}$ находится на середине амплитуды импульса (*Рис. 25,а*) ширина импульсов функции **F(t, τ)** равна **0.5**. Если уровень $\boldsymbol{\gamma}$ повышается, то пропорционально растет и ширина импульса – на *Рис. 25,б* он равна **0.8**, а если $\boldsymbol{\gamma}$ уменьшается, то импульсы сужаются, как на *Рис. 25,в*. , где ширина импульса $\boldsymbol{\gamma}$ равна **0.1**. При этом амплитуда, частота, период и фаза сигнала не меняются.

Создаваемая таким образом функция ***F(t,τ)*** называется *ШИМ-сигнал* и используется для организации автоматического управления каким либо переключаемым процессом (см. *Рис. 26*). Какой-либо *регулируемый параметр* системы (например, температура) измеряется датчиком-измерителем и, в качестве электрического сигнала, *скважность* ***γ*** передается по *обратной связи* в *систему автоматического управления* (САУ). Там по принципам, приведенным на *Рис. 25*, строится *ШИМ-сигнал*, подаваемый на переключающее устройство системы, регулируя таким образом работу нагревателя или охладителя. Поскольку ширина импульсов функции ***F(t,τ)*** зависит от скважности ***γ***, то в зависимости от её уровня производится дозирование энергии, подаваемой на регулируемый параметр.

Рис. 26 – Система автоматического управления с ШИМ

Импульсная последовательность

При математическом описании двоичной последовательности символов в виде «потенциального кода» (нуль задается высоким потенциалом (напряжением) а единица – низким), используются *импульсные функции* различного вида.

$$S := (1\ 0\ 1\ 1\ 0\ 0\ 0\ 1) \qquad D(t,\tau) := \left| n \leftarrow \mathrm{floor}\left(\frac{t}{\tau}\right) \right. \\ \left. S_{0,n} \right.$$

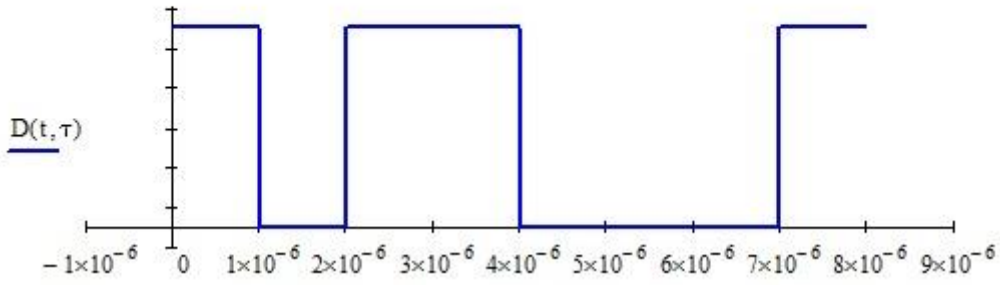


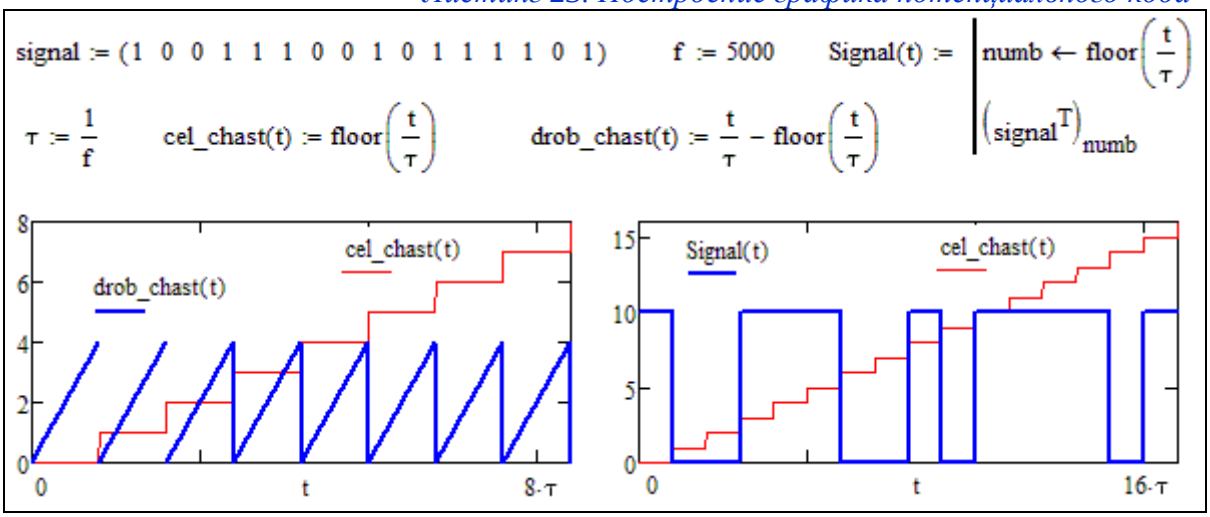
Рис. 27 – График двоичной последовательности S с тактовым периодом τ = 10^{-6}[c]

Например, на *Рис. 27* изображен график двоичной последовательности S с тактовым периодом τ = 10^{-6}[c]. Здесь, как и в примере на *Рис. 22*, функция **floor**(*t*/τ) определяет номер такового интервала, в котором находится текущий момент времени t, и в соответствии с этим номером выбирается компонент $S_{0,n}$ последовательности S.

Листинг 23 содержит другой пример построения на графике функции потенциального кода – последовательности битов исходного сигнала *S*(*t*), формируемого с тактовой частотой (частотой квантования) f_d, необходимо определять номер тактового интервала. Воспользуемся встроенной функцией **floor**( ), вычисляющей целую часть числа. Рассмотрите, как построены функции целой части *cel_chast*(*t*/τ) и дробной части *drob_chast*(*t*/τ) числа *t*/τ. При формировании функции

Signal(t) в локальную переменную *numb* заносится номер тактового периода, строящийся аналогично функции *cel_chast(t/τ)*.

Листинг 23. Построение графика потенциального кода



Аналоговая модуляция

Аналоговая модуляция является таким способом преобразования цифрового сигнала в аналоговый, при котором двоичная информация кодируется изменением амплитуды (U), частоты (ω) или фазы (φ) гармонического сигнала $F(t) = U \cdot \sin(\omega t + \varphi)$.

а)

"0" → +5 В

"1" → 0 В

б)

$F_1(t) = U_1 \cdot \sin(\omega t + \varphi)$

$F_2(t) = U_2 \cdot \sin(\omega t + \varphi)$

в)

$F_1(t) = U \cdot \sin(\omega_1 t + \varphi)$

$F_2(t) = U \cdot \sin(\omega_2 t + \varphi)$

г)

$F_1(t) = U \cdot \sin(\omega t + \varphi_1)$

$F_2(t) = U \cdot \sin(\omega t + \varphi_2)$

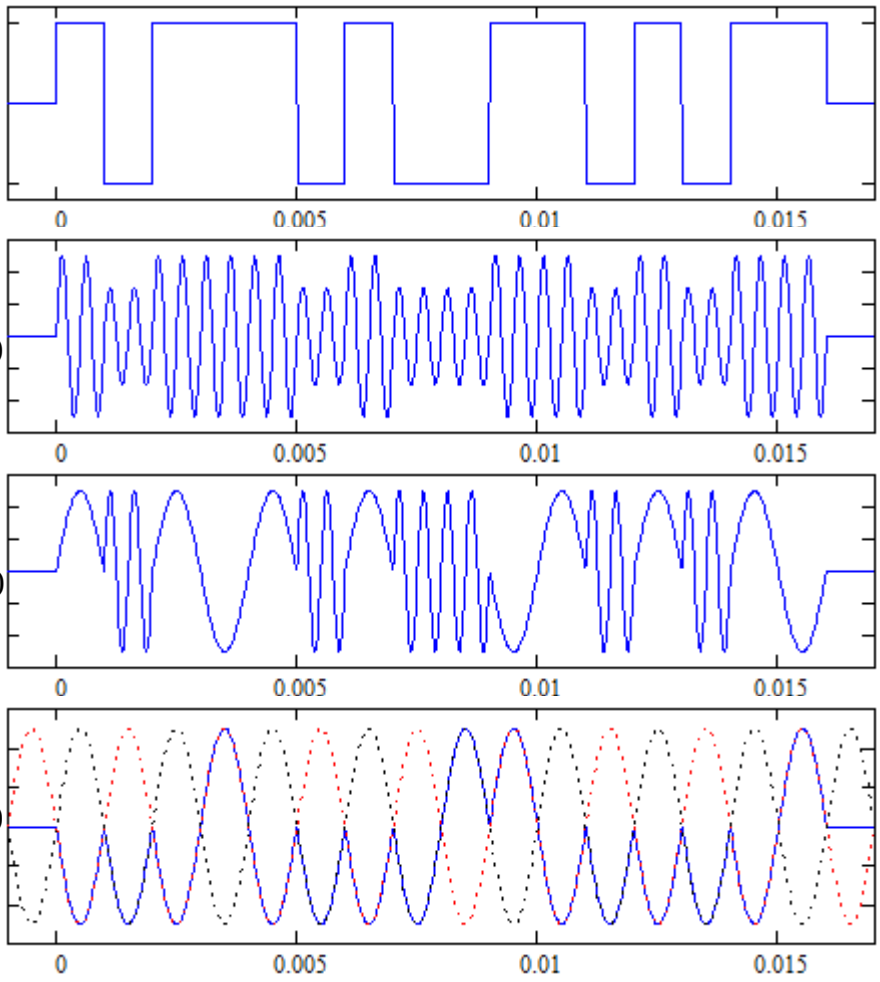


Рис. 28 – Виды аналоговой модуляции

Основные способы аналоговой модуляции – *амплитудная* (*б*), *частотная* (*в*) и *фазовая* (*г*) показаны на *Рис. 28*. На графике (*Рис. 28, а*) приведена последовательность битов исходного сигнала **S(t)**, представленная *потенциальным кодом* – логическому нулю соответствует +5 В, логической единице – 0 В. Программа построения графика этой последовательности приведена ниже на *Рис. 29*.

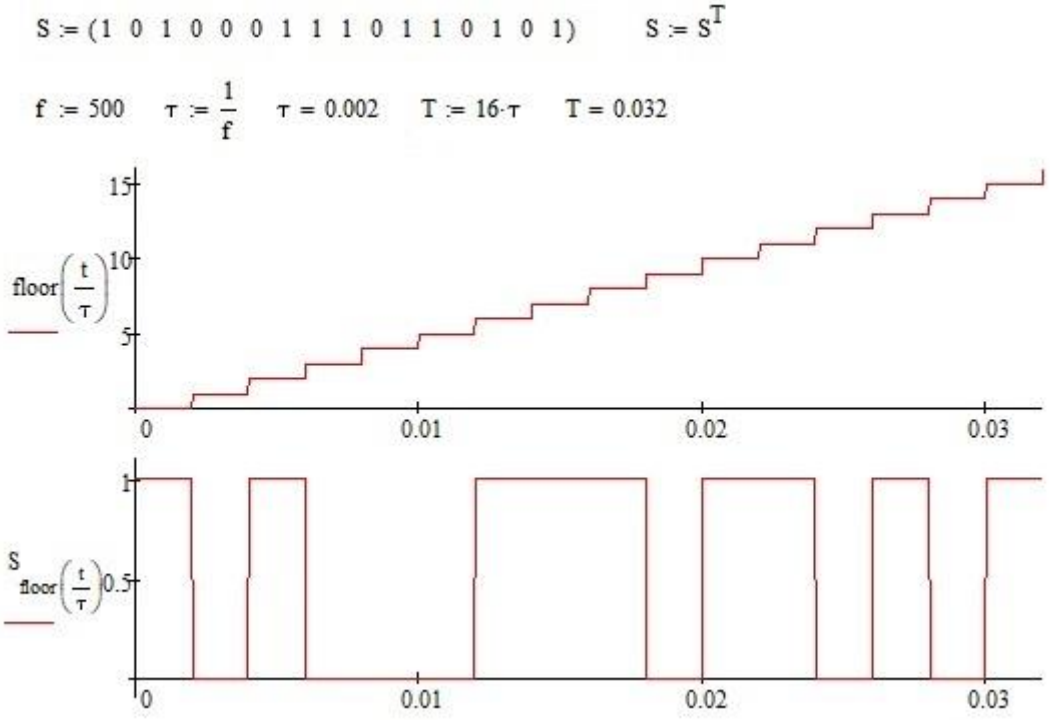


Рис. 29 – Импульсная последовательность

Амплитудная модуляция. Как следует из названия, изменяемым параметром гармонического сигнала является *амплитуда*. В примере, рассмотренном ниже (*Рис. 30*) задается два гармонических сигнала с одинаковыми частотой и фазой колебаний, но с различными амплитудами - 1 [В] и 3 [В] соответственно. Функция **M(*t*)** строится следующим образом: для заданного момента времени ***t*** переменной ***q*** присваивается номер тактового интервала, в который попадает момент ***t***. В зависимости от того, равняется **1** или **0** значение элемента S_q (элемент последовательности ***S*** с номером ***q***), вычисляется гармоническая функция с соответствующей амплитудой.

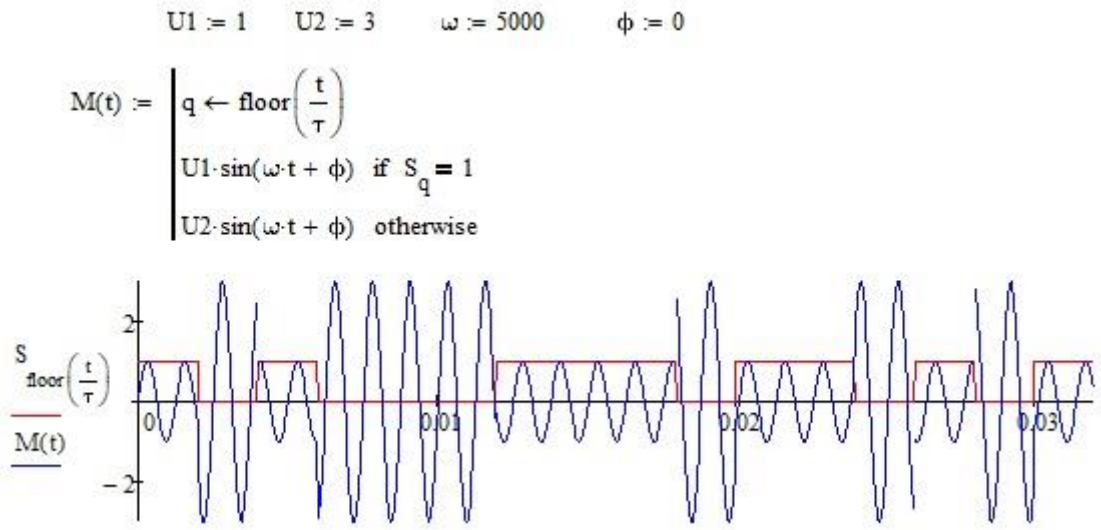


Рис. 30 – Амплитудная модуляция

Частотная модуляция. Здесь изменяемым параметром гармонического сигнала является *частота*. В примере, рассмотренном ниже (*Рис. 31*) задается два гармонических сигнала с одинаковыми амплитудой и фазой колебаний, но с различными частотами - 1000 [рад/с] и 5000 [рад/с] соответственно. Функция **M(*t*)** строится аналогично: для заданного момента времени ***t*** переменной ***q*** присваивается номер тактового интервала, в который попадает момент ***t***. В зависимости от того, равняется **1** или **0** значение элемента S_q (элемент последовательности ***S*** с номером ***q***), вычисляется гармоническая функция с соответствующей частотой.

$$U := 1 \qquad \omega 2 := 1000 \qquad \omega 1 := 5000 \qquad \phi := 0$$

$$M(t) := \left| \begin{array}{l} q \leftarrow \text{floor}\left(\frac{t}{\tau}\right) \\ U \cdot \sin(\omega 1 \cdot t + \phi) \quad \text{if } S_q = 1 \\ U \cdot \sin(\omega 2 \cdot t + \phi) \quad \text{otherwise} \end{array} \right.$$

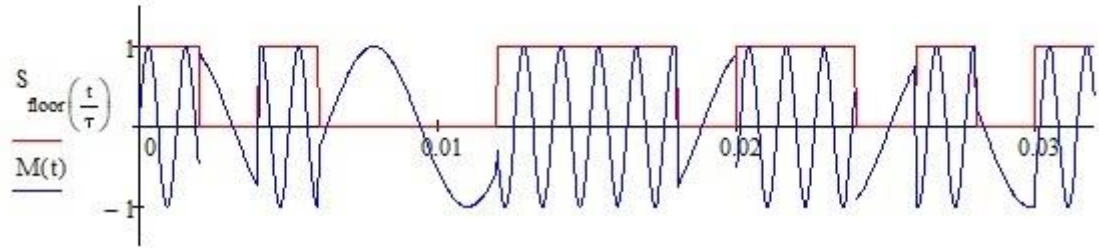


Рис. 31 – Частотная модуляция

Фазовая модуляция. Здесь изменяемым параметром гармонического сигнала является *фаза*. В примере, рассмотренном ниже (*Рис. 32*) задается два гармонических сигнала с одинаковыми амплитудой и частотой колебаний, но с различными фазами - 0 [рад] и π/3 [рад] соответственно. Функция **M(*t*)** строится аналогично предыдущим, с то лишь разницей, что для заданного момента времени ***t***, попадающего в определенный тактовый интервал, вычисляется значение гармонической функции с соответствующей фазой.

$$U := 1 \qquad \omega := 1000 \qquad \phi 1 := 0 \qquad \phi 2 := \frac{\pi}{3}$$

$$M(t) := \left| \begin{array}{l} q \leftarrow \text{floor}\left(\frac{t}{\tau}\right) \\ U \cdot \sin(\omega \cdot t + \phi 1) \quad \text{if } S_q = 1 \\ U \cdot \sin(\omega \cdot t + \phi 2) \quad \text{otherwise} \end{array} \right.$$

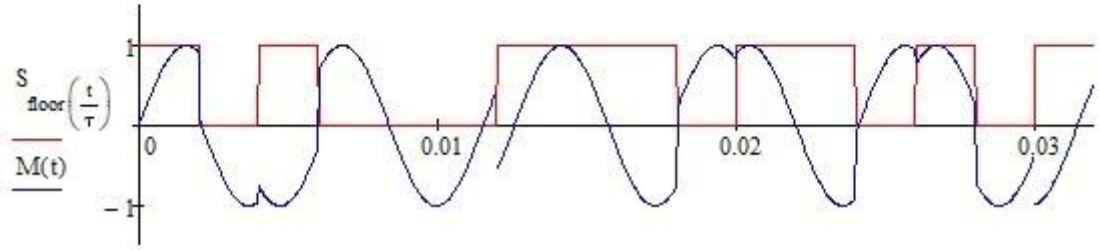


Рис. 32 – Фазовая модуляция

Таким образом единицы и нули импульсной последовательности заменяются фрагментами двух различных гармонических функций. Или иначе говоря, импульсная последовательность "склеивается" из фрагментов соответствующих синусоид, как показано на *Рис. 33*.

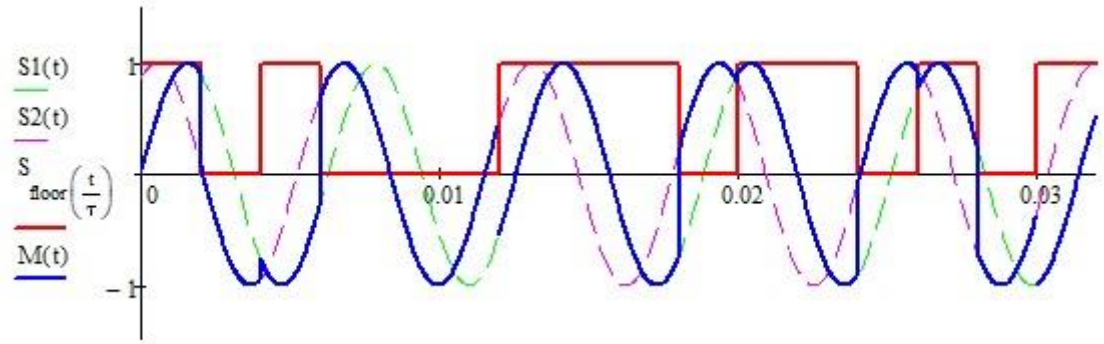


Рис. 33 – Фазовая модуляция

Контрольные вопросы

1. Какие виды аналоговой модуляции вы знаете?
2. Записать формулу гармонической функции, перечислить ее параметры.
3. Как построить цифровой сигнал на графике?
4. Существуют ли в природе чисто цифровые сигналы?
5. Для гармонической функции описать амплитуду, частоту и период, круговую частоту, начальную фазу (в радианах и в градусах).
6. Для линейно-нарастающей и линейно-спадающей функции дать определение понятиям тактовый период и тактовая частота, амплитуда, начальный сдвиг (фаза).
7. Для широтно-импульсной модуляции определить амплитуду, тактовую частоту и тактовый период, скважность.
8. Для импульсной последовательности описать тактовую частоту и тактовый период, импульсную последовательность в двоичном формате.
9. Для аналоговой модуляции дать определение понятиям: вид модуляции, амплитуда, частота и фаза модулирующих функций.
10. Что необходимо учитывать, чтобы по графику импульсной последовательности восстановить импульсную последовательность в двоичном формате.

3. ОПЕРАЦИИ МАТРИЧНОЙ АЛГЕБРЫ И ИХ РЕАЛИЗАЦИЯ В СРЕДЕ MATHCAD

В курсе линейной алгебры студенты знакомятся с такими понятиями как *матрица* и *вектор*, рассмотрим, какие инструменты для работы с матричными объектами предоставляет *MathCAD*. Под матрицей понимается прямоугольная таблица чисел размерности $n \times m$.

Вектор – это такая матрица у которой $n=1$ или $m=1$. В средней школе давалось определение вектора как направленного отрезка прямой, важно понимать, что эти определения эквивалентны.

Над векторами и матрицами определены операции *сложения и умножения на число* – эти операции выполняются *поэлементно* и обладают свойствами *линейности*. Дополнительно определены операции произведения матриц и умножения матрицы на вектор. Свойства этих операций рассмотрены ниже.

Кроме того, для матриц введено понятие *определителя* – числа, некоторым образом характеризующего матрицу и понятие *ранга* – размерности линейно-независимой системы векторов, составляющих матрицу. Строгое определение этих понятий дается в курсе высшей математики, здесь мы рассмотрим, как вычислять эти показатели и как ими пользоваться.

3.1. Матричные операции MathCAD

В среде *MathCAD* операции над матрицами производятся в соответствии с правилами математики. Комплекс *MathCAD* содержит достаточно представительную библиотеку встроенных функций, в том числе и функций, работающих с матрицами. Ниже рассмотрены некоторые функции, осуществляющие матричные операции, используемые при выполнении лабораторных и практических работ.

Ввод и редактирование формата матриц с помощью шаблона

Интерфейс процессора *MathCAD* позволяет вводить матрицы с помощью шаблона, задаваемого в диалоговом окне (*Рис. 34,б*). Диалоговое окно вызывается командой главного меню *Insert\Matrix*, либо комбинацией клавиш-акселераторов *Ctrl+M*, либо нажатием на пиктограмму палитры матричных операций (*Рис. 34,а*).

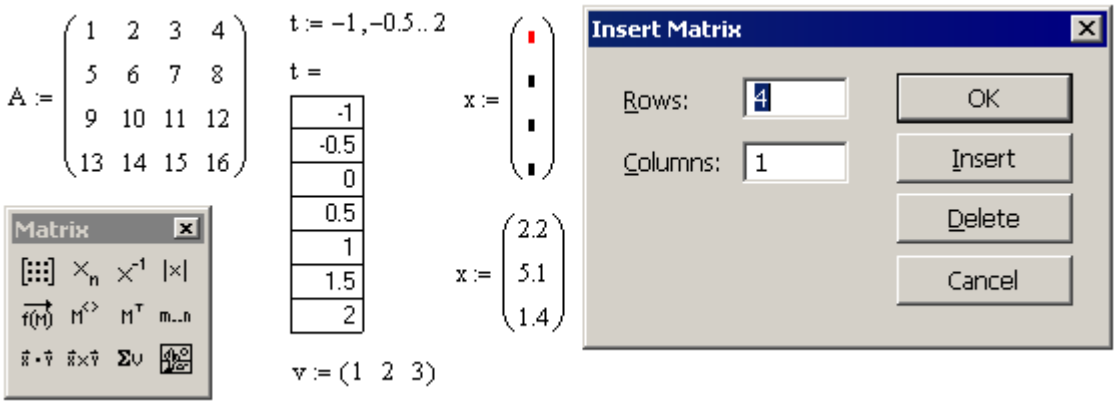


Рис. 34 – Матрицы и вектора в MathCAD. Диалоговое окно ввода матрицы

С помощью этого же диалога можно редактировать формат уже созданных матриц, добавляя/удаляя строки и столбцы. Эти действия вызываются кнопками *Insert* и *Delete*.

Нумерация строк и столбцов в *MathCAD* по умолчанию начинается ***с нуля*** (*Рис. 35*). Принцип нумерации элементов матриц можно переопределить с помощью команды **ORIGIN:=1** (после этого матрицы и вектора нумеруются не с 0, а с 1).

Интерпретирующая система блокирует некорректную матричную операцию. При этом некорректный, нереализуемый оператор выделяется измененным цветом шрифта и комментарием в виде всплывающей подсказки. Таким образом, программист может контролировать принципиальную осуществимость матричных операций, связанную, в частности, и с размерностью операндов.

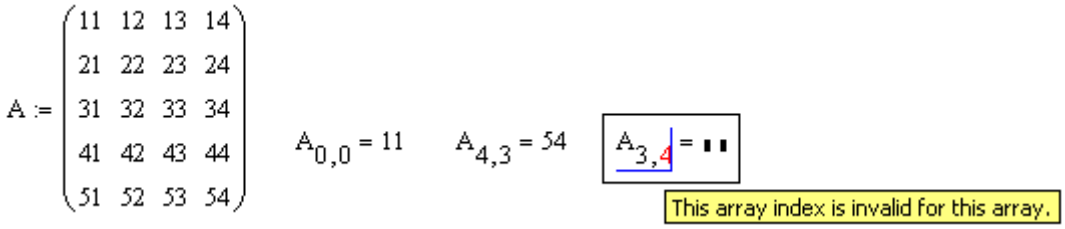
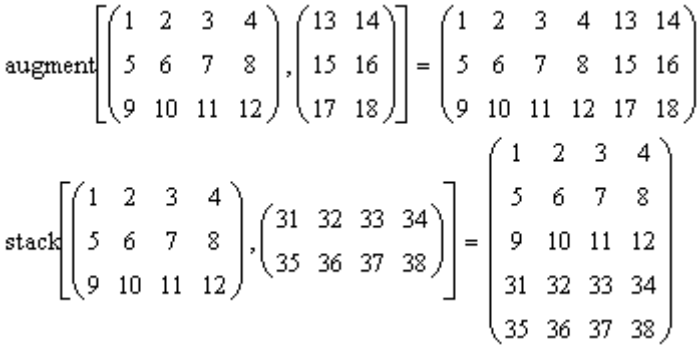


Рис. 35 – Некорректное обращение к элементу матрицы блокируется

Функции компоновки и декомпозиции матриц и векторов

В составе библиотеки встроенных функций *MathCAD* имеются функции, позволяющие производить, формировать матрицу из блоков-подматриц и выделять из заданной матрицы подматрицы нужного размера. Компоновку матрицы из блоков осуществляют функции ***augment, stack***:

Листинг 24. Функции объединения матриц по строкам и столбцам

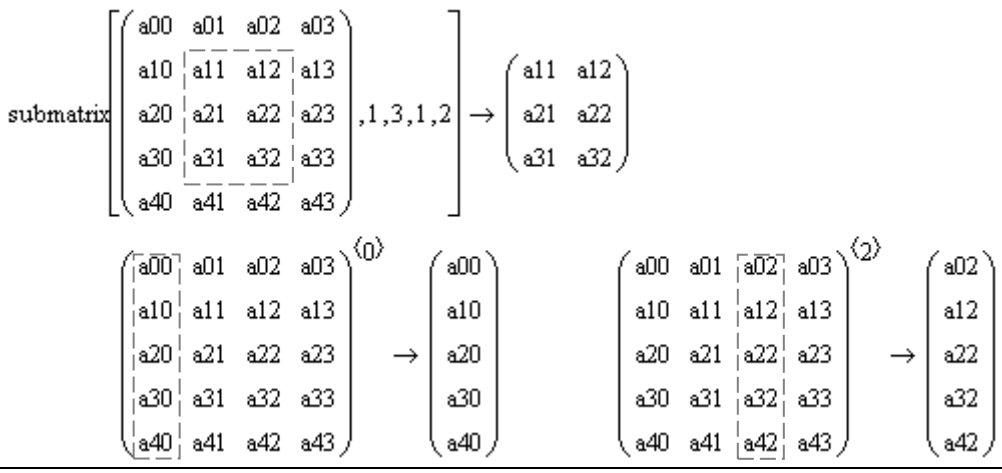


Выделение подматрицы выполняется функцией *submatrix* и оператором выделения столбца матрицы $\mathbf{M}^{\langle\rangle}$. Функция выделения подматрицы *submatrix* имеет следующий синтаксис вызова:

$$\text{submatrix}(\mathbf{A}, row_1, row_2, col_1, col_2),$$

где **A** – исходная матрица, из которой извлекается фрагмент; row_1 и row_2– номера начальной и конечной выбираемых строк в матрице; col_1 и col_2– номера начального и конечного столбца в матрице A, включаемого в выделяемую подматрицу.

Листинг 25. Функции извлечения подматрицы



Для частного случая выделения подматрицы – выделения столбца с заданным номером – в *MathCAD* имеется специальный оператор $\mathbf{M}^{\langle\,\rangle}$. Строку матрицы **A** можно выделить как при помощи функции submatrix, так и при помощи оператора $\mathbf{M}^{\langle\,\rangle}$ (*Листинг 26*).

Приведенный ниже пример (*Листинг 26*) иллюстрирует применение оператора $\mathbf{M}^{\mathrm{T}}$ – транспонирования матрицы или вектора. *Оператор транспонирования* делает строки матрицы столбцами, а столбцы – строками, визуально эта операция соответствует зеркальному отражению относительно диагонали.

Листинг 26. Выделение столбца и строки матрицы

$$v1 := \begin{pmatrix} 11 & 12 & 13 & 14 \\ 21 & 22 & 23 & 24 \\ 31 & 32 & 33 & 34 \\ 41 & 42 & 43 & 44 \\ 51 & 52 & 53 & 54 \end{pmatrix}^{\langle 1 \rangle} \qquad v1 = \begin{pmatrix} 12 \\ 22 \\ 32 \\ 42 \\ 52 \end{pmatrix} \qquad v2 := \begin{pmatrix} 11 & 12 & 13 & 14 \\ 21 & 22 & 23 & 24 \\ 31 & 32 & 33 & 34 \\ 41 & 42 & 43 & 44 \\ 51 & 52 & 53 & 54 \end{pmatrix}^{\mathrm{T}\,\langle 2 \rangle} \qquad v2 = \begin{pmatrix} 31 \\ 32 \\ 33 \\ 34 \end{pmatrix}$$

$$v2 := v2^{\mathrm{T}} \qquad v2 = (31 \;\; 32 \;\; 33 \;\; 34)$$

Матричные и векторные арифметические операции

В среде *MathCAD* операции над матрицами производятся в соответствии с правилами математики.

При умножении матрицы (или вектора) на константу *K*, все элементы матрицы (вектора) умножаются на *K*.

При сложении (вычитании) матриц (или векторов) суммирование производится покоординатно. Складывать и вычитать можно только матрицы (вектора) совпадающей размерности.

Произведение матриц производится по правилу ***«строка на столбец»*** – каждый элемент матрицы-произведения есть скалярное произведение вектора-строки первой матрицы на вектор-столбец второй:

$$\mathbf{C} = \mathbf{A} \times \mathbf{B}: \quad \mathbf{c}_{i,j} = \sum_{k=0}^{n-1} \mathbf{a}_{i,k} \cdot \mathbf{b}_{k,j}\,, \tag{5}$$

где $\mathbf{A}[m \times n]$, $\mathbf{B}[n \times p]$, $\mathbf{C}[m \times p]$. Напомним, что элементы матриц в *MathCAD* нумеруются с нуля, поэтому индексы изменяются так: $i = 0...m-1$, $j = 0...p-1$.

Естественно в формуле (5) требовать, чтобы количество столбцов матрицы **A** совпадало с количеством строк матрицы **B** – такие матрицы называются *согласованными* или *сцепленными*.

