МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ»
(ТУСУР)

УТВЕРЖДАЮ
Заведующий кафедрой ЭМИС
_____
И. Г. Боровской
«___» _____
2018 г.

Е.А. ШЕЛЬМИНА

ПРИКЛАДНАЯ ИНФОРМАТИКА

Методические указания по выполнению лабораторных и самостоятельных работ для студентов направлений 09.03.01 «Информатика и вычислительная техника» и 09.03.02 «Информационные системы и технологии»

2018

Шельмина Е.А. Прикладная инфорамтика: методическое пособие – Томск: Изд-во ТУСУР, 2018. – 35 с.

Методическое пособие для студентов ВУЗов посвящено изучению математических пакетов Smath Studio и Scilab. Описываются основные правила работы с математическими пакетами, особенности работы в Smath Studio и Scilab.

СОДЕРЖАНИЕ

Краткое содержание тем и результатов их освоения

Тема лабораторных занятий	Деятельность студента. Решая задачи, студент:
Структура окон Smath Studio и Scilab	• *изучает* интерфейс пакетов Smath Studio и Scilab;
Арифметические операции. Целые и рациональные числа, константы	• *знакомится* с основными арифметическими операциями, которые можно осуществлять в пакетах Smath Studio и Scilab; • *учится* выполнять простейшие математические вычисления в пакетах Smath Studio и Scilab;
Синтаксис команд. Стандартные функции	• *получает навыки* работы со стандартными функциями Smath Studio и Scilab; • *решает* самостоятельно задачи, базирующиеся на использовании стандартных функций Smath Studio и Scilab;
Матричные вычисления	• *получает навыки* обработки матриц в Studio и Scilab;
Преобразование математических выражений	• *знакомится* с основными командами математических пакетов для символьного преобразования алгебраических выражений; • *получает навыки* преобразования математических выражений в пакетах Smath Studio и Scilab;
Решение уравнений	• *получает навыки* решения уравнений с использованием математических пакетов; • *выполняет* индивидуальное задание на тему «Решение уравнений»;
Построение 2D и 3D графиков	• *получает* навыки работы по созданию двух- и трехмерных графиков в математических пакетах; • *создает* графики функций, зависящих от одной и нескольких переменных;
Дифференциальное и интегральное исчисление	• *получает навыки* нахождения производных функций и решения интегралов с использованием математических пакетов; • *выполняет* индивидуальное задание на тему «Дифференциальное и интегральное исчисление»;
Программирование в Smath Studio и Scilab	• *знакомится* с основными возможностями программирования с использованием математических пакетов Smath Studio и Scilab; • *выполняет* индивидуальное задание на тему «Программирование в Smath Studio и Scilab»;

Введение

Этот курс предназначен для практического изучения математических пакетов Smath Studio и Scilab. В методических указаниях основное внимание уделяется принципиальным моментам, которые необходимы для успешного выполнения практических и лабораторных работ.

Раздел 1. Лабораторные работы

Лабораторная работа №1
Структура окон Smath Studio и Scilab

Цель работы: изучение интерфейса пакетов Smath Studio и Scilab.

Структура окна Smath Studio

Программа SMath Studio предназначена для численного и аналитического решения математических задач (решения уравнений и систем, нахождения экстремумов функций, вычисления производных и интегралов, решения дифференциальных уравнений) [3].

Позволяет работать с формулами, текстами, графиками, а также выполнять программирование вычислительных процессов и производить расчеты как в численном, так и в аналитическом виде.

Программа состоит из 3 областей: основное меню, инструментальная панель, рабочее поле (рис. 1).

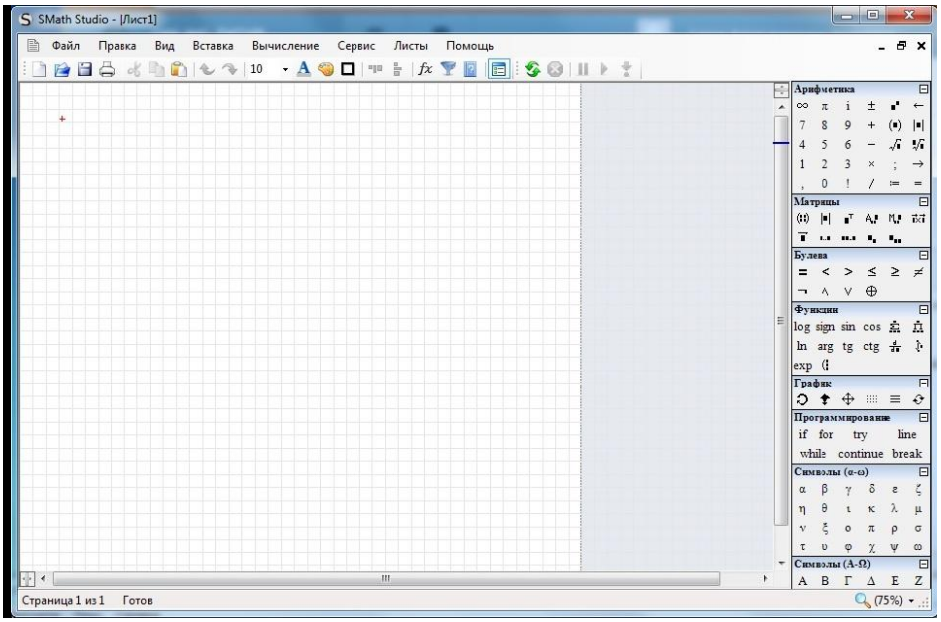


Рисунок 1 - Окно пакета Smath Studio

Основное меню состоит из основных команд для работы с документом в целом, такие как: вставить, вырезать, открыть, сохранить… а также содержит математический справочник и набор примеров [3].

Панель инструментов (ПИ) разделена по категориям:

а) панель «Арифметика» содержит цифры, математические символы, и основные операции:

– оператор присвоения «:=» служит для присвоения переменным каких-либо значений, численных либо символьных;

– оператор численного вычисления « = » служит для получения численного результата, он применим как к выражениям, так и к переменным;

– оператор символьного вычисления « → » позволяет вычислять символьный результат;

б) панель «Матрицы» содержит команды для работы с матрицами. Позволяет находить определитель матрицы, транспонировать ее, находить минор, а также содержит команду векторного умножения, потому что векторы программа рассматривает как матрицу с одним столбцом (или строкой);

в) панель «Булева» содержит набор для команд для булевой алгебры, а также позволяет задавать логические операции в командах ветвления и циклах;

г) панель «Функции» содержит набор часто используемых функций, таких как: sin, cos, log и т. п., а также 2 кнопки «2d» и «3d», эти кнопки позволяют вставить соответственно 2-мерные и 3-мерные графики;

д) панель «График» позволяет вращать, перемещать, увеличивать/уменьшать графики функций;

е) панель «Программирование» содержит 4 функции программирования, таких как: ветвление «IF», цикл с предусловием «WHILE», цикл со счетчиком «FOR», вспомогательная функция «LINE»;

ж) последние две панели называются одинаково «Символы» и содержат греческие символы.

Рабочее поле занимает самую большую часть программы, здесь выполняются все расчеты. Основным элементом поля является курсор ввода, он выглядит как красный крестик.

Структура окна Scilab

Рассмотрим основные компоненты окна программы Scilab.

После запуска Scilab на экране появиться основное окно приложения. Окно содержит меню, панель инструментов и рабочую область. Признаком того, что система готова к выполнению команды, является наличие знака приглашения -->, после которого расположен активный (мигающий) курсор. Рабочую область со знаком приглашения обычно называют *командной строкой*. Ввод команд в Scilab осуществляется с клавиатуры. Нажатие клавиши *Enter* заставляет систему выполнить команду и вывести результат (рис. 2).

Понятно, что все выполняемые команды не могут одновременно находиться в поле зрения пользователя. Поэтому, просмотреть ту информацию, которая покинула видимую часть окна можно, если воспользоваться стандартными средствами просмотра, например, полосами прокрутки или клавишами перемещения курсора Page Up, Page Down.

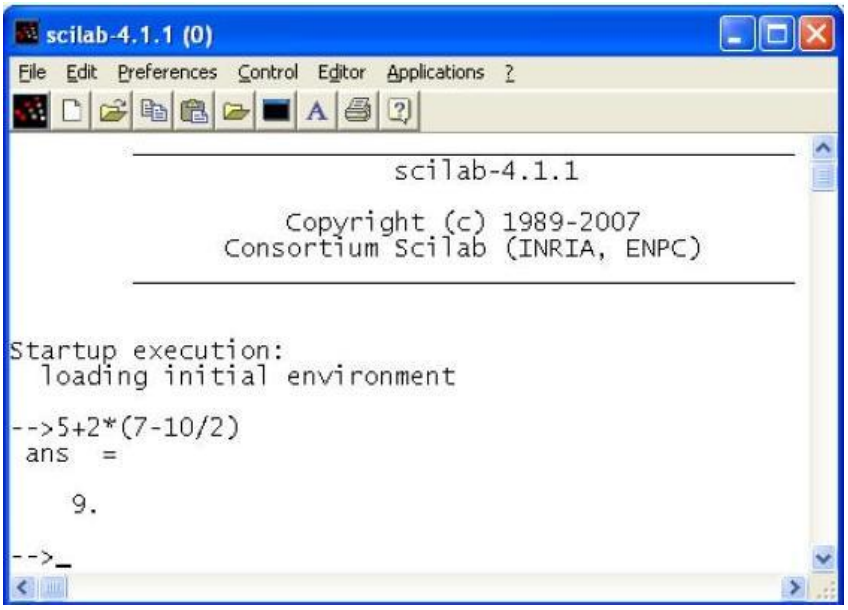


Рисунок 2 - Окно программы Scilab

Клавиши «Стрелка вверх» и «Стрелка вниз» так же управляют курсором, однако в Scilab они имеют другое назначение. Эти клавиши позволяют вернуть в командную строку ранее введенные команды или другую входную информацию, так как вся эта информация сохраняется в специальной области памяти. Так, если в пустой активной командной строке нажать клавишу «Стрелка вверх», то появится последняя вводимая команда, повторное нажатие вызовет предпоследнюю и так далее. Клавиша «Стрелка вниз» выводит команды в обратном порядке. Таким образом, можно сказать, что вся информация в рабочей области находится или в зоне просмотра или в зоне редактирования. Важно знать, что в зоне просмотра нельзя ничего исправить или ввести. Единственная допустимая операция, кроме просмотра, это выделение информации с помощью мыши и копирование ее в буфер обмена, например, для дальнейшего помещения в командную строку.Зона редактирования это фактически командная строка [4,5].

Кроме того, существуют особенности ввода команд. Если команда заканчивается точкой с запятой «;», то результат ее действия не отображается в командной строке. В противном случае, при отсутствии знака «;», результат действия команды сразу же выводится в рабочую область [5]:

```
-->2.7*3+3.14/2
 ans  =
    9.67
-->2.7*3+3.14/2;
-->
```

Текущий документ, отражающий работу пользователя с системой Scilab, содержащий строки ввода, вывода и сообщения об ошибках принято называть *сессией*. Значения всех переменных, вычисленные в течение текущей сессии, сохраняются в специально зарезервированной области памяти, называемой рабочим пространством системы. При желании, определения всех переменных и функций, входящих в текущую сессию, можно сохранить в виде файла, саму сессию сохранить нельзя.

Основные команды главного меню Scilab

Главное меню системы содержит команды предназначенные для работы с файлами, настройки среды, редактирования команд текущей сессии и получения справочной информации. Кроме того, с помощью главного меню можно создавать, редактировать, выполнять отладку и запускать на выполнение так называемые файлы-сценарии Scilab, а так же работать с графическими приложениям пакета.

Работа с файлами

Пункт меню File предназначен для работы с файлами. Рассмотрим назначение представленных в нем команд:

- New Scilab - открывает новое окно Scilab, фактически пакет запускается повторно;
- Exec... - запуск на выполнение созданной ранее Scilab-программы (файлы с расширением sce или sci);
- Open - открывает окно для загрузки созданного ранее файла, рисунка или модели;
- Load - открывает окно для загрузки файлов, информация в которых хранится в виде машинных кодов, при их открытии в память компьютера загружаются определенные ранее переменные и функции;
- Save - сохранение всех определенных в данной сессии переменных и функций в виде файла с расширением sav или bin;
- Change Directory - смена текущего каталога, выводит окно настройки путей файловой системы;
- Get Change Directory - выводит в командную строку имя текущего каталога;
- Print Setup... - выводит окно настройки параметров печати;
- Print - печать текущей сессии;

- Exit - выход из системы Scilab.

Редактирование команд текущей сессии

Пункт меню Edit содержит следующие команды:

- Select All - выделение всех команд текущей сессии;
- Copy - копирование выделенного объекта в буфер;
- Paste - вставка объекта из буфера;
- Empty Clipboard - очистка буфера обмена;
- History - группа команд предназначенных для редактирования командной строки.

Настройка среды

Команды настройки среды пакета представлены в меню Preferences:

- Language - предлагает выбрать из списка язык интерфейса (английский, французский);
- Colors - позволяет установить цвет шрифта (Text), цвет фона (Background) или цвета принятые по умолчанию (Default System Colors);
- Toolbar - выводит или удаляет панель инструментов;
- Files Association- предлагает установить типы поддерживаемых файлов;
- Choose Font - выполняет настройки шрифта (гарнитура, начертание, размер);
- Clear History -очищает рабочее пространство;
- Clear Command Window - очищает рабочее окно;
- Consol - активизирует консольное приложение.

Справочная система

Команда главного меню *?* открывает доступ к справочной системе Scilab. В справочной системе информацию можно искать, воспользовавшись содержанием, в списке, упорядоченном по алфавиту, по ключевому слову или фразе.

С помощью команды Scilab Demos можно осуществить просмотр демонстрационных примеров.

Редактирование и отладка файлов-сценариев

Файл-сценарий - это список команд Scilab сохраненный на диске. Для подготовки, редактирования и отладки файлов-сценариев служит специальный редактор SciPad, который можно вызвать, выполнив команду главного меню Editor. В результате работы этой команды будет создан новый файл-сценарий. По умолчанию он имеет имя Untitled1.sce.

Окно редактора файлов-сценариев выглядит стандартно, то есть имеет заголовок, меню, панели инструментов, строку состояния. Ввод текста в окно редактора файла-сценария осуществляется по правилам принятым для команд Scilab [5].

Задание. Изучите пункты меню пакетов Smath Studio и Scilab. Изучите подпункт «Справочник» пункта «Помощь».

Лабораторная работа №2
Арифметические операции. Целые и рациональные числа, константы. Синтаксис команд. Стандартные функции.

Цель работы: познакомится с основными арифметическими операциями пакетов Smath Studio и Scilab, научится выполнять простейшие математические вычисления.

Арифметические операции. Целые и рациональные числа. Константы. Синтаксис команд. Стандартные функции в Smath Studio

Для проведения каких-либо математических вычислений, в математическом пакете Smath Studio, нужно установить курсор на рабочем поле, ввести выражение, поставить знак "=" и нажать Enter. Линия подчёркивания указывает на те символы, к которым будет

применено действие арифметического знака или функции. Изменить её положение можно с помощью клавиши «пробел» [3].

Определение переменных

Присваивание значения переменной осуществляется знаком «:=», который можно ввести из панели «Арифметика» или с клавиатуры. На рис. 3 представлен вид программы Smath Studio с объявлением переменных и простейшими вычислениями.

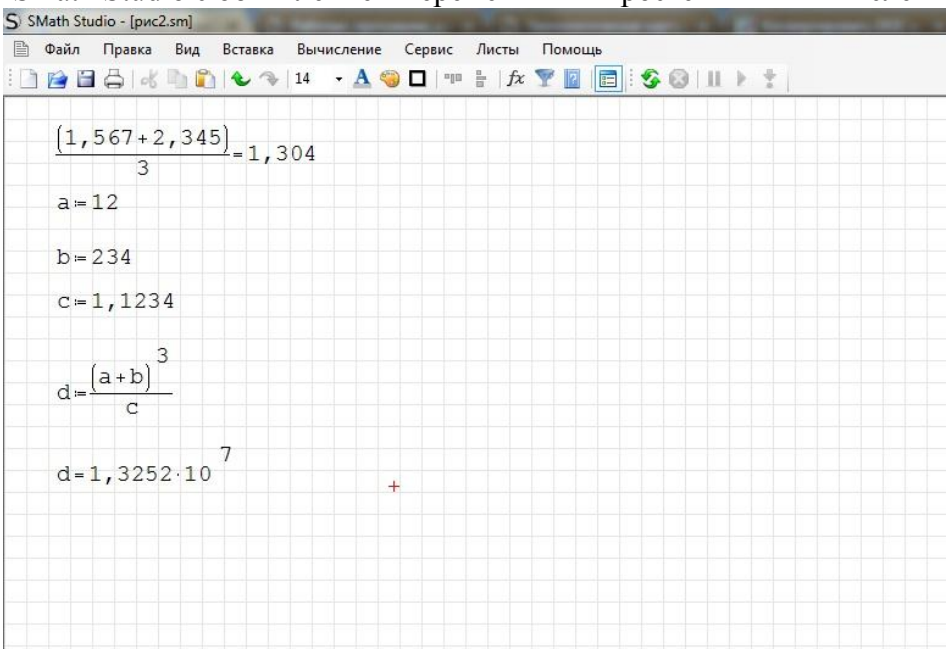


Рисунок 3 - Окно Smath Studio с простейшими арифметическими вычислениями

Согласно синтаксису рассматриваемого пакета, введённая заранее переменная или функция должна быть записана левее или выше того выражения, в котором она используется. При этом её изменение приведёт к тому, что весь лист будет пересчитан (незаменимое свойство при организации многократных объёмных вычислений). Промежуточные расчёты можно скрыть из поля зрения с помощью инструмента «Область» (меню Вставка — Область).

Текстовый комментарий может быть введён в любой области рабочего окна, при этом никакие специальные знаки не требуются. Пользователь может менять цвет шрифта, фона, а также выделять выражение рамкой.

Программа знает многие математические и физические константы, умеет работать с размерностями. Весь лист с расчётами может быть сохранён и в дальнейшем открыт для просмотра и редактирования.

Для пакета Smath Studio есть определенные правила записи выражений:

1. используемая переменная или функция должна быть объявлена левее или выше того выражения, где она используется в вычислении;

2. если переменная переобъявлялась, то будет использовано то значение, которое встретилось самым последним перед использованием в вычислениях;

3. при объявлении переменной в выражении можно использовать встроенные и ранее объявленные функции, ранее объявленные переменные и их сочетания. Если используемые в выражении переменные ранее не объявлялись, то результат можно будет получить только в символьном виде;

4. переменная не обязательно должна вычисляться как числовое значение, допускается присваивать имена выражениям, дающим при вычислении матрицу;

5. для символьных вычислений объявлять переменные заранее не требуется, если не нужно, чтобы при преобразовании выражений были подставлены их значения.

При работе с вещественными числами иногда требуется настраивать точность ответа (количество знаков после запятой). Для этого надо выполнить команду: Сервис -

Опции - вкладка «Вычисление» - «Точность ответа». Дробная часть в числах отделяется запятой.

Рассмотрим подробно синтаксис различных команд в Smath Studio.

Правила ввода текста на рабочем листе Smath Studio

В SMath Studio есть возможность вставлять текстовые области на рабочем листе. Например, Вы можете сделать примечания о проделанных вычислениях. Для того, чтобы ввести текст, необходимо [3]:

1. Щелкнуть в выбранном месте рабочего листа.

2. Выбрать Текстовая область из меню Вставка, или нажать клавишу ", чтобы сказать SMath Studio, что Вы собираетесь ввести текст. После этого, SMath Studio изменяет курсор в виде крестика в вертикальную линию. Контур окружает точку вставки, указывая, что Вы находитесь теперь в текстовой области. Этот контур называют текстовым окном. Для того чтобы ввести вторую линию текста, нажмите Shift + Ввод и продолжайте печатать текст.