Листинг 27. Арифметические операции с матрицами

$$A := \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \qquad B := \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \end{pmatrix} \qquad \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \end{pmatrix} = \begin{pmatrix} 53 & 56 & 59 \\ 117 & 124 & 131 \\ 181 & 192 & 203 \end{pmatrix}$$

$$A \cdot B = \begin{pmatrix} 53 & 56 & 59 \\ 117 & 124 & 131 \\ 181 & 192 & 203 \end{pmatrix} \qquad |A \cdot B| = 0 \qquad \begin{pmatrix} 11 & 21 \\ 31 & 41 \\ 51 & 61 \end{pmatrix} - 2\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} = \begin{pmatrix} 9 & 17 \\ 25 & 33 \\ 41 & 49 \end{pmatrix} \qquad 2A = \begin{pmatrix} 2 & 4 \\ 6 & 8 \\ 10 & 12 \end{pmatrix}$$

Нельзя забывать, что в отличие от произведения чисел, матричное произведение некоммутативно, т.е., в общем случае $\mathbf{A} \times \mathbf{B} \neq \mathbf{B} \times \mathbf{A}$, подтверждение этому легко построить (см., например *Листинг 28*):

Листинг 28. Матричное произведение некоммутативно

$$\begin{pmatrix}1&2\\3&1\end{pmatrix}\cdot\begin{pmatrix}3&2\\2&3\end{pmatrix}=\begin{pmatrix}7&8\\11&9\end{pmatrix}\qquad\begin{pmatrix}3&2\\2&3\end{pmatrix}\cdot\begin{pmatrix}1&2\\3&1\end{pmatrix}=\begin{pmatrix}9&8\\11&7\end{pmatrix}$$

$$\begin{pmatrix}1&2\\3&1\end{pmatrix}\cdot\begin{pmatrix}3&2\\6&4\end{pmatrix}=\begin{pmatrix}15&10\\15&10\end{pmatrix}\qquad\begin{pmatrix}3&2\\6&4\end{pmatrix}\cdot\begin{pmatrix}1&2\\3&1\end{pmatrix}=\begin{pmatrix}9&8\\18&16\end{pmatrix}$$

Для создания единичной и диагональной матриц заданного размера $n\times n$ в *MathCAD* можно использовать функции ***identity***(*n*). и ***diag***(***x***) соответственно (*Листинг 29*), здесь вектор ***x*** – вектор диагональных элементов создаваемой матрицы.

Единичная матрица **E** выполняет роль единицы при матричном умножении, а нулевая матрица **0** – роль нуля. Обучаемому предлагается проверить свойства этих матриц: $\mathbf{A}\times\mathbf{E}=\mathbf{A}$, $\mathbf{E}\times\mathbf{A}=\mathbf{A}$ и $\mathbf{A}\times\mathbf{0}=\mathbf{0}$, $\mathbf{0}\times\mathbf{A}=\mathbf{0}$.

Листинг 29. Создание единичной и диагональной матриц

$$\text{identity}(3)=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}\quad\text{identity}(2)=\begin{pmatrix}1&0\\0&1\end{pmatrix}\quad D:=\text{diag}\left(\begin{pmatrix}1\\2\\3\end{pmatrix}\right)\quad D=\begin{pmatrix}1&0&0\\0&2&0\\0&0&3\end{pmatrix}\quad |D|=6$$

Функции определения размеров матриц

Возможны ситуации, когда для вычислительного процесса необходимы сведения о размерах матриц, которые становятся известны только после их формирования в ходе вычислений. Для получения такой информации используются функции определения количества строк и столбцов в матрицах ***rows*** и ***cols.***

Листинг 30. Использование функций rows и cols

$$A:=\begin{pmatrix}1&2\\3&4\\5&6\end{pmatrix}\qquad B:=\begin{pmatrix}11&12&13\\21&22&23\end{pmatrix}$$

$$C:=\text{augment}(A\cdot B,\text{identity}(\text{rows}(A\cdot B)))\qquad \text{rows}(C)=3\quad \text{cols}(C)=6$$

$$C=\begin{pmatrix}53&56&59&1&0&0\\117&124&131&0&1&0\\181&192&203&0&0&1\end{pmatrix}$$

$$C:=\text{stack}\left(C,\text{identity}(\text{cols}(C))^{\langle 0\rangle^T}\right)\qquad \text{rows}(C)=4\quad \text{cols}(C)=6$$

$$C=\begin{pmatrix}53&56&59&1&0&0\\117&124&131&0&1&0\\181&192&203&0&0&1\\1&0&0&0&0&0\end{pmatrix}$$

На *Листинг 30* приведен пример довольно сложного формирования матрицы **C** из матриц **A** и **B** с использованием функций *rows* и *cols* для определения размеров промежуточных матриц.

Применение функций *rows* и *cols* повышает универсальность выражений, использующих матричные операции, они становятся менее зависимыми от конкретных размеров исходных матриц.

При программировании нестандартных пользовательских операций с элементами матриц особенно важно знать их размерность для организации циклов.

Оператор | x |

Листинг 31. Вычисление определителя квадратной матрицы

$$D:=\text{diag}\left(\begin{pmatrix}1\\2\\3\end{pmatrix}\right)\quad D=\begin{pmatrix}1&0&0\\0&2&0\\0&0&3\end{pmatrix}\quad |D|=6\qquad \left|\begin{pmatrix}3&2\\6&4\end{pmatrix}\right|=0$$

Расчет определителя квадратной матрицы в *MathCAD* производится оператором **|A|.** Попытка применить этот оператор к неквадратной матрице с количеством столбцов более одного вызывает ошибку.

Формирование матрицы или вектора на основе заданной функции

В ряде случаев значения элемента матрицы $\mathbf{a}_{i,j}$ можно связать со значениями его индексов некоторой функцией от двух целочисленных аргументов: $\mathbf{a}_{i,j} = F(i, j)$. В этом случае матрицу произвольного размера можно сформировать при помощи встроенной функции ***matrix***(*m*, *n*, *F*), где *m* и *n* – число строк и столбцов в создаваемой матрице, *F* – имя функции.

Листинг 32. Формирование матриц функцией matrix

$$F(x,y) := 2^{x+y} \qquad F(3,4) = 128$$

$$\text{matrix}(4,5,F) = \begin{pmatrix} 1 & 2 & 4 & 8 & 16 \\ 2 & 4 & 8 & 16 & 32 \\ 4 & 8 & 16 & 32 & 64 \\ 8 & 16 & 32 & 64 & 128 \end{pmatrix} \qquad A := \text{matrix}(3,3,F) \quad A = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 4 & 8 \\ 4 & 8 & 16 \end{pmatrix}$$

Матрицу произвольного размера можно сформировать, задавая значение каждому элементу с помощью оператора назначения (*Листинг 33*). Перебор множества индексов может быть произведен с помощью *ранжированных переменных* (создание матрицы **A**), либо с использованием *операторов цикла в программном блоке* (создание матрицы **B**). Как видно из приведенного примера, результат одинаков.

Листинг 33. Поэлементное формирование матриц

$$i := 0..2 \qquad j := 0..5 \qquad A_{i,j} := i + j \qquad A = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$$

$$B := \left| \begin{array}{l} \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..5 \\ \quad\quad d_{i,j} \leftarrow i + j \\ d \end{array} \right| \qquad B = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$$

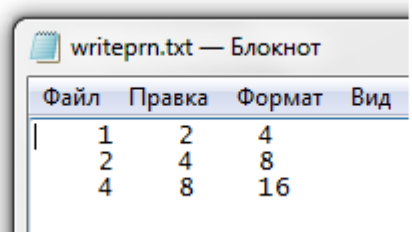
Сохранение матриц и векторов в файле

Для сохранения матриц в *текстовых файлах* и чтения их них в системе *MathCAD* имеется семейство встроенных функций:

- ***READPRN***("*путь_к_файлу*") – чтение данных в матрицу из текстового файла;
- ***WRITEPRN***("*путь_к_файлу*") – запись данных из матрицы в новый текстовый файл;
- ***APPENDPRN***("*путь_к_файлу*") – дозапись данных из матрицы в уже существующий текстовый файл.

$$A := \begin{pmatrix} 1 & 2 & 4 \\ 2 & 4 & 8 \\ 4 & 8 & 16 \end{pmatrix}$$

$$\text{WRITEPRN}(\text{"G:\writeprn.txt"}) := A$$



$$B := \text{READPRN}(\text{"G:\writeprn.txt"})$$

$$B = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 4 & 8 \\ 4 & 8 & 16 \end{pmatrix}$$

Рис. 36 – Работа с файлами в MathCAD

Функции WRITEPRN и APPENDPRN отличаются тем, что при записи матрицы в файл первой функцией прежнее содержимое файла (если оно есть) будет стерто, а функция APPENDPRN допишет матрицу в конец адресуемого текстового файла.

Текстовая строка "*путь_к_файлу*" может содержать как полный путь к файлу, так и сокращенный – только имя файла, который должен в этом случае размещаться в рабочем каталоге.

Целочисленные степени квадратных матриц

В среде *MathCAD* целочисленная степень матрицы (в том числе отрицательная) задается точно так же, как и степень обычных числовых переменных (*Листинг 34*). В том случае, когда матрица не имеет обратной, попытка вычислить ее какую-либо отрицательную степень блокируется с выдачей соответствующего сообщения. Нулевая и положительная степень квадратной матрицы вычисляются всегда.

Листинг 34. Целочисленная степень матрицы

$$\begin{pmatrix}1 & 2\\ 3 & 1\end{pmatrix}^2 \rightarrow \begin{pmatrix}7 & 4\\ 6 & 7\end{pmatrix} \qquad \left|\begin{pmatrix}1 & 2\\ 3 & 1\end{pmatrix}\right| = -5 \qquad \left|\begin{pmatrix}7 & 4\\ 6 & 7\end{pmatrix}\right| = 25 \qquad \begin{pmatrix}1 & 2\\ 3 & 1\end{pmatrix}\cdot\begin{pmatrix}1 & 2\\ 3 & 1\end{pmatrix} = \begin{pmatrix}7 & 4\\ 6 & 7\end{pmatrix}$$

Матричные ряды

Существование целочисленных степеней квадратной матрицы (*Листинг 34*) позволяет ввести матричные степенные ряды с числовыми коэффициентами. Из математики известно, что некоторые функции одной переменной $f(x)$ представляются степенными рядами, в общем случае бесконечными, по неотрицательным степеням x:

$$f(x) = \sum_{k=0}^{\infty} \alpha_k \cdot x^k \ . \tag{6}$$

Если в таком ряду формально заменить скалярную переменную x на квадратную матрицу $\mathbf{A}$, то получим обобщение функции $f(x)$ в матричную функцию матричного аргумента:

$$f(\mathbf{A}) = \sum_{k=0}^{\infty} \alpha_k \cdot \mathbf{A}^k \ , \tag{7}$$

где α_k – постоянные коэффициенты.

Здесь мы не будем касаться принципиального вопроса сходимости матричного ряда, поскольку это специальный раздел теории матриц, изложение которого выходит за пределы настоящего курса.

В приложении к решению систем дифференциальных уравнений и в теории устойчивости А.М. Ляпунова, которые будут изучаться на последующих семестрах курса «Высшая математика», важную роль играет матричная экспонента:

$$\mathbf{e}^{\mathbf{A}t} = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k \cdot t^k}{k!}, \tag{8}$$

где $\mathbf{A}$ – квадратная матрица, t – скалярная переменная.

Через матричную экспоненту выражается аналитическое решение системы однородных дифференциальных уравнений с постоянными коэффициентами.

Оператор векторизации функций

В наборе инструментов матричных операций имеется оператор векторизации функций. Пусть $f(x)$ определена как функция скалярного аргумента и задана $\mathbf{M}$ – матрица произвольных размеров с элементами $\mathbf{M}_{i,j}$.

Листинг 35. Целочисленная степень матрицы

$$f(x) := x^2 + 4 \qquad f\left(\overrightarrow{\begin{pmatrix}2 & 0\\ 0 & 1\end{pmatrix}}\right) = \begin{pmatrix}8 & 4\\ 4 & 5\end{pmatrix} \qquad \sin\left(\overrightarrow{\begin{pmatrix}\pi & \frac{-\pi}{4}\\ \frac{\pi}{3} & \frac{\pi}{2}\end{pmatrix}}\right) = \begin{pmatrix}0 & -0.707\\ 0.866 & 1\end{pmatrix} \qquad \left|\begin{pmatrix}-3 & 1\\ 2-i & 4\end{pmatrix}\right| = -14 + i$$

Операция векторизации функции $\vec{\mathbf{f}}(x)$ вызывает появление матрицы с элементами $f(\mathbf{M}_{i,j})$, т.е. происходит поэлементное функциональное преобразование матрицы-аргумента в матрицу-результат того же размера.

Как видно из приведенных примеров, векторизованный оператор |**x**| выполняет замену элементов матрицы их абсолютными величинами, причем для комплексных чисел берется модуль. Матрица-аргумент может быть произвольного размера.

3.2. Векторная геометрия

Математическое представление о векторе как о координатном наборе из *n* чисел, позволяет использовать его для описания физических явлений природы, обладающих *направленностью и величиной*. Таких как перемещение, скорость и ускорение, характеристики гравитационных, магнитных и электрических полей, ток и напряжение электрических схем и др.

Базис и координаты. Тройка $\{\bar{e}_1, \bar{e}_2, \bar{e}_3\}$ линейно независимых векторов в пространстве R^3 называется *базисом*. Любой вектор $\bar{\mathbf{x}}$ может быть единственным образом разложен по базисным векторам, то есть, представлен в виде

$$\bar{x} = x_1 \cdot \bar{e}_1 + x_2 \cdot \bar{e}_2 + x_3 \cdot \bar{e}_3 . \qquad (9)$$

Числа (x_1, x_2, x_3) в разложении (9) называются *координатами вектора* $\bar{x}$ в базисе $\{\bar{e}_1, \bar{e}_2, \bar{e}_3\}$.

Ортонормированный базис. Если векторы $\{\bar{e}_1, \bar{e}_2, \bar{e}_3\}$ попарно перпендикулярны и длина каждого из них равна единице, то базис называется *ортонормированным*, а координаты (x_1, x_2, x_3) – прямоугольными. Базисные векторы ортонормированного базиса будем обозначать $\{\bar{i}, \bar{j}, \bar{k}\}$.

В векторной геометрии определены три вида произведений векторов – это скалярное произведение, векторное и смешанное:

Скалярное произведение – это операция над двумя векторами, результатом которой является число (скаляр), не зависящее от системы координат и характеризующее длины векторов-сомножителей и угол между ними. Данной операции соответствует умножение длины вектора $\bar{x}$ на проекцию вектора $\bar{y}$ на вектор $\bar{x}$. Скалярное произведение может быть вычислено по следующим формулам:

$$\bar{x} \cdot \bar{y} = |\bar{x}| \cdot |\bar{y}| \cdot \cos(\angle \bar{x}\bar{y}) \qquad (10)$$

$$\bar{x} \cdot \bar{y} = x_1 \cdot y_1 + x_2 \cdot y_2 + x_3 \cdot y_3 \qquad (11)$$

Соотнесение формул (10) и (11) позволяет легко вычислить угол $\angle \bar{x}\bar{y}$ между векторами.

Через скалярное произведение определяется ***длина вектора***, под которой обычно понимается его *модуль* или *абсолютное значение*:

$$|\bar{x}| = \sqrt{\bar{x} \cdot \bar{x}} = \sqrt{x_1^2 + x_2^2 + x_3^2} . \qquad (12)$$

Векторным произведением вектора $\bar{x}$ на вектор $\bar{y}$ называется вектор $\bar{x} \times \bar{y}$, удовлетворяющий следующим требованиям:

• длина вектора $|\bar{x} \times \bar{y}|$ равна произведению длин векторов $|\bar{x}|$ и $|\bar{y}|$ на синус угла между ними:

$$|\bar{x} \times \bar{y}| = |\bar{x}| \cdot |\bar{y}| \cdot \sin(\angle \bar{x}\bar{y}); \qquad (13)$$

• вектор $\bar{x} \times \bar{y}$ ортогонален каждому из векторов $\bar{x}$ и $\bar{y}$;

• вектор $\bar{x} \times \bar{y}$ направлен так, что тройка векторов $\bar{x}$, $\bar{y}$, $\bar{x} \times \bar{y}$; является правой (*Рис. 37*).

Векторное произведение рассчитывается так (как определитель записанной ниже матрицы, составленной из базисных ортонормированных векторов $\{\bar{i}, \bar{j}, \bar{k}\}$ и координат векторов $\bar{x}$ и $\bar{y}$):

$$\bar{x} \times \bar{y} = \begin{vmatrix} \bar{i} & \bar{j} & \bar{k} \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix} = (x_2 y_3 - x_3 y_2,\ x_3 y_1 - x_1 y_3,\ x_1 y_2 - x_2 y_1). \quad (14)$$

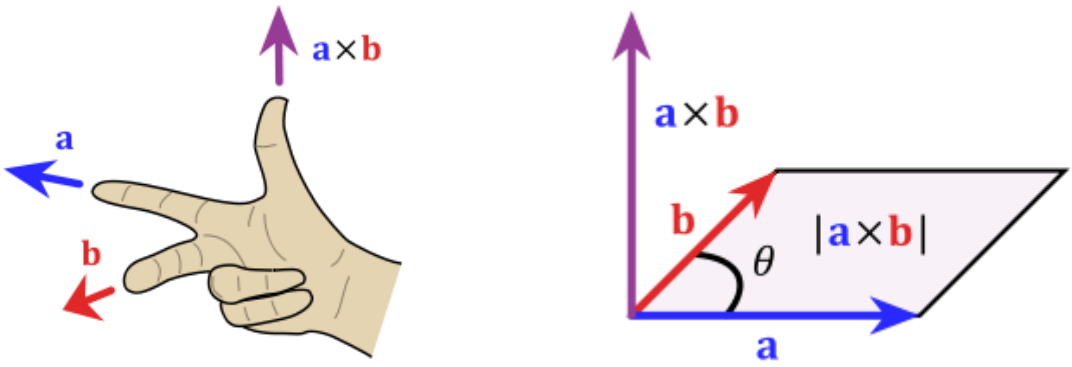


Рис. 37 – Результат векторного произведения направлен так, чтобы образовывать правую тройку векторов

Смешанным произведением трех векторов $\bar{a}$, $\bar{b}$ и $\bar{c}$ называется число, равное скалярному произведению вектора $\bar{a} \times \bar{b}$ на вектор $\bar{c}$: $(\bar{a}, \bar{b}, \bar{c}) = (\bar{a} \times \bar{b}, \bar{c}) = (\bar{a}, \bar{b} \times \bar{c})$.

Иначе произведение $(\bar{a}, \bar{b}, \bar{c})$ может быть задано так:

$$(\bar{a}, \bar{b}, \bar{c}) = \begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix}. \quad (15)$$

Смешанное произведение помимо линейности обладает следующими свойствами:
$(\bar{a}, \bar{b}, \bar{c}) = (\bar{b}, \bar{c}, \bar{a}) = (\bar{c}, \bar{a}, \bar{b}) = -(\bar{b}, \bar{a}, \bar{c}) = -(\bar{c}, \bar{b}, \bar{a}) = -(\bar{a}, \bar{c}, \bar{b})$.

Три вектора называются *компланарными* (или иначе говоря – *линейно зависимыми*) тогда и только тогда, когда $(\bar{a}, \bar{b}, \bar{c}) = 0$, следует из (15).

Тройка векторов является правой (*Рис. 37*) тогда и только тогда, когда $(\bar{a}, \bar{b}, \bar{c}) > 0$. Если же $(\bar{a}, \bar{b}, \bar{c}) < 0$, то векторы $\bar{a}$, $\bar{b}$ и $\bar{c}$ образуют левую тройку векторов.

Смешанное произведение имеет простое геометрическое толкование – это число, по модулю равное объему параллелепипеда, построенного на трех данных векторах.

3.3. Линейная зависимость (независимость) векторов

Важную роль в математике играет понятие линейной зависимости векторов. *Линейная зависимость* – это возможность выразить один вектор через другие вектора. Говорят, что вектор $\bar{z}$ является линейной комбинацией других векторов, например $\bar{x}$ и $\bar{y}$, если существуют такие числа a и b, что $\bar{z} = a \cdot \bar{x} + b \cdot \bar{y}$. Вектор $\bar{z}$ может быть выражен через $\bar{x}$ и $\bar{y}$.

Если теперь из линейно зависимых векторов $\bar{z}$, $\bar{x}$ и $\bar{y}$ составить матрицу **A**, то *определитель* этой матрицы будет равен нулю, а ее *ранг* – будет равен двум.

Здесь важно, что ранг – это количество линейно независимых векторов матрицы, а вектор $\bar{z}$ - зависимый.

Рассматривая матрицу как набор из трех векторов-строк или трех векторов-столбцов, можно сделать вывод, что если в такой матрице только два столбца линейно независимые, то и строк линейно независимых не больше двух.

Листинг 36. Определитель и ранг матрицы, состоящей из ЛЗС векторов

$$v1 := \begin{pmatrix}1\\2\\3\end{pmatrix} \quad v2 := \begin{pmatrix}4\\5\\6\end{pmatrix} \quad v3 := \begin{pmatrix}7\\8\\9\end{pmatrix} \quad A := \text{augment}(v1, v2, v3) \quad A = \begin{pmatrix}1&4&7\\2&5&8\\3&6&9\end{pmatrix} \quad \text{rref}(A) \rightarrow \begin{pmatrix}1&0&2\\0&1&0\\0&0&0\end{pmatrix}$$

$$|A| = 0 \qquad \text{rank}(A) = 2$$

В примере (*Листинг 37*) матрица ***A*** составляется из линейно-зависимых векторов ***v1***, ***v2*** и ***v3***, что определяет величину 2 ранга матрицы. И равенство нулю ее определителя.

Функция *rref( )* в данном примере служит для приведения матрицы к треугольной форме. Поскольку система линейно-зависима, в полученной треугольной матрице *rref(A)* третья строка нулевая.

В листинге ниже доказывается, что вектор-столбец ***v3*** является линейной комбинацией векторов ***v1*** и ***v2***. Здесь подобраны коэффициенты (a= -1, b= 2) линейного разложения вектора ***v3*** через векторы ***v1*** и ***v2***.

Листинг 37. Построение линейной комбинации векторов-столбцов матрицы A

$$a \cdot v1 + b \cdot v2 = v3$$

$$aa := \begin{pmatrix}1&4\\2&5\end{pmatrix} \quad bb := \begin{pmatrix}7\\8\end{pmatrix} \quad \text{rref}(\text{augment}(aa, bb)) = \begin{pmatrix}1&0&-1\\0&1&2\end{pmatrix} \quad -1 \cdot v1 + 2v2 = \begin{pmatrix}7\\8\\9\end{pmatrix}$$

Поскольку система линейно-зависима, в полученной треугольной матрице *rref(A)* третья строка нулевая (см. *Листинг 36*). Это значит, что и среди строк ***s1***, ***s2*** и ***s3*** имеется одна – являющаяся линейной комбинацией двух других.

Листинг 38. Построение линейной комбинации векторов-строк матрицы A

$$s1 := \text{submatrix}(A, 0, 0, 0, 2) \quad s1 = (1 \;\; 4 \;\; 7) \quad s2 := \text{submatrix}(A, 1, 1, 0, 2) \quad s2 = (2 \;\; 5 \;\; 8)$$

$$a \cdot s1 + b \cdot s3 = s2 \qquad s3 := \text{submatrix}(A, 2, 2, 0, 2) \quad s3 = (3 \;\; 6 \;\; 9)$$

$$aa := \begin{pmatrix}1&3\\4&6\end{pmatrix} \quad bb := \begin{pmatrix}2\\5\end{pmatrix} \quad \text{rref}(\text{augment}(aa, bb)) = \begin{pmatrix}1&0&0.5\\0&1&0.5\end{pmatrix} \quad 0.5 \cdot s1 + 0.5 \cdot s3 = (2 \;\; 5 \;\; 8)$$

В этом листинге рассчитаны коэффициенты (a= 0.5 и b= 0.5) линейного разложения вектора-строки ***s2*** через векторы-строки ***s1*** и ***s3***.

3.4. Собственный вектор и собственное значение

Особое значение в алгебре имеет представление о матрице как о некотором ***линейном преобразовании*** (функции) над векторами. Действительно, умножение матрицы на вектор в результате дает некоторый новый вектор, т.е. матрица как бы преобразует вектор. На изучении свойств таких преобразований базируется вся компьютерная графика, часть теории дифференциальных уравнений и все операционное исчисление. Здесь мы лишь коснемся двух базовых понятий теории линейных пространств – *собственный вектор* и *собственное число* матрицы (линейного преобразования).

Собственный вектор $\bar{\mathbf{x}}$ – это вектор, умножение матрицы **A** на который даёт тот же самый вектор, умноженный на некоторое скалярное число λ, называемое *собственным числом* матрицы:

$$\mathbf{A} \cdot \bar{\mathbf{x}} = \lambda \cdot \bar{\mathbf{x}}. \qquad (16)$$

Понятия собственного вектора и собственного числа являются одними из ключевых в линейной алгебре, на их основе строится множество конструкций. Множество всех собственных векторов линейного преобразования называется *собственным*

подпространством, множество всех собственных значений матрицы или линейного преобразования – *спектром матрицы* или преобразования.

В *MathCAD* существуют функции *eigenvals( )* и *eigenvec( )* для нахождения собственных значений и собственных векторов матрицы (16), кроме того имеется функция *eigenvecs( )* для получения всех собственных векторов сразу.

В примере, приведенном ниже (*Листинг 39*) для матрица A, заданной произвольно, рассчитываются собственные числа при помощи функции *eigenvals( )*. В нашем случае – два из них комплексно-сопряженные, а одно – вещественное. Затем для них строятся собственные векторы *eigenvec( )* и *eigenvecs( )*, после чего производится проверка правильности вычисления собственных значений и векторов по формуле (16).

Листинг 39. Собственные вектора и собственные значения матрицы

$$A := \begin{pmatrix} 1 & 2 & 1 \\ -3 & 3 & 2 \\ 3 & -3 & -1 \end{pmatrix} \quad |A| = 9 \quad c := \text{eigenvals}(A) \quad \text{eigenvals}(A) = \begin{pmatrix} 0.718 + 2.289i \\ 0.718 - 2.289i \\ 1.564 \end{pmatrix}$$

$$v0 := \text{eigenvec}(A, c_0) \quad v0 = \begin{pmatrix} -0.227 - 0.231i \\ 0.331 - 0.566i \\ -0.07 + 0.679i \end{pmatrix} \quad v2 := \text{eigenvec}(A, c_2) \quad v2 = \begin{pmatrix} 0.421 \\ -0.308 \\ 0.853 \end{pmatrix}$$

$$\text{eigenvecs}(A) = \begin{pmatrix} -0.206 + 0.249i & -0.206 - 0.249i & 0.421 \\ -0.597 - 0.271i & -0.597 + 0.271i & -0.308 \\ 0.682 & 0.682 & 0.853 \end{pmatrix}$$

Проверка:

$$A \cdot v1 = \begin{pmatrix} 0.366 + 0.684i \\ 1.533 - 0.351i \\ -1.603 - 0.327i \end{pmatrix} \quad c_1 \cdot v1 = \begin{pmatrix} 0.366 + 0.684i \\ 1.533 - 0.351i \\ -1.603 - 0.327i \end{pmatrix}$$

3.5. Сила Лоренца в электромагнитном поле

Понятие вектора используется в электрофизике для работы с объектами имеющими ***направление и величину***. Рассмотрим задачи на вычисление силы Лоренца в электромагнитном поле. Сила $\bar{F}$ [Н], действующая на частицу с электрическим зарядом q [Кл], движущуюся со скоростью $\bar{v}$ [м/с], во внешнем электрическом $\bar{E}$ [В/м] и магнитном $\bar{B}$ [Тл] полях, такова:

$$\bar{F} = q \cdot \bar{E} + q \cdot \bar{v} \times \bar{B}. \qquad (17)$$

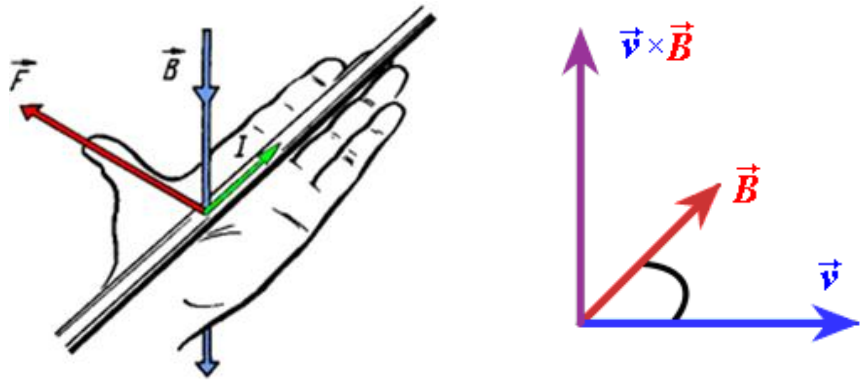


Рис. 38 – Результат векторного произведения направлен так, чтобы образовывать правую тройку векторов

При решении задач такого рода, предполагается, что направления силовых линий электрического ($\bar{E}$) и магнитного ($\bar{B}$) поля, а также направление движения

заряда ($\bar{v}$) описываются соответствующими векторами, имеющими три координаты в пространстве. Направление результирующей силы вычисляется в соответствии с законом (17).

Сила Лоренца в магнитном поле (напряженность электрического поля $\bar{E}$ равна 0)

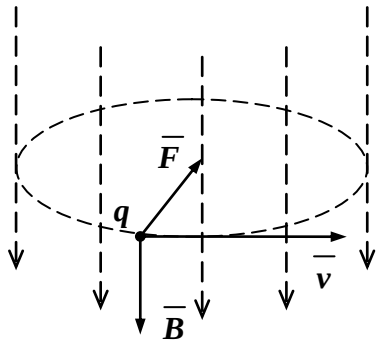
Магни́тная инду́кция $\bar{B}$ [Тл] – векторная величина, являющаяся силовой характеристикой магнитного поля (его действия на заряженные частицы) в данной точке пространства. Иначе, $\bar{B}$ – это такой вектор, что сила Лоренца $\bar{F}$, действующая со стороны магнитного поля на заряд q, движущийся со скоростью $\bar{v}$, равна

$$\bar{F} = q \cdot \bar{v} \times \bar{B}, \tag{18}$$

а ее модуль $$\left|\bar{F}\right| = q \cdot \left|\bar{v}\right| \cdot \left|\bar{B}\right| \cdot \sin(\alpha). \tag{19}$$

α — угол между векторами скорости и магнитной индукции (направление вектора $\bar{F}$ перпендикулярно им обоим и направлено по правилу векторного произведения, *см. Рис. 38*).

В однородном магнитном поле (*Рис. 39*) частица будет двигаться по окружности, при этом сила Лоренца будет центростремительной силой. Работа при этом не будет совершаться.



В данном случае сила Лоренца направлена к центру окружности и ускорение, ею создаваемое, направлено туда же, то есть это и есть центростремительное ускорение.

$\bar{F} = m\bar{a}$

Рис. 39 – Сила Лоренца частицы, движущаяся по окружности

Определить силу, действующую на протон, движущуюся со скоростью $\left(9.8 \cdot 10^5, 10^5, 1.2 \cdot 10^6\right)$ [м/с] в электромагнитном поле с напряженностью $\left(2 \cdot 10^{12}, 3 \cdot 10^{12}, 3.5 \cdot 10^{12}\right)$ [В/м] и магнитной индукцией $\left(2.2 \cdot 10^6, 3 \cdot 10^7, 5.7 \cdot 10^6\right)$ [Тл].

Листинг 40. Пример расчета силы Лоренца в электромагнитном поле

$q := 1.602176565 \cdot 10^{-19}$ [Кл] (заряд протона)

$$v := \begin{pmatrix} 9.8 \cdot 10^5 \\ 10^5 \\ 1.2 \cdot 10^6 \end{pmatrix} \text{[м/с]} \qquad B := \begin{pmatrix} 2.2 \cdot 10^6 \\ 3 \cdot 10^7 \\ 5.7 \cdot 10^6 \end{pmatrix} \text{[Тл]} \qquad E := \begin{pmatrix} 2 \cdot 10^{12} \\ 3 \cdot 10^{12} \\ 3.5 \cdot 10^{12} \end{pmatrix} \text{[В/м]}$$

$F := q \cdot E + q \cdot (v \times B)$ Сила Лоренца в электро-магнитном поле

$$F = \begin{pmatrix} -5.356 \times 10^{-6} \\ 8.652 \times 10^{-9} \\ 5.236 \times 10^{-6} \end{pmatrix} \text{[Н]} \qquad |F| = 7.490154 \times 10^{-6}$$

Угол между векторами v и B

$\alpha := \operatorname{acos}\left(\frac{v \cdot B}{|v| \cdot |B|}\right)$ $\alpha = 75.382 \cdot \text{deg}$

Абсолютное значение силы Лоренца в магнитном поле

$absF := |q| \cdot |v| \cdot |B| \cdot \sin(\alpha)$ $absF = 7.36903 \times 10^{-6}$

магнитная составляющая силы Лоренца

$FB := q \cdot (v \times B)$

$$FB = \begin{pmatrix} -5.677 \times 10^{-6} \\ -4.72 \times 10^{-7} \\ 4.675 \times 10^{-6} \end{pmatrix}$$

$|FB| = 7.36903 \times 10^{-6}$

электрическая составляющая силы Лоренца

$FE := q \cdot E$

$$FE = \begin{pmatrix} 3.204 \times 10^{-7} \\ 4.807 \times 10^{-7} \\ 5.608 \times 10^{-7} \end{pmatrix}$$

$$FB + FE = \begin{pmatrix} -5.356 \times 10^{-6} \\ 8.652 \times 10^{-9} \\ 5.236 \times 10^{-6} \end{pmatrix}$$

$|FB + FE| = 7.490154 \times 10^{-6}$

Контрольные вопросы

1. Как вы понимаете понятие матрица? Вектор? Что такое вектор-строка и вектор-столбец матрицы?
2. Что такое транспонирование матрицы?
3. Является ли матричное произведение коммутативным? А для квадратных матриц? Как вычислить определитель матрицы?
4. Что такое линейные операции с матрицами, векторами?
5. Как вычислить определитель матрицы разложением по строке (столбцу)?
6. Что такое обратная матрица? Когда она существует? Что такое линейная комбинация векторов?
7. Как с помощью средств MathCAD выделить нужную строку матрицы?
8. Что такое обратная матрица? Как это понятие связано с обратным числом?
9. Как выделить требуемый фрагмент (подматрицу) заданной матрицы в MathCAD?
10. Как понятие вектор n-мерного пространства связано со школьным понятием вектора (направленного отрезка)?
11. Какие операции над строками (столбцами) матрицы называются «эквивалентные преобразования»? Почему?
12. Что такое сила Лоренца? Чему она равна?
13. Дайте определение скалярного произведения векторов. Опишите его свойства.
14. Какие векторы называются линейно зависимыми? Дайте строгое определение.
15. Как с помощью средств MathCAD выделить нужный столбец матрицы?
16. Дайте определение векторного произведения векторов. Опишите его свойства.
17. Как определить число линейно независимых векторов в системе?
18. Что такое смешанное произведение векторов?
19. Как задать матрицу (вектор) в MathCAD? Различаются ли эти понятия?
20. Какие векторы называются собственными векторами матрицы? Что такое собственные числа матрицы?
21. Какие векторы называются линейно независимыми? Дайте строгое определение.
22. Что такое базис? Ортонормированный базис?
23. Какая система векторов может считаться базисом?
24. Как вычислить силу Лоренца в магнитном поле? В электромагнитном поле?
25. Как вы понимаете понятие линейный оператор? Как это понятие связано с матрицей?
26. Как выглядит матричная единица? Матричный ноль?
27. Как изменяется определитель матрицы при эквивалентных преобразованиях строк, столбцов матрицы?
28. Как вычислить обратную матрицу через алгебраические дополнения?
29. Что такое определитель матрицы?