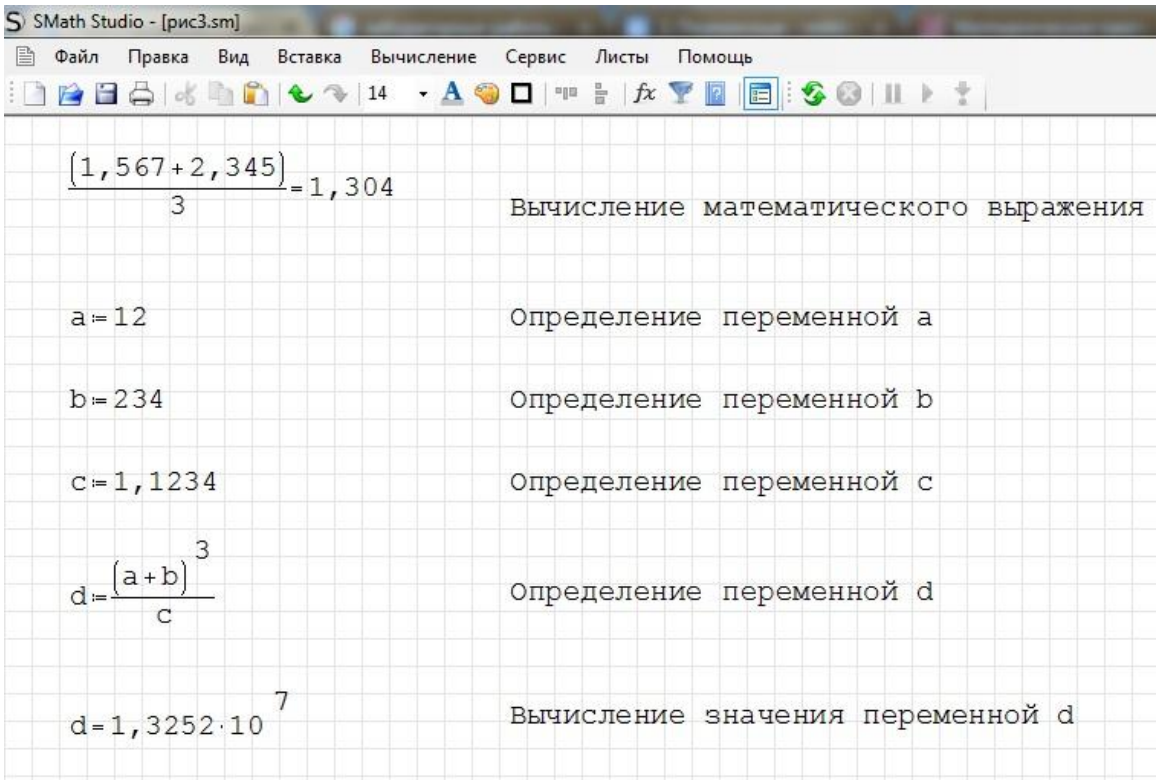


Рисунок 4 - Пример использования текстовой области

Ранжированные переменные

Ранжированные переменные или переменные диапазона в Smath Studio являются разновидностью векторов и предназначены, главным образом, для создания циклов или итерационных вычислений. Простейший пример ранжированной переменной — это массив с числами, лежащими в некотором диапазоне с некоторым шагом.

SMath Studio позволяет сформировать вектор с заданным диапазоном значений. Для формирования такого вектора сначала нужно создать переменную диапазона. Например, сформируем вектор A для диапазона значений от 1 до 10 с шагом 0.5. Для этого нужно выполнить следующую последовательность действий [3]:

1. Наберите на клавиатуре A:=;
2. Наберите на клавиатуре 1, затем .. (диапазон значений с ПИ "Матрицы"), затем 10. Если вам необходимо, чтобы величина изменялась с шагом 1, можно на этом закончить. Если нужен шаг, отличный от единицы, то наберите на клавиатуре ; (точка с

запятой) после 1, затем 1.5, диапазон значений и 10. Ниже на рабочем листе нужно ввести A= и нажать Enter (рис. 5).

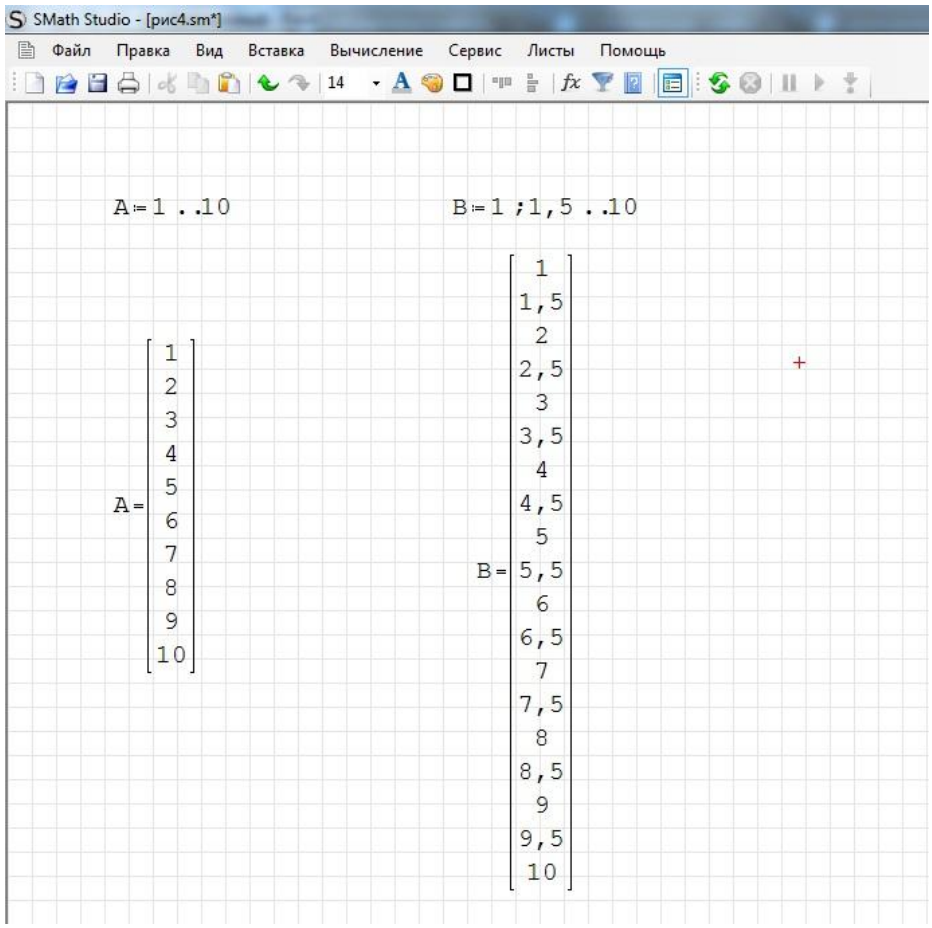


Рисунок 5 - Пример использования ранжированной переменной

Определение пользовательских функций и использование встроенных функций

Система компьютерной математики Smath Studio имеет ряд встроенных функций, которые можно использовать при решении различных задач. Встроенные функции можно вводить вручную, либо использовать панель инструментов Функции. Но на этой панели приведены далеко не все имеющиеся в пакете функции. Поэтому есть ещё один способ доступа к функциям: Основное меню - Вставка - Функция [3].

Для справки по встроенным функциям (и их ввода) используется значок *f(x)*. Ряд встроенных функций после ввода открывающей скобки изменяют свой внешний вид и выглядят так же, как при письме «по бумаге». Перечислим некоторые встроенные алгебраические функции:

abs(a) – модуль числа a;

sqrt(a) – квадратный корень числа a;

exp(a) – возвращает число e в степени a;

ln(a) – натуральный логарифм числа a;

lg(a) – десятичный логарифм числа a;

xy2pol(x;y) – переводит координаты точки из декартовой системы в полярную;

pol2xy(r;fi) – переводит координаты точки из полярной системы в декартову;

random(n) – возвращает случайно выбранное натуральное число в диапазоне от 0 до n;

m! – возвращает факториал числа m;

Кроме использования стандартных функций, пользователь может создавать свои собственные функции. Синтаксис описания пользовательской функции следующий:

имя_функции (параметр1 , параметр2, .. , параметрN) :=выражение,

где:

имя_функции - название функции;

параметр1 , параметр2, .. ,параметрN - парметры, с которыми будет вызываться функция;

выражение - любое правильно написанное выражение, которое может использовать параметры функции.

Функции пользователя очень гибкий и удобный инструмент, они позволяют существенно уменьшить объем расчета путем замены часто повторяющихся участков. Но при работе с пользовательскими функциями стоит помнить, что

1. числу параметров, указанному при описании функции должно соответствовать число параметров при вызове.
2. параметрами функции могут быть Матрицы, Числа, и Строки. Но при вызове функции должен быть учтен порядок записи переменных. Т.е. если в описании функции 1я переменная - матрица, 2я - число, а 3я - строка, то и при вызове функции 1й параметр должен быть матрицей, 2й - числом, 3й - строкой.
3. в правой части описания функции могут использоваться элементы панели Программирование.

На рис. 6 приведен пример объявления и вызова пользовательской функции.

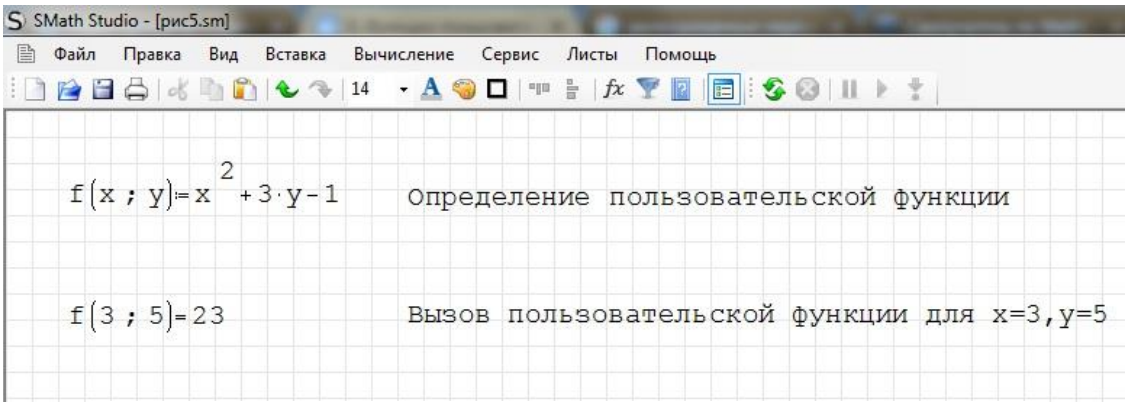


Рисунок 6 - Пример объявления и вызова пользовательской функции

Арифметические операции. Целые и рациональные числа. Константы. Синтаксис команд. Стандартные функции. Определение переменных в Scilab

Пользовательские переменные в Scilab

В рабочей области Scilab можно определять переменные для дальнейшего их использования в различных выражениях. Определить переменную - значит присвоить ей какое-либо значение. Оператором присваивания в Scilab является «=». Процедура присваивания оформляется следующим образом: имя переменной=значение переменной. Каждая переменная перед использованием в выражении должна быть определена. В противном случае Scilab выдаст предупреждение об ошибке [4].

Переменным могут быть присвоены не только численные значения, например переменная может быть определена как строка символов. При определении переменной ей может быть присвоено значение результата вычисления. Тогда в результате выполнения команды на экран будет выведено: имя переменной=результат вычисления [4] (рис. 7).

```
-->d='stroka'
 d  =
stroka
```

```
--> d=6+8
 d  =
   14.
```

Рисунок 7 - Определение переменной в Scilab

Если в течение одного сеанса присвоить некоторое значение ранее определенной в этом сеансе переменной, то в дальнейшем именно это значение будет использоваться программой во всех вычислениях, содержащих переменную. Говорят, что переменная переопределена [4].

Системные переменные и константы

При выполнении операции, Scilab обязательно присваивает ее результат какой-либо переменной, если в командной строке нет оператора присваивания, то результат будет присвоен системной переменной с именем ans. Переменную ans можно использовать в последующих вычислениях, но ее значение будет изменяться каждый раз после выполнения команды без оператора присваивания [4]:

```
-->3+4
 ans  =
 7.
```

Ans — первый пример системной переменной. Имена других системных переменных в Scilab начинаются с символа %:

- %pi – число π (3.141592653589793);
- %e – число e=2.7182818.

Эти переменные используются в математических операциях в качестве констант. Их значения не могут быть изменены пользователем [4].

Математические операции в Scilab

Для выполнения простейших арифметических операций Scilab использует следующие операторы [4]:

+ сложение;
- вычитание;
* умножение;
/ деление слева направо;
\ деление справа налево;
^ возведение в степень.

Чтобы вычислить значение арифметического выражения, необходимо ввести его в командную строку и нажать «Enter». Результат вычисления появится в рабочей области.

Например для вычисления выражения $4\times8-\frac{45.6}{5.67}+2^3$ в Scilab необходимо записать следующие команды [4]:

```
-->4*8-5.67\45.6+2^3
 ans  =
   31.957672
```

При вводе выражения применена операция деления слева «\». Поскольку значение выражения не присвоено пользовательской переменной, то результат присваивается системной переменной ans. Если вычисляемое выражение длинное и желательно перенести его запись на следующую строку, то в конце незавершенной строки необходимо ввести три (или более) точки. После этого можно нажать «Enter» и продолжать набор оставшейся части на следующей строке [4].

Если после ввода команды нажать клавишу «Enter», то в рабочей области появится результат выполнения этой команды. Если результат отображать не нужно, то набор команды следует завершить символом «;» [4].

Правила работы с вещественными числами в Scilab

Числовые результаты могут быть представлены с плавающей (например, 3.2E - 6, 6.42E+2), или с фиксированной (например, 4.12, 6.05, 17.5489) точкой. Числа в формате

с плавающей точкой представлены в экспоненциальной форме mE±p, где m - мантисса (целое или дробное число с десятичной точкой), p - порядок (целое число). Для того, чтобы перевести число в экспоненциальной форме к обычному представлению с фиксированной точкой, необходимо мантиссу умножить на десять в степени порядок [5].

При вводе вещественных чисел для отделения дробной части используется точка. Примеры ввода и вывода вещественных чисел [5]:

```
-->0.123
 ans  =   0.123
-->-6.42e+2
 ans  =   - 642.
-->3.2e-6
 ans  =   0.0000032
```

Scilab в качестве результата выводит только восемь значащих цифр. Это формат вывода вещественного числа по умолчанию. Для того, чтобы контролировать количество выводимых на печать разрядов применяют команду printf с заданным форматом, который соответствует правилам принятым для этой команды в языке C:

```
-->printf("%1.12f",%pi)
3.141592653590
-->printf("%1.15f",%pi)
3.141592653589793
-->printf("%1.2f",q)
123.46
-->printf("%1.10f",q)
123.4567890123
-->//По умолчанию 6 знаков после запятой
-->printf("%f",q)
123.456789
```

Задание 1. Вычислить значение выражения для данного набора исходных данных:

$a = r^2 t - b^2$, при *r=-0,2 t=2 b=-1,3*

Задание 2. Вычислить значение арифметического выражения для заданного набора исходных данных. Установить формат результата N знаков после запятой.

Для вывода определенного количества знаков после запятой необходимо выделить результат вычислений – нажать правую кнопку мыши – выбрать пункт «Точность ответа» - выбрать количество знаков после запятой.

$\frac{2\sin\left(\frac{d}{b}\right)}{b+d^{3-b}} + \frac{1}{2e^{b+d}} + (\cos(b))^2$ при b=2,12 d=3,13 (формат результата: 3 знака после запятой).

Задание 3. Найдите значения функций при x=1:

$$f(x) = \frac{3}{5}x^5 - \frac{1}{2x^4} - \frac{2}{\sqrt[4]{x^3}} + 7$$

Задание 4. Вычислить значение арифметического выражения:

$$\frac{\left(13.75+9\frac{1}{6}\right)\cdot 1.2}{\left(10.3-8\frac{1}{2}\right)\cdot\frac{5}{9}} + \frac{\left(6.8-3\frac{3}{5}\right)\cdot 5\frac{5}{6}}{\left(3\frac{2}{3}-3\frac{1}{6}\right)\cdot 56} - 27\frac{1}{6}$$

Лабораторная работа №3
Матричные вычисления

Цель работы: получить навыки обработки матриц в пакетах Smath Studio и Scilab.

Матричные вычисления в Smath Studio

Синтаксис команд при работе с матрицами и векторами

Матрица – это прямоугольный набор элементов, который может со-держать числа (целые, вещественные, комплексные), строковые значения, другие матрицы. Создать матрицу можно несколькими способами:

1. при помощи команды mat() или кнопки  с ПИ «Матрицы». В появившемся диалоговом окне задаются количества строк и столбцов, в результате проделанных действий получится пустая матрица:

2. командой matrix(arg1, arg2), где arg1 – число строк, arg2 – число столбцов. Так, matrix(10, 20) создаст матрицу, заполненную нулями, состоящую из 10 строк и 20 столбцов (рис. 8).

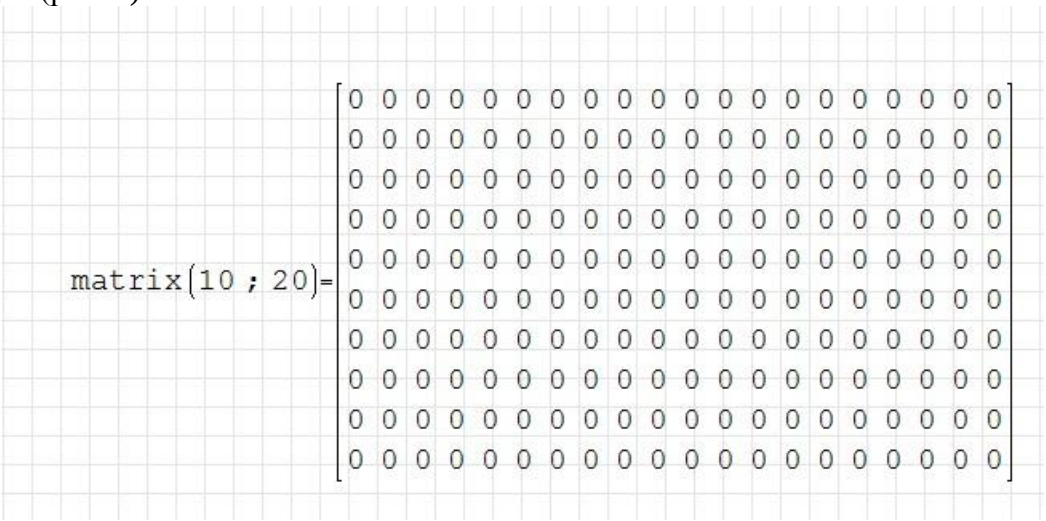


Рисунок 8 - Создание матрицы с помощью функции matrix

Операторы для работы с матрицами находятся на ПИ «Матрицы» (рис. 9).

Рисунок 9 - Панель инструментов Матрицы

Показанные на рисунке 9 символы, означают в указанной последовательности»: вставка матрицы; нахождение определителя матрицы; транспонирование матрицы; алгебраическое дополнение; минор; векторное умножение; функция векторизации; диапазон значений; диапазон с указанием второго значения; элемент вектора (вставка нижнего индекса); элемент матрицы.

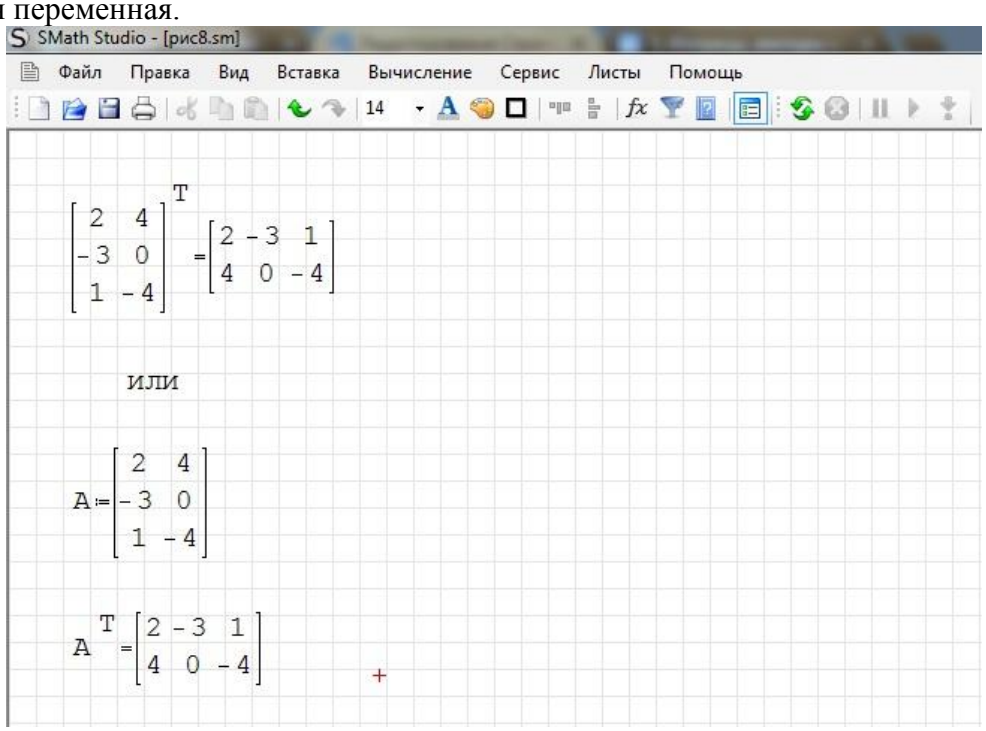
В Smath Studio реализованы простейшие операции матричной алгебры. Рассмотри их.

Транспонирование. Транспонированием называют операцию, переводящую матрицу размером *mxn* в матрицу размером *nxm*, делая столбцы исходной матрицы строками, а строки - столбцами.

Ввод символа транспонирования осуществляется с помощью панели инструментов Матрицы. Возможны два варианта [3]:

1. выбирается инструмент  и в предложенное пустое поле с помощью инструмента  вставляется матрица, затем ставится знак равенства и получается результат;

2. вводится переменная, которой с помощью инструмента присваивается матрица, затем выбирается инструмент и в предложенное пустое поле вводится указанная переменная.