30. Какая матрица называется диагональной? Треугольной?
31. Какие матрицы можно перемножать?
32. Как величина определителя связано с числом линейно независимых строк (столбцов) квадратной матрицы?
33. Что такое ранг матрицы? Как значение ранга связано с определителем?
34. Как связано существование обратной матрицы и величина определителя?
35. Как изменяется определитель матрицы при ее умножении на число?
36. Что такое ранг матрицы? Как его рассчитать? Как он вычисляется в MathCAD?

4. СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Системы линейных алгебраических уравнений (СЛАУ) и матрицы весьма тесно связаны. Матричная алгебра возникла в связи развитием методов решения линейных уравнений (алгебраических, дифференциальных) и отражает основные законы преобразования систем таких уравнений. Система линейных алгебраических уравнений в матричной форме выглядит так:

$$\begin{cases} a_{1,1}\cdot x_1 + a_{1,2}\cdot x_2 + a_{1,3}\cdot x_3 = b_1; \\ a_{2,1}\cdot x_1 + a_{2,2}\cdot x_2 + a_{2,3}\cdot x_3 = b_2; \\ a_{3,1}\cdot x_1 + a_{3,2}\cdot x_2 + a_{3,3}\cdot x_3 = b_3; \end{cases} \quad \mathbf{A}\cdot\mathbf{x}=\mathbf{b}, \tag{20}$$

где **A** – матрица с *m* строками и *n* столбцами с известными элементами; **x** – вектор искомых решений размера *n*; **b** — вектор правых частей с известными элементами размера *m*:

$$\mathbf{A}=\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}, \ \mathbf{x}=\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \ \mathbf{b}=\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}, \tag{21}$$

При решении СЛАУ матрицу системы **A** нельзя, разумеется, рассматривать отдельно от вектора правых частей **b**, поэтому в решении СЛАУ главную роль играет ***расширенная матрица*** системы **Ab**, составляемая из элементов **A** и **b**:

$$\mathbf{Ab}=\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & b_1 \\ a_{2,1} & a_{2,2} & a_{2,3} & b_2 \\ a_{3,1} & a_{3,2} & a_{3,3} & b_3 \end{pmatrix} \tag{22}$$

4.1. Существование решений СЛАУ

В разделе матричной алгебры курса «*Высшая математика*» рассматривается вопрос о том, когда система вида (20)-(21) имеет решение. Здесь мы не будем разбирать его подробно, отметим лишь самое главное:

Рис. 40 – Существование и единственность решений СЛАУ

1. Если матрица **A** квадратная (*n=m*) и *невырожденная*, т.е. её определитель не равен нулю **det(A)≠0**, а следовательно ранг этой матрицы равен её размерности **rank(A)=n=m**, а значит и равен рангу расширенной матрицы. Для такой матрицы(см. *Рис. 40*) решение существует и единственно.
2. Если определитель квадратной матрицы **A** системы **A·x=b** равен нулю **det(A)=0**, то это означает, что линейно независимых строк в матрице системы меньше, чем ее размерность. Т.е. существует хотя бы одна линейно зависимая строка (а сколько их точно - определяется по рангу матрицы **A**).
    a. Теперь следует рассмотреть расширенную матрицу системы **augment(A,b)** если в ней столько же линейно независимых строк, сколько и в матрице **A** (их ранги совпадают), то в системе решения существуют и их бесконечно много.
    b. Если же число линейно независимых строк матрицы **A** и расширенной матрицы не совпадают (их ранги не равны), то решения в такой системе уравнений нет.

4.2. Поиск решений СЛАУ через обратную матрицу

Если матрица **A** квадратная (***n=m***) и *невырожденная*, т.е. её определитель не равен нулю **det(A)≠0,** а следовательно ранг этой матрицы равен её размерности **rank(A)=n=m**, а значит и равен рангу расширенной матрицы **rank(A)=rank(augment(A,b))**. Для такой матрицы (см. *Рис. 40*) решение существует и единственно. Построим это решение. Для чего домножим обе части системы уравнений **A·x=b** слева на обратную матрицу $\mathbf{A^{-1}}$. Здесь важно понимать, что *обратная матрица* существует только для невырожденной матрицы **A**, кроме того, умножать обе части уравнений необходимо с одной стороны (слева), так как мы помним, что матричное произведение не является коммутативным.

Получим $\mathbf{A^{-1}\cdot A\cdot x=A^{-1}\cdot b}$. Но мы знаем из свойств обратной матрицы, что $\mathbf{A^{-1}\cdot A=E}$. Тогда наше уравнение примет вид $\mathbf{E\cdot x=A^{-1}\cdot b}$. Из свойств единичной матрицы **E** следует, что **E·x=x.** Отсюда получается, что $\mathbf{x=A^{-1}\cdot b,}$ т.е. вектор-решение **x** удалось записать в явном виде.

Искомое единственное решение системы уравнений **A·x=b** можно построить по формуле $\mathbf{x=A^{-1}b}$ – найти его непосредственно при помощи умножения слева на обратную матрицу.

Кроме того, его можно отыскать при помощи встроенной функции **lsolve(A,b):**

Листинг 41. СЛАУ непосредственный поиск решения. Функция lsolve

Матрица системы:

$$A := \begin{pmatrix} 3 & -1 & 0 \\ -2 & 1 & 1 \\ 2 & -1 & 4 \end{pmatrix}$$

Матрица правой части:

$$b := \begin{pmatrix} 5 \\ 0 \\ 15 \end{pmatrix}$$

Вычисление определителя

$$|A| = 5$$

Вычисление решения системы

$$x := A^{-1} \cdot b \qquad x = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

Решение системы с помощью функции lsolve

$$x := \text{lsolve}(A, b) \qquad x = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

Проверка правильности решения

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

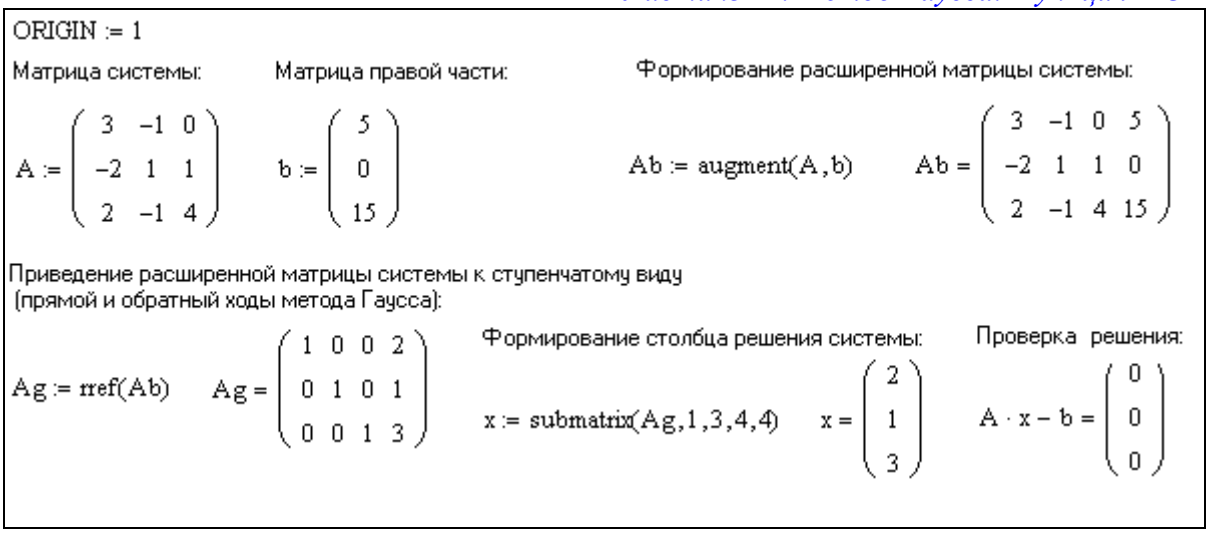
4.3. Метод Гаусса

Системы линейных алгебраических уравнений (20)-(21) встречались обучаемому еще в курсе средней школы. Там для решения подобных систем использовался метод

последовательного выражения неизвестных x_i, $i = 1..n$ и подстановки полученного равенства в оставшиеся уравнения.

Этот метод в математике называется методом Гаусса, и в матричном виде сводится к эквивалентному домножению на константу и суммированию строк (но не столбцов) расширенной матрицы $\mathbf{Ab} = augment(\mathbf{A}, \mathbf{b})$ с целью приведения ее к ступенчатому виду – как матрица **Ag** на листинге (*Листинг 42*). Крайний правый столбец матрицы **Ag** – решение.

Листинг 42. Метод Гаусса. Функция **rref**



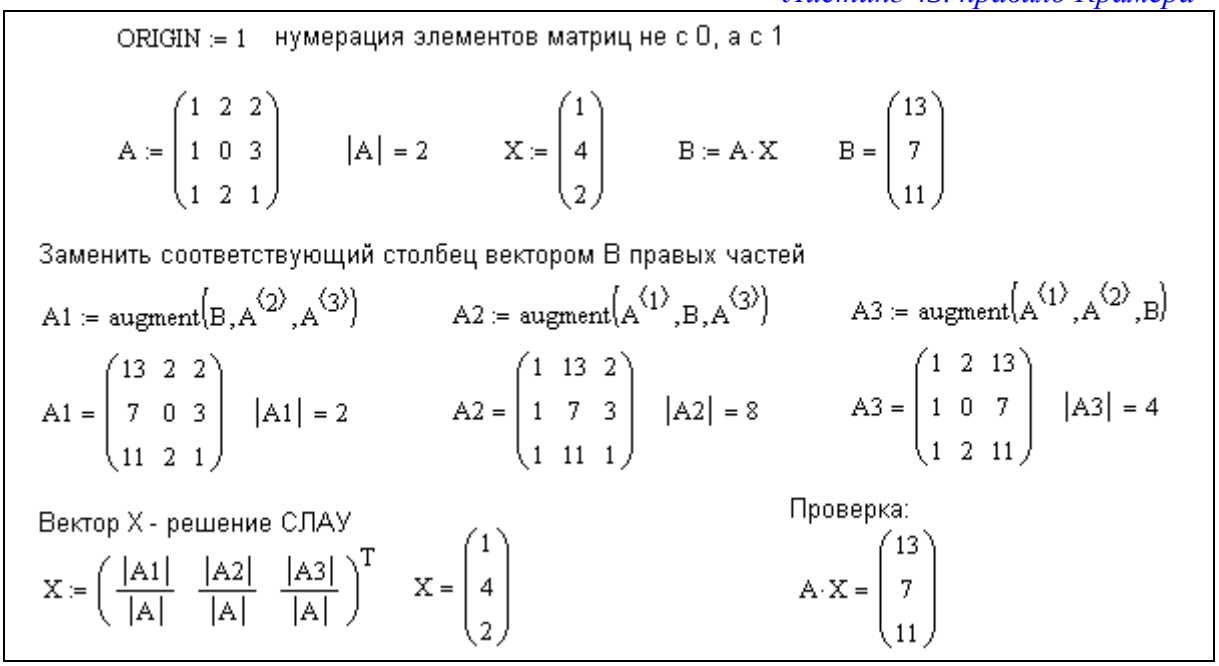
Команда **ORIGIN:=1** переопределяет принцип нумерации элементов матриц – теперь они нумеруются не с 0, а с 1.

Метод Гаусса можно реализовать вручную преобразуя строки матрицы, а можно использовать встроенную функцию **rref()**, как показано на примере. Можно доказать, что полученная матрица **Ag** эквивалентна матрице **Ab** и описывает одну и ту же исходную систему линейных алгебраических уравнений.

4.4. Правило Крамера

Еще один широко известный способ построения решения системы (20) – правило Крамера (*Листинг 43*).

Листинг 43. правило Крамера



Этот метод состоит в последовательном построении вспомогательных матриц $\mathbf{A}_1$, $\mathbf{A}_2$, … $\mathbf{A}_n$ путем замены i-того столбца матрицы вектором правых частей **b** и вычисления неизвестных переменных x_i, $i = 1..n$ как отношение определителей матриц $\mathbf{A}_i$ и **A**.

Разумеется, корни системы уравнений (20) можно отыскать известными нам из предыдущих занятий операторами *Given-Find* и *Given-Minerr*.

4.5. СЛАУ вырожденный случай

Главным критерием существования решения СЛАУ является *ранг* матрицы. В примерах, рассмотренных выше *Листинг 41 - Листинг 43,* ранг основной матрицы равнялся рангу расширенной матрицы – самый простой тип СЛАУ, для которого *существует единственное решение*. Это объясняется тем, что количество линейно-независимых уравнений системы (20) равняется количеству неизвестных.

Если в СЛАУ количество уравнений больше числа неизвестных, ясно, что такая система избыточна. И может быть противоречивой, в этом случае решения у системы может и не быть. Вопрос о существовании решения в этом случае сводится к поиску непротиворечивого подмножества всех уравнений – такого, для которого ранг основной матрицы равен рангу расширенной и равен числу неизвестных.

В этом случае множество решений строится следующим образом. Поскольку количество уравнений m меньше числа неизвестных n ($n>m$), нам необходимо выбрать m ***зависимых переменных*** (обозначим их вектором $\mathbf{x}_1$) и выразить их через оставшиеся ($n–m$) переменных – ***свободные переменные*** (обозначим их вектором $\mathbf{x}_2$). Это возможно, если матрица **A** имеет m независимых строк, иными словами ни одно из уравнений не является линейной комбинацией других. Систему уравнений можно преобразовать к виду:

$$\mathbf{A}_1 \cdot \mathbf{x}_1 = \mathbf{b} - \mathbf{A}_2 \cdot \mathbf{x}_2. \qquad (23)$$

где $\mathbf{A}_1$ — квадратная матрица $m \times m$, состоящая из коэффициентов при компонентах вектора $\mathbf{x}_1$, а $\mathbf{A}_2$ — матрица, содержащая m строк и ($n–m$) столбцов, образованная коэффициентами при компонентах вектора $\mathbf{x}_2$.

Задав свободные переменные $\mathbf{x}_2$ какими-либо значениями мы получим новый вектор правых частей $\mathbf{b}_1 = \mathbf{b} - \mathbf{A}_2 \cdot \mathbf{x}_2$, и можем записать систему уравнений для зависимых переменных $\mathbf{x}_1$:

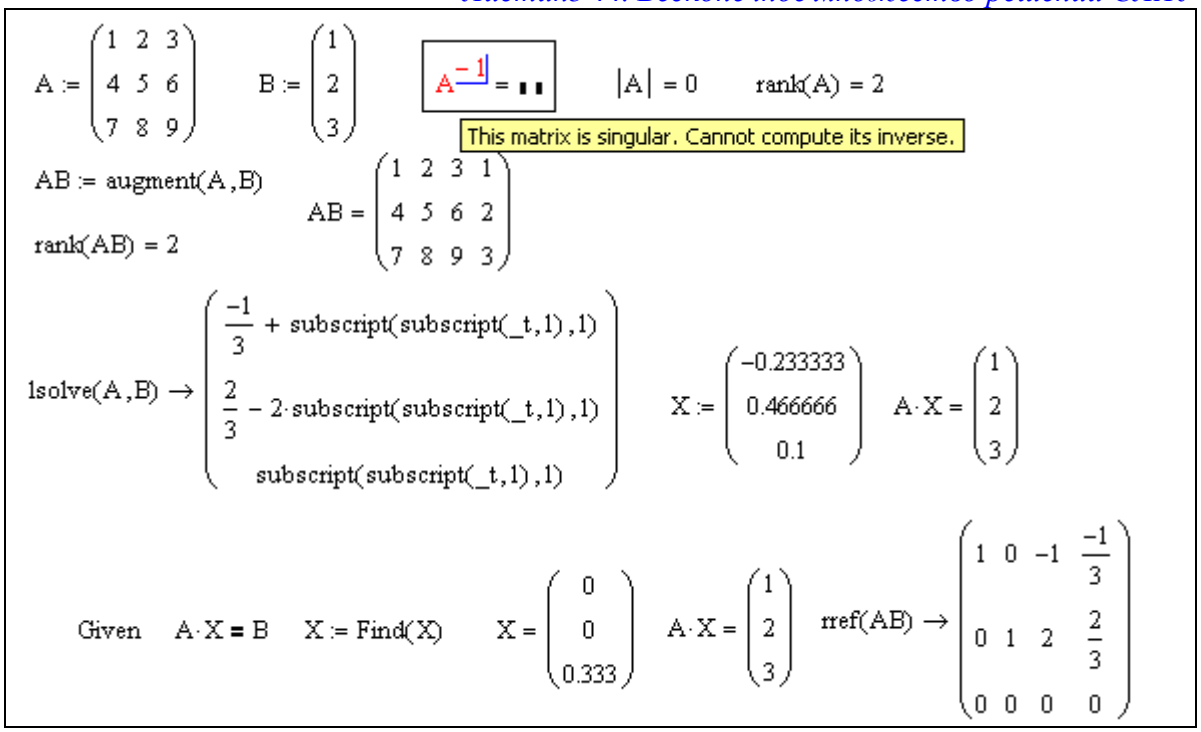
$$\mathbf{A}_1 \cdot \mathbf{x}_1 = \mathbf{b}_1, \quad \text{где } \mathbf{b}_1 = \mathbf{b} - \mathbf{A}_2 \cdot \mathbf{x}_2. \qquad (24)$$

У этой системы матрица $\mathbf{A}_1$ невырожденная и решение существует и единственное (при заданных ранее свободных переменных $\mathbf{x}_2$).

Ясно, что решение такой системы будет представлять собой некоторую матричную функцию от свободных переменных $\bar{\mathbf{x}}_1 = \mathbf{F}(\bar{\mathbf{x}}_2)$, которую можно записать таким образом: $\mathbf{A}_1 \cdot \mathbf{x}_1 = \mathbf{b}_1$. Реализация таких вычислений сводится к компоновке матриц $\mathbf{A}_1$ и $\mathbf{A}_2$ из столбцов исходной матрицы **A** и взятию обратной матрицы $\mathbf{A}_1^{-1}$, выполнение этих действий в среде *MathCAD* не составляет проблем.

Пример ниже (*Листинг 44*) иллюстрирует неоднозначность решений СЛАУ вида (23) в зависимости от того, как задана свободная переменная. Можно видеть, что разные методы решения – *lsolve, Given-Find* и *rref* выдают различные результаты. В связи с этим, к поиску решения СЛАУ нельзя подходить «бездумно», *MathCAD* –это всего лишь инструмент, он помогает в изучении математики, но не заменяет её.

Листинг 44. Бесконечное множество решений СЛАУ

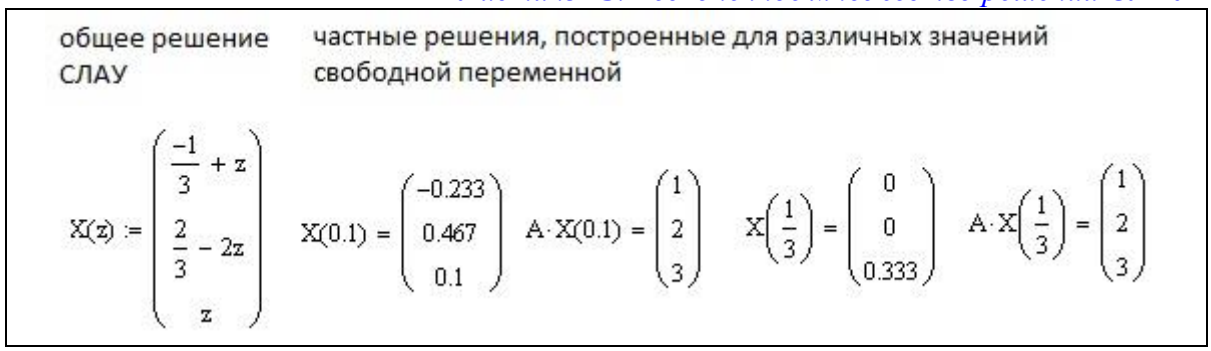


В приведенном выше примере функция *lsolve* возвращает символьные зависимости, показывающие, что решение может быть выражено через параметр t_. А функция *rref* возвращает матрицу, эквивалентную **A**, последняя строка которой состоит из нулей. Запишем, какой системе уравнений соответствует матрица *rref*(AB):

$$\begin{cases} x - z = -\frac{1}{3}, \\ y + 2z = \frac{2}{3}. \end{cases} \qquad \begin{cases} x = -\frac{1}{3} + z, \\ y = \frac{2}{3} - 2z. \end{cases}$$

Теперь понятно, почему переменная *z* называется *свободной переменной*, а *x* и *y* – *зависимыми*. Здесь переменные *x* и *y* – могут быть выражены через *z*, и в зависимости от неё будут принимать различные значения:

Листинг 45. Бесконечное множество решений СЛАУ



Свободным переменным присваиваются произвольно различные значения. Далее по закону, называемому ***общим решением СЛАУ*** (или её ***фундаментальным решением***), строится частный случай этого закона - ***частное решение*** исходной СЛАУ. Каждое из них удовлетворяет СЛАУ, а значит является решением. Отсюда и бесконечное число таких частных решений.

Если же число линейно независимых строк матрицы **A** и расширенной матрицы **augment(A,b)** не совпадают (их ранги не равны), то решения в такой системе уравнений не существует.

4.6. Однородная СЛАУ

Рассмотрим частный случай СЛАУ, когда все компоненты вектора **b** равны нулю. Такая система уравнений называется однородной:

$$\mathbf{A \cdot x = 0} \qquad (25)$$

Как строятся её решения? Совершенно ясно, что однородная система ***всегда имеет решение***. Например, это так называемое ***тривиальное решение*** $x_1=x_2=\ldots x_m=0$. Существуют ли другие, нетривиальные решения?

Теорема: однородная система линейных уравнений имеет *единственное только тривиальное решение*, если ранг матрицы системы равен количеству переменных.

*Рис. 41 – Существование и единственность решений **однородной** СЛАУ*

Однако на практике гораздо более распространен случай, когда строки матрицы системы *линейно зависимы*. И тогда неизбежно появление общего решения, зависящего от свободных переменных, аналогично тому, как строилось такое решение в предыдущем параграфе.

Листинг 46. Однородная нетривиальная СЛАУ - общее решение по методу Гаусса

$$A = \begin{pmatrix} 1 & 2 & 3 & 1 \\ 2 & 3 & -1 & 0 \\ 0 & -1 & 1 & -1 \\ -1 & -2 & 5 & 0 \end{pmatrix} \qquad b := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad |A| = 0 \qquad \operatorname{rank}(A) = 3$$

$$\operatorname{rref}(A) \rightarrow \begin{pmatrix} 1 & 0 & 0 & -\frac{13}{8} \\ 0 & 1 & 0 & \frac{9}{8} \\ 0 & 0 & 1 & \frac{1}{8} \\ 0 & 0 & 0 & 0 \end{pmatrix} \qquad R(z) := \begin{pmatrix} \frac{13 \cdot z}{8} \\ -\frac{9 \cdot z}{8} \\ -\frac{z}{8} \\ z \end{pmatrix}$$

Листинг 47. Однородная нетривиальная СЛАУ - частные решения с проверкой

$$R(1)=\begin{pmatrix}1.625\\-1.125\\-0.125\\1\end{pmatrix}\quad R(2)=\begin{pmatrix}3.25\\-2.25\\-0.25\\2\end{pmatrix}\quad R(3)=\begin{pmatrix}4.875\\-3.375\\-0.375\\3\end{pmatrix}\quad R(80)=\begin{pmatrix}130\\-90\\-10\\80\end{pmatrix}$$

$$A\cdot R(1)=\begin{pmatrix}0\\0\\0\\0\end{pmatrix}\quad A\cdot R(2)=\begin{pmatrix}0\\0\\0\\0\end{pmatrix}\quad A\cdot R(3)=\begin{pmatrix}0\\0\\0\\0\end{pmatrix}\quad A\cdot R(80)=\begin{pmatrix}0\\0\\0\\0\end{pmatrix}$$

В примере выше (*Листинг 47*) общее решение **R(z)** однородной нетривиальной СЛАУ строилось вручную. Ниже рассмотрен алгоритм, разбивающий переменные на зависимые и независимые (независимая одна - обозначим ее **z**). Тогда вектор правых частей для зависимой (и уже неоднородной) СЛАУ будет $\mathbf{b_1(z)}$, а матрица этой системы $\mathbf{A_1}$. Разрешим неоднородную СЛАУ для зависимых переменных при помощи обратной матрицы и припишем к ним независимые - получим общее решение **R(z)**:

Листинг 48. Однородная нетривиальная СЛАУ - алгоритм построения общего решения

$$A1 := \text{submatrix}(A,1,3,1,3)$$

$$b1(z) := -z\cdot\text{submatrix}(A,1,3,4,4)$$

$$R(z) := \text{stack}\left(A1^{-1}\cdot b1(z), z\right)$$

$$R(z)\rightarrow\begin{pmatrix}\frac{13\cdot z}{8}\\-\frac{9\cdot z}{8}\\-\frac{z}{8}\\z\end{pmatrix}$$

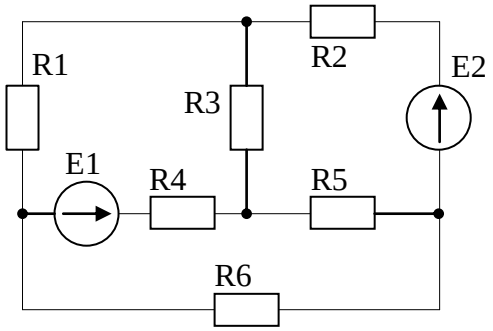
Контрольные вопросы

1. Как определить, разрешима ли однородная система алгебраических уравнений?
2. Если система имеет решения, то как определить, сколько их?
3. Как отделить зависимые переменные от свободных? Сколько их?
4. Что такое расширенная матрица системы?
5. В каких случаях решение неоднородной СЛАУ единственно?
6. Что такое ранг СЛАУ? Чем он отличается от ранга матрицы СЛАУ?
7. В чем смысл метода Гаусса решения СЛАУ?
8. В каких случаях не существует решения неоднородной СЛАУ?
9. Как решить СЛАУ по методу Крамера? Можно ли решить вырожденную систему?
10. Какие методы поиска решений СЛАУ вам известны?
11. Как решить СЛАУ, если определитель матрицы равен 0?
12. Как определить, разрешима ли неоднородная система алгебраических уравнений?
13. В каких случаях не существует решения однородной СЛАУ?
14. В каких случаях решение однородной СЛАУ единственно?

5. РАСЧЕТ ЭЛЕКТРИЧЕСКИХ ЦЕПЕЙ ПОСТОЯННОГО ТОКА

Умение находить решение систем линейных алгебраических уравнений востребовано в различных разделах естественных и технических наук. Рассмотрим, как применяются данное умение в электротехнике. Одна из задач состоит в расчете направлений протекания и величин токов в электрических цепях постоянного тока (и напряжения).

Рассмотрим, в качестве примера, простейшую задачу из теории цепей постоянного тока.



E1= 250 [В]
E2= 100 [В]
R1= 35 [Ом]
R2= 80 [Ом]
R3= 20 [Ом]
R4= 100 [Ом]
R5= 150 [Ом]
R6= 40 [Ом]

Рис. 42 – Задача на баланс мощностей в цепи постоянного тока

На рисунке изображена электрическая цепь, включающая *идеальные источники ЭДС* E1, E2 и *резисторы* R1-R6. В правой части рисунка приведены номиналы этих элементов. Ставится задача при помощи закона Ома и законов Кирхгофа определить токи, протекающие по всем элементам цепи и вычислить генерируемую и потребляемую схемой мощность. Вспомним эти законы:

Первый закон Кирхгофа

Алгебраическая сумма токов в каждом узле любой цепи равна нулю, при этом втекающий в узел ток принято считать положительным, а вытекающий – отрицательным.

$$\sum_{k=1}^{n} i_k = 0 \qquad (26)$$

Иными словами, сколько тока втекает в узел, столько из него и вытекает. Это правило следует из фундаментального закона сохранения заряда.

Второй закон Кирхгофа

Алгебраическая сумма падений напряжений во всех замкнутых контурах цепи, равна алгебраической сумме ЭДС этого контура. Если в контуре нет источников ЭДС, то суммарное падение напряжений в контуре равно нулю.

$$\sum_{k=1}^{n} E_k = \sum_{k=1}^{m} U_k \qquad (27)$$

Иными словами, при полном обходе контура потенциал, изменяясь, возвращается к исходному значению. Это правило вытекает из 3-го уравнения Максвелла для стационарного магнитного поля.

Частным случаем второго закона Кирхгофа для цепи, состоящей из одного контура, является закон Ома для этой цепи.

Закон Ома

Сила тока в участке цепи прямо пропорциональна напряжению *и обратно пропорциональна электрическому сопротивлению данного участка цепи.*

$$I = \frac{U}{R} \qquad (28)$$

Рассмотрим применение законов (26)-(28) на примере расчета токов в цепи (*Рис. 42*).

При построении уравнений и проведении расчетов необходимо учитывать тот факт, что схема содержит *идеальные источники* напряжения E1 и E2, т.е. вводится допущение, что внутреннее сопротивление этих источников равно нулю.

Алгоритмом расчета цепи постоянного тока

ШАГ 1. Произвольно выбрать положительные направления токов и обозначить их на схеме.

ШАГ 2. Составить уравнения по первому закону Кирхгофа (26), причем на одно уравнение меньше числа узлов (т.к. для последнего узла уравнение будет линейно зависимым от предыдущих уравнений).

ШАГ 3. Выбрать независимые (главные) контуры и направление их обхода. Удобно для всех контуров выбрать одинаковое направление обхода.

ШАГ 4. Записать уравнения по второму закону Кирхгофа (27) для выбранных контуров.

ШАГ 5. Решить полученную систему уравнений – определяют искомые токи.

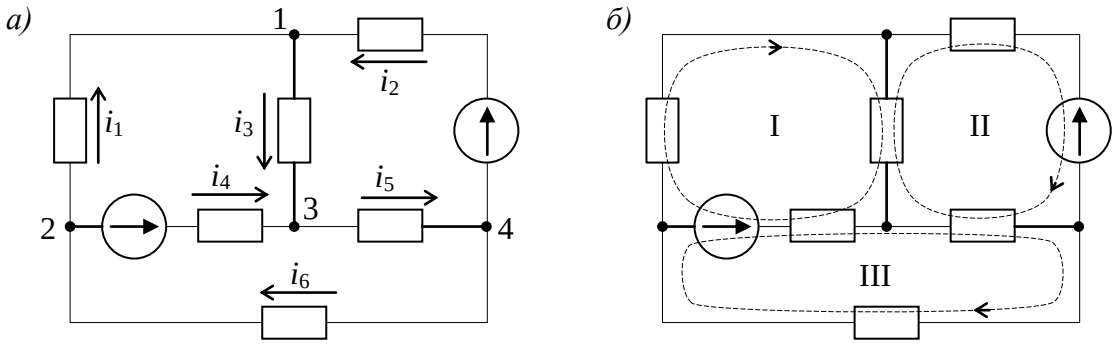


Рис. 43 – а) условные направления токов в ветвях схемы; б) направления обхода контуров

Решение:

1) В цепи имеется 4 узла, обозначенных на *Рис. 43, а* цифрами 1-4, значит, потребуется составить 3 уравнения по первому закону Кирхгофа. Стрелками обозначены произвольно выбранные направления протекания токов $i_1 - i_6$.

2) Составим 3 уравнения токов в узлах по первому закону Кирхгофа (26):

узел 1: $i_1 + i_2 - i_3 = 0$ (токи i_1 и i_2 втекают в узел 1, а i_3 вытекает из узла 1);

узел 2: $-i_1 - i_4 + i_6 = 0$ (ток i_6 втекает в узел 2, а i_1 и i_4 вытекает из узла 2);

узел 3: $i_3 + i_4 - i_5 = 0$ (токи i_4 и i_3 втекают в узел 3, а i_5 вытекает из узла 3).

3) На *Рис. 43, б* предложены выбранные произвольно независимые контуры I – III и направление их обхода.