Сложение и вычитание матриц. В Smath Studio можно как складывать матрицы, так и вычитать их друг из друга. Для этих операций применяются инструменты "+" или "-". Матрицы должны иметь одинаковый размер . Каждый элемент суммы двух матриц равен сумме соответствующих элементов матриц-слагаемых.

Кроме сложения матриц, Smath Studio поддерживает операцию сложения матрицы со скаляром. Каждый элемент результирующей матрицы равен сумме соответствующего элемента исходной матрицы и скалярной величины.

Умножение матриц. При умножении матриц следует помнить, что матрицу размером *mxn* допустимо умножать только на матрицу размером *nxp*. В результате получается матрица размером *mxp*.

Умножение матриц проводится двумя способами:

1. вводятся переменные, которым присваиваются с помощью инструмента матрицы, затем находят их произведение с помощью операции умножения;

2. с помощью инструмента вводится матрица, затем ставится знак умножения, далее снова с помощью инструмента вводится матрица и, наконец, ставится знак равенства (рис. 10).

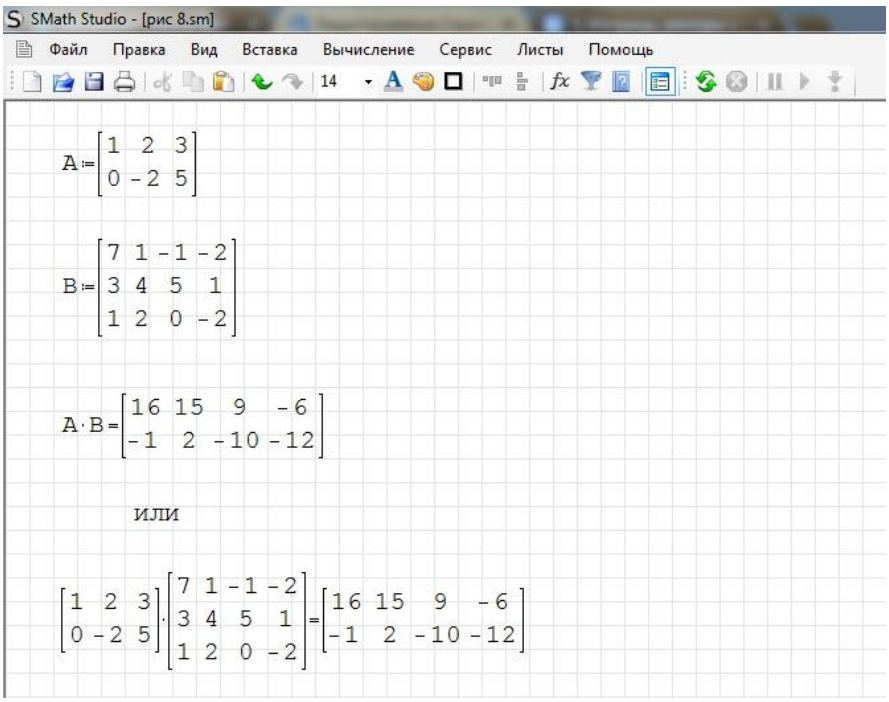


Рисунок 10 - Пример умножения матриц

Определитель квадратной матрицы. Определитель квадратной матрицы - это её числовая характеристика. В Smath Studio он обозначается стандартным математическим символом. Чтобы ввести оператор нахождения определителя матрицы, необходимо нажать кнопку |▪| на панели Матрицы. В результате появляется местозаполнитель, в который либо следует поместить матрицу, либо ввести имя матрицы, которая была определена ранее.

Дополнительные функции при работе с матрицами

В Smath Studio есть ряд дополнительных функций, предназначенных для работы с матрицами:

alg(матрица, число, число) - определяет алгебраическое дополнение элемента матрицы;

augment() - возвращает массив, образованный последовательным размещением аргументов друг рядом с другом. Аргументы могут быть скалярами, векторами или матрицами с одинаковым количеством строк;

col(матрица, число) - возвращает указанный столбец матрицы;

row(матрица, число) - возвращает указанную строку матрицы;

cols(матрица) - возвращает количество столбцов матрицы или вектора;

rows(матрица) - возвращает количество строк матрицы или вектора;

rank(матрица) - определяет ранг матрицы;

tr(матрица) - определяет след матрицы;

invert(матрица) - определение обратной матрицы;

max(матрица) - находит максимальное значение в матрице;

min(матрица) - находит минимальное значение в матрице.

И в завершении изложения хотелось бы отметить, что с точки зрения Smath Studio вектор - это матрица состоящая из одного столбца и 3 строчек, т.е. с векторами можно делать все те же действия, что и с матрицами.

Массивы и матрицы в Scilab. Функции их обработки

Для работы с множеством данных удобно использовать массивы. Например, можно создать массив для хранения числовых или символьных данных. В этом случае,

вместо создания переменной, для хранения каждого данного, достаточно создать один массив, где каждому элементу будет присвоен порядковый номер.

Таким образом, массив - множественный тип данных, состоящий из фиксированного числа элементов. Как и любой другой переменной, массиву должно быть присвоено имя.

Переменную, представляющую собой просто список данных, называют одномерным массивом или вектором. Для доступа к данным, хранящимся в определенном элементе массива, необходимо указать имя массива и порядковый номер этого элемента, называемый индексом.

Если возникает необходимость хранения данных в виде таблиц, в формате строк и столбцов, то необходимо использовать двумерные массивы (матрицы). Для доступа к данным, хранящимся в таком массиве, необходимо указать имя массива и два индекса, первый должен соответствовать номеру строки, а второй номеру столбца в которых хранится необходимый элемент.

Значение нижней границы индексации в Scilab равно единице. Индексы могут быть только целыми положительными числами [5].

Ввод и формирование массивов и матриц

Самый простой способ задать одномерный массив в Scilab имеет вид: *[name]=Xn:dX:Xk1,* где *name* - имя переменной, в которую будет записан сформированный массив, *Xn* - значение первого элемента массива, *Xk* - значение последнего элемента массива, *dX* - шаг, с помощью которого формируется каждый следующий элемент массива, то есть значение второго элемента составит *Xn+dX*, третьего *Xn+ dX+dX* и так далее до *Xk.*

Если параметр *dX* в конструкции отсутствует, это означает, что по умолчанию он принимает значение равное единице, то есть каждый следующий элемент массива равен значению предыдущего плюс один: *[name]=Xn:Xk.* Переменную, заданную как массив можно использовать в арифметических выражениях и в качестве аргумента математических функций [5].

Еще один способ задания векторов и матриц в Scilab это их поэлементный ввод. Так для определения вектор строки следует ввести имя массива, а затем после знака присваивания, в квадратных скобках через пробел или запятую перечислить элементы массива:

[name]=x1 x2 … xn

или

[name]=x1, x2, …, xn

Элементы вектора столбца вводятся через точку с запятой: *[name]=x1; x2; …; xn.*

Обратиться к элементу вектора можно, указав имя массива и порядковый номер элемента в круглых скобках: *name(индекс)* [5].

Ввод элементов матрицы так же осуществляется в квадратных скобках, при этом элементы строки отделяются друг от друга пробелом или запятой, а строки разделяются между собой точкой с запятой: *[name]=[x11, x12, …, x1n; x21, x22, …, x2n;…; xm1, xm2, …, xmn;* Обратиться к элементу матрицы можно, указав после имени матрицы, в круглых скобках, через запятую, номер строки и номер столбца на пересечении которых элемент расположен: name(индекс1, индекс2) [5].

Действия над матрицами

Для работы с матрицами и векторами в Scilab предусмотрены следующие операции:

- + сложение;
- – вычитание;
- * матричное умножение;
- ^ возведение в степень;

- \ левое деление;
- / правое деление и др.

Матричные функции

Для работы с матрицами и векторами в Scilab существуют специальные функции. Рассмотрим наиболее часто используемые из них [5]:

- matrix(A [,n,m]) - преобразует матрицу A в матрицу другого размера;
- ones(m,n) - создает матрицу единиц из m строк и n столбцов;
- zeros(m,n) - создает нулевую матрицу из m строк и n столбцов;
- eye(m,n) - формирует единичную матрицу из m строк и n столбцов;
- rand(n1,n2,…nn[,fl]) - формирует многомерную матрицу случайных чисел, необязательный параметр *p* - это символьная переменная, с помощью которой можно задать тип распределения случайной величины ('uniform' равномерное , 'normal' гауссовское);
- rand(m,n) - формирует матрицу m на n случайных чисел;
- rand(M) - формирует матрицу случайных чисел, размер которой совпадает с размером матрицы M;
- sparse([i1 j1;i2 j2;…;in jn],[n1,n2,…,nn]) - формирует разреженную матрицу; для создания матрицы такого типа необходимо указать индексы ее ненулевых элементов [i1 j1,i2 j2,…,in jn], и их значения [n1,n2,…,nn], индексы одного элемента отделяются друг от друга либо пробелом, либо запятой, а пары индексов соответственно точкой с запятой, значения элементов разделяются запятыми; при попытке просмотреть матрицу подобного типа пользователю будет предоставлено сообщение о ее размерности, а так же значения ненулевых элементов и их местоположение в матрице;
- full(M) - вывод разреженной матрицы M в виде таблицы;
- hypermat(D[,V]) - создание многомерной матрицы с размерностью заданной вектором D и значениями элементов, хранящихся в векторе V (использование параметра V необязательно);
- diag(V[,k]) - возвращает квадратную матрицу с элементами V на главной диагонали или на k-й; функция diag(A [, k]), где A ранее определенная матрица, в качестве результата выдаст вектор столбец, содержащий элементы главной или k ой диагонали матрицы A;
- cat(n, A, B, [C, …]) - объединяет матрицы A и B или все входящие матрицы, при n=1 по строкам, при n=2 по столбцам; то же что [A; B] или [A, B];
- tril(A[,k]) - формирует из матрицы A нижнюю треугольную матрицу начиная с главной или с k-й диагонали;
- triu(A[,k]) - формирует из матрицы A верхнюю треугольную матрицу начиная с главной или с k й диагонали;
- sort(X) - выполняет упорядочивание массива X, если X - матрица, сортировка выполняется по столбцам;
- length(X) - определяет количество элементов массива X, если X - вектор, его длину; если X - матрица, вычисляет общее число ее элементов;
- sum(X) - вычисляет сумму элементов массива X;
- prod(X) - вычисляет произведение элементов массива X;
- max(M) - вычисляет наибольший элемент в массиве M;
- min(M) - вычисляет наименьший элемент в массиве M;
- mean(M) - вычисляет среднее значение массива M;
- det(M) - вычисляет определитель квадратной матрицы M;
- rank(M[,tol]) - вычисление ранга матрицы M с точностью tol.

Задание 1. Вычислить определители:

$$\begin{vmatrix} 1 & 2 & 2 & 4 \\ 1 & 6 & 0 & 5 \\ 1 & 2 & -1 & 3 \\ 1 & 2 & -1 & 2 \end{vmatrix}$$

$$\begin{vmatrix} -2 & 2 & -2 & 1 \\ -1 & 2 & 2 & 4 \\ -1 & 1 & 0 & 1 \\ -3 & 3 & 0 & 4 \end{vmatrix}$$

Задание 2. Найти матрицу обратную заданной, транспонировать матрицу:

$$\begin{pmatrix} 1 & -1 & 0 \\ 2 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Задание 3. Найти ранги матриц, выделить из матриц подматрицы, состоящие из 2й и 3й строк и 2го и 3го столбца:

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 3 & 1 \\ 0 & 2 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 & 7 & 4 \\ 4 & 5 & 7 & 2 \\ 6 & 8 & 14 & 6 \\ -2 & -2 & 0 & 2 \end{pmatrix}$$

Задание 4. Задать матрицы произвольной размерности с произвольными элементами. Выполнить над матрицами указанные действия: max(A), A*B

Лабораторная работа №4
Преобразование математических выражений

Цель работы: получить навыки работы с основными командами математических пакетов для символьного преобразования алгебраических выражений.

Символьные преобразования

Программа SMath Studio умеет выполнять некоторые символьные преобразования, такие как факторизация (выделение общего множителя), работа с показательными и логарифмическими выражениями, матрицами, взятие производных.

Для символьных преобразований необходимо записать выражение и нажать «→» (либо Ctrl+.). Если ввести этот знак после математического выражения, ответ будет получен в виде обыкновенной дроби.

Задание 1. Упростить выражения:

$$\frac{\sqrt{abc+4}}{a}+4\frac{\sqrt{bc}}{a};$$

$$\frac{(\sin^2(x)+\cos^2(x))\cos(x)}{\sin(2x)};$$

$$x(y^2-z^2)+y(z^2-x^2)+z(x^2-y^2);$$

Лабораторная работа №5
Решение уравнений и систем уравнений

Цель работы: получение навыков решения уравнений и неравенств с использованием математических пакетов Smath Studio и Scilab.

Решение уравнений и систем уравнений в пакете Smath Studio

Решение нелинейных уравнений

Для решения алгебраических уравнений с одним неизвестным в Smath Studio предусмотрена функция*solve*, которая, в зависимости от типа задачи, может включать либо два, либо четыре аргумента и, соответственно, использует разные алгоритмы поиска корней [1]:

1. *solve(уравнение, переменная)* - находит действительные корни уравнения относительно указанной переменной;
2. *solve(уравнение, переменная, число, число)* - находит действительные корни уравнения на промежутке [число, число] относительно указанной переменной.

Когда функция *solve* имеет четыре аргумента, следует помнить о двух её особенностях [1]:

1. внутри интервала может находиться более одного корня, тогда они будут найдены все;
2. значения *f(a*) и *f(b)* должны иметь разный знак.

Решение полиномиальных уравнений

Если функция *f(x)* является полиномом, то все его корни можно определить, используя встроенную функцию *polyroots(v)*, где *v* - вектор, составленный из коэффициентов полинома [1].

Так как полином *n*-й степени имеет ровно n корней, вектор v должен состоять из (*n+1)* элемента. В основе встроенной функции *polyroots(v)* лежат специальные численные алгоритмы, а результатом её действия является вектор, составленный из n корней рассматриваемого полинома [1].

Решение систем нелинейных уравнений

Решением системы нелинейных уравнений называется группа чисел, которые, будучи подставлены на место неизвестных, обращают каждое уравнение системы в тождество.

В Smath Studio для решения используют функцию *roots(уравнения;переменные)* [1].

Как уравнения, так и переменные задаются с помощью вектора (ПИ «Матрицы»). Уравнения можно задать непосредственно в первоначальном виде, где используют булево равно. Можно также предварительно привести уравнения к виду *f(x) = 0*. Функция *roots* находит только действительные решения.

Решение систем линейных уравнений

Один из методов решения систем линейных уравнений - метод обратной матрицы. Для решения систем таким способом используются операторы ПИ «Матрицы».

Также возможно решение систем линейных уравнений методом Крамера с использованием ПИ «Матрицы» и «Арифметика».

Решение уравнений и систем уравнений в Scilab

Решение алгебраических уравнений

Любое уравнение P(x) = 0, где P(x) - это многочлен, отличный от нулевого, называется алгебраическим уравнением или полиномом. Всякое алгебраическое уравнение относительно x можно записать в виде $a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x+a^n=0$ [3].

Решение алгебраического уравнения в Scilab состоит из двух этапов. Необходимо задать полином P(x) с помощью функции poly, а затем найти его корни, применив функцию roots [2,3].

Итак, определение полиномов в Scilab осуществляет функция poly(a, "x ["fl"]), где a - это число или матрица чисел, x - символьная переменная, fl - необязательная символьная переменная, определяющая способ задания полинома. Символьная переменная fl может принимать только два значения "roots" или "coeff" (соответственно "r" или "c"). Если fl=c,

то будет сформирован полином с коэффициентами, хранящимися в параметре a. Если же fl=r, то значения параметра a воспринимаются функцией как корни, для которых необходимо рассчитать коэффициенты соответствующего полинома [2,3].

Функция roots(p) предназначена для решения алгебраического уравнения. Здесь p - это полином, созданный функцией poly и представляющий собой левую часть уравнения P(x) = 0 [2,3].

Трансцендентные уравнения

Уравнение f(x) = 0, в котором неизвестное входит в аргумент трансцендентных функций, называется трансцендентным уравнением. К трансцендентныуравнениям принадлежат показательные, логарифмические и тригонометрические. В общем случае аналитическое решение уравнения f(x) = 0 можно найти только для узкого класса функций. Чаще всего приходится решать это уравнение численными методами [3].

Численное решение нелинейного уравнения проводят в два этапа. В начале отделяют корни уравнения, т.е. находят достаточно тесные промежутки, в которых содержится только один корень. Эти промежутки называют интервалами изоляции корня, определить их можно, изобразив график функции f(x) или любым другим методом. На втором этапе проводят уточнение отделенных корней, или, иначе говоря, находят корни с заданной точностью [2,3].

Для решения трансцендентных уравнений в Scilab применяют функцию fsolve(x0,f), где x0 - начальное приближение, f - функция, описывающая левую часть уравнения y(x) = 0.

Системы уравнений

Если заданы m уравнений с n неизвестными и требуется найти последовательность из n чисел, которые одновременно удовлетворяют каждому из m уравнений, то говорят о системе уравнений. Для решения систем уравнений в Scilab также применяют функцию fsolve(x0,f) [3].

Задание. Решить уравнения в пакетах Smath Studio и Scilab.

1. Дано уравнение $\frac{x+2}{x+1}+\frac{2-x}{1-x}+\frac{4}{x-1}$. Определить один из корней с помощью задания начального значения, другой с помощью задания интервала, которому принадлежит корень. Найти вектор корней уравнения. Все вычисления производить с точностью 0.00001.

2. Найти все корни полиномиального уравнения: $2x^4-21x^3+74x^2-105x+50=0$.

3. Найти одно из решений системы уравнений численно: $\begin{cases}\cos(x)+y=1.5\\ 2x-\sin(y-0.5)=1\end{cases}$

4. Решить систему линейных уравнений:

$$\begin{cases}0.65x1-0.93x2+0.45x3=-0.72\\ 1.15x1+0.43x2-0.72x3=1.24\\ 0.56x1-0.18x2+1.03x3=2.15\end{cases}.$$

Лабораторная работа №6
Построение 2D и 3D графиков

Цель работы: получение навыков работы по созданию двух- и трехмерных графиков в математических пакетах.

Построение 2D и 3D графиков в Smath Studio

Построение двумерных графиков

Графики функции в Smath Studio бывают 2х видов: двухмерные (2d) и трехмерные (3d). Для работы с ними есть специальная панель "График". Вставить график в расчет можно несколькими способами:

- 2d график можно вставить комбинацией shift + @;
- при помощи меню Вставка - График.

Таким образом, при создании графика необходимо выбрать в меню Вставка – График – Двумерный (2D) (рис. 11) или нажать shift + @. В нижней части появившегося окна вводится имя функции или сама функция. Аргументом должна быть только переменная «*x*» (т.е. функция имеет вид *f(x)*).

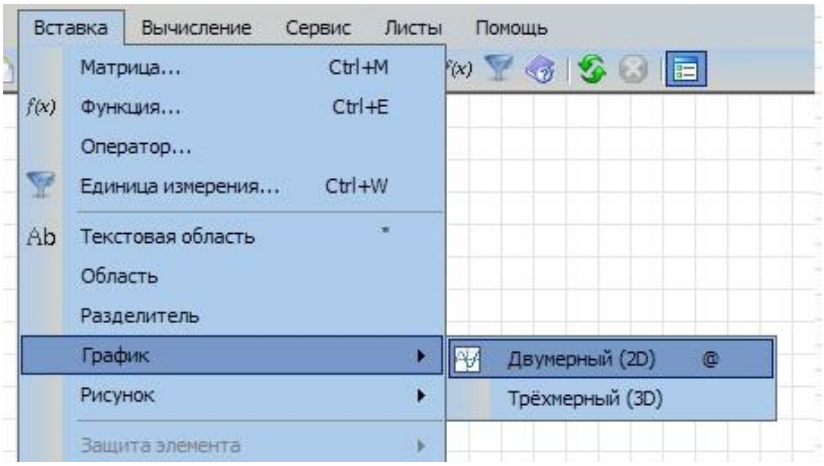


Рисунок 11 - Построение графика в Smath Studio

Чтобы построить графики нескольких функций в одной системе координат, необходимо добавить фигурные скобки, нажать пробел, растянуть их за чёрную точку в правом нижнем углу до требуемого размера и затем ввести функции (рис. 12).