4) Для выбранных контуров запишем уравнения по второму закону Кирхгофа (27):

контур I: $u_1 + u_3 + u_4 = -E1$ (включен навстречу направлению обхода контура I);

контур II: $u_2 + u_3 + u_5 = -E2$ (включен навстречу направлению обхода контура II);

контур III: $u_4 + u_5 + u_6 = E1$ (включен по направлению обхода контура III).

5) В полученных уравнениях перейдем от напряжений к токам, пользуясь законом Ома (28) и учитывая направления токов (если ток направлен против обхода контура, он отрицательный)

контур I: $i_1 \cdot R1 + i_3 \cdot R3 - i_4 \cdot R4 = -E1$ (ток i_4 направлен против обхода контура I);

контур II: $-i_2 \cdot R2 - i_3 \cdot R3 - i_5 \cdot R5 = -E2$ (токи i_2 , i_3 и i_5 – против обхода контура II);

контур III: $i_4 \cdot R4 + i_5 \cdot R5 + i_6 \cdot R6 = E1$ (все токи по направлению обхода контура III).

6) Из первых трех и последних трех уравнений составим матрицы СЛАУ и разрешим её любым известным способом относительно токов $i_1 - i_6$ и получим вектор $\boldsymbol{X} = (i_1,\ i_2,\ i_3,\ i_4,\ i_5,\ i_6)^{\mathrm{T}}$.

В программе (*Листинг 49*) Получены следующие значения токов: $i_1 = -1.672$ А, $i_2 = 0.525$ А, $i_3 = -1.146$ А, $i_4 = 1.686$ А, $i_5 = 0.539$ А, $i_6 = 0.014$ А. Выбирая случайным образом направления протекания токов в цепях схемы, мы угадали направление токов i_2, i_4, i_5 и i_6, а токи i_1 и i_3 имеют отрицательное значение, следовательно в реальной схеме они направлены в противоположную сторону.

Листинг 49. Решение полученной системы уравнений

$$ORIGIN := 1 \qquad R := (35\ \ 80\ \ 20\ \ 100\ \ 150\ \ 40)^{T} \qquad E := (250\ \ 100)^{T}$$

$$A := \begin{pmatrix} 1 & 1 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 1 & -1 & 0 \\ R_1 & 0 & R_3 & -R_4 & 0 & 0 \\ 0 & -R_2 & -R_3 & 0 & -R_5 & 0 \\ 0 & 0 & 0 & R_4 & R_5 & R_6 \end{pmatrix} \qquad B := \begin{pmatrix} 0 \\ 0 \\ 0 \\ -E_1 \\ -E_2 \\ E_1 \end{pmatrix}$$

$$|A| = 4.534 \times 10^{6} \qquad \mathrm{rank}(A) = 6$$

$$X := \mathrm{rref}(\mathrm{augment}(A, B))^{\langle 7 \rangle} \qquad X = \begin{pmatrix} -1.67181 \\ 0.52547 \\ -1.14634 \\ 1.6856 \\ 0.53926 \\ 0.01378 \end{pmatrix}$$

7) Проверить полученные правильность вычисления результатов можно подставив значения найденных токов в уравнения, построенные по 1 закону Кирхгофа.

8) Зная сопротивления элементов цепи [R] и токи [A], через них протекающие легко построить значения падений напряжений [B] на каждом из резисторов по закону Ома. После этого можно произвести проверку по уравнениям, построенным по 2 закону Кирхгофа.

Листинг 50. Расчет падения напряжений в цепи постоянного тока

$$k := 1..6 \qquad U_k := X_k \cdot R_k \qquad U^{T} = (-58.51345\ \ 42.03794\ \ -22.92678\ \ 168.55977\ \ 80.88884\ \ 0.55139)\ \ [\mathrm{B}]$$

9) Проверить полученные результаты можно опираясь на закон сохранения энергии. То есть, вся суммарная мощность, выработана источниками ЭДС нашей схемы, должна быть потреблена нагрузкой.

Из школьного курса физики известно, что мощность источника ЭДС может быть вычислена как произведение напряжения на ток: $P_{ист} = E \cdot i$ [Вт]. Таким образом, все источники схемы вырабатывают суммарную мощность, равную

$$P_{\Sigma ист} = \sum_{k=1}^{n} E_k \cdot i_k \ \ [\mathrm{Вт}]. \qquad (29)$$

С другой стороны, мощность, потребляемая нагрузкой может быть подсчитана $P_{потр} = R \cdot i^2$ [Вт], отсюда можно вычислить суммарную мощность, рассеянную на сопротивлениях R1-R6:

$$P_{\Sigma потр} = \sum_{k=1}^{m} R_k \cdot i_k^2 \ \ [\mathrm{Вт}]. \qquad (30)$$

Пользуясь формулами (29) и (30), вычислим сгенерированную и потребленную мощность в исследуемой схеме.

Листинг 51. Баланс мощностей в цепи постоянного тока

$Pist := E_1 \cdot X_4 + E_2 \cdot X_2$ $Pist = 473.94685$ [Вт] $Ppotr := \sum_{k=1}^{6} \left[R_k \cdot (X_k)^2 \right]$ $Ppotr = 473.94685$ [Вт]

Можем обоснованно предполагать, что токи рассчитаны верно, поскольку *баланс мощностей* в исследуемой цепи постоянного тока сошелся – потребленная элементами цепи мощность равняется мощности произведенной источниками.

Контрольные вопросы

1. Сформулируйте первый закон Кирхгофа.
2. Как связано напряжение, сопротивление и ток в резисторе?
3. Сформулируйте закон сохранения энергии в приложении к балансу мощностей.
4. Мощность. По каким двум формулам можно вычислить мощность?
5. Сформулируйте закон Ома.
6. Могут ли токи иметь отрицательное значение? Почему?
7. Назовите единицы измерения тока, напряжения, сопротивления и мощности.
8. Сформулируйте второй закон Кирхгофа.
9. Расскажите алгоритм расчета токов в цепи постоянного тока.

6. КОМПЛЕКСНЫЕ ВЫЧИСЛЕНИЯ

6.1. Теория чисел. Краткий обзор

Развитие цивилизации требует развития абстрактных научных взглядов на объекты природы, в первую очередь это относится к математике. Развитие теории чисел определяется теми *операциями*, которые необходимо производить с числами.

На заре человечества требовалось лишь операция «*счет*» для которого было достаточно *натуральных чисел* $\{1,\ 2,\ 3,\ \ldots\}$. Натуральные числа можно было только складывать, однако, появление операции «*вычитание*» определило множество *целых чисел* $\{0,\ 1,\ -1,\ 2,\ -2,\ \ldots\}$. Для целых чисел применимы операции «умножения», «деления нацело» и «остаток от деления». Появление операции «*дробь*» или «*деление*» обусловило появление множества *рациональных чисел* $\{a/b,\ где\ a,\ b - целые\ числа\}$. Рациональные числа можно умножать, делить и возводить в степень, но извлекать корень – нельзя. Операция «*извлечение корня из положительного числа*» определила множество *иррациональных* (или *действительных*) *чисел* $\{-\infty,\ +\infty\}$, изображаемых обычно в виде числовой прямой.

$$\mathbb{N} \longrightarrow \mathbb{Z} \longrightarrow \mathbb{Q} \longrightarrow \mathbb{R} \longrightarrow \mathbb{C}$$

Ясно, что каждое последующее числовое множество включает в себя все предыдущие. Но как же быть с корнем из отрицательного числа? Придется построить очередное расширение множества вещественных чисел, но в математике существует строгая теорема, доказывающая, что никаких других чисел кроме действительных на числовой прямой нет.

Подойдем с другой стороны. Понятно, что $\sqrt{-16} = \sqrt{16\cdot(-1)} = \sqrt{16}\cdot\sqrt{-1} = 4\sqrt{-1}$ и понятно, что любой корень из отрицательного числа может быть представлен в виде произведения некоторого действительного числа на $\sqrt{-1}$. Таким образом, корни из положительных и корни из отрицательных чисел могут быть получены друг из друга умножением на $\sqrt{-1}$, но при этом не смешиваются. Ясно, что определив $\sqrt{-1}$ мы сможем расширить множество чисел до множества *комплексных чисел*.

6.2. Комплексные числа

Комплексным числом называется число вида $z = x + i\cdot y$, где $x = \mathrm{Re}(z)$ и $y = \mathrm{Im}(z)$ – вещественные числа, а i – *мнимая единица*, определяемая равенством $i^2 = -1$ или $i = \sqrt{-1}$. Геометрическая интерпретация комплексного числа очевидна – каждому комплексному числу z соответствует точка на *комплексной плоскости* с координатами (x, y). Ось абсцисс называется *действительной осью*, ось ординат – *мнимой*.

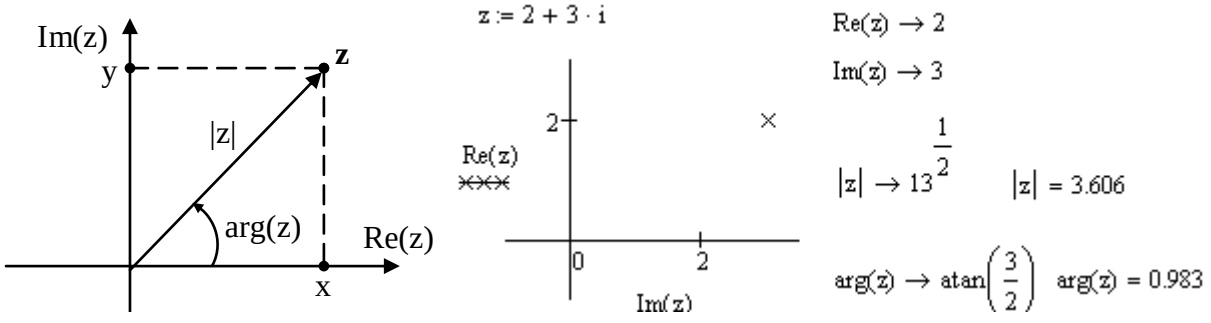


Рис. 44 – Представление комплексного числа в MathCAD

При вводе комплексных чисел в программе на *MathCAD* для ввода комплексной единицы необходимо всегда набирать **1i** или **1j**, в противном случае *MathCAD* истолкует i или j как переменную. Когда курсор покидает выражение, содержащее **1i** или **1j**, *MathCAD* скрывает невидимую 1 и оставляет только **i**.

Понятно, что любую точку на плоскости можно записать в *декартовых* (x, y) координатах или в *полярных* (ρ, φ). Расстояние ρ от точки z до начала координат называется *модулем числа* z и обозначается $|z|$. Угол φ между положительным направлением действительной оси и радиус-вектором точки z называется *аргументом числа* z и обозначается arg(z). По определению,

$$\rho = |z| = \sqrt{x^2 + y^2}, \quad tg(\varphi) = y/x \tag{31}$$

Если радиус-вектор числа z совершит полное число оборотов вокруг начала координат, то он совпадает с первоначальным положением, а угол φ увеличится на число кратное 2π. Совокупность всех этих углов называется *полным значением аргумента* и обозначается

$$Arg(z) = \varphi + 2k\pi, \ k\text{- целое}. \tag{32}$$

Из множества значений аргумента особо выделяется *главное значение аргумента* arg(z), удовлетворяющее неравенству $-\pi < \arg(z) \le \pi$, при этом:

$$\arg(z) = \begin{cases} arctg(y/x), & x > 0; \\ arctg(y/x) + \pi, & x < 0, y \ge 0; \\ arctg(y/x) - \pi, & x < 0, y < 0. \end{cases} \tag{33}$$

6.3. Тригонометрическая и показательная форма комплексного числа

Кроме алгебраической формы $z = x + iy$ часто используют *тригонометрическую форму* записи того же самого комплексного числа (здесь $\varphi = \arg(z)$):

$$z = |z| \cdot (\cos(\varphi) + i \cdot \sin(\varphi)). \tag{34}$$

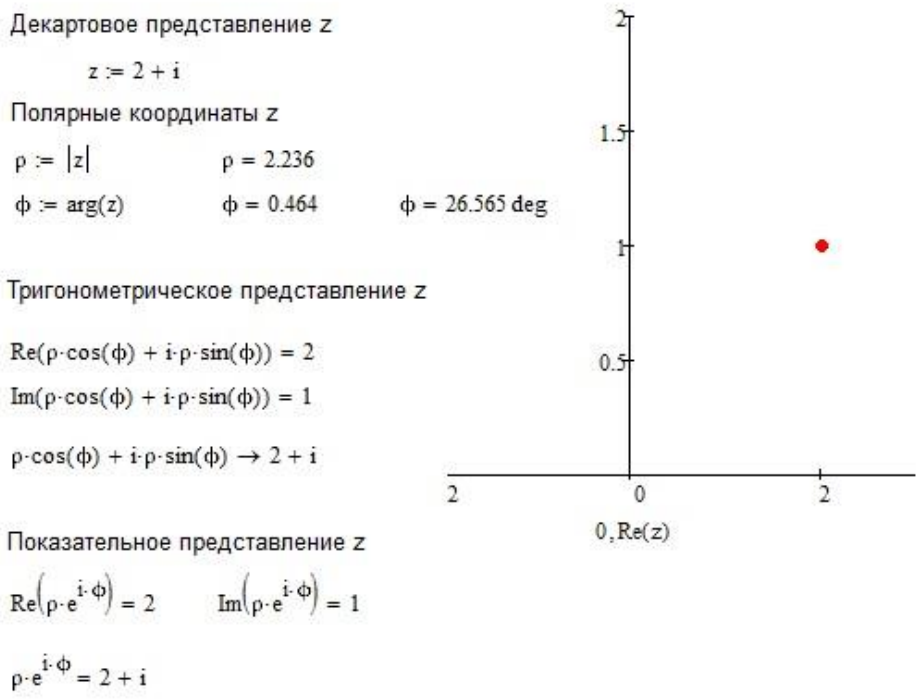


Рис. 45 – Тригонометрическая и показательная форма комплексного числа

Пользуясь формулой Эйлера $e^{i\varphi}=\cos(\varphi)+i\sin(\varphi)$, можно получить *показательную форму* любого комплексного числа, кроме $z=0+i\cdot 0$:

$$z=|z|e^{i\varphi}. \tag{35}$$

Нужно отдавать себе отчет, что классическое *декартовое представление* комплексного числа $z=x+i\cdot y$, *тригонометрическая форма* (34) и *показательная форма* (35) определяют ***одно и то же комплексное число***!

Рис. 45 иллюстрирует возможность перехода между этими тремя формами записи одного и того же числа.

6.4. Операции над комплексными числами

Сложение и вычитание комплексных чисел $z_1=x_1+iy_1$ и $z_2=x_2+iy_2$ осуществляются по формулам $z_1\pm z_2=(x_1\pm x_2)+i(y_1\pm y_2)$. Умножение, деление и возведение в степень удобнее производить в показательной форме. Пусть $z_1=|z_1|e^{i\varphi_1}$ и $z_2=|z_2|e^{i\varphi_2}$, тогда произведение и частное:

$$z_1z_2=|z_1|\cdot|z_2|e^{i(\varphi_1+\varphi_2)},\qquad z_1/z_2=|z_1|/|z_2|\cdot e^{i(\varphi_1-\varphi_2)}. \tag{36}$$

$$(x_1+i\cdot y_1)\cdot(x_2+i\cdot y_2)=(x_1x_2-y_1y_2)+i\cdot(y_2x_1+y_1x_2)$$

$$\frac{x_1+i\cdot y_1}{x_2+i\cdot y_2}=\frac{x_1x_2+y_1y_2}{x_2^2+y_2^2}+i\frac{x_2y_1-x_1y_2}{x_2^2+y_2^2}$$

С помощью формулы Муавра $(\cos(\varphi)+i\sin(\varphi))^n=\cos(n\varphi)+i\sin(n\varphi)$ можно получить формулу для степени числа:

$$z^n=\left(|z|e^{i\varphi}\right)^n=|z|^n e^{in\varphi}=|z|^n\left(\cos(n\varphi)+i\sin(n\varphi)\right). \tag{37}$$

Корень n-ной степени из комплексного числа имеет n различных значений ($k=0,1,\ldots,n-1$):

$$w_k=\sqrt[n]{|z|\left(\cos(\varphi)+i\sin(\varphi)\right)}=\sqrt[n]{|z|}\left(\cos\frac{\varphi+2k\pi}{n}+i\sin\frac{\varphi+2k\pi}{n}\right). \tag{38}$$

Геометрически эти n значений корня изображаются вершинами правильного n-угольника с полярными координатами $\left(\sqrt[n]{|z|}\quad\frac{\arg(z)+2k\pi}{n}\right)$, (см. *Рис. 48*).

$z1:=2+3\cdot i$ $\quad z2:=3+2\cdot i$

$z1+z2\rightarrow 5+5\cdot i$	$\arg(z1+z2)=0.785$	$\lvert z1+z2\rvert=7.071$
$z1-z2\rightarrow -1+i$	$\arg(z1-z2)=2.356$	$\lvert z1-z2\rvert=1.414$
$z1\cdot z2\rightarrow 13\cdot i$	$\arg(z1\cdot z2)=1.571$	$\lvert z1\cdot z2\rvert=13$
$\frac{z1}{z2}\rightarrow\frac{12}{13}+\frac{5}{13}\cdot i$	$\arg\left(\frac{z1}{z2}\right)=0.395$	$\left\lvert\frac{z1}{z2}\right\rvert=1$

Рис. 46 – Операции над комплексными числами

6.5. Понятие комплексных функций

С точки зрения *MathCAD* нет разницы между вычислением вещественнозначной или комплексной функции. Трудности состоят в том, что обычный человек не может визуально представить себе функцию размерности больше 3, а комплексные функции отображают точку из двумерной комплексной плоскости в двумерную комплексную

плоскость, т.е. четырехмерное изображение. Поэтому при работе с комплексными функциями пользуются аналитическими или табличными представлениями функций, но не графиками.

На рисунке (*Рис. 47*) схематично изображено множество *B*, называемое *образом множества A при воздействии на все его точки функцией* $\psi(z) \to w,\ z \in A,\ B=\psi(A)$. Множество *A* называется *прообразом* множества *B*. От функции $\psi(z)$.

По виду результирующего множества $\psi(A) = B$, (*образа множества A*) можно догадываться о поведении функции ψ и о ее свойствах.

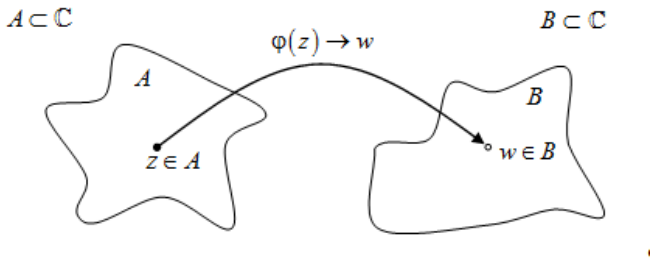


Рис. 47 – Представление комплексных функций

Комплексная величина $w = \psi(z) = u(z) + i \cdot v(z)$ называется *функцией комплексной переменной* $z = x + i \cdot y$, если каждому значению z из комплексной плоскости соответствует одно или несколько значений *w*. Здесь $u(z) = u(x, y)$ – *вещественная* и $v(z) = v(x, y)$ – *мнимая* части функции $w = \psi(z)$.

Многозначные функции

При использовании в комплексной области многие функции, о которых мы привыкли думать как о возвращающих одно значение, становятся *многозначными*, как, например на *Рис. 48* для функции «корень *n*-ной степени из комплексного числа».

Общее правило состоит в том, что для многозначной функции *MathCAD* всегда возвращает значение, составляющее на комплексной плоскости самый маленький положительный угол с положительным направлением действительной оси. Оно называется *главным значением аргумента*.

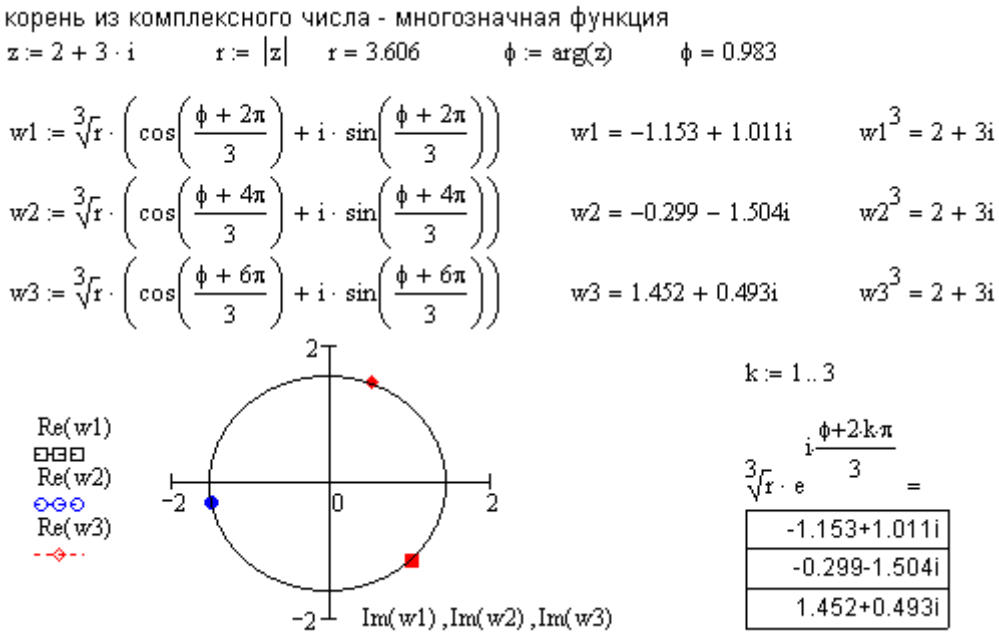


Рис. 48 – Пример многозначной функции $\sqrt[3]{2+3\mathbf{i}}$

Например, если требуется вычислить $\sqrt[3]{-1}$, *MathCAD* вернёт -1, хотя кубическим корнем из -1, кроме того, являются числа 0.5+0.866**i** и 0.5-0.866**i.**

Корни полинома n-ной степени.

Методика отыскания корней *n*-ной степени из комплексного числа легко расширяется на поиск корней полинома n-ной степени. Если в пространстве действительных чисел гарантировано можно найти корни полинома 3-ей степени (кубической параболы), поиск корней полинома 4-ой степени уже аналитически сложная задача, а корни параболы 5-ой и более старших степеней аналитически найти практически невозможно, то в пространстве комплексных чисел эта задача решается легко и красиво - у полинома *n*-ной степени всегда ровно *n* корней. Этот материал будет рассмотрен в курсе математики.

Рассмотрим свойства некоторых элементарных функций комплексной переменной.

Линейная функция

$w(z) = a \cdot z + b$, где a и b – постоянные комплексные числа. Это отображение преобразует прямые в прямые (углы между прямыми сохраняются) и окружности в окружности.

Дробно-линейная функция

$w(z) = \dfrac{a \cdot z + b}{c \cdot z + d}$, где a, b, c и d – постоянные комплексные числа. Это отображение преобразует окружность или прямую в окружность или прямую. Внутренняя область отображаемой окружности переходит во внутреннюю или во внешнюю область образа. Частный случай этой функции (a=d=0, b=c=1) – функция $w(z) = 1/z$, называется *инверсией*.

Экспонента

Экспоненциальная функция $e^z = e^{x+iy} = e^x\left(\cos(y) - i \cdot \sin(y)\right)$ обладает следующими свойствами: $\left|e^z\right| = e^x$, $\arg(e^z) = y$. Линии $x = const$ преобразуются в окружности $|w| = const$, а линии $y = const$ – в лучи $\arg(w) = const$.

Тригонометрические и гиперболические функции

$$\cos(z) = \frac{1}{2}\left(e^{i\cdot z} + e^{-i\cdot z}\right),\ \sin(z) = \frac{1}{2i}\left(e^{i\cdot z} - e^{-i\cdot z}\right),\ \mathrm{tg}(z) = \frac{\sin(z)}{\cos(z)},\ \mathrm{ctg}(z) = \frac{\cos(z)}{\sin(z)},$$

$$\mathrm{ch}(z) = \frac{1}{2}\left(e^{z} + e^{-z}\right),\ \mathrm{sh}(z) = \frac{1}{2}\left(e^{z} - e^{-z}\right),\ \mathrm{th}(z) = \frac{\mathrm{sh}(z)}{\mathrm{ch}(z)},\ \mathrm{cth}(z) = \frac{\mathrm{ch}(z)}{\mathrm{sh}(z)}.$$

Логарифмическая функция

$\mathrm{Ln}(z) = \ln|z| + i \cdot (\arg(z) + 2k\pi),\ k \in \square$ определяется как обратная к экспоненциальной, бесконечнозначная функция. Дополнительно принимается, что $\mathrm{Ln}(0) = \infty$, и $\mathrm{Ln}(\infty) = \infty$.

Значение логарифма при k=0 называется главным значением логарифма и обозначается $\ln(z) = \ln|z| + i \cdot \arg(z)$. Поэтому $\mathrm{Ln}(z) = \ln(z) + 2k\pi i.$.

Обратные гиперболические и тригонометрические функции

$$\mathrm{Arcsin}(z)=\frac{1}{i}\mathrm{Ln}\left(iz+\sqrt{1-z^2}\right),\ \mathrm{Arccos}(z)=\frac{1}{i}\mathrm{Ln}\left(z+\sqrt{z^2-1}\right),\ \mathrm{Arctg}(z)=\frac{1}{2i}\mathrm{Ln}\frac{1+iz}{1-iz},$$

$$\mathrm{Arcctg}(z)=\frac{1}{2i}\mathrm{Ln}\frac{z+i}{z-i},\ \mathrm{Arsh}(z)=\mathrm{Ln}\left(z+\sqrt{z^2+1}\right),\ \mathrm{Arch}(z)=\mathrm{Ln}\left(z+\sqrt{z^2-1}\right),$$

$$\mathrm{Arth}(z)=\frac{1}{2}\mathrm{Ln}\frac{1+z}{1-z},\ \mathrm{Arcth}(z)=\frac{1}{2}\mathrm{Ln}\frac{z+1}{z-1}.$$

Степенная и показательная функции

Если α и β – два комплексных числа, $\alpha\neq 0$, то степенная функция $w(\alpha,\beta)=\alpha^{\beta}$ в силу основного логарифмического тождества $\alpha^{\beta}=e^{\beta\cdot\mathrm{Ln}(\alpha)}=e^{\beta\cdot(\ln\alpha+2k\pi i)}$.

Когда β – целое вещественное число, то степень имеет одно значение, т.к. $e^{2\beta k\pi i}=1$. Если β – несократимая рациональная дробь p/q, $q>1$, то степень имеет ровно q различных значений. Во всех других случаях степень имеет бесконечное множество значений.

6.6. Производная функции комплексного переменного

Если для функции $w=f(z)=u(x,y)+i\cdot v(x,y)$ существует конечный предел. $\lim\limits_{z\to z_0}\dfrac{f(z)-f(z_0)}{z-z_0}$, то он называется производной от функции $f(z)$ в точке z_0.

Необходимым и достаточным условием дифференцируемости функции $w=f(z)=u(x,y)+i\cdot v(x,y)$ являются *условия Коши-Римана*:

$$\frac{\partial u}{\partial x}=\frac{\partial v}{\partial y}\ \text{и}\ \frac{\partial u}{\partial y}=-\frac{\partial v}{\partial x}. \qquad (39)$$

Если функция задана через свои вещественную и мнимую части $u(x,y)$ и $v(x,y)$, то после проверки условий (39) производную $w'(z)$ можно найти по одной из равносильных формул:

$$w'=\frac{\partial u}{\partial x}+i\frac{\partial v}{\partial x},\quad w'=\frac{\partial u}{\partial x}-i\frac{\partial u}{\partial y},\quad w'=\frac{\partial v}{\partial y}+i\frac{\partial v}{\partial x}\quad\text{и}\quad w'=\frac{\partial v}{\partial y}-i\frac{\partial u}{\partial y}. \qquad (40)$$

Если же дана зависимость $f(z)$, то после выполнения проверки условий Коши-Римана (39) производную $w(z)=df(z)/dz$ можно найти непосредственным дифференцированием по таблице производных (такой же, как и для функций вещественного аргумента).

Функция $f(z)$ называется *аналитической* (или *голоморфной*) в точке z_0, если она дифференцируема в каждой точке некоторой окрестности z_0.

Восстановление аналитической функции по ее действительной или мнимой части

Кроме того, доказано, что вещественная и мнимая части аналитической функции удовлетворяют *уравнению Лапласа*:

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0,\ \text{и}\ \frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}=0. \qquad (41)$$

Рассмотрим обратную задачу: Дана действительная часть $u(z)=u(x,y)$ некоторой неизвестной функции комплексной переменной $w=f(z)$. Требуется найти мнимую часть $v(z)=v(x,y)$ этой функции. Найти тем самым саму функцию $f(z)$, используя некоторое *начальное условие*. Или наоборот – восстановить неизвестную $u(x,y)$ по известной $v(x,y)$.

Эта задача решается исходя из (39) –(41).

Контрольные вопросы

1. Как вычислить корень из отрицательного числа?
2. Как вы понимаете комплексное число? Как его представить / изобразить?
3. Как представляется комплексное число в декартовой и полярной системе координат?
4. Умеете ил вы строить модуль и аргумент комплексного числа?
5. Как извлечь корень из комплексного числа?
6. Сколько значений имеет корень пятой степени из 1? Нарисуйте их.
7. Как изучать комплексные функции?
8. Умеете ли вы записывать комплексное число $4+3i$ в тригонометрической форме?
9. Как вычислить вещественную и мнимую части комплексного числа аналитически и численно? Возможно ли это без MathCAD?
10. Как записать комплексное число $4+3i$ в показательной форме?
11. Какие арифметические операции с элементами комплексного пространства можно производить?
12. Что такое многозначная функция? Приведите пример.
13. Как выглядит пространство комплексных чисел?

Как удобнее всего возвести комплексное число в степень?

7. РАСЧЕТ ЭЛЕКТРИЧЕСКИХ ЦЕПЕЙ ПЕРЕМЕННОГО ТОКА

7.1. Представление гармонической функции комплексным числом

Более детально данный материал будет изучаться в курсе теоретических основ электротехники. В электротехнических системах, как правило, ток имеет синусоидальную (гармоническую) форму:

$$i(t) = I \cdot \sin(\omega \cdot t + \varphi) \text{ [A]}, \qquad (42)$$

где $\boldsymbol{I}$ – амплитуда [A], $\omega = 2\pi/T$ – угловая частота [рад/с], φ – начальная фаза [рад], T – период колебаний [с]. График функции $i(t)$ приведен на *Рис. 49*.

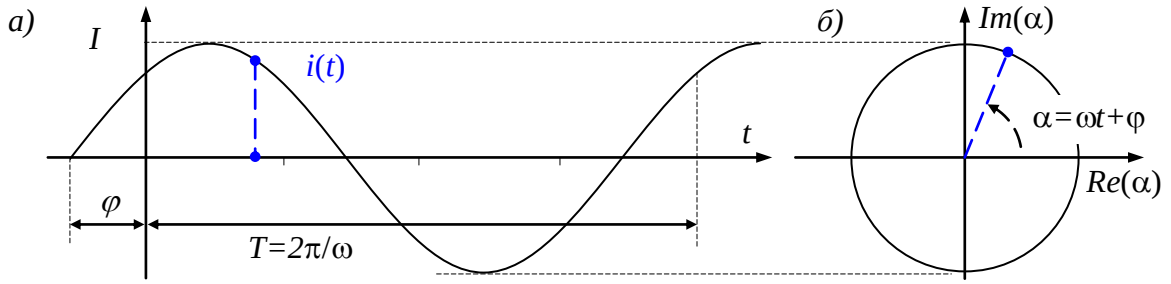


Рис. 49 – Представление (а) гармонического сигнала в виде (б) комплексной величины

Любая *гармоническая функция* полностью определяется тремя величинами: амплитудой, частотой и фазой.

Рассмотрим комплексную функцию $I \cdot e^{\mathbf{i}\cdot\alpha(t)}$, где $\alpha(t)$ – угол α (*Рис. 49,б*), линейно зависящий от времени t и начальной фазы φ: $\alpha(t) = \omega \cdot t + \varphi$. Выражение $\alpha(t) = \omega \cdot t + \varphi$ часто называется *мгновенной (полной) фазой* гармонического сигнала.

По формуле Эйлера это выражение может быть представлено в виде:

$$I \cdot e^{\mathbf{i}\cdot(\omega t+\varphi)} = I \cdot \cos(\omega \cdot t + \varphi) + \mathbf{i} \cdot I \cdot \sin(\omega \cdot t + \varphi), \qquad (43)$$

Мнимая часть выражения (43) представляет собой выражение (42), так что можно записать его в виде

$$i(t) = \mathrm{Im}\left(I \cdot e^{\mathbf{i}\cdot(\omega t+\varphi)}\right) = I \cdot \sin(\omega \cdot t + \varphi) \text{ [A]}, \qquad (44)$$

здесь функция Im(z) – взятие мнимой части комплексного числа. Выражение (44) фактически представляет собой проекцию вращающегося вектора $I \cdot e^{\mathbf{i}\cdot\alpha(t)}$ на мнимую ось (ординат), таким образом можно представить любой гармонический сигнал.

7.2. Элементы цепи переменного тока

Рассмотрим следующие элементы цепи переменного тока – катушка индуктивности, конденсатор и резистор. На данном этапе обучения будем рассматривать их в виде идеальных (абстрактных) моделей.

С точки зрения оказываемого переменному току *сопротивления*, элементы цепи разделяются на *активные* (резистор) и *реактивные* элементы (катушка индуктивности и конденсатор). Соответственно, сопротивление резистора называется *активным сопротивлением* – энергия на нем выделяется только в виде теплоты, а в реактивных элементах энергия в виде тепла не выделяется, но периодически запасается в электрическом или магнитном полях, сопротивление реактивных элементов называется *реактивным сопротивлением*.

Резистор (активное сопротивление)

При протекании по резистору R синусоидального тока вида (42), напряжение на резисторе (по закону Ома) равно

$$u(t) = R \cdot i(t) = R \cdot I \cdot \sin(\omega \cdot t + \varphi) \text{ [В]}. \quad (45)$$

Ток $i(t)$ и напряжение $u(t)$ имею одну и ту же начальную фазу, при этом говорят что они *совпадают по фазе* (*Рис. 50*). При протекании тока через активное сопротивление происходит потребление энергии из источника и выделение ее в виде тепла.

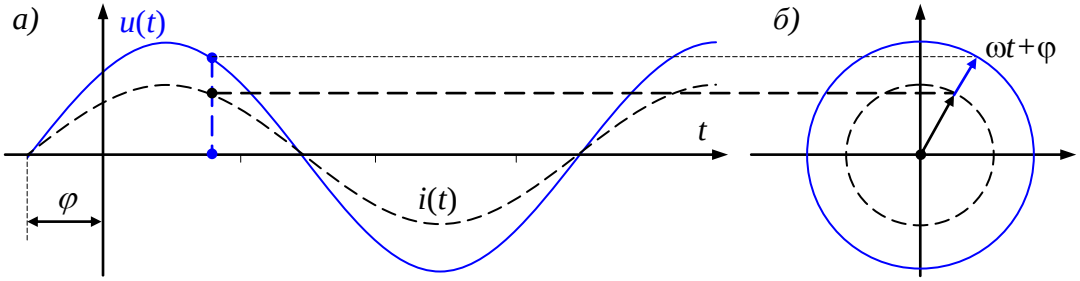


Рис. 50 – Ток и напряжение на резисторе

Катушка индуктивности

Для идеальной катушки индуктивности предполагается, что её активное сопротивление равно нулю и катушка обладает только индуктивностью L. При протекании по катушке индуктивности переменного тока вида (42), в ней наводится э.д.с. самоиндукции, равная

$$e_L(t) = -L\frac{di_L}{dt} = -\omega \cdot L \cdot I \cdot \cos(\omega \cdot t + \varphi) = X_L \cdot I \cdot \sin\left(\omega \cdot t + \varphi - \frac{\pi}{2}\right) \text{ [В]}.$$

Здесь вводится величина X_L [Ом] называемая реактивным сопротивлением катушки индуктивности, или кратко – *индуктивным сопротивлением*.

$$X_L = \omega \cdot L \text{ [Ом]} \quad (46)$$

Для того, чтобы через катушку индуктивности протекал переменный ток необходимо, чтобы на ее выводах было напряжение, равное по величине и противоположное по знаку наведенной э.д.с.:

$$u_L(t) = U \cdot \sin(\omega \cdot t + \varphi - 90°) \text{ [В]}, \qquad U = X_L \cdot I \text{ [В]}. \quad (47)$$

Напряжение на индуктивности опережает ток по фазе на 90°, что иллюстрирует векторная диаграмма (*Рис. 51,б*).

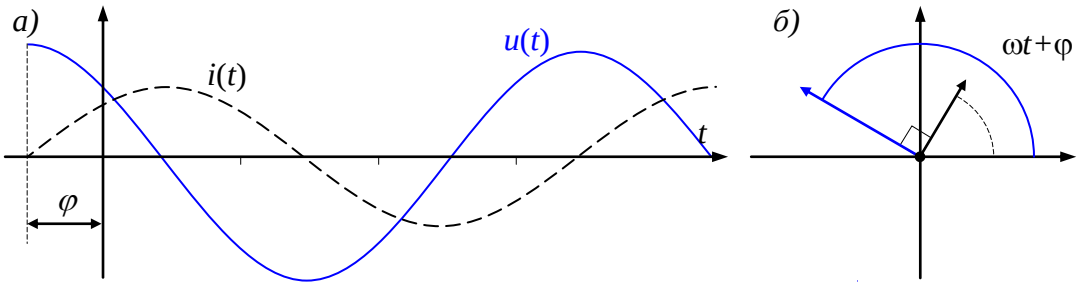


Рис. 51 – Ток и напряжение в катушке индуктивности

Конденсатор

Если к конденсатору С приложено напряжение $u_C(t) = U \cdot \sin(\omega \cdot t + \varphi)$, то конденсатор будет периодически перезаряжаться, что вызовет протекание тока через конденсатор:

$$i_C(t) = C\frac{du_C}{dt} = \omega \cdot C \cdot U \cdot \cos(\omega \cdot t + \varphi) = \frac{U}{X_C} \cdot \sin\left(\omega \cdot t + \varphi + \frac{\pi}{2}\right) \text{ [А]}. \quad (48)$$

Здесь вводится величина X_C [Ом] называемая реактивным сопротивлением конденсатора, или кратко – *емкостным сопротивлением*.

$$X_C = \frac{1}{\omega \cdot C} \text{ [Ом]} \qquad (49)$$

Обозначив $I = U / X_C$ [А], получим:

$$i_C(t) = I \cdot \sin(\omega \cdot t + \varphi + 90°) \text{ [А]}.$$

Ток, протекающий через конденсатор, опережает напряжение по фазе на 90°, что иллюстрирует векторная диаграмма (*Рис. 52,б*).

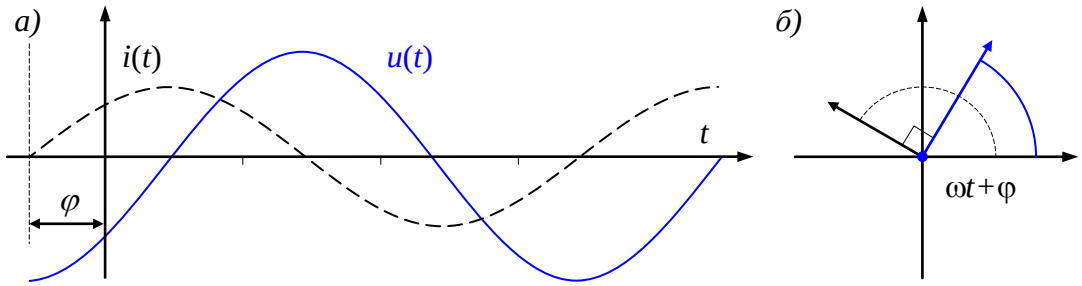


Рис. 52 – Ток и напряжение на конденсаторе

За первую четверть периода конденсатор потребляет от источника энергию, которая идет на создание электрического поля в нем. Во вторую четверть периода напряжение на конденсаторе уменьшается от максимума до нуля – запасенная в электрическом поле энергия отдается источнику, затем процесс повторяется.

7.3. Полное комплексное сопротивление участка цепи

Комплексное сопротивление Z для участка цепи переменного тока (с частотой ω), обладающего активным сопротивлением *R*, индуктивностью *L* и емкостью *C* в общем случае может быть записано в виде:

$$Z = Z_m \cdot e^{\mathbf{i}\cdot\omega} = R + \mathbf{i} \cdot \left(\omega \cdot L - \frac{1}{\omega \cdot C} \right) = R + \mathbf{i} \cdot \left(X_L - X_C \right) \text{ [Ом]}. \qquad (50)$$

Ясно, что для участков цепей, в которых отсутствует резистор, конденсатор или катушка индуктивности, в формуле (50) будет отсутствовать соответствующее слагаемое – активное, емкостное или индуктивное сопротивление.

7.4. Мощность цепи переменного тока

Закон сохранения энергии определяет *баланс мощностей* в цепи переменного тока. Полная мощность ***S*** источников ЭДС (допустим, что их *m* штук) вычисляется как сумма произведений напряжения источников E_k на протекающие через них токи i_k. Естественно, полная мощность ***S*** измеряется в *вольт-амперах* [ВА].

$$\boldsymbol{S} = \sum_{k=1}^{m} E_k \cdot i_k \text{ [ВА]}. \qquad (51)$$

Поскольку токи представляются комплексными величинами, то и мощность будет комплексной, а значит, ее можно представить как сумму действительной и мнимой частей (52):

$$\boldsymbol{S} = \boldsymbol{P} + \boldsymbol{Q} \cdot \mathbf{i} \text{ [ВА]}. \qquad (52)$$

Действительная часть $\boldsymbol{P} = \mathrm{Re}(\boldsymbol{S})$ [Вт] называется активной мощностью и измеряется в *ваттах* [Вт], а мнимая часть полной мощности $\boldsymbol{Q} = \mathrm{Im}(\boldsymbol{S})$ [ВАр] называется *реактивной мощностью* и измеряется в *варах* [ВАр] – «*вольт-амперах реактивной мощности*».

Активной потребляемой мощностью, рассеиваемой элементами схемы называется мощность, потраченная на активных элементах – резисторах. Вычисляется она, естественно, так же как и для цепей постоянного тока (30):

$$\boldsymbol{P}_{потр} = \sum_{k=1}^{n} R_k \cdot \left|i_k\right|^2 \ [\text{Вт}]. \qquad (53)$$

Реактивной потребляемой мощностью называется мощность, потраченная на реактивных элементах – конденсаторах и катушках индуктивности. Она не связана с выполнением полезной работы, а расходуется на создание электромагнитных полей в электродвигателях, трансформаторах, индукционных печах, сварочных трансформаторах, дросселях и пр. Вычисляется она, аналогично активной мощности (53), но вместо активных сопротивлений учитываются, разумеется, реактивные:

$$\boldsymbol{Q}_{потр} = \sum_{j=1}^{n1} X_{Lj} \cdot \left|i_j\right|^2 - \sum_{k=1}^{n2} X_{Ck} \cdot \left|i_k\right|^2 \ [\text{ВАр}]. \qquad (54)$$

Полная потребляемая мощность рассчитывается по формуле (52).

Таблица 2. Мощность цепи переменного тока

		Генерируемая мощность	Потребляемая мощность
Полная	***S* [ВА]**	***S=E·I*** напряжение, генерируемое источником умножить на ток, в нем протекающий	Сумма комплексных сопротивлений (***Z***) всех участков цепи, умноженная на модули соответствующих токов в квадрате
Активная	***P* [Вт]**	***P=Re(S)*** Действительная часть полной мощности источников	Сумма активных сопротивлений (***R***) всех резисторов, умноженная на модули соответствующих токов в квадрате
Реактивная	***Q* [ВАр]**	***Q=Im(S)*** Мнимая часть полной мощности источников	Сумма реактивных сопротивлений (***X_L, X_C***) всех емкостных и индуктивных элементов цепи, умноженная на модули соответствующих токов в квадрате

Баланс мощностей состоит в том, что произведенная активная мощность должна равняться сумме мощностей, потребляемых активными элементами цепи, а генерируемая реактивная мощность должна равняться сумме мощностей, расходуемых в реактивных элементах схемы.

7.5. Расчет электрической цепи синусоидального переменного тока

Расчет электрической цепи синусоидального переменного тока может быть сделан аналогично расчету токов цепи постоянного тока, однако, в качестве сопротивлений реактивных элементов нужно брать, разумеется, их комплексные сопротивления, а токи и напряжения представлять в комплексной форме, по аналогии с (44). Значения установившихся токов и напряжений цепи переменного тока можно рассчитать, основываясь на законах Кирхгофа и законе Ома, справедливых также и для комплексных сопротивлений в цепи переменного тока (50).

Рассмотрим этот метод на примере.

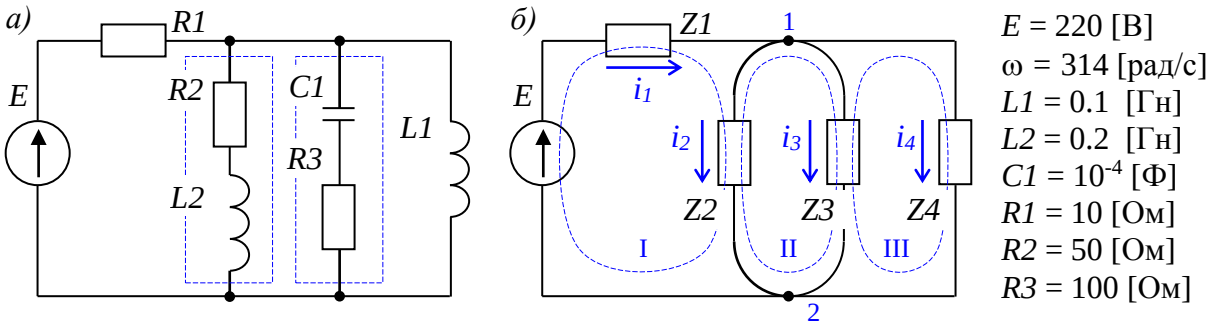


Рис. 53 – Расчет цепи переменного тока

1) Построим комплексные сопротивления Z участков цепи по определению (50). На *Рис. 53,а* пунктирными прямоугольниками отмечены участки электрической цепи на которых комплексное сопротивление имеет и активную (R) и реактивную (X) составляющие. $Z1 = R1$; $Z2 = R2 + \mathbf{i} \cdot X_{L2}$; $Z3 = R3 - \mathbf{i} \cdot X_{C1}$; $Z4 = \mathbf{i} \cdot X_{L1}$. Зададим параметры схемы и вычислим реактивные сопротивления X_{L1}, X_{L2} и X_{C1} по формулам (46), (49):

ORIGIN := 1 E1 := 220 [В] ω := 314 [рад/с] R1 := 10 [Ом] R2 := 100 [Ом]

L1 := 0.1 [Гн] L2 := 0.2 [Гн] C1 := 10^{-4} [Ф] R3 := 50 [Ом]

реактивные сопротивления элементов			комплексные сопротивления		
XL1 := ω·L1	XL1 = 31.4	[Ом]	Z1 := R1	Z1 = 10	[Ом]
XL2 := ω·L2	XL2 = 62.8	[Ом]	Z2 := R2 + i·XL2	Z2 = 100 + 62.8i	[Ом]
XC1 := $\frac{1}{\omega \cdot C1}$	XC1 = 31.847	[Ом]	Z3 := R3 – i·XC1	Z3 = 50 – 31.847i	[Ом]
			Z4 := i·XL1	Z4 = 31.4i	[Ом]

2) Построим уравнения (и матрицы) для нахождения токов по законам Кирхгофа:
(1): i_1-i_2-i_3-i_4=0; (I): U_{Z1}+U_{Z2}=E; (II): U_{Z3}-U_{Z2}=0; (III): U_{Z4}-U_{Z3}=0.

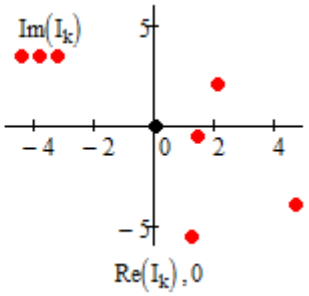
3) Выпишем матрицы и вычислим токи на участках рассматриваемой цепи переменного тока:

$$A := \begin{pmatrix} 1 & -1 & -1 & -1 \\ Z1 & Z2 & 0 & 0 \\ 0 & -Z2 & Z3 & 0 \\ 0 & 0 & -Z3 & Z4 \end{pmatrix} \quad B := \begin{pmatrix} 0 \\ E1 \\ 0 \\ 0 \end{pmatrix} \quad I := A^{-1} \cdot B \quad I = \begin{pmatrix} 4.75 - 3.878i \\ 1.412 - 0.499i \\ 2.103 + 2.115i \\ 1.235 - 5.494i \end{pmatrix} \text{ [A]}$$

Рассчитаем модули и аргументы токов в комплексном представлении

$k := 1..4$

$$|I_k| = \begin{pmatrix} 6.131 \\ 1.497 \\ 2.983 \\ 5.631 \end{pmatrix} \quad \arg(I_k) = \begin{pmatrix} -0.685 \\ -0.34 \\ 0.788 \\ -1.35 \end{pmatrix}$$



На графике справа красными точками обозначены 4 комплексных числа – искомые токи в комплексной интерпретации. Амплитуда переменных токов определяется как в формуле (44) их модулями $|I_k|$, а начальная фаза – аргументами $\arg(I_k)$.

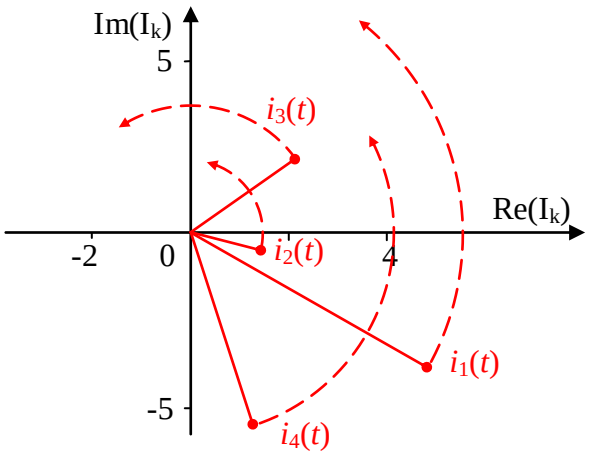


Рис. 54 – Токи в цепи переменного тока представленные в виде вращающихся комплексных векторов

Обратите внимание, что выражение (44) описывает *переменные токи*, поэтому представляет собой проекции *вращающихся векторов* $|I_k| \cdot e^{i\cdot(\omega\cdot t+\arg(I_k))}$ как функции от времени. Красные точки задают *начальную фазу* движения (в начальный момент времени $t_0 = 0$), в дальнейшем, при изменении времени t фаза вектора будет изменяться по закону $\omega \cdot t + \arg(I_k)$ с частотой ω, как показано на *Рис. 54*.

Таким образом можно представить любой гармонический сигнал. Отобразим переменные токи как функции от времени (*Рис. 55*):

$k := 1..4$ $\quad Ik(t,k) := |I_k| \cdot \sin(\omega \cdot t + \arg(I_k))$

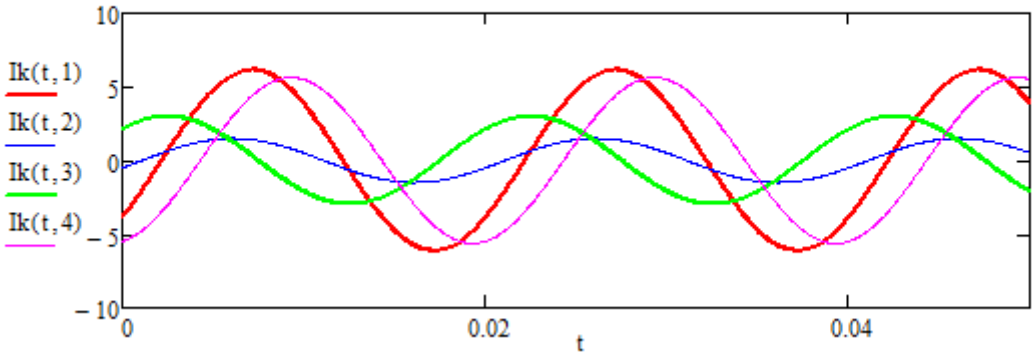


Рис. 55 – Токи в цепи переменного тока

4) Рассчитаем и представим на графике напряжения на всех активных и реактивных элементах исследуемой цепи в виде векторной диаграммы комплексных величин (*Рис. 56,а*) и в виде функций от времени (*Рис. 56,б*).

$$U := \begin{pmatrix} E1 \\ R1\cdot I_1 \\ R2\cdot I_2 \\ i\cdot XL2\cdot I_2 \\ -i\cdot XC1\cdot I_3 \\ R3\cdot I_3 \\ i\cdot XL1\cdot I_4 \end{pmatrix} \text{ [B]} \qquad U = \begin{pmatrix} 220 \\ 47.496 - 38.776i \\ 141.177 - 49.883i \\ 31.327 + 88.659i \\ 67.356 - 66.973i \\ 105.148 + 105.749i \\ 172.504 + 38.776i \end{pmatrix}$$

$k := 1..7$

$Uk(t,k) := |U_k| \cdot \sin(\omega \cdot t + \arg(U_k))$ [B]

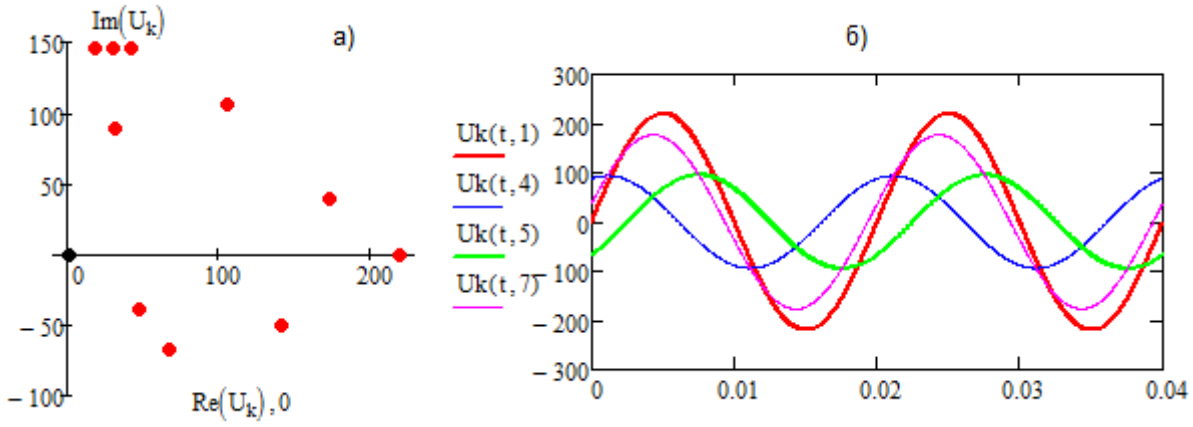


Рис. 56 – Напряжения в цепи переменного тока

5) Проверим правильность произведенных вычислений двумя способами. Во-первых, можно проверить выполнимость законов Кирхгофа для любого момента времени – сумма токов в узле должна равняться нулю, а сумма напряжений в любом контуре – сумме источников ЭДС. Для примера рассчитаем и построим на графике следующую функцию:

$$sumU(t) := Uk(t,2) + Uk(t,3) + Uk(t,4) - Uk(t,1)$$

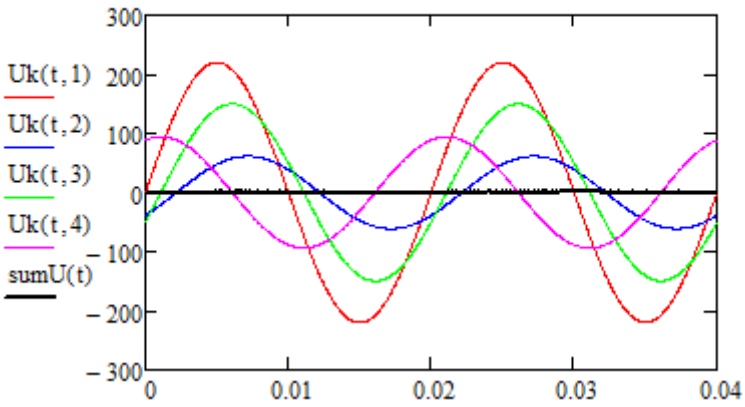


Рис. 57 – Сумма падений напряжения в 1 контуре в любой момент времени

Как видно из графика (*Рис. 57*), функция *sumU*(*t*), представляющая собой сумму четырех различных по амплитуде и фазе синусоид, в любой момент времени имеет нулевое значение.

Проверка законов Кирхгофа для исследуемой схемы *(а)* в комплексном представлении *(б)* и в мгновенных значениях *(в)* – в любой заданный момент времени *t*:

(а)	*(б)*	*(в)*
$U_{R2}+U_{L2}-U_{C1}-U_{R3}=0$	$U_3 + U_4 - U_5 - U_6 = 0$	$t := 0.02$
$U_{C1}+U_{R3}-U_{L1}=0$	$U_5 + U_6 - U_7 = 0$	$Ik(t,1) - Ik(t,2) - Ik(t,3) - Ik(t,4) = 0$
$U_{R1}+U_{R2}+U_{L2}=E$	$U_2 + U_3 + U_4 = 220$	$Uk(t,5) + Uk(t,6) - Uk(t,7) = -0$
$i_1-i_2-i_3-i_4=0;$	$I_1 - I_2 - I_3 - I_4 = 0$	$Uk(t,2) + Uk(t,3) + Uk(t,4) - Uk(t,1) = 0$

При проверке уравнений, составленных по законам Кирхгофа для исследуемой схемы в мгновенных значениях *(в)* случайным образом задан момент $t = 0.02$ [с], однако, можно произвести проверку в любой другой момент времени.

6) Во-вторых, рассчитаем *баланс мощностей* в схеме по формулам (51)-(54), построив полную мощность источников энергии ***Sист*** [ВА] и полную суммарную мощность всех потребителей ***Sпотр*** [ВА].

$Sист := U_1 \cdot I_1$ $\quad Sист = 1044.918 - 853.072i$ [ВА]

$Pист := Re(Sист)$ $\quad Pист = 1044.918$ [Вт]

$Qист := Im(Sист)$ $\quad Qист = -853.072$ [ВАр]

$Pпотр := R1 \cdot \left(\left|I_1\right|\right)^2 + R2 \cdot \left(\left|I_2\right|\right)^2 + R3 \cdot \left(\left|I_3\right|\right)^2$ $\quad Pпотр = 1044.918$ [Вт]

$Qпотр := XL2 \cdot \left(\left|I_2\right|\right)^2 - XC1 \cdot \left(\left|I_3\right|\right)^2 + XL1 \cdot \left(\left|I_4\right|\right)^2$ $\quad Qпотр = 853.072$ [ВАр]

$Sпотр := Pпотр + i \cdot Qпотр$ $\quad Sпотр = 1044.918 + 853.072i$ [ВА]

Можно видеть, что активная мощность источника равняется сумме мощностей, потребляемых активными элементами цепи, а реактивная мощность источника равняется по модулю сумме мощностей, потребляемых реактивными элементами схемы.

Расчет цепи переменного тока окончен. Нам известны:
1. ***Сопротивления*** каждого элемента схемы
2. ***Токи,*** протекающие через каждый элемент схемы в любой момент времени
3. ***Напряжения,*** прикладываемые к каждому элементу схемы в любой момент времени
4. ***Мощности***, потребляемые любым элементом схемы
5. ***Мощность***, генерируемая источником схемы

Переходный процесс

Нужно понимать, что приведенные выше расчеты электрической схемы показывают какие токи и напряжения будут на её элементах при достаточно долгом времени работы системы - ***установившиеся*** токи и напряжения. Но в начальный момент времени, пока источник не включен и еще не начал подавать энергию в цепи, все токи и напряжения равны нулю. После включения схемы происходит постепенный (плавный) переход от нулевых значений токов и напряжений к установившимся - тем, которые рассчитаны изученным способом. Процесс перехода от нулевых состояний токов и напряжений схемы к установившимся называется ***переходный процесс*** или ***процесс установления***. Переходный процесс, разумеется, занимает некоторое время.

Методика расчета цепи, описанная выше не учитывает переходный процесс, а сразу позволяет рассчитать ***установившийся процесс.*** Для того, чтобы производить расчет переходного процесса требуется изучить такой математический аппарат как методы решения *дифференциальных уравнений*. Позже, когда в курсе математики этот инструментарий будет изучен, в курсе *"Теоретические основы электротехники"* мы на новом уровне вернемся к расчетам электрических цепей переменного тока с учетом переходного процесса (см. *Главу 10 «Обыкновенные дифференциальные уравнения»*).

Контрольные вопросы
1. Сформулируйте первый закон Кирхгофа для цепи переменного тока.
2. Как связано напряжение, сопротивление и ток в резисторе?
3. Что такое активное сопротивление, как оно связано с полным сопротивлением участка цепи переменного тока?
4. Сформулируйте закон сохранения энергии в приложении к балансу мощностей цепи переменного тока.
5. Что такое полное сопротивление участка цепи? Запишите формулу.
6. Как связано векторное (комплексное) представление напряжения и форма его гармонического сигнала?

7. Назовите единицы измерения полной, активной и реактивной мощностей. Как они связаны?
8. Как вычисляется активная мощность источника?
9. Сформулируйте баланс мощностей в цепи переменного тока.
10. Полная мощность. По каким формулам можно вычислить мощность?
11. Сформулируйте закон Ома для цепи переменного тока.
12. Могут ли токи иметь отрицательное значение в цепи переменного тока?
13. На что расходуется энергия активного сопротивления участка цепи? Как вычисляется величина потраченной мощности?
14. Как изменяется вектор комплексного представления тока при изменении времени?
15. Назовите единицы измерения тока, напряжения, сопротивления и мощности.
16. Сформулируйте второй закон Кирхгофа для цепи переменного тока.
17. Как связано напряжение, сопротивление и ток в конденсаторе?
18. Что такое реактивное сопротивление, какое оно бывает? Как оно связано с полным сопротивлением участка цепи переменного тока?
19. В чем состоит идея представления гармонического сигнала комплексной переменной? Как изменяется угол?
20. Как связано векторное (комплексное) представление тока и форма его гармонического сигнала?
21. Как связано напряжение, сопротивление и ток в катушке индуктивности?
22. На что расходуется энергия реактивного элемента цепи? Как вычисляется величина потраченной мощности?
23. Чем отличается мгновенная (полная) фаза сигнала от начальной его фазы? Запишите формулу.

8. СИМВОЛЬНЫЕ ВЫЧИСЛЕНИЯ

Ядро символьных вычислений MathCAD предназначено для аналитических преобразований математических выражений. Некоторые из этих возможностей *MathCAD* (версии до 13.1 включительно) основаны на подмножестве системы компьютерной алгебры *Maple* (*MKM, Maple Kernel Mathsoft).* Начиная с 14 версии *MathCAD* использует символьное ядро *MuPAD.*

8.1. Символьные вычисления в MathCAD

Символьные вычисления – это удобный инструмент, который существенно облегчает работу с математическими преобразованиями. Символьные вычисления позволяют упростить математическое выражение (*Листинг 52*), в то время как численные расчеты возвращают просто значение выражения в точке.

Листинг 52. Символьное упрощение выражения

$$f(x) := \sum_{k=0}^{3} \left(\frac{3!}{k! \cdot (3-k)!} \cdot x^{k} \cdot 2^{3-k} \right) \qquad \sum_{k=0}^{3} \left(\frac{3!}{k! \cdot (3-k)!} \cdot x^{k} \cdot 2^{3-k} \right) \text{simplify} \rightarrow 8 + 12 \cdot x + 6 \cdot x^{2} + x^{3}$$

$$f(3) = 125 \qquad f(0) = 8$$

Воспользоваться возможностями символьного ядра можно вызвав одну из функций символьных преобразований (*Таблица 3*) из инструментальной панели шаблонов «*Symbolic*» или вызвав одноименную вкладку главного меню (*Рис. 58*).

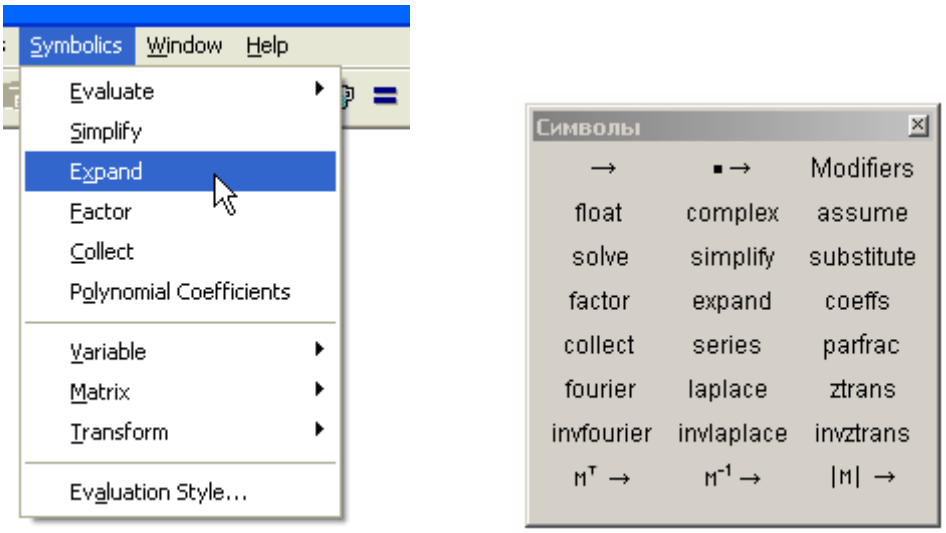


Рис. 58 – Вызов символьных преобразований MathCAD

Таблица 3. Функции символьных вычислений MathCAD

Функции	Выполняемые операции
float	Численное вычисление
complex	Комплексное вычисление
assume	Символьное вычисление с некоторыми предположениями
solve	Решение уравнения, неравенства, системы уравнений
simplify	Упрощение выражений
substitute	Замена переменной
factor	Разложение на множители
expand	Перемножение степеней и произведений
coeffs	Определение коэффициентов полинома
collect	Группировка слагаемых по степеням переменной

series	Разложение в ряд Тейлора или Лорана
parfrac	Разложение на элементарные дроби
fourier	Преобразование Фурье
invfourier	Обратное преобразование Фурье
laplace	Преобразование Лапласа
invlaplace	Обратное преобразование Лапласа
ztrans	Z-преобразование
invztrans	Обратное Z-преобразование

На самом деле, оператор индикации символьных значений →, который мы активно использовали в предыдущих работах, тоже относится к символьным операторам и вызывает алгоритмы символьных вычислений (см. *Листинг 53*).

Многие задачи математического анализа, например, суммирование ряда, взятие производных, интегрирование, взятие несобственного интеграла и т.п. могут быть легко разрешены при помощи ядра символьных вычислений *MathCAD*.