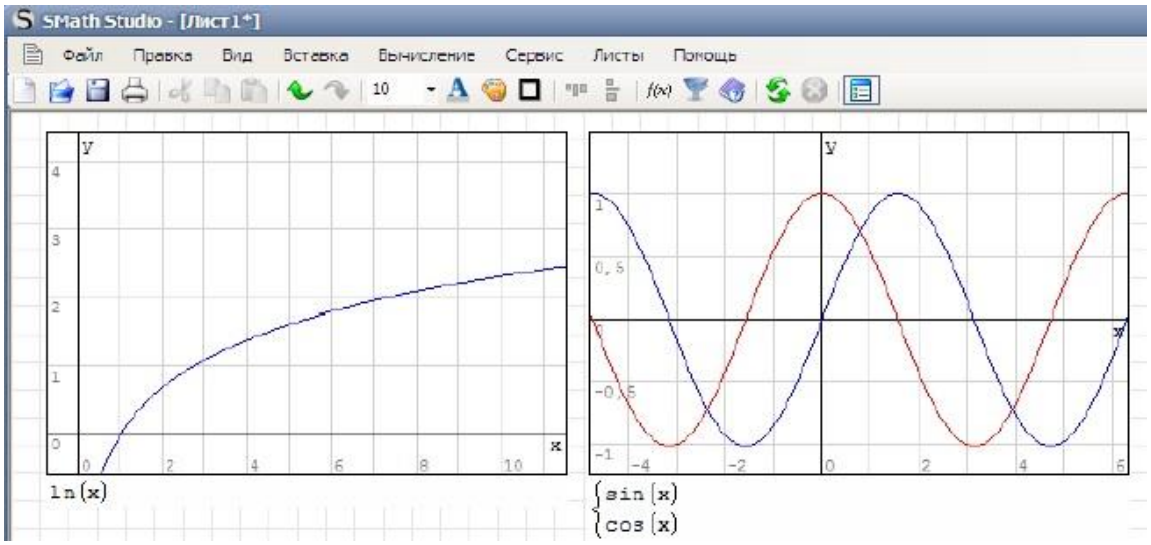


Рисунок 12 - Графики в Smath Studio

Для изменения окна графика необходимо растянуть его за одну из трёх чёрных точек. Вращение ролика мыши приводит к масштабированию всего графика в целом. Для масштабирования по оси абсцисс необходимо при вращении ролика держать нажатой клавишу Shift, по оси ординат – Ctrl. Движение мыши при нажатой левой клавише ведёт к разному результату в зависимости от выбранного значка на панели: перемещает систему координат («Перемещать», «Вращать», включено по умолчанию), либо масштабирует график («Масштабировать»). Пункты «График точками» и «График линиями» изменяют

соответствующий вид отображения. Нажатие «Обновить» сбрасывает все сделанные изменения.

Стоит обратить внимание, что в объявлении функции можно использовать любые имена переменных.Но график строиться только относительно "x" и "y". Т.е внутри графика функция должна быть вызвана с использованием этих переменных (рис. 13).

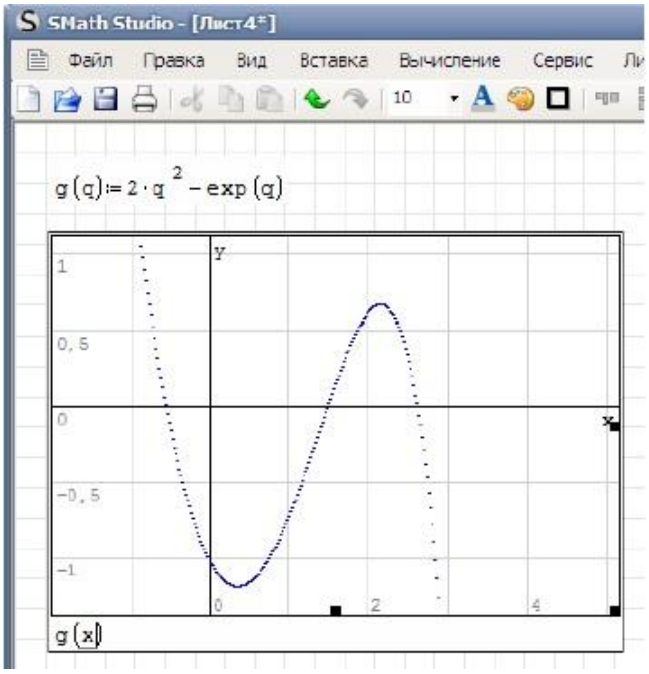


Рисунок 13 - Двумерный график в Smath Studio

Построение трёхмерных графиков

Трёхмерный график строится через меню Вставка – График – Трёхмерный. В нижней части окна нужно ввести имя функции или саму функцию. Аргументами должны быть только «x» и «y». Графики нескольких функций также строятся с помощью фигурных скобок.

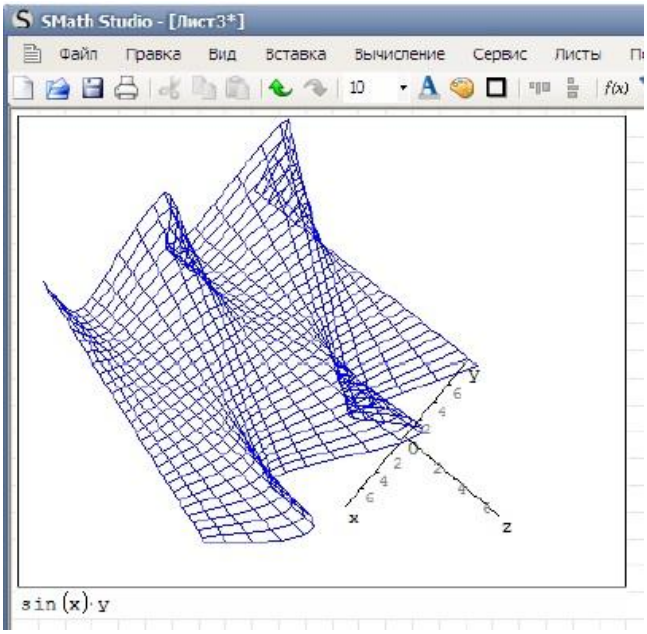


Рисунок 14 - Трехмерный график в Smath Studio

Для форматирования графика нужно сделать его активным и установить желаемый размер окна. Масштабирование вращением ролика осуществляется для всех осей

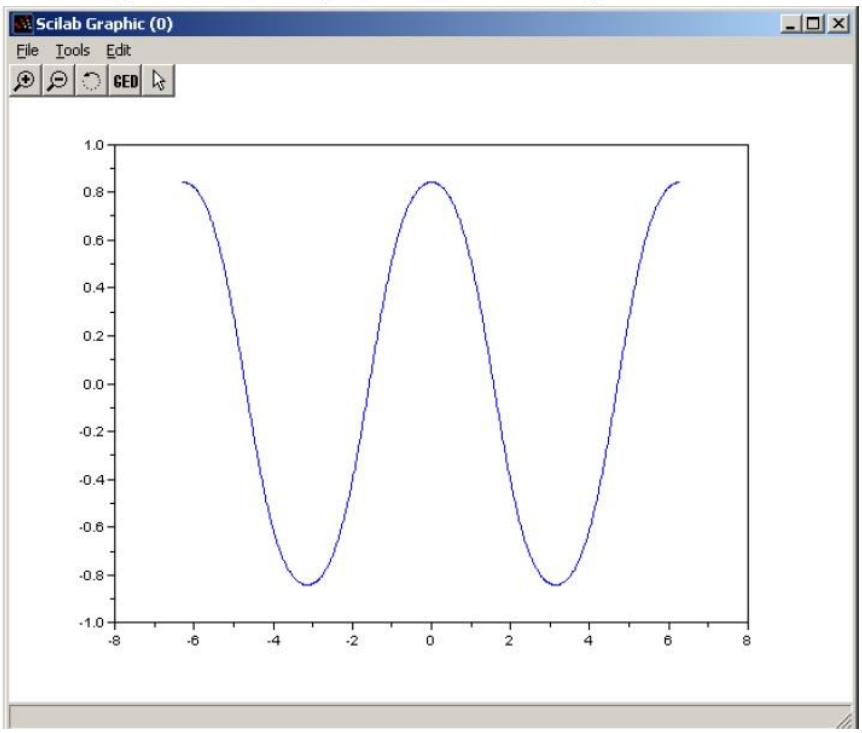
одновременно. В зависимости от выбранного значка на панели график можно движением мыши вращать (по умолчанию), масштабировать или перемещать. Также доступен выбор отображения линиями или точками. Нажатие «Обновить» сбрасывает все сделанные изменения.

Построение 2D и 3D графиков в Scilab

Построение двумерных графиков
Функция plot

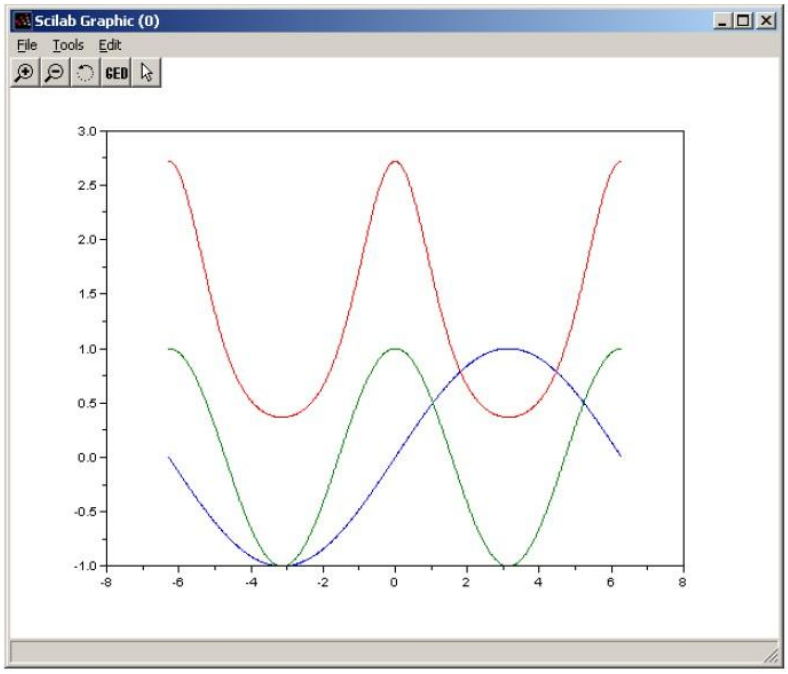
Для построения графиков вида y=f(x) в Scilab существует функция *plot*. Эта функция была значительно модифицирована начиная с 4-й версии пакета. Так, при простейшем обращении к функции *plot(x,y)* будет создано окно, в котором будет построен график функции y(x) [5]:

```
x=-2*%pi:0.1:2*%pi;
y=sin(cos(x));
plot(x,y,'X','Y','plot function y=sin(cos(x))');
```