Листинг 53. Оператор → индикации символьных значений

$$\sum_{i=1}^{n} i^2 \to \frac{1}{3}\cdot(n+1)^3 - \frac{1}{2}\cdot(n+1)^2 + \frac{1}{6}\cdot n + \frac{1}{6} \qquad \frac{d}{dt}\sin\left(10t + \frac{\pi}{4}\right) \to 10\cdot\cos\left(10\cdot t + \frac{1}{4}\cdot\pi\right)$$

$$\int \ln(x)\,dx \to x\cdot\ln(x) - x \qquad \int_a^{\infty} \frac{1}{x^3}\,dx \to \frac{1}{2\cdot a^2} \qquad \int_a^{b} \frac{1}{x^3}\,dx \to \frac{-1}{2\cdot b^2} + \frac{1}{2\cdot a^2} \qquad \prod_{n=1}^{\infty} \frac{1}{n^2+1} \to 0$$

8.2. Аналитическое решение уравнений

Символьный процессор программы *MathCAD* позволяет решать уравнения и неравенства аналитически, для этого используется встроенная функция **solve** (*решить*):

Листинг 54. Решение уравнений и неравенств - символьные преобразования

$$x^3 - 5x^2 - 4x + 20 \text{ solve},x \to \begin{pmatrix} 5 \\ 2 \\ -2 \end{pmatrix} \qquad x^3 - 5x^2 - 4x + 20 > 0 \text{ solve},x \to \begin{pmatrix} -2 < x \cdot x < 2 \\ 5 < x \end{pmatrix}$$

Можно осуществлять символьные подстановки в выражение (*Листинг 55*). Для этого используется функция **substitute** (*подстановка*):

Листинг 55. Символьная подстановка

$$f(k,x) := \cos(k\cdot x) + 4x^{2-k}$$

$$f(k,x) \text{ substitute},k = \sqrt{x} \to \cos\left(x^{\frac{3}{2}}\right) + 4\cdot x^{2-x^{\frac{1}{2}}}$$

Ограничения на переменную

При решении уравнений и неравенств, при разложении на множители и во множестве других задач, бывает необходимо *искусственно задать ограничения* на область определения той или иной переменной. Для этих целей используется встроенная функция **assume** (*принять*). Введенные ограничения действительны только в области данного оператора – в остальных частях программы они не учитываются.

В примере (*Листинг 56*) аналитически вычислен определенный интеграл. Символьный процессор *MathCAD* настолько совершенен, что вычислил три решения интеграла, в зависимости от значений *a* и *b* – пределов интегрирования. Принимая

ограничения на переменные *a* и *b*, можно получить одно из этих решений, как показано в нижней части примера (*Листинг 56*).

Листинг 56. Ограничения на переменную

$$\int_a^b \frac{1}{x^3}\,dx \rightarrow \begin{cases} -\infty & \text{if } a \neq 0 \wedge b = 0 \\ \infty & \text{if } b \neq 0 \wedge a = 0 \\ -\dfrac{a^2 - b^2}{2 \cdot a^2 \cdot b^2} & \text{if } (a \geq 0 \vee 0 \geq b) \wedge a \neq 0 \wedge b \neq 0 \end{cases}$$

$$\int_a^b \frac{1}{x^3}\,dx \text{ assume}, a \neq 0 \wedge b = 0 \rightarrow -\infty \qquad \int_a^b \frac{1}{x^3}\,dx \text{ assume}, a > 0 \wedge b \neq 0 \rightarrow \frac{1}{2 \cdot a^2} - \frac{1}{2 \cdot b^2}$$

Используя команды **float** (*вещественные*) и **complex** (*комплексные*), можно переключать символьные вычисления из одного режима в другой.

8.3. Упрощение выражений

Возможности символьного процессора *MathCAD* сводят на нет любимое развлечение учителей средней школы – упрощение выражений. Теперь упрощение громоздких и сложных формул можно выполнить одной командой – **simplify** (*упростить*).

Как видно на примере ниже (*Листинг 57*), можно заставить операцию **simplify** (и другие функции символьного процессора) возвращать вещественные значения вместо символьных, для этого необходимо один из аргументов функции задать вещественным числом. Как это сделано в примере ниже при вычислении asin(1.). Нужно понимать при этом, что расчеты всё равно проводятся аналитически, а затем вычисленное значение π/2 переводится в формат с десятичной точкой.

Листинг 57. Символьные операции. Упростить выражение

$$\sin(x)^2 + \cos(x)^2 \text{ simplify} \rightarrow 1$$

$$\text{asin}(1) \text{ simplify} \rightarrow \frac{1}{2} \cdot \pi \qquad \text{asin}(1.) \text{ simplify} \rightarrow 1.5707963267948966192$$

$$\frac{x^2 - 3x - 4}{x - 4} + 2x - 5 \text{ simplify} \rightarrow 3 \cdot x - 4 \qquad \sqrt{75 x^4 y} \text{ simplify} \rightarrow 5 \cdot 3^{\frac{1}{2}} \cdot \left(x^4 \cdot y\right)^{\frac{1}{2}}$$

8.4. Разложение по степеням

В *MathCAD* имеется встроенная возможность аналитически раскрывать скобки в сложных выражениях, это делается функцией **expand** (*раскрыть*) и раскладывать на множители (группировать по степеням) – **collect** (*сбор*), см. *Листинг 58*.

Листинг 58. Разложение по степеням. Разложение на множители

$$f(x,y,a) := 3 \cdot x^3 \cdot y \cdot \left(4 \cdot x^2 - 5 \cdot a^3 \cdot y^2\right) + 2a + 4x^5 - 12a \cdot y^3$$

$$f(x,y,a) \text{ expand}, x \rightarrow 12 \cdot x^5 \cdot y - 15 \cdot x^3 \cdot y^3 \cdot a^3 + 2 \cdot a + 4 \cdot x^5 - 12 \cdot a \cdot y^3$$

$$f(x,y,a)\ \text{collect},x \rightarrow (12\cdot y + 4)\cdot x^5 - 15\cdot x^3\cdot y^3\cdot a^3 - 12\cdot a\cdot y^3 + 2\cdot a$$

$$f(x,y,a)\ \text{collect},y \rightarrow \left[(-15)\cdot x^3\cdot a^3 - 12\cdot a\right]\cdot y^3 + 12\cdot x^5\cdot y + 2\cdot a + 4\cdot x^5$$

Коэффициенты разложения многочлена

Операция разложения многочлена по степеням эквивалентна вычислению коэффициентов разложения многочлена перед соответствующими степенями переменных, в этом можно убедится, сравнив *Листинг 58* и *Листинг 59*. Получение этих коэффициентов производится при помощи функции **coeffs** (*коэффициенты*).

Листинг 59. Вычисление коэффициентов многочлена

$$f(x,y,a) := 3\cdot x^3\cdot y\cdot\left(4\cdot x^2 - 5\cdot a^3\cdot y^2\right) + 2a + 4x^5 - 12a\cdot y^3$$

$$f(x,y,a)\ \text{coeffs},y \rightarrow \begin{bmatrix} 2\cdot a + 4\cdot x^5 \\ 12\cdot x^5 \\ 0 \\ (-15)\cdot x^3\cdot a^3 - 12\cdot a \end{bmatrix} \qquad f(x,y,a)\ \text{coeffs},x \rightarrow \begin{bmatrix} 2\cdot a - 12\cdot a\cdot y^3 \\ 0 \\ 0 \\ (-15)\cdot y^3\cdot a^3 \\ 0 \\ 12\cdot y + 4 \end{bmatrix}$$

Самый простой и известный со школы способ разложения на степени – это представление полинома в виде произведения сомножителей. Действие это в аналитическом виде может быть проделано при помощи функции **factor** (*факторизация*). Такое преобразование многочлена сразу показывает все его корни (*Листинг 60*).

Листинг 60. Разложение на множители

$$x^4 + x^3 - 13\cdot x^2 - 25\cdot x - 12\ \text{factor} \rightarrow (x + 3)\cdot(x - 4)\cdot(x + 1)^2$$

Разложение в степенной ряд

Кроме того, разложение многочлена по степеням можно выполнить, используя встроенную функцию **series** (*разложение в степенной ряд*). Сравните *Листинг 61* с примерами, приведенными выше (*Листинг 58* и *Листинг 59*).

Листинг 61. Разложение полинома в степенной ряд

$$f(x,y,a)\ \text{series},x,6 \rightarrow 2\cdot a - 12\cdot a\cdot y^3 + (-15)\cdot y^3\cdot a^3\cdot x^3 + (12\cdot y + 4)\cdot x^5$$

$$f(x,y,a)\ \text{series},y,5 \rightarrow 2\cdot a + 4\cdot x^5 + 12\cdot x^5\cdot y + \left[(-15)\cdot x^3\cdot a^3 - 12\cdot a\right]\cdot y^3$$

Функция **series** представляет в виде ряда *Тейлора* (или ряда *Лорана*) любую функцию, для этого справа от слова **series** необходимо указать раскладываемую функцию, а слева через запятую – в окрестности какой точки производить разложение (если этот параметр не указан, то – в окрестности нуля) и до какой степени раскладывать (по умолчанию – 5).

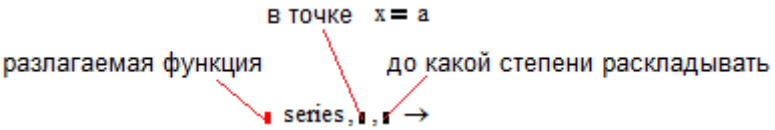


Рис. 59 – Функция **series** *– разложение в степенной ряд*

Данная функция очень востребована в задаче разложения громоздкого выражения на простые дроби. Эта задача актуальна для вычисления интегралов через вычеты, с которой вы встретитесь в курсе теории автоматического управления.

Листинг 62. Разложение в степенной ряд в окрестности точки

$$\ln(x+2)\ \text{series},x,5 \rightarrow \ln(2)+\frac{1}{2}\cdot x-\frac{1}{8}\cdot x^2+\frac{1}{24}\cdot x^3-\frac{1}{64}\cdot x^4$$

$$\ln(x+2)\ \text{series},x=2,5 \rightarrow \ln(4)+\frac{1}{4}\cdot(x-2)-\frac{1}{32}\cdot(x-2)^2+\frac{1}{192}\cdot(x-2)^3-\frac{1}{1024}\cdot(x-2)^4$$

Пользуясь функциями **expand, collect** и **series** можно преобразовать любую функцию к полиномиальному виду, а с полиномами легко работать.

8.5. Задача разложения на простые дроби

Одной из важных задач, часто встречающихся в математике (в теории комплексных функций, теории вычетов, вариационном исчислении) и в прикладных науках является задача разложения дробно-рациональной функции $R(x)$ вида (55) на сумму простейших рациональных дробей.

$$R(x)=\frac{Q_m(x)}{P_n(x)}=\frac{b_m\cdot x^m+b_{m-1}\cdot x^{m-1}+\ldots+b_0}{a_n\cdot x^n+a_{n-1}\cdot x^{n-1}+\ldots+a_0} \qquad (55)$$

Знаменатель этой дроби $P_n(x)$ может быть представлен в виде элементарных множителей (разложен на множители). Об этом в курсе алгебры строго доказывается *теорема 1*:

Всякий действительный многочлен $P_n(x)=a_n\cdot x^n+a_{n-1}\cdot x^{n-1}+\ldots+a_0$ степени n может быть представлен в виде:

$$P_n(x)=a_n(x-x_1)^{\lambda_1}\times\ldots\times(x-x_r)^{\lambda_r}\times(x^2+p_1x+q_1)^{\mu_1}\times\ldots\times(x^2+p_sx+q_s)^{\mu_s}, \qquad (56)$$

где $x_1,\ldots,x_r$ - различные действительные корни кратностей $\lambda_1,\ldots,\lambda_r$ многочлена $P_n(x)$, а $(x^2+p_kx+q_k)^{\mu_k}$, $k=1\ldots s$ - квадратные трехчлены, имеющие комплексные корни соответствующих кратностей $\mu_1,\ldots,\mu_s$. Причем, $n=\lambda_1+\ldots+\lambda_r+2(\mu_1+\ldots+\mu_s)$.

В отношении разложения дробно-рациональной функции $R(x)$ вида (55) на сумму простейших рациональных дробей доказывается следующая *теорема 2*:

Всякую правильную рациональную дробь $R(x)$ вида (55), со знаменателем, представленным в виде (56), можно разложить на сумму дробей вида (57) - (58):

$$\frac{A}{(x-x_i)^{\lambda_i}}\ \text{и}\ \frac{Mx+N}{(x^2+p_kx+q_k)^{\mu_k}}.$$

В этом разложении каждому действительному корню x_i кратности λ_i, $i=1,\ldots,r$ соответствует сумма λ_i дробей вида:

$$\frac{A_1}{x-x_i}+\frac{A_2}{(x-x_i)^2}+\ldots+\frac{A_{\lambda i}}{(x-x_i)^{\lambda_i}}, \qquad (57)$$

а каждой паре комплексно-сопряженных корней, задаваемых квадратным трехчленом $(x^2+p_kx+q_k)^{\mu_k}$, $k=1\ldots s$, соответствует сумма дробей вида:

$$\frac{M_1x+N_1}{x^2+p_kx+q_k}+\frac{M_2x+N_2}{(x^2+p_kx+q_k)^2}+\ldots+\frac{M_{\mu k}\cdot x+N_{\mu k}}{(x^2+p_kx+q_k)^{\mu_k}}. \qquad (58)$$

Какова же методика разложения рациональной дроби $R(x)$ вида (55) на сумму дробей вида (57) - (58)?

Алгоритм очень прост, но громоздок:

1) представить знаменатель Pn(x) в виде (56);
2) выписать разложение дроби R(x) в виде суммы элементарных дробей (57) - (58), полагая коэффициенты $A_1,...,A_{\lambda i}$, $M_1,...,M_{\mu k}$ и $N_1,...,N_{\mu k}$ неопределенными;
3) привести полученную сумму элементарных дробей к общему знаменателю;
4) полученный числитель, включающий неопределенные коэффициенты $A_1,...,A_{\lambda i}$, $M_1,...,M_{\mu k}$ и $N_1,...,N_{\mu k}$ приравнять к исходному числителю $Q_m(x)$ из (55);
5) для нахождения неопределенных коэффициентов нужно приравнять выражения при одинаковых степенях x, получится система линейных алгебраических уравнений, решением которой, будут коэффициенты $A_1,...,A_{\lambda i}$, $M_1,...,M_{\mu k}$ и $N_1,...,N_{\mu k}$.

Легко убедиться, что при использовании системы *MathCAD*, данная громоздкая задача решается сравнительно просто (*Листинг 63*):

Листинг 63. Разложение на простые дроби

Задача:
Разложить на сумму элементарных дробей функцию:

$$\frac{8\cdot x^3 - 23\cdot x^2 + 37\cdot x - 70}{x^4 - 5\cdot x^3 + 6\cdot x^2 + 16\cdot x - 48}$$

Решение:
В соответствии с *теоремой 1* представим знаменатель дроби в виде (56) при помощи функции **factor**:

$$x^4 - 5\cdot x^3 + 6\cdot x^2 + 16\cdot x - 48 \text{ factor} \rightarrow (x-3)\cdot(x+2)\cdot\left(x^2 - 4\cdot x + 8\right)$$

В соответствии с *теоремой 2* разложение будем искать в виде суммы:

$$\frac{C}{x-3} + \frac{D}{x+2} + \frac{Ex+F}{x^2 - 4x + 8}$$

Приведем данную сумму простейших дробей к общему знаменателю при помощи функции **factor**:

$$\frac{C}{x-3} + \frac{D}{x+2} + \frac{E\cdot x + F}{x^2 - 4x + 8} \text{ factor} \rightarrow$$

$$\rightarrow \frac{16\cdot C - 24\cdot D - 6\cdot F - 2\cdot C\cdot x^2 + C\cdot x^3 - 7\cdot D\cdot x^2 + D\cdot x^3 - E\cdot x^2 + E\cdot x^3 + F\cdot x^2 + 20\cdot D\cdot x - 6\cdot E\cdot x - F\cdot x}{(x-3)\cdot(x+2)\cdot\left(x^2 - 4\cdot x + 8\right)}$$

Рассмотрим числитель полученной дроби:

$$ch(x) := 16\cdot C - 24\cdot D - 6\cdot F - 2\cdot C\cdot x^2 + C\cdot x^3 - 7\cdot D\cdot x^2 + D\cdot x^3 - E\cdot x^2 + E\cdot x^3 + F\cdot x^2 + 20\cdot D\cdot x - 6\cdot E\cdot x - F\cdot x$$

Построим коэффициенты разложения числителя по степеням x при помощи функции **coeffs.** Получим систему линейных алгебраических уравнений относительно неизвестных коэффициентов C, D, E и F. Построим A – матрицу СЛАУ:

$$ch(x)\ coeffs, x \rightarrow \begin{pmatrix} 16\cdot C - 24\cdot D - 6\cdot F \\ 20\cdot D - 6\cdot E - F \\ F - 7\cdot D - E - 2\cdot C \\ C + D + E \end{pmatrix} \qquad A := \begin{pmatrix} 16 & -24 & 0 & -6 \\ 0 & 20 & -6 & -1 \\ -2 & -7 & -1 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Построим коэффициенты разложения числителя по степеням x при помощи функции **coeffs,** получим вектор B правых частей СЛАУ. Решим систему.

$$B := 8\cdot x^3 - 23\cdot x^2 + 37\cdot x - 70\ coeffs \rightarrow \begin{pmatrix} -70 \\ 37 \\ -23 \\ 8 \end{pmatrix} \qquad X := A^{-1}\cdot B \qquad X = \begin{pmatrix} 2 \\ 3 \\ 3 \\ 5 \end{pmatrix}$$

Полученный вектор X содержит искомые коэффициенты C, D, E и F. Подставим их в

разложение и получим решение исходной задачи:

$$\frac{2}{x-3}+\frac{3}{x+2}+\frac{3x+5}{x^2-4x+8}$$

Для проверки правильности разложения рациональной дроби $R(x)$ на сумму элементарных дробей вида (57) - (58), достаточно привести полученное разложение к общему знаменателю и раскрыть скобки, это легко сделать, используя функции **factor** и **expand.**

Листинг 64. Разложение на простые дроби. Проверка

$$\frac{2}{x-3}+\frac{3}{x+2}+\frac{3x+5}{x^2-4x+8}\ \text{factor}\ \rightarrow \frac{8\cdot x^3-23\cdot x^2+37\cdot x-70}{(x-3)\cdot(x+2)\cdot\left(x^2-4\cdot x+8\right)}$$

$$(x-3)\cdot(x+2)\cdot\left(x^2-4\cdot x+8\right)\ \text{expand}\ \rightarrow x^4-5\cdot x^3+6\cdot x^2+16\cdot x-48$$

В *MathCAD* имеется встроенная функция **parfrac** для реализации разложений на элементарные дроби вида (57) - (58), она позволяет решить задачу, разобранную в примере выше (*Листинг 63*), в одно действие:

Листинг 65. Разложение на простые дроби. Встроенная функция **parfrac**

$$\frac{8\cdot x^3-23\cdot x^2+37\cdot x-70}{x^4-5\cdot x^3+6\cdot x^2+16\cdot x-48}\ \text{parfrac}\ \rightarrow \frac{3\cdot x+5}{x^2-4\cdot x+8}+\frac{3}{x+2}+\frac{2}{x-3}$$

Контрольные вопросы:

1. Как символьно упростить громоздкое выражение?
2. Воспроизведите алгоритм разложения на простые дроби.
3. Имеет ли MathCAD средства разложения функции в степенные дроби?
4. Как символьно вычислить коэффициенты многочлена?

9. ИНТЕРПОЛЯЦИЯ И РЕГРЕССИЯ

Этот раздел посвящен методам обработки выборочных данных – *интерполяции, экстраполяции* и *регрессии*. Основным объектом исследования будет *выборка экспериментальных данных*, которые, представляются в виде массива пар чисел (x_i, y_i), $i = 1..n$. Возникает задача аппроксимации дискретной зависимости $y(x_i)$ непрерывной функцией $f(x)$. Функция $f(x)$, в зависимости от специфики задачи, может отвечать различным требованиям:

- $f(x)$ должна проходить через точки (x_i, y_i), т.е. $f(x_i) = y_i$, $i = 1..n$. В этом случае говорят об ***интерполяции*** функции $f(x)$ во внутренних точках $x \in [x_1, x_n]$ или *экстраполяции* во внешних точках: $x < x_1$ или $x > x_n$ – за пределами заданного интервала.
- $f(x)$ должна некоторым образом (например, в виде определенной аналитической зависимости) приближать $y(x_i)$, не обязательно проходя через точки (x_i, y_i). Такова постановка задачи ***регрессии***.

MathCAD имеет целый арсенал встроенных функций, позволяющих осуществлять различные виды регрессии, интерполяции и экстраполяции.

9.1. Интерполяция

Для построения интерполяции-экстраполяции в *MathCAD* имеется несколько встроенных функций, позволяющих «соединить» точки выборки данных (x_i, y_i) кривой разной степени гладкости. По определению интерполяция означает построение функции $\varphi(x)$, аппроксимирующей зависимость $y(x_i)$ в промежуточных точках (между x_i). Поэтому интерполяцию называют кроме того *аппроксимацией*.

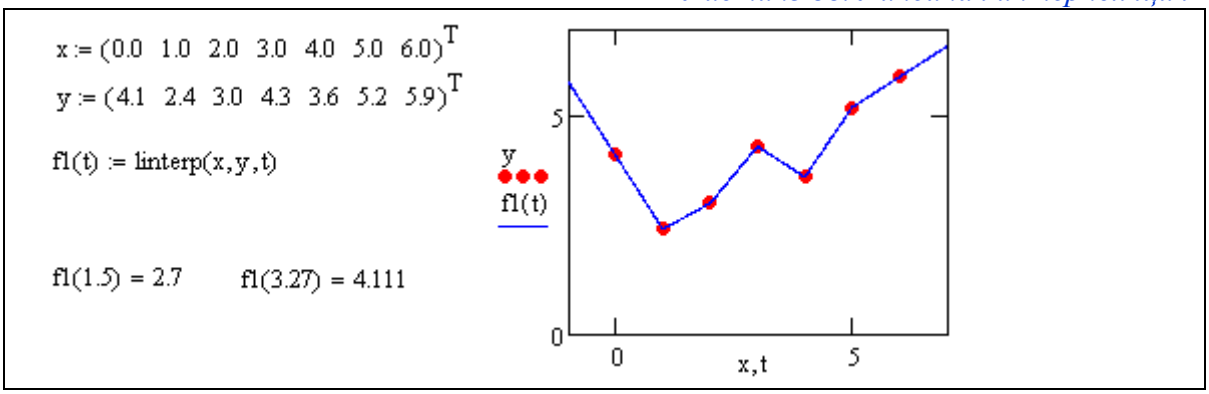
Линейная интерполяция

Самый простой вид интерполяции – линейная, которая представляет искомую зависимость $\varphi(x)$ в виде ломаной линии (*Листинг 66*).

Для построения линейной интерполяции служит встроенная функция **linterp**(*x, y, t*) – функция, аппроксимирующая данные векторов ***x*** и ***y*** кусочно-линейной зависимостью. Здесь ***x*** – вектор аргументов, а ***y*** – вектор соответствующих значений того же размера; *t* – координата, в которой вычисляется интерполирующая функция.

При составлении вектора *x* следует помнить, что его элементы должны быть определены в порядке возрастания, т.е. $x_1 < x_2 < ... < x_n$.

Листинг 66. Линейная интерполяция



Кубическая сплайн-интерполяция

В большинстве практических приложений желательно соединить экспериментальные точки не ломаной линией, а гладкой кривой. Лучше всего для этих

целей подходит интерполяция кубическими сплайнами, т.е. отрезками кубических парабол:

interp(*s*, *x*, *y*, *t*) – функция, аппроксимирующая данные векторов *x* и *y* кубическими сплайнами; *s* – вектор вторых производных, созданный одной из сопутствующих функций **cspline, pspline** или **lspline**:

s:=**lspline**(*x*, *y*) – вектор значений коэффициентов линейного сплайна;

s:=**pspline**(*x*, *y*) – вектор значений коэффициентов квадратичного сплайна;

s:=**cspline**(*x*, *y*) – вектор значений коэффициентов кубического сплайна.

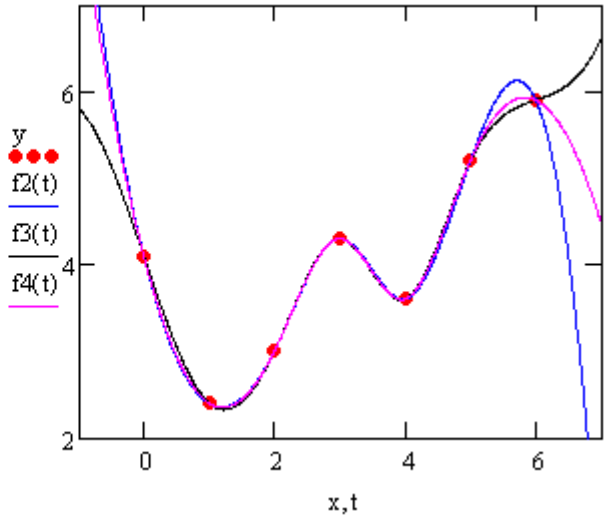
x, *y* — векторы данных (x_i, y_i).

Выбор конкретной функции сплайновых коэффициентов влияет на интерполяцию вблизи конечных точек интервала (*Листинг 67*).

Листинг 67. Кубическая сплайн-интерполяция

```
vc := cspline(x,y)      f2(t) := interp(vc,x,y,t)
vl := lspline(x,y)      f3(t) := interp(vl,x,y,t)
vp := pspline(x,y)      f4(t) := interp(vp,x,y,t)

f1(1.5) = 2.7           f1(3.27) = 4.111
f2(1.5) = 2.4544642857  f2(3.27) = 4.1717865786
f3(1.5) = 2.4164423077  f3(3.27) = 4.1565649715
f4(1.5) = 2.4468301435  f4(3.27) = 4.1627213033
```

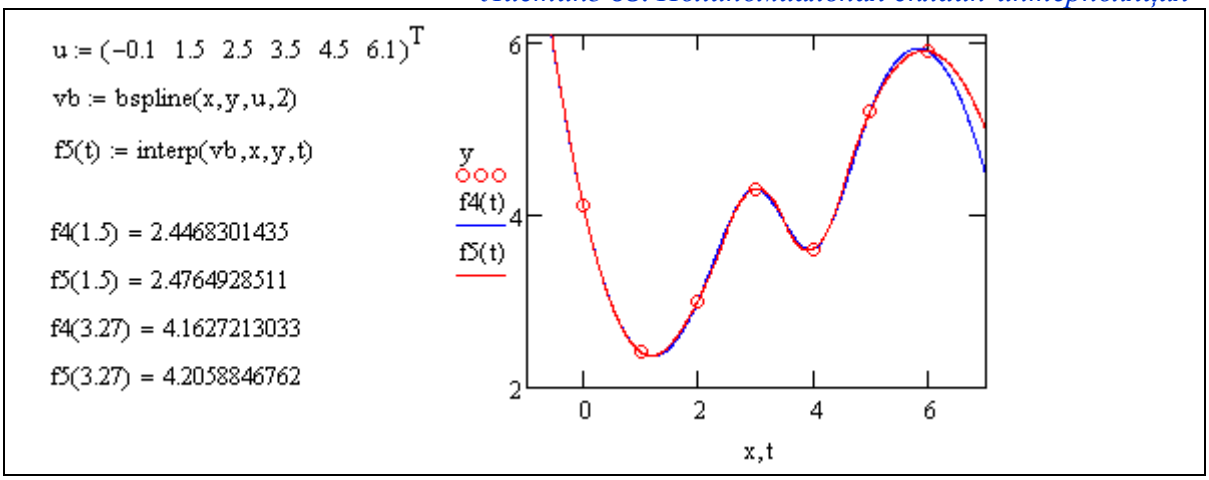


Смысл сплайн-интерполяции заключается в том, что в промежутках между точками осуществляется аппроксимация в виде зависимости $\varphi(t) = a \cdot t^3 + b \cdot t^2 + c \cdot t + d$. Коэффициенты *a, b, c, d* рассчитываются независимо для каждого промежутка, исходя из значений y_i в соседних точках. Этот процесс скрыт от пользователя, поскольку смысл задачи интерполяции состоит в выдаче значения $\varphi(x)$ в любой точке *t* (*Листинг 67*). Чтобы подчеркнуть различия, соответствующие разным вспомогательным функциям **cspline**, **pspline**, **lspline,** графики приведены на одном рисунке. Как видно, выбор вспомогательных функций существенно влияет на поведение $\varphi(x)$ вблизи граничных точек рассматриваемого интервала и особенно разительно меняет результат экстраполяции данных за его пределами.

Полиномиальная сплайн-интерполяция

Более сложный тип интерполяции – так называемая интерполяция *B-сплайнами* (*Листинг 68*). В отличие от обычной сплайн-интерполяции сшивка элементарных *B-сплайнов* производится не в точках x_i а в других точках u_i, координаты которых предлагается ввести пользователю. Сплайны могут быть полиномами 1, 2 или 3 степени (линейные, квадратичные или кубические). Применяется интерполяция *B-сплайнами* точно так же, как и обычная сплайн-интерполяция, различие состоит только в определении вспомогательной функции коэффициентов сплайна.

Листинг 68. Полиномиальная сплайн-интерполяция



interp(*s*, *x*, *y*, *t*) – функция, аппроксимирующая данные векторов *x* и *y* с помощью B-сплайнов;

s:=**bspline**(*x*, *y*, *u*, *n*) – вектор значений коэффициентов *B-сплайна*; где *s* – вектор вторых производных, созданный функцией **bspline;** *x, y* — векторы данных (x_i, y_i); *t* – точка, в которой вычисляется интерполирующая функция; *u* – вектор значений аргумента, в которых производится сшивка *B-сплайнов*; *n* – порядок полиномов онлайновой интерполяции (1, 2 или 3).

Размерность вектора u должна быть на 1, 2 или 3 меньше размерности векторов **x** и **y**. Первый элемент вектора u должен быть меньше или равен первому элементу вектора **x**: $u_1 \le x_1$, а последний – больше или равен последнему элементу x.

Сплайн-экстраполяция

Все описанные выше функции интерполяции работают также и как функции экстраполяции данных. Для вычисления экстраполяции достаточно просто указать соответствующее значение аргумента, которое лежит за границами рассматриваемого интервала. С этой точки зрения разницы в применении в *MathCAD* между интерполяцией и экстраполяцией нет.

На практике при построении экстраполяции следует соблюдать известную осторожность, не забывая о том, что ее успех определяется значимостью ближайших к границе интервала точек – чем дальше от них, тем сомнительнее будет результат.

Многомерная интерполяция

Двумерная сплайн-интерполяция приводит к построению поверхности $z(x, y)$, проходящей через массив точек, описывающий сетку на координатной плоскости (x_i, y_i), $i = 1..n$. Поверхность создается участками двумерных кубических сплайнов, являющихся функциями (x_i, y_i) и имеющих непрерывные первые и вторые производные по обеим координатам.

Многомерная интерполяция строится с помощью тех же встроенных функций, что и одномерная, но имеет в качестве аргументов не векторы, а соответствующие матрицы.

interp(*s*, *x*, *z*, *v*) – скалярная функция, аппроксимирующая данные выборки двумерного поля по координатам *x* и *y* кубическими сплайнами, *s* – вектор вторых производных, созданный одной из сопутствующих функций **cspline, pspline** или **lspline**; *x* – матрица размерности $n \times 2$, определяющая диагональ сетки значений аргумента (элементы обоих столбцов соответствуют меткам *x* и *y* и расположены в порядке возрастания); z – матрица значений функции размерности $n \times n$; *v* – вектор из двух элементов, содержащий значения аргументов *x* и *y*, для которых вычисляется интерполяция.

Вспомогательные функции построения вторых производных имеют те же матричные аргументы: **lspline**(*x*, *y*), **pspline**(*x*, *y*), **cspline**(*x*, *y*).

Листинг 69. Формирование матриц многомерной интерполяции

$$X := \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 3 & 3 \\ 4 & 4 \end{pmatrix} \quad Y := \begin{pmatrix} 1 & 1 & 0 & 1.1 & 1.2 \\ 1 & 5 & 3 & 2.1 & 1.5 \\ 1.3 & 3.3 & 5 & 1.7 & 2 \\ 1.3 & 3 & 3.7 & 2.1 & 2.9 \\ 1.5 & 2 & 2.5 & 2.8 & 1 \end{pmatrix}$$

$S := \text{lspline}(X, Y)$

$i := 0..40 \qquad j := 0..40 \qquad A_{i,j} := \text{interp}\left[S, X, Y, \begin{pmatrix} \frac{i}{10} \\ \frac{j}{10} \end{pmatrix}\right]$

$t := \begin{pmatrix} 2 \\ 1.5 \end{pmatrix} \qquad \text{interp}(S, X, Y, t) = 4.6462053571$

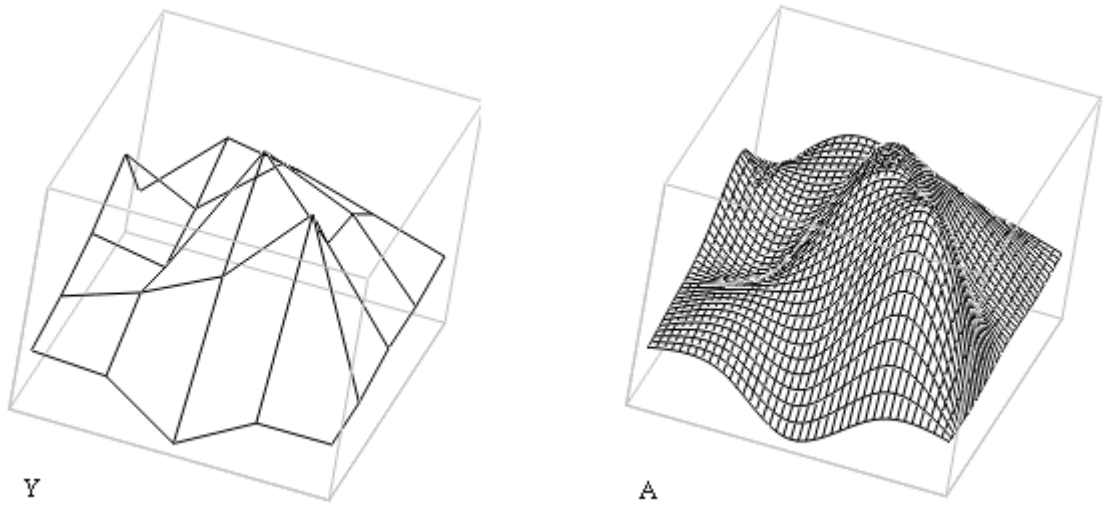


Рис. 60 – Результат двумерной интерполяции

9.2. Регрессия

Задачи математической *регрессии* сводятся к приближению выборки данных (x_i, y_i), $i = 1..n$. некоторой функцией $f(x)$, определенным образом минимизирующей совокупность *ошибок* $\varepsilon_i = |f(x_i) - y_i|$. Регрессия сводится к подбору типа функции $f(x)$ и 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) – тип функции $f(x)$. *MathCAD* позволяет реализовать практически любой алгоритм регрессии, кроме того здесь имеются встроенные функции реализующие *регрессию одним полиномом*, *отрезками нескольких полиномов*, а также *двумерная регрессия* массива данных.

Регрессия одним полиномом

Полиномиальная регрессия означает приближение данных (x_i, y_i) полиномом *k*-й степени $f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_k x^k$, для построения такого полинома коэффициенты a_0, a_1, a_2, ..., a_k требуется рассчитать. Для построения регрессии полиномом *k*-й степени необходимо наличие, по крайней мере, *k*+1 точек данных. При *k* = 1 полином является прямой линией, при *k* = 2 – параболой, и т.д.

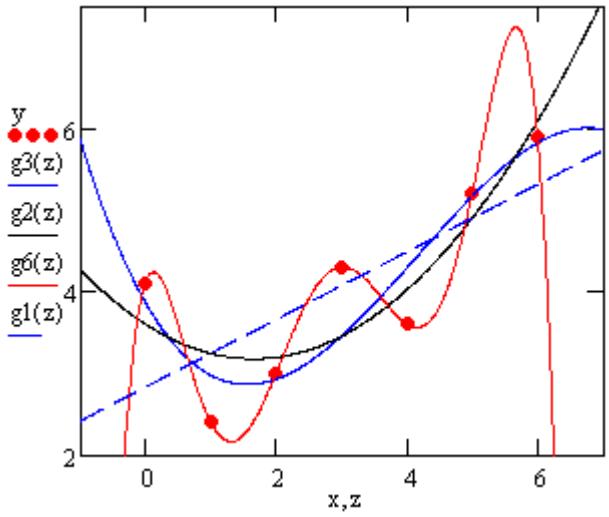
В *MathCAD* полиномиальная регрессия осуществляется комбинацией встроенной функции **regress** и полиномиальной интерполяции:

s=**regress**(*x, y, k*) – расчет вектора коэффициентов для построения полиномиальной регрессии данных;
interp(*s, x, y, t*) – результат полиномиальной регрессии.

Здесь *x, y* – вектор экспериментальных данных аргумента в порядке возрастания и вектор значений того же размера; *k* – степень полинома регрессии (целое положительное число); *t* – значение аргумента полинома регрессии.

Листинг 70. Регрессия полиномом

```
x := (0.0  1.0  2.0  3.0  4.0  5.0  6.0)^T
y := (4.1  2.4  3.0  4.3  3.6  5.2  5.9)^T
g1(z) := interp(regress(x,y,1),x,y,z)
g2(z) := interp(regress(x,y,2),x,y,z)
g3(z) := interp(regress(x,y,3),x,y,z)
g4(z) := interp(regress(x,y,4),x,y,z)
g5(z) := interp(regress(x,y,5),x,y,z)
g6(z) := interp(regress(x,y,6),x,y,z)
```



Точность регрессии

Качество регрессии можно оценить величиной среднеквадратической ошибки аппроксимации в выборочных точках (x_i, y_i). Рассмотрим такую оценку на примере полиномиальной регрессии с полиномами различной степени (*Листинг 70*).

Листинг 71. Среднеквадратичная ошибка регрессии

$n := \mathrm{rows}(x) \quad n = 7$

$$\sqrt{\frac{1}{n}\cdot\sum_{k=0}^{n-1}\left(y_k - g1\left(x_k\right)\right)^2} = 0.75862 \qquad \sqrt{\frac{1}{n}\cdot\sum_{k=0}^{n-1}\left(y_k - g2\left(x_k\right)\right)^2} = 0.53675$$

$$\sqrt{\frac{1}{n}\cdot\sum_{k=0}^{n-1}\left(y_k - g3\left(x_k\right)\right)^2} = 0.4766 \qquad \sqrt{\frac{1}{n}\cdot\sum_{k=0}^{n-1}\left(y_k - g6\left(x_k\right)\right)^2} = 0$$

Полином 1-го порядка $g1(z)$ – прямая линия, его график проходит с минимальным среднеквадратическим удалением от выборочных точек. Полином 6-го прядка $g6(z)$ имеет 7 параметров – коэффициентов, поэтому его график точно прошел через все выборочные точки, регрессия превратилась в интерполяцию. Для каждого из полиномов во всех узловых точках (x_i, y_i) построены величины среднеквадратической ошибки аппроксимации.

Здесь функция **rows**( ) используется для определения размерности массива *x*.

Как следует из приведенных выше результатов, «точность» регрессии растет с увеличением порядка аппроксимирующего полинома. В том случае, когда число коэффициентов полинома (оно на единицу больше его степени) сравняется с размером

выборки график полинома начинает проходить через все выборочные точки, ошибка становиться нулевой (с точностью до округления). Полином регрессии превращается в интерполяционный полином, дальнейшее наращивание степени бесполезно.

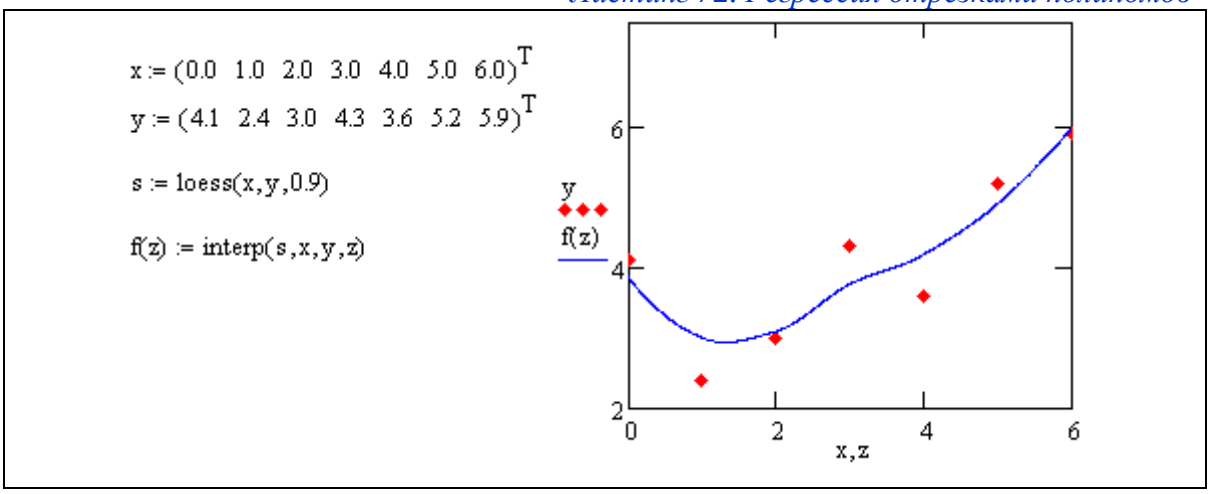
С повышением порядка полинома регрессии ухудшаются «сглаживающие» свойства аппроксимации, кривая графика начинает осциллировать, стремясь «достать» каждую выборочную точку. Это обстоятельство негативно сказывается на результате, если в выборке имеется шумовая компонента.

Регрессия отрезками полиномов

Помимо приближения массива данных одним полиномом имеется возможность осуществить регрессию сшивкой отрезков (точнее говоря, участков, т.к. они имеют криволинейную форму) нескольких полиномов. Для этого имеется встроенная функция **loess**, применение которой аналогично функции **regress:**

s=**loess***(x, y, span)* – вектор коэффициентов для построения регрессии данных отрезками полиномов; **interp***(s, x, y, t)* – результат полиномиальной регрессии. Здесь *span* – параметр, определяющий размер отрезков полиномов (положительное число, хорошие результаты дает значение порядка *span=0.75*). Параметр *span* задает степень сглаженности данных, при больших значениях *span* регрессия практически не отличается от регрессии одним полиномом.

Листинг 72. Регрессия отрезками полиномов



Двумерная полиномиальная регрессия

По аналогии с одномерной полиномиальной регрессией и двумерной интерполяцией, *MathCAD* позволяет приблизить множество точек $z_{i,j} = Z(x_i, y_j)$ поверхностью, которая определяется многомерной полиномиальной зависимостью. В качестве аргументов встроенных функций для построения полиномиальной регрессии должны стоять в этом случае не векторы, а соответствующие матрицы.

s=**regress***(x, z, k)* – вектор коэффициентов для построения полиномиальной регрессии данных (одним полиномом).
s=**loess***(x, z, span)* — вектор коэффициентов для построения регрессии данных отрезками полиномов.
interp(*s, x, z, v*) – скалярная функция, аппроксимирующая данные выборки двумерного поля по координатам *x* и *z* кубическими сплайнами.
Здесь s – вектор вторых производных, созданный одной из сопутствующих функций **loess** или **regress**; *x* – матрица размерности $n \times 2$, определяющая пары значений аргумента – столбцы соответствуют точкам $\left(x_i, y_j\right)$; *z* – вектор данных размерности *n*; *span* – параметр, определяющий размер отрезков полиномов; *k* – степень регрессии полинома (целое положительное число); *v* - вектор из двух элементов, содержащий

значения аргументов $\left(x_i, y_j\right)$, для которых вычисляется интерполяция.

Листинг 73. Двумерная полиномиальная регрессия

$$X := \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 \\ 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 \end{pmatrix}^T$$

$$Y := (1 \;\; 2 \;\; 0 \;\; 1.1 \;\; 1.2 \;\; 1 \;\; 5 \;\; 3 \;\; 2.1 \;\; 1.5 \;\; 1.3 \;\; 3.3 \;\; 5 \;\; 1.7 \;\; 2 \;\; 1.3 \;\; 3 \;\; 3.7 \;\; 2.1 \;\; 2.9 \;\; 1.5 \;\; 2 \;\; 2.5 \;\; 2.8 \;\; 1)^T$$

$S := regress(X, Y, 3)$

$i := 0..40 \quad j := 0..40$

$$A_{i,j} := interp\left[S, X, Y, \begin{pmatrix} \frac{i}{10} \\ \frac{j}{10} \end{pmatrix}\right]$$

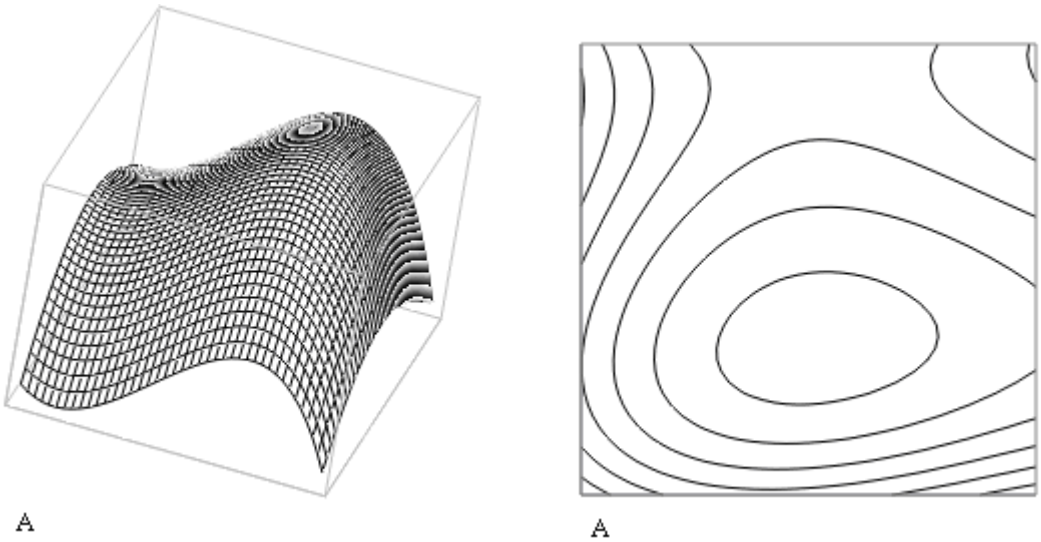


Рис. 61 – Двумерная полиномиальная регрессия

Для построения регрессии не предполагается никакого предварительного упорядочивания данных, в связи с этим данные представляются как вектор.

Другие типы регрессии

Кроме рассмотренных, в *MathCAD* встроено еще несколько видов двумерной регрессии. Их реализация несколько отличается от приведенных выше вариантов регрессии тем, что для них, помимо массива данных, требуется задать некоторые начальные значения коэффициентов *a, b, c*. Используйте соответствующий вид регрессии, если хорошо представляете себе, какой зависимостью описывается ваш массив данных. Когда тип регрессии плохо отражает последовательность данных, то ее результат часто бывает неудовлетворительным и даже сильно различающимся в зависимости от выбора начальных значений. Каждая из функций выдает вектор уточненных параметров *a, b, c*.

expfit(*x, y, g*)	регрессия экспонентой	$f(x) = a \cdot e^{bx} + c$
igsfit(*x, y, g*)	регрессия логистической функцией	$f(x) = a/1 + b \cdot e^{-cx}$
sinfit(*x, y, g*)	регрессия синусоидой	$f(x) = a \cdot \sin(x + b) + c$
pwfit(*x, y, g*)	регрессия степенной функцией	$f(x) = a \cdot x^b + c$
logfit(*x, y, g*)	регрессия логарифмической функцией	$f(x) = a \cdot \ln(x + b) + c$
lnfit(*x, y*)	двухпараметрической логарифмической функцией	$f(x) = a \cdot \ln(x) + b$

Контрольные вопросы:

1. Чем отличаются понятия интерполяция и регрессия?
2. Какими средствами решения задач регрессии обладает MathCAD?
3. Как оценит погрешность интерполяции? Регрессии?
4. В чем идея сплайн-интерполяции?
5. Связана ли интерполяционная точность с количеством точек разбиения? Как?
6. Какими средствами решения задач интерполяции обладает MathCAD?
7. Связана ли интерполяционная точность с величиной шага сетки?

10. ОБЫКНОВЕННЫЕ ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ

Большинство научно-технических задач в области электроники и наноэлектроники (особенно, относящихся к анализу динамических систем и их математическому моделированию) базируются на решении систем обыкновенных дифференциальных уравнений (ОДУ). Практически любой физический закон, включающее величины, *изменяющиеся с течением времени* содержит информацию о *скорости изменения* этих величин (производные), а следовательно описывается дифференциальными уравнениями.

Дифференциальным уравнением (ДУ) называется уравнение, включающее в себя не только неизвестные функции, но и их производные относительно некоторых свободных переменных.

Решить дифференциальное уравнение – это записать в явном виде все функции, входящие в ДУ и все их производные, сводящие уравнение к тождеству. Иногда решения ищутся в виде *явных* формул, но чаще их удается представить лишь *в приближенном виде* или же получить о них качественную информацию. Часто бывает трудно установить, существует ли решение вообще, не говоря уже о том, чтобы найти его, в этой связи использование математических пакетов незаменимо.

Если в уравнение входит только первая производная, то оно называется ДУ *первого порядка*, если еще и вторая производная, то – второго, и т.д.

Если уравнение имеет производные относительно только одной *свободной переменной*, то оно называется ***обыкновенным дифференциальным уравнением*** (ОДУ), а если оно включает в себя производные по различным переменным, то называется ***дифференциальным уравнением в частных производных*** (ДУЧП). В этом курсе ДУЧП не рассматриваются.

Общий вид ОДУ n-ного порядка: $F\left(t, y(t), \frac{dy(t)}{dt}, \frac{d^2y(t)}{dt^2}, ..., \frac{d^ny(t)}{dt^n}\right) = 0$. (59)

Рассмотрим для начала ОДУ первого порядка:

$$F\left(t, y(t), y'(t)\right) = 0, \text{ где } y'(t) = \frac{dy(t)}{dt}. \quad (60)$$

Если в уравнении (60) удается явно выразить производную $y'(t)$ и вынести ее в левую часть, то такое дифференциальное уравнение называется *явно заданным*:

$$y'(t) = f(t, y(t)). \quad (61)$$

Процесс решения дифференциального уравнения называется ***интегрированием*** и это понятно, ведь для нахождения первообразной функции по известной производной требуется ее проинтегрировать. Но этим решение ДУ *не ограничивается (!)*, поскольку известно, что первообразную можно найти только *с точностью до константы*:

$$\int\left(\frac{dy(t)}{dt}\right)dt = y(t) + C. \quad (62)$$

Таким образом, решением задачи (60) является не одна функция $y(t)$, а целое *семейство* таких функций, отличающихся константой (а в некоторых случаях – целый набор таких семейств функций). Каждая отдельная функция называется *интегральной кривой* (или *частным решением*), в то время как зависимость описывающая *все (!)* возможные частные решения – называется *общим решением*.

Решить ДУ, это значит описать именно общее решение, однако на практике, в задачах электродинамики и т. п., зачастую требуется выбрать из семейства решений одно единственное, обладающее *дополнительными свойствами*. Эти свойства позволяют задаться точным значением константы C из (62). Такая постановка задачи называется *задачей Коши*, а дополнительные свойства частного решения как правило задаются *начальными условиями*.

Задача Коши для ОДУ первого порядка формулируется таким образом:

Найти функцию $y(t)$, удовлетворяющую ОДУ $\frac{dy(t)}{dt} = f(t, y)$ (63)

и начальным условиям $y(t_0) = y_0$,

где y_0 – значение функции $y(t)$ в момент времени t_0.

Из курса высшей математики (раздел «Теория дифференциальных уравнений») известно, что ОДУ n-ного порядка (59) может быть без ограничения общности сведено к *системе n дифференциальных уравнений* первого порядка. Для этого достаточно произвести следующие обозначения (замены):

$$y_1(t) \equiv y(t),\ y_2(t) \equiv \frac{dy(t)}{dt},\ y_3(t) \equiv \frac{d^2y(t)}{dt^2},\ \ldots,\ y_n(t) \equiv \frac{d^{n-1}y(t)}{dt^n}. \quad (64)$$

И, подставив их в задачу Коши (63), получить n уравнений относительно n функций

$$\begin{cases} y_1'(t) = y_2(t); \\ y_2'(t) = y_3(t); \\ \ldots \\ y_{n-1}'(t) = y_n(t); \\ y_n'(t) = f(t, y_1, y_2, \ldots y_{n-1}). \end{cases} \text{ с начальными условиями } \begin{cases} y_1'(t_0) = y_{0,1}; \\ y_2'(t_0) = y_{0,2}; \\ \ldots \\ y_{n-1}'(t_0) = y_{0,n-1}; \\ y_n'(t_0) = y_{0,n}. \end{cases} \quad (65)$$

Проблема аналитического поиска решений задач (63) и (65) рассматривается в курсе высшей математики в разделе «Теория дифференциальных уравнений», а методы поиска численных решений задач (63) и (65) изучаются в разделе «Вычислительная математика».

10.1. Численные методы решения ОДУ

Численные методы решения ОДУ основаны на замене производных, входящих в уравнение, разностными функциями различного вида и поиске приближенных значений функции $y(t)$ на некотором наборе точек t_i, $i = 0..m$ – сетке. Интервал между соседними точками называется шагом интегрирования $\Delta t = t_{i+1} - t_i$. Значения функции $y(t)$ в точке t_0 задается начальными условиями задачи Коши $y(t_0) = y_0$.

Метод Эйлера.

Зададим интервал, на котором будем искать решение $[t_0, t_m]$, где m – число точек разбиения интервала на узлы сетки с шагом $\Delta t = (t_m - t_0)/m$. Построим сетку: $t_{i+1} = t_i + \Delta t$. Для каждого узла сетки t_{i+1} будем искать значение $y(t_{i+1})$ как некоторую рекуррентную зависимость от значений функции $y(t_i)$ в предыдущей точке t_i.

По определению, производная функции – это предел отношения приращения функции $y(t + \Delta t) - y(t)$ к приращению аргумента Δt, когда последний стремится к нулю. Заменим производную, входящую в (63) рекуррентным соотношением в соответствии с определением:

$$\frac{dy(t)}{dt} \approx \frac{y(t + \Delta t) - y(t)}{\Delta t}. \quad (66)$$

Запишем теперь Задачу Коши (63) в точках t_i и t_{i+1} сетки с учетом замены (66):

$$\frac{dy(t)}{dt} \approx \frac{y(t + \Delta t) - y(t)}{\Delta t} \approx \frac{y_{i+1} - y_i}{\Delta t} \approx f(t_i, y_i),\ i = 0,..,m-1. \quad (67)$$

Отсюда следует:

$$\begin{aligned} &y_{i+1} \approx y_i + f(t_i, y_i) \cdot \Delta t \\ &y(t_0) = y_0; \\ &t_{i+1} = t_i + \Delta t, \quad i = 0,..,m-1 \end{aligned} \qquad (68)$$

Выражение (68) называется *явным методом Эйлера*. Решим в *MathCAD* простейшее дифференциальное уравнение $y'(t) = t \cdot y(t)$ по методу (68) на участке $[0,1]$ с начальными условиями $y(0) = 1$. Аналитическое решение заданного ОДУ известно:

$$y(t) = C \cdot e^{\frac{t}{2}}.$$

С учетом начальных условий вычислим C= 1. Точное решение позволит оценить погрешность данного численного метода, которая, в общем, пропорциональна $(\Delta t)^2$.

Листинг 74. Решение ОДУ по явному методу Эйлера

$n := 10$

$\Delta t := \frac{(1-0)}{n}$ $\quad \Delta t = 0.1$

$f(t,y) := t \cdot y$

$i := 0..n-1$

$\begin{pmatrix} t_0 \\ y_0 \end{pmatrix} := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ $\quad \begin{pmatrix} t_{i+1} \\ y_{i+1} \end{pmatrix} := \begin{pmatrix} t_i + \Delta t \\ y_i + \Delta t \cdot f(t_i, y_i) \end{pmatrix}$ $\quad z(t) := e^{\frac{t^2}{2}}$

t =

0	0
1	0.1
2	0.2
3	0.3
4	0.4
5	0.5
6	0.6
7	0.7
8	0.8
9	0.9
10	1

y =

0	1
1	1
2	1.01
3	1.03
4	1.061
5	1.104
6	1.159
7	1.228
8	1.314
9	1.419
10	1.547

Вектор t содержит узлы сетки, а вектор y – найденные значения функции $y(t_i)$ в узловых точках сетки. Построим эти точки на графике и для сравнения – значения аналитического решения $z(t_i)$ заданного ОДУ в этих же точках:

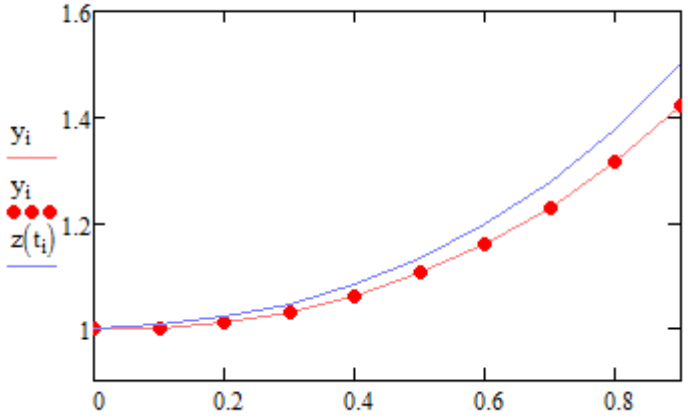
Листинг 75. Погрешность решения ОДУ по явному методу Эйлера

$z(t_i) =$

1
1.005
1.02
1.046
1.083
1.133
1.197
1.278
1.377
1.499

$|y_i - z(t_i)| =$

0
$5.013 \cdot 10^{-3}$
0.01
0.016
0.022
0.03
0.038
0.049
0.063
0.08



Анализ полученных результатов подтверждает, что ошибка увеличивается квадратично с ростом t. Для уменьшения погрешности следует уменьшать шаг интегрирования или применять другие итерационные методы решения ОДУ.

Модифицированный метод Эйлера:

$$y_{i+1} = y_i + f\left(t_i + \frac{\Delta t}{2}, y_i + \frac{\Delta t \cdot f(t_i, y_i)}{2}\right) \cdot \Delta t, \quad . \qquad (69)$$
$$t_{i+1} = t_i + \Delta t, \quad i = 0,..,m-1$$

Метод Рунге-Кутты четвертого порядка:

$$y_{i+1} = y_i + \frac{k(t_i, y_i)}{6}, \quad k_1(t,y) = \Delta t \cdot f(t,y), \quad k_2(t,y) = \Delta t \cdot f\left(t + \frac{\Delta t}{2}, y + \frac{k_1(t,y)}{2}\right),$$
$$k_3(t,y) = \Delta t \cdot f\left(t + \frac{\Delta t}{2}, y + \frac{k_2(t,y)}{2}\right), \quad k_4(t,y) = \Delta t \cdot f\left(t + \Delta t, y + k_3(t,y)\right), \quad . \qquad (70)$$
$$k(t,y) = k_1(t,y) + 2k_2(t,y) + 2k_3(t,y) + k_4(t,y);$$
$$t_{i+1} = t_i + \Delta t, \quad i = 0,..,m-1.$$

Встроенные методы решения ОДУ

В вычислительное ядро встроенной функции решения ОДУ – **Odesolve** заложен метод Рунге-Кутты четвертого порядка (70). Данная функция имеет следующий формат:

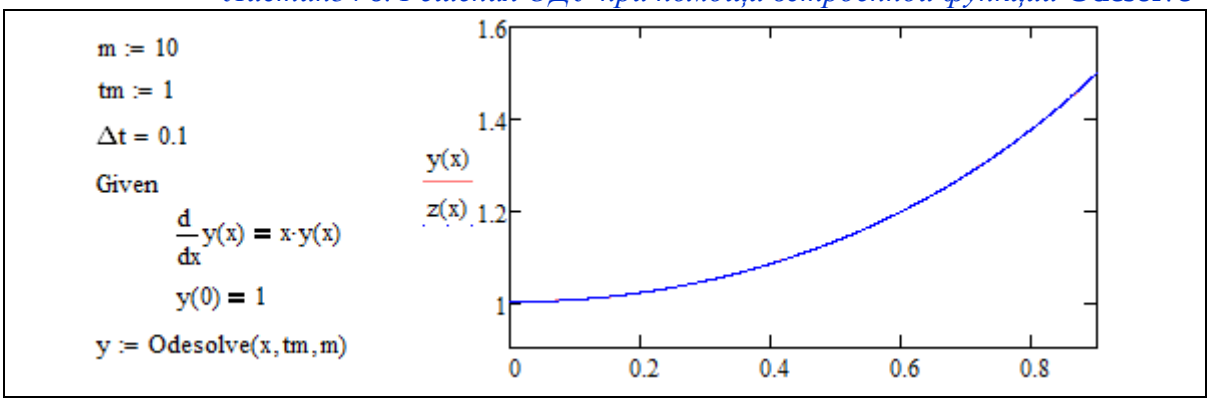
Given

ОДУ или система ОДУ *Равенства должны быть записаны в виде*
Начальные условия *логических операторов*

Odesolve(*t*, *tm*, *m*)

Здесь *t* – имя переменной, относительно которой решается уравнение, *tm* – конец интервала интегрирования, *m* – число шагов интегрирования.

Листинг 76. Решения ОДУ при помощи встроенной функции **Odesolve**



Визуально точное и численное решения полностью совпали. Читателю предлагается самостоятельно вычислить погрешность данного метода по аналогии с примером выше (см. *Листинг 75*).

Встроенные методы решения систем ОДУ

Для численного решения систем ОДУ (65) в MathCAD введен большой набор функций:

Функция MathCAD	*Численные методы*
rkfixed(*y*0, *t*0, *tm*, *m*, *D*)	Метод Рунге-Кутты с постоянным шагом.
rkadapt(*y*0, *t*0, *tm*, *m*, *D*)	Метод Рунге-Кутты с переменным шагом. В зависимости от скорости изменения функции шаг интегрирования автоматически «подбирается».
bulstoer(*y*0, *t*0, *tm*, *m*, *D*)	Метод Булирша-Штера – позволяет получать более точные решения, чем *rkfixed*, затрачивая на это меньшее число шагов (для гладких, медленно меняющихся систем)
BDF(*y*0, *t*0, *tm*, *m*, *D*)	Неявный многошаговый метод для решения жестких,

	быстро изменяющихся систем ОДУ
AdamsBDF(*y*0, *t*0, *tm*, *m*, *D*)	Комбинированный метод, использующий BDF для жестких систем и метод Адамса – для нежестких
Stiffr(*y*0, *t*0, *tm*, *m*, *D*, *AJ*)	Метод Розенброка с расширенной функцией Якоби *AJ* для жестких систем

Здесь *t*0, *tm* – границы интервала интегрирования, *y*0 – вектор начальных условий (65), *m* – число шагов интегрирования. *D* – векторная функция размера *N*×1 скалярного аргумента *t* и векторного *y*, причем, вектор *y*0 и искомая функция *y*(*t*) так же имеют размерность *N*×1.
N – порядок системы уравнений.

Расширенная матрица Якоби AJ имеет размерность *N*×(*N*+1). Её первый столбец содержит производные $\partial D_i / \partial t$, остальные строки и столбцы представляют собой матрицу Якоби $\partial D_i / \partial y_k$ системы ОДУ. Например, если

$$D(t,y)=\begin{pmatrix} ty_1 \\ -2y_1y_2 \end{pmatrix}, \text{ то } AJ(t,y)=\begin{pmatrix} y_1 & t & 0 \\ 0 & -2y_2 & -2y_1 \end{pmatrix}.$$

Каждая из приведенных функций возвращает решение в виде матрицы (*m*+1)×(*N*+1): в ее левом столбце находятся значения узлов сетки t_i, а в остальных *N* столбцах – значения искомых функций $y_1(t)$, $y_2(t)$, …, $y_N(t)$, рассчитанные в этих узлах. Поскольку всего точек помимо начальной *m*, то строк в матрице – *m*+1.

10.2. Переходный процесс в электрической схеме

Переходные процессы – это процессы, возникающие в электрических цепях при различных воздействиях, приводящих к изменению их режима работы, то есть при действии различного рода коммутационной аппаратуры, например, ключей, переключателей для включения или отключения источника или приёмника энергии, при обрывах в цепи, при коротких замыканиях отдельных участков цепи и т. д.

Физическая причина возникновения переходных процессов в цепях — наличие в них *катушек индуктивности и конденсаторов*, то есть *индуктивных и ёмкостных элементов* в соответствующих схемах замещения. Объясняется это тем, что энергия магнитного и электрического полей этих *реактивных элементов* не может изменяться скачком при коммутации (процесс замыкания или размыкания выключателей) в цепи.

Реактивные элементы – элементы, способные накапливать электрическую энергию и отдавать ее либо источнику, от которого эта энергия была получена, либо передавать другому элементу. В любом случае этот элемент не превращает электрическую энергию в тепловую.

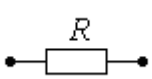
Переходный процесс в цепи описывается системой дифференциальных уравнений. Переходный процесс бывает *неоднородным*, если схема замещения цепи содержит источники ЭДС и тока, и *однородным* – если не содержит. Переходный процесс называется *линейным (нелинейным)* для линейной (нелинейной) электрической цепи.

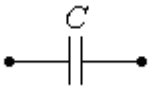
Подчеркнем еще раз, что *переходные процессы не могут протекать мгновенно*, так как невозможно в принципе мгновенно изменять энергию, накопленную в электромагнитном поле цепи. Теоретически переходные процессы заканчиваются за время t→∞. Практически же переходные процессы являются быстропротекающими, и их длительность обычно составляет доли секунды. Так как энергия магнитного W_M и электрического полей $W_Э$ описывается выражениями

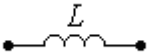
$$W_M = L\cdot\frac{i^2}{2} \quad \text{и} \quad W_Э = C\cdot\frac{u^2}{2}, \tag{71}$$

то ток в индуктивности и напряжение на емкости не могут изменяться мгновенно. На этом основаны законы изменения тока и напряжения.

Законы изменения тока и напряжения в активных (резистор) реактивных (катушка индуктивности и конденсатор) элементах описываются формулами (72):





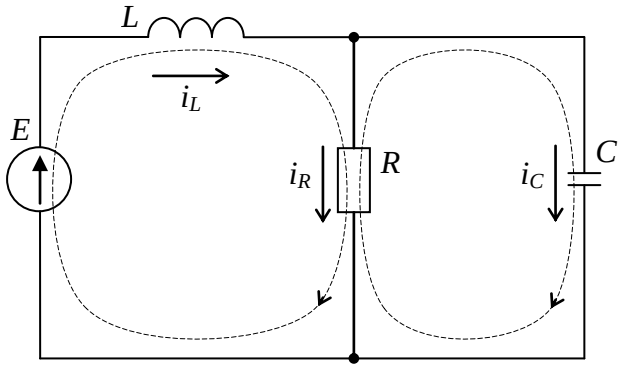


$$i_R(t)=\frac{u_R(t)}{R} \qquad i_C(t)=C\frac{du_C(t)}{dt} \qquad i_L(t)=\frac{1}{L}\int u_L(t)dt \qquad (72)$$

$$u_R(t)=R\cdot i_R(t) \qquad u_C(t)=\frac{1}{C}\int i_C(t)dt \qquad u_L(t)=L\frac{di_L(t)}{dt}$$

С использованием известных законов Ома и Кирхгофа, с учетом формул (72), строится система обыкновенных дифференциальных уравнений, описывающая процесс изменения токов и напряжений в реактивных элементах – *переходный процесс от начальных условий X_0 до установившегося режима работы*.

В качестве примера решения ОДУ рассчитаем переходный процесс в электрической схеме, приведенной на *Рис. 62*.



$L = 0.01\ Гн$
$C = 10^{-6}\ Ф$
$R = 100\ Ом$
$E = 100\ В$

Рис. 62 – Схема замещения

Система дифференциальных уравнений, описывающих данную схему строится на основе законов (26)-(28), а так же зависимостей относительно тока и напряжения в L и C (72).

Выражая из полученных зависимостей относительно I и II законов Кирхгофа токи в активных элементах (резисторах) до тех пор, пока не останется столько дифференциальных уравнений, сколько реактивных элементов в схеме (в нашем примере – 2).

$$\begin{cases} \dfrac{du_C}{dt}=\dfrac{1}{C}\left(i_L-\dfrac{u_C}{R}\right); \\ \dfrac{di_L}{dt}=\dfrac{1}{L}\left(E-u_C\right). \end{cases} \quad X(t)=\begin{bmatrix} i_L(t) \\ u_C(t) \end{bmatrix}, \quad X(0)=\begin{bmatrix} 0 \\ 0 \end{bmatrix}. \qquad (73)$$

После того, как система ОДУ выписана в явном виде – в левой части только производные, в правой (73) – все остальное, строится матрица системы $D(t,x)$. И решается система ОДУ изученными ранее методами.

Решим полученную систему дифференциальных уравнений первого порядка с заданными начальными условиями, результат вычислений будет представлять собой графики переходного процесса тока и напряжения в электрической схеме, приведенной на *Рис. 62*. Решение будем искать численно по методу Рунге-Кутта.

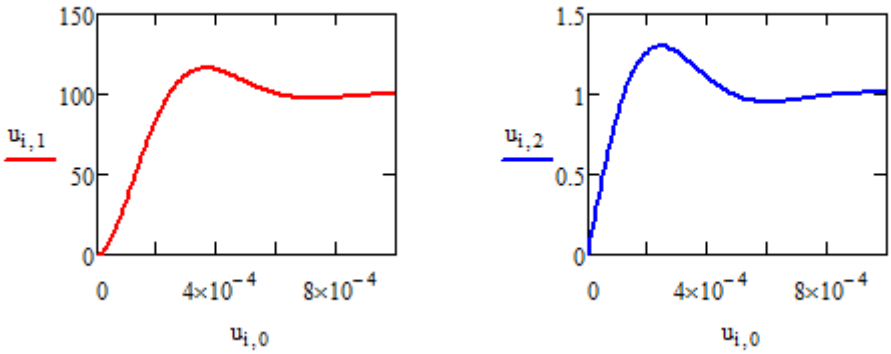
Полученный результат будет представлять собой массив точек, который, при необходимости, можно сгладить методами интерполяции.

Листинг 77. Решения ОДУ. Построение переходного процесса

$L := 0.01 \quad C := 10^{-6} \quad R := 100 \quad E := 100 \quad m := 100$

$$D(t,x) := \begin{pmatrix} \dfrac{x_1 - \dfrac{x_0}{R}}{C} \\ \dfrac{E - x_0}{L} \end{pmatrix} \qquad x0 := \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad u := \mathrm{rkfixed}\left(x0, 0, 10^{-3}, m, D\right)$$

$i := 0..m - 1$



В данном примере переходный процесс имеет перерегулирование и завершается в течении приблизительно $2*10^{-4}$ с, сходясь к постоянному значению тока и напряжения, которые не имеют гармонических колебаний.

Контрольные вопросы:

1. Какое уравнение называется дифференциальным?
2. Что понимается под термином решение дифференциального уравнения?
3. Что такое задача Коши?
4. Как влияют начальные условия на решение дифференциального уравнения?
5. Что такое общее (полное) решение дифференциального уравнения?
6. Что является частным решением задачи Коши?
7. Какие численные методы решения ДУ вы знаете? В чем их смысл?
8. Что такое интегральная кривая?
9. Какое уравнение называется обыкновенным дифференциальным уравнением?
10. Что называется дифференциальным уравнением в частных производных?
11. Что такое система дифференциальных уравнений?
12. Что является решением системы дифференциальных уравнений?
13. Как задаются начальные условия для системы дифференциальных уравнений?
14. Как связана система дифференциальных уравнений первого порядка и дифференциальное уравнений высшего порядка?
15. В чем суть численных методов решения ДУ?
16. Что такое переходный процесс электрической цепи?
17. Могут ли переходные процессы в электрической схеме протекать мгновенно? Почему?
18. Как влияют реактивные элементы схемы на дляительность переходного процесса?

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ПРОВЕРОЧНЫХ РАБОТ

Для студентов направления *11.03.04 Электроника и наноэлектроника (профиль - Промышленная электроника)* и *09.03.01 Информатика и вычислительная техника (профиль - Микропроцессорные системы обработки информации и управления)* с Томского университета систем управления и радиоэлектроники данный курс входит в образовательную программу и содержит следующие дидактические единицы:

- лекционную составляющую (*аудиторно*);
- практические занятия (*аудиторно*);
- самостоятельная работа студентов (*электронный курс*).

Учебно-методический комплект дидактических (УМКД) материалов по настоящему курсу включает:

- настоящее учебное пособие *«Введение в профессию: Учебное пособие / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019»*
- *«Введение в профессию: Учебно-методическое пособие для проведения практических занятий / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019»* [5].
- *«Введение в профессию: Учебно-методическое пособие для организации самостоятельной работы студентов / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019»* [6].
- электронный курс (в среде *moodle*), размещенный на сайте ТУСУР.

Лекции №№ 1-7 завершаются выполнением ***практических работ*** №№ 1-7, содержание которых, а также описание с примерами решения задач и индивидуальные задания для студентов содержит одноименное учебно-методическое пособие [5].

Материал, излагаемый в лекциях №№ 8-10 не подразумевает выполнения практических работ в данном учебном курсе, однако, может использоваться в качестве дополнительной литературы при проведении занятий по дисциплинам «Математика», «Теоретические основы электротехники» и «Теория автоматического управления».

Организация и контроль за проведением ***самостоятельной работы студентов*** по настоящей дисциплине обеспечивает ***электронный курс***, выполненный в среде *moodle*), размещенный на сайте ТУСУР. Доступ к материалам электронного курса студенты получают автоматически в своем личном кабинете.

Также организации проведения самостоятельной работы студентов сопутствует учебно-методическое пособие по СРС [6].

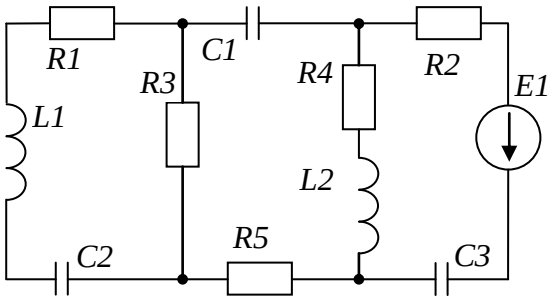
ЛИТЕРАТУРА

1. Mathematica. Система компьютерной алгебры компании Wolfram Research. Официальный сайт компании Wolfram Research http://www.wolfram.com. Способ доступа: http://www.wolfram.com/mathematica/.
2. Maple. Программный пакет компьютерной алгебры компании Waterloo Maple Inc. Официальный сайт: http://www.maplesoft.com/. Способ доступа: http://www.maplesoft.com/products/Maple/index.aspx.
3. MatLab. Пакет математических и инженерных вычислений. Официальный сайт компании-разработчика MathWorks http://www.mathworks.com/. Способ доступа: http://www.mathworks.com/products/matlab/.
4. MathCAD. Система компьютерных вычислений. Официальный сайт компании-разработчика Mathsoft http://www.mathsoft.com/, в составе PTC Community http://communities.ptc.com. Способ доступа: http://www.mathcad.com/, http://communities.ptc.com/community/mathcad
5. Михальченко, Сергей Геннадьевич. Введение в профессию: Учебно-методическое пособие по проведению практических работ / С.Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019. – 102 с. : ил., табл., прил. – Библиогр.: с. 96.
6. Михальченко, Сергей Геннадьевич. Введение в профессию: Учебно-методическое пособие по проведению практических работ / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019. – 87 с. : ил., табл. – Библиогр.: с. 86.
7. Дьяконов В. П. Mathematica 5/6/7. Полное руководство. — М.: «ДМК Пресс», 2009. — С. 624. — ISBN 978-5-94074-553-2
8. Дьяконов В. П. Mathematica 5.1/5.2/6 в математических и научно-технических расчетах. Изд-е второе дополненное и переработанное. — М.: «СОЛОН-Пресс», 2008. — С. 744. — ISBN 978-5-91359-045-9
9. Говорухин В. Н., Цибулин В. Г. Введение в Maple. Математический пакет для всех. — М.: Мир, 1997. — С. 208. — ISBN 5-03-003255-X
10. Дьяконов В. П. Maple 9.5/10 в математике, физике и образовании. — М.: СОЛОН Пресс, 2006. — С. 720. — ISBN 5-98003-258-4
11. Васильев А. Н. Maple 8. Самоучитель. — М.: Диалектика, 2003. — С. 352. — ISBN 5-8459-0452-8
12. Матросов А. В. Maple 6: Решение задач высшей математики и механики: Практическое руководство. 2001 г. 528 с. ISBN 5-94157-021-X
13. Дьяконов В. П. MATLAB R2006/2007/2008 + Simulink 5/6/7. Основы применения. Изд-е 2-е, переработанное и дополненное. Библиотека профессионала. — Москва.: «СОЛОН-Пресс», 2008. — С. 800. — ISBN 978-5-91359-042-8
14. Дьяконов В. П. MATLAB и SIMULINK для радиоинженеров. — Москва.: «ДМК-Пресс», 2011. — С. 976. — ISBN 978-5-94074-492-4
15. Курбатова Екатерина Анатольевна. MATLAB 7. Самоучитель. — М.: «Диалектика», 2005. — С. 256. — ISBN 5-8459-0904-X

16. Джон Г. Мэтьюз, Куртис Д. Финк. Численные методы. Использование MATLAB = Numerical Methods: Using MATLAB. — 3-е изд. — М.: «Вильямс», 2001. — С. 720. — ISBN 0-13-270042-5
17. В.Ф. Очков. MathCAD 14 для студентов и инженеров. СПб.: BHV, 2009.
18. А. Васильев. MathCAD 13 на примерах (+ CD-ROM). С-Пб: БХВ-Петербург, 2006, 512с. ISBN: 5-94157-880-6.
19. Д. В. Кирьянов. MathCAD 12. Наиболее полное руководство (+ CD-ROM). С-Пб: БХВ-Петербург, 2005, 566 с. ISBN: 5-94157-407-X.
20. Дьяконов В. MathCAD 2001: Учебный курс. — СПб.: Питер, 2001.
21. А. Черняк, Ж. Черняк, Ю. Доманова. Высшая математика на базе MathCAD. Общий курс С-Пб.:БХВ-Петербург, 2004.
22. В.А. Охорзин. Прикладная математика в системе MATHCAD Учебное пособие. 3-е изд. СПб.: Лань, 2009, 352с. ISBN: 978-5-8114-0814-6.
23. Р.Ивановский. Теория вероятностей и математическая статистика. Основы, прикладные аспекты с примерами и задачами в среде MathCAD. М.: БХВ-Петербург, 2008, 528с. ISBN978-5-9775-0199-6.
24. Е. Р. Алексеев, О. В. Чеснокова. Решение задач вычислительной математики в пакетах MathCAD 12, MATLAB 7, Maple 9. М: НТ Пресс, 2006, 496с. ISBN: 5-477-00208-5.
25. В. В. Фриск. MathCAD. Расчеты и моделирование цепей на ПК. Москва: Солон-Пресс, 2006. ISBN: 5-98003-242-8.
26. Д. В. Кирьянов, Е. Н. Кирьянова. Вычислительная физика (с курсом лекций на CD) +. М.: Полибук Мультимедиа, 2006.
27. О.Гольдберг. Переходные процессы в электрических машинах и аппаратах и вопросы их проектирования. М.: Высшая школа. 2001.
28. С.Поршнев. Компьютерное моделирование физических процессов с использованием пакета MathCAD. Учебное пособие. М.: Горячая линия - Телеком, 2002.
29. М.Панько. Расчет и моделирование автоматических систем регулирования в среде MathCAD. Издательство МЭИ. 2001.
30. Чарльз Генри Эдвардс, Дэвид Э. Пенни. Дифференциальные уравнения и проблема собственных значений: моделирование и вычисление с помощью Mathematica, Maple и MATLAB = Differential Equations and Boundary Value Problems: Computing and Modeling. — 3-е изд. — М.: «Вильямс», 2007. — ISBN 978-5-8459-1166-7
31. Коновалов, Борис Игоревич. Теоретические основы электротехники. Часть 1 [Электронный ресурс] : Учебное методическое пособие / Б. И. Коновалов; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники. – Электрон. текстовые дан. – Томск : [б. и.], 2006. – on-line, 75 с. Способ доступа: http://ie.tusur.ru/content.php?id=444
32. Коновалов, Борис Игоревич. Теоретические основы электротехники. Часть 1 [Электронный ресурс] : Учебное пособие / Б. И. Коновалов; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники. – Электрон. текстовые дан. – Томск : [б. и.], 2006. – on-line, 145 с. Способ доступа: http://ie.tusur.ru/content.php?id=444

Приложение 1

РАСЧЕТ ЦЕПИ ПЕРЕМЕННОГО ТОКА. ПРИМЕР



E1=360 [В]	*ω= 314 [рад/с]*
R1= 30 [Ом]	*L1= 0,005 [Гн]*
R2= 50 [Ом]	*L2= 0,055 [Гн]*
R3= 10 [Ом]	*C1= 0,5 [Ф]*
R4= 80 [Ом]	*C2= 0,025 [Ф]*
R5= 30 [Ом]	*C3= 0,01 [Ф]*

Сопротивления реактивных элементов цепи равны:

$$X_C = \frac{1}{\omega \cdot C}, \qquad X_L = \omega \cdot L \quad [\text{Ом}] \qquad (46)\ (49)$$

Зададим параметры схемы

E1 := 360 [В] ω := 3.14 [рад/сек]

R1 := 30 [Ом] R2 := 50 [Ом] R3 := 10 [Ом] R4 := 80 [Ом] R5 := 30 [Ом]

L1 := 0.005 [Гн] L2 := 0.055 [Гн]

C1 := 0.5 [Ф] C2 := 0.025 [Ф] C3 := 0.01 [Ф]

Вычислим сопротивления реактивных элементов

$XC1 := \frac{1}{\omega \cdot C1}$ XC1 = 0.637 [Ом] XL1 := ω·L1 XL1 = 0.016 [Ом]

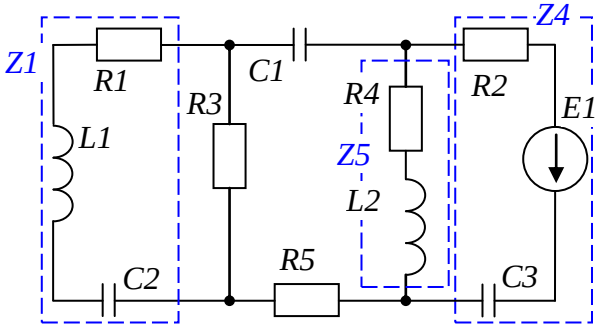
$XC2 := \frac{1}{\omega \cdot C2}$ XC2 = 12.739 [Ом] XL2 := ω·L2 XL2 = 0.173 [Ом]

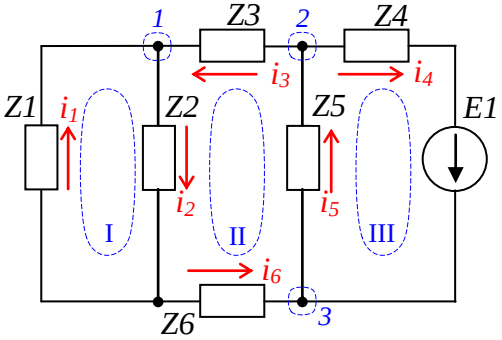
$XC3 := \frac{1}{\omega \cdot C3}$ XC3 = 31.847 [Ом]

Комплексное сопротивление Z для участка цепи переменного тока (с частотой ω), обладающего активным сопротивлением *R*, индуктивностью *L* и емкостью *C* в общем случае может быть записано в виде:

$$Z = Z_m \cdot e^{\mathbf{i}\cdot\omega} = R + \mathbf{i}\cdot\left(\omega \cdot L - \frac{1}{\omega \cdot C}\right) = R + \mathbf{i}\cdot\left(X_L - X_C\right) \ [\text{Ом}]. \qquad (50)$$

Вычислим комплексные сопротивления *Z1..Z6* для всех участков цепи:





$$Z1 = R1 + \mathbf{i}\cdot\omega\cdot L1 - \frac{\mathbf{i}}{\omega\cdot C2};\qquad Z2 = R3;\qquad Z3 = -\frac{\mathbf{i}}{\omega\cdot C2};$$

$$Z4 = R2 - \frac{\mathbf{i}}{\omega\cdot C3};\qquad Z5 = R4 + \mathbf{i}\cdot\omega\cdot L2;\qquad Z6 = R5$$

Произвольно выберем направления протекания токов $i_1..i_6$ в сегментах схемы (на рисунке выше обозначены красными стрелками) и направления обхода контуров. По законам Кирхгофа в выбранных контурах и узлах цепи выпишем уравнения для поиска токов $i_1..i_6$:

$$\begin{cases}(I): & i_1\cdot Z1 + i_2\cdot Z2 = 0\\ (II): & i_2\cdot Z2 + i_3\cdot Z3 + i_5\cdot Z5 + i_6\cdot Z6 = 0\\ (III): & i_5\cdot Z5 + i_4\cdot Z4 = E1\end{cases}\qquad \begin{cases}(1): & i_1 - i_2 + i_3 = 0\\ (2): & -i_3 - i_4 + i_5 = 0\\ (3): & i_4 - i_5 + i_6 = 0\end{cases}$$

Запишем матрицу комплексных сопротивлений системы Z и вектор внешних воздействий E:

$$Z := \begin{pmatrix} R1 + i\cdot\omega\cdot L1 + \frac{-i}{\omega\cdot C2} & R3 & 0 & 0 & 0 & 0\\ 0 & R3 & \frac{-i}{\omega\cdot C1} & 0 & R4 + i\cdot\omega\cdot L2 & R5\\ 0 & 0 & 0 & R2 + \frac{-i}{\omega\cdot C3} & R4 + i\cdot\omega\cdot L2 & 0\\ 1 & -1 & 1 & 0 & 0 & 0\\ 0 & 0 & -1 & -1 & 1 & 0\\ 0 & 0 & 0 & 1 & -1 & 1 \end{pmatrix}\qquad E := \begin{pmatrix} 0\\ 0\\ E1\\ 0\\ 0\\ 0\end{pmatrix}$$

Найдем решение системы уравнений – значения токов $i_1..i_6$, протекающих в сегментах схемы в виде комплексных значений [А]:

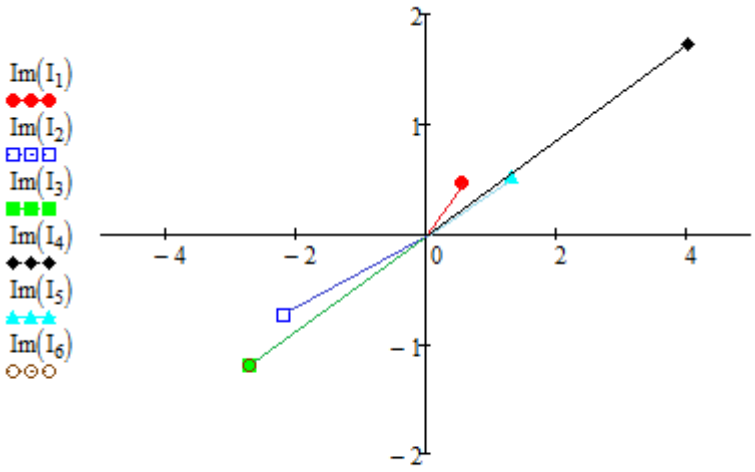
$$I := Z^{-1}\cdot E\qquad I = \begin{pmatrix} 0.5296826 + 0.4697468i\\ -2.1867138 - 0.7353172i\\ -2.7163964 - 1.2050641i\\ 4.0191 + 1.7244382i\\ 1.3027035 + 0.5193741i\\ -2.7163964 - 1.2050641i\end{pmatrix}$$

$$i1(t) := |I_1|\cdot\sin(\omega\cdot t + \arg(I_1))\qquad i4(t) := |I_4|\cdot\sin(\omega\cdot t + \arg(I_4))$$

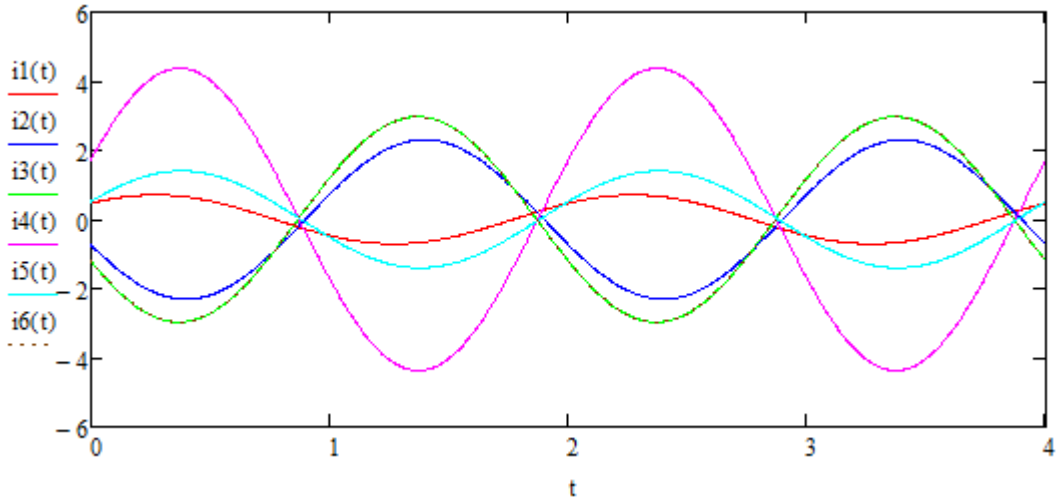
$$i2(t) := |I_2|\cdot\sin(\omega\cdot t + \arg(I_2))\qquad i5(t) := |I_5|\cdot\sin(\omega\cdot t + \arg(I_5))$$

$$i3(t) := |I_3|\cdot\sin(\omega\cdot t + \arg(I_3))\qquad i6(t) := |I_6|\cdot\sin(\omega\cdot t + \arg(I_6))$$

На рисунке ниже изображена векторная диаграмма токов:



Изобразим токи в виде функции от времени и выведем их на график [A]:



Проверим правильность нахождения токов по первому закону Кирхгофа:

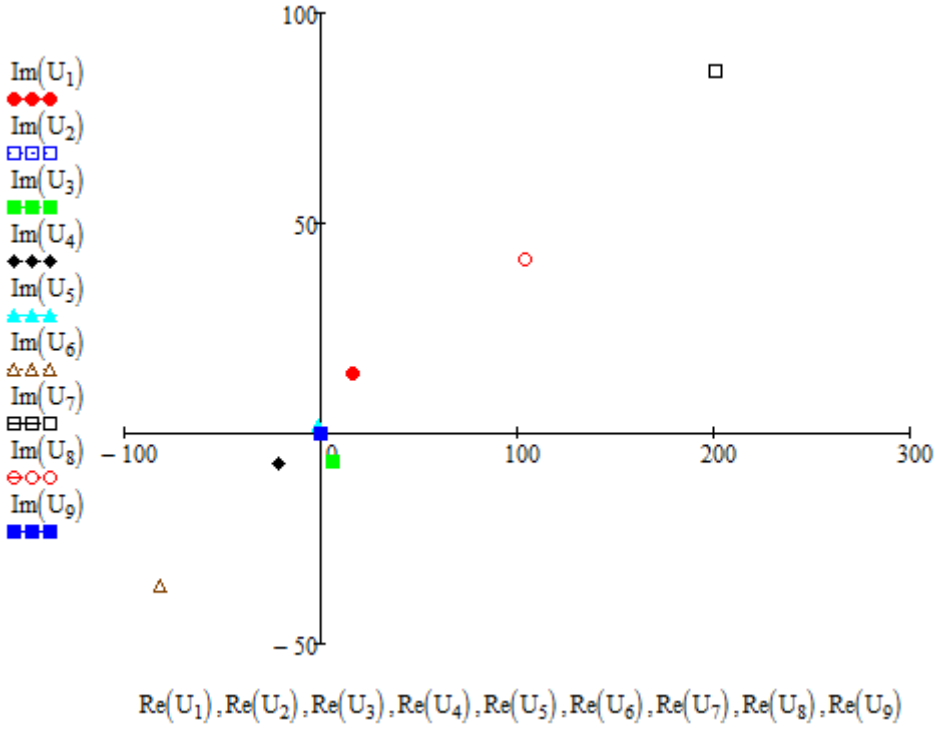
$$I_1 - I_2 + I_3 = 0$$
$$-I_3 - I_4 + I_5 = 0$$
$$-I_1 + I_2 - I_6 = 0$$
$$I_4 - I_5 + I_6 = 0$$

$$tt := 1.7$$
$$i1(tt) - i2(tt) + i3(tt) = 0$$
$$-i3(tt) - i4(tt) + i5(tt) = 0$$
$$-i1(tt) + i2(tt) - i6(tt) = 0$$
$$i4(tt) - i5(tt) + i6(tt) = 0$$

Можно видеть, что сумма токов в узлах цепи равна нулю для любого момента времени (здесь произвольно выбрано *tt=1.7 c*).

Вычислим напряжения на активных и реактивных элементах цепи переменного тока [B]:

$$U := \begin{pmatrix} R1 \cdot I_1 \\ i \cdot \omega \cdot L1 \cdot I_1 \\ \frac{-i}{\omega \cdot C2} \cdot I_1 \\ R3 \cdot I_2 \\ \frac{-i}{\omega \cdot C1} \cdot I_3 \\ R5 \cdot I_6 \\ R2 \cdot I_4 \\ R4 \cdot I_5 \\ i \cdot \omega \cdot L2 \cdot I_5 \\ \frac{-i}{\omega \cdot C3} \cdot I_4 \end{pmatrix} \qquad U = \begin{pmatrix} 15.8904773654 + 14.0924050865i \\ -0.0073750253 + 0.0083160165i \\ 5.9840361302 - 6.7475487751i \\ -21.8671384702 - 7.3531723279i \\ -0.7675567319 + 1.7301888063i \\ -81.4918927759 - 36.1519220702i \\ 200.9549987222 + 86.2219088809i \\ 104.2162838865 + 41.549928689i \\ -0.0896959086 + 0.2249769028i \\ 54.9184132999 - 127.9968144727i \end{pmatrix}$$



Изобразим напряжения в виде функции от времени и выведем их на график [В]:

$$uE1(t) := E1 \cdot \sin(\omega \cdot t)$$

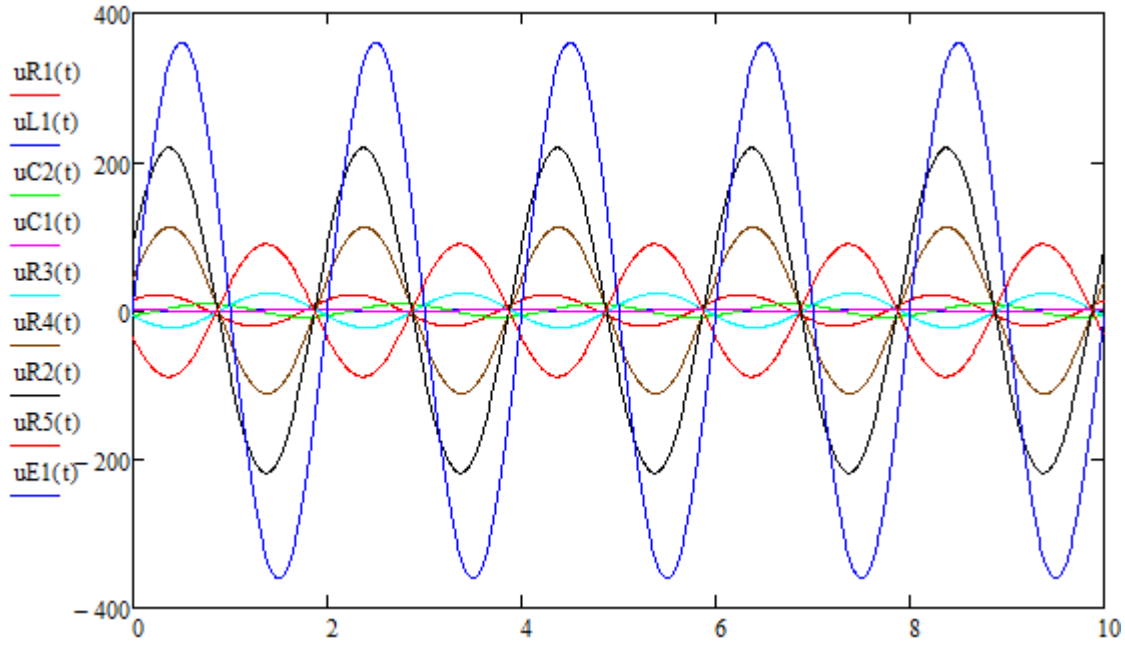
$$uR1(t) := |U_1| \cdot \sin(\omega \cdot t + \arg(U_1)) \qquad uR5(t) := |U_6| \cdot \sin(\omega \cdot t + \arg(U_6))$$

$$uL1(t) := |U_2| \cdot \sin(\omega \cdot t + \arg(U_2)) \qquad uR2(t) := |U_7| \cdot \sin(\omega \cdot t + \arg(U_7))$$

$$uC2(t) := |U_3| \cdot \sin(\omega \cdot t + \arg(U_3)) \qquad uR4(t) := |U_8| \cdot \sin(\omega \cdot t + \arg(U_8))$$

$$uR3(t) := |U_4| \cdot \sin(\omega \cdot t + \arg(U_4)) \qquad uL2(t) := |U_9| \cdot \sin(\omega \cdot t + \arg(U_9))$$

$uC1(t) := |U_5| \cdot \sin(\omega \cdot t + \arg(U_5))$ $uC3(t) := |U_{10}| \cdot \sin(\omega \cdot t + \arg(U_{10}))$



Проверим правильность нахождения напряжений по второму закону Кирхгофа:

$uR1(tt) + uL1(tt) + uR3(tt) + uC2(tt) = 0$

$uC1(tt) + uR3(tt) + uR5(tt) + uL2(tt) + uR4(tt) = 0$

$uR4(tt) + uL2(tt) + uR2(tt) + uC3(tt) - uE1(tt) = 0$

Можно видеть, что сумма напряжений в контурах цепи равна сумме ЭДС источника для произвольного момента времени.

Рассчитаем баланс мощностей:

$Sgen := E1 \cdot I_4$ $Sgen = 1446.875991 + 620.797744i$

$Ppotr := (|I_1|)^2 \cdot R1 + (|I_4|)^2 \cdot R2 + (|I_2|)^2 \cdot R3 + (|I_5|)^2 \cdot R4 + (|I_6|)^2 \cdot R5 = 1446.875991$

$Qpotr := (|I_1|)^2 \cdot \omega \cdot L1 + (|I_1|)^2 \cdot \frac{-1}{\omega \cdot C2} + (|I_3|)^2 \cdot \frac{-1}{\omega \cdot C1} + (|I_5|)^2 \cdot \omega \cdot L2 + (|I_4|)^2 \cdot \frac{-1}{\omega \cdot C3} = -620.797744$

Можно видеть, что полная мощность источника *Sgen* [ВА] в комплексном виде имеет две составляющие – активную (вещественная часть) и реактивную (мнимая часть). Активная мощность потребителей равна *Ppotr* [Вт], а реактивная мощность потребителей – *Qpotr* [ВАр].

Результаты расчетов демонстрируют полное совпадение потребленной и сгенерированной активной и реактивной мощностей. Это подтверждает правильность расчетов и неизменность закона сохранения энергии в природе.

Михальченко Сергей Геннадьевич

Введение в профессию: Учебное пособие / С. Г. Михальченко; Томский государственный университет систем управления и радиоэлектроники, Кафедра промышленной электроники – Томск: ТУСУР, 2019. – 117 с. : ил., табл., прил. – Библиогр.: с. 110.