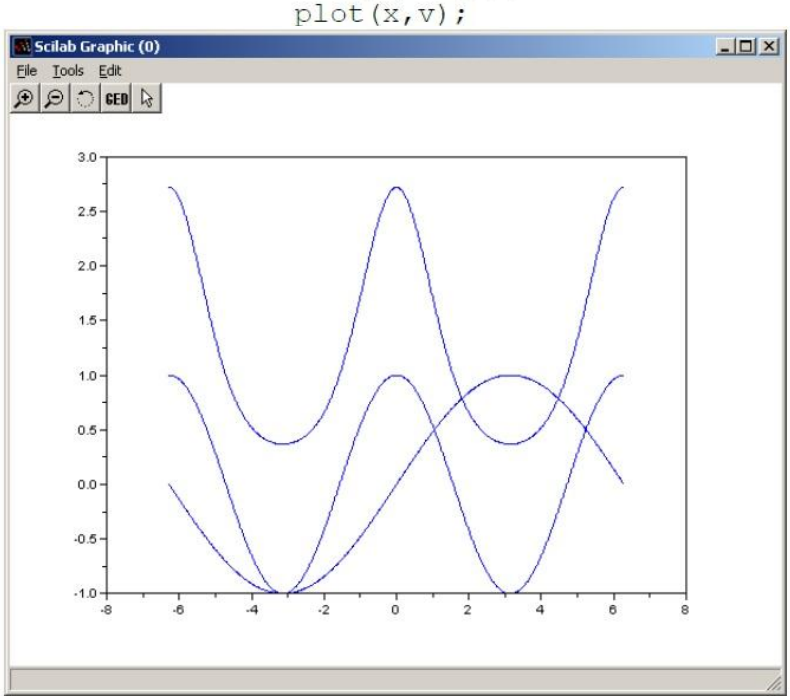
Если повторно обратиться к функции *plot*, то будет создано новое графическое окно и в нем будет построен новый график. Для построения нескольких графиков в одной системе координат можно поступить одним из следующих способов:
1. Обратиться к функции plot следующим образом plot(x1,y1,x2,y2,…xn,yn), где x1, y1 – массивы абсцисс и ординат первого графика; x2, y2 – массивы абсцисс и ординат второго графика; …, xn, yn – массивы абсцисс и ординат n-ого графика. Например [5]:

```
x=-6.28:0.02:6.28;
y=sin(x/2);
z=cos(x);
v=exp(cos(x));
plot(x,y,x,z,x,v);
```



Каждый график можно изображать с помощью функции plot(x,y), но перед обращением к функциям plot(x2,y2), plot(x3,y3), …, plot(xn,yn) вызвать команду mtlb_hold(' on') , которая блокирует режим создания нового окна [4,5].

```
x=-6.28:0.02:6.28;
y=sin(x/2);
plot(x,y);
mtlb_hold(' on');
z=cos(x);
plot(x,z);
mtlb_hold(' on');
v=exp(cos(x));
plot(x,v);
```



Обратите внимание, что при построении графиков первым способом Scilab автоматически изменяет цвета изображаемых в одной системе координат графиков. Однако управлять цветом и видом каждого из изображаемых графиков может и

пользователь, для чего необходимо воспользоваться полной формой функции plot: plot(x1, y1, s1, x2, y2, s2, …, xn, yn, sn), где x1, x2, …, xn – массивы абсцисс графиков; y1, y2, …, yn – массивы ординат графиков; s1, s2, …, sn – строка, состоящая из трех символов, которые определяют цвет линии, тип маркера и тип линии графиков (см. табл. 4.1-4.3), в строке могут использоваться один или два символа [4,5].

Параметры функции plot. Символы цветов линий

- y - желтый;
- m - розовый;
- c - голубой;
- r - красный;
- g - зеленый;
- b - синий;
- w - белый;
- k - черный;

Параметры функции plot. Символы типов маркера:

- *. - точка;*
- *o - кружок;*
- *x - крестик;*
- *+ - знак "плюс";*
- ** - звездочка;*
- *s - квадрат;*
- *d - ромб;*
- *v - треугольник вершиной вниз;*
- *^ - треугольник вершиной вверх;*
- *< - треугольник вершиной влево;*
- *> - треугольник вершиной вправо;*
- *p - пятиконечная звезда;*
- *h - шестиконечная звезда;*

Параметры функции plot. Символы типов линий.

- *- сплошная;*
- *: пунктирная;*
- *-. штрихпунктирная;*
- *-- штриховая.*

Функция plot2d

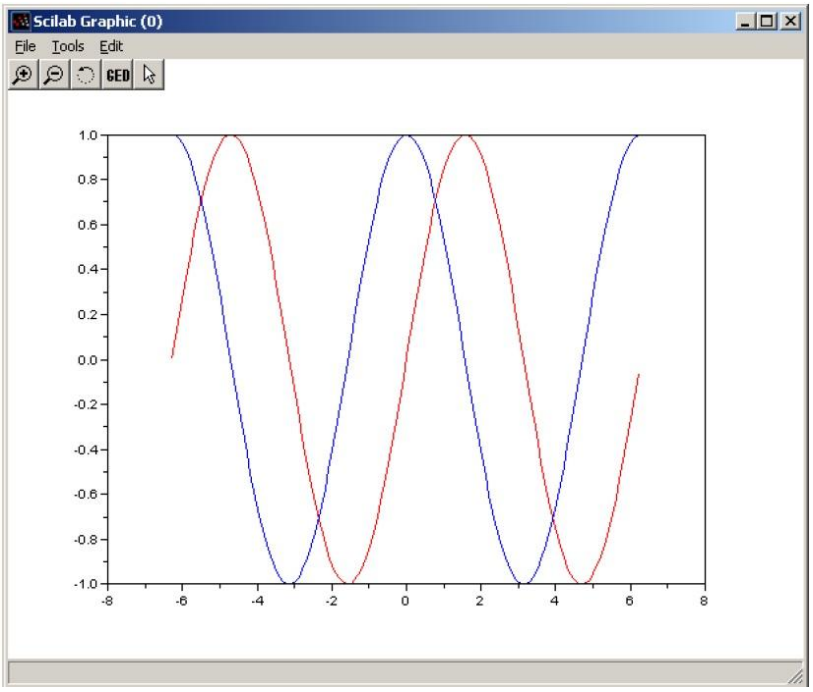
В общем виде обращение к функции имеет вид [5]: *plot2d([loglog],x,y,[key1=value1,key2=value2, ..., keyn=valuen]*, где

- logflag – строка из двух символов, каждый из которых определяет тип осей (n- нормальная ось, l – логарифмическая ось), по умолчанию "nn";
- x – массив абсцисс;
- y – массив ординат(или матрица, каждый столбец которого содержит массив ординат очередного графика) (количество элементов в массиве x и y должно быть одинаковым), если x и y – являются матрицами одного размера, то в этом случае, каждый столбец матрицы y отображается относительно соответствующего столбца матрицы x;
- keyi=valuei – последовательность значений параметров графиков, возможны следующие значения параметров: style – определяет массив (mas) числовых значений цветов графика (id цвета), количество элементов массива совпадает с количеством изображаемых графиков, по умолчанию, по умолчанию представляет собой массив masi=i, цвет i-й линии совпадает с номером i, для формирования id соответствующего цвета (кода цвета) можно воспользоваться функцией color, которая по названию (color("цвет")) или коду grb (color(r,g,b)) цвета формирует нужный id (код) цвета. Если значение стиля отрицательное то это будет точечный график без соединения точек между собой линиями.

- rect – этот вектор [xmin, ymin, xmax, ymax] определяет размер окна вокруг графика;
- frameflag – параметр определяет окно в котором, будет изображаться график, он может принимать следующие значения: 0 – не вычислять размеры окна, использовать значения по умолчанию или значения из предыдущего графика, 1 – размер окна определяется параметром rect, 2 – размер окна определяется из соотношения между минимальным или максимальным значениями x и y, 3 – размер окна определяется параметром rect в в изометрическом масштабе, 4– размер окна определяется из соотношения между минимальным или максимальным значениями x и y в изометрическом масштабе;
- axesflag - параметр, который определяет рамку вокруг графика, следует выделить следующие значения этого параметра: 0 – нет рамки вокруг графика; 1 – изображение рамки, ось y слева; 3 – изображение рамки, ось y справа; 5 – изображение осей проходящих через точку (0,0);
- nax – этот параметр используют, если параметр axesflag равен 1, nax представляет массив из четырех значений [nx, Nx, ny, Ny] – где Nx (Ny) – число основных делений с подписями под осью X (Y), nx (ny) – число промежуточных делений;
- leg – строка, определяющая легенды для каждого графика, структура строки такая: "leg1@leg2@leg3@...@legn", где leg1 – легенда первого графика, ..., legn – легенда первого графика.

Пример построения нескольких графиков различного цвета приведен ниже:

```
x=[-2*%pi:0.1:2*%pi];
y=[sin(x); cos(x)];
plot2d(x,y',style=[color("red"), color("blue")]);
```



Функция plot2d

Функцию plot2d [5] можно использовать для построения точечных графиков. В этом случае обращение к функции имеет вид *plot2d(x,y,d)*, здесь d – отрицательное число, определяющее тип маркера.

Оформление графиков

Рассмотрим основные возможности Scilab по оформлению графиков.

Изображение сетки на графике

Для изображения сетки следует воспользоваться функцией xgrid(color), color определяет id цвета линии сетки.

Заголовки на графике

Для вывода заголовков на графике служит функция xtitle(title, xstr, ystr), здесь title – название графика, xstr – название оси X, ystr - название оси Y.

Нанесение легенд на график

Для нанесения легенд на график служит функция legend(leg1,leg2,...,legn[,pos][,boxed]), где leg1 – имя первой легенды, leg2 – имя второй легенды, ... , legn – имя n-й легенды; pos – месторасположение легенды: 1 – верхнем правом углу (по умолчанию), 2 – верхнем левом углу, 3 – нижнем левом углу, 4 – нижнем правом углу, 5 - определяется пользователем после изображения графика; boxed – логическая переменная (по умолчанию %t), которая определяет рисовать или нет рамку вокруг легенды.

Кроме того, Scilab позволяет после вывода графика перейти в режим форматирования с помощью команды Edit – Figure proterties в окне график.

Построение трехмерных графиков в Scilab

В целом процесс построения графика функции вида Z(x,y) можно разделить на три этапа [5]:

1. Создание в области построения графика прямоугольной сетки. Для этого формируются прямые линии, параллельные координатным осям x_i и y_j , где

$$x_i = x_0 + ih,\ h = \frac{x_n - x_0}{n},\ i = 0,1\ldots,n \quad \text{и} \quad y_j = y_0 + jh,\ h = \frac{y_k - y_0}{k},\ j = 0,1\ldots,k\ .$$

2. Вычисление значений функции $z_{ij} = f(x_i , y_j)$ во всех узлах сетки.

3. Обращение к функции построения трехмерной графики.

Функция plot3d и plot3d1

В Scilab поверхность можно построить с помощью функций plot3d или plot3d1. Их отличие состоит в том, что plot3d строит поверхность и заливает ее одним цветом, а plot3d1 строит поверхность, каждая ячейка которой имеет цвет, зависящий от значения функции в каждом соответствующем узле сетки.

Обращение к функциям следующее:

plot3d(x,y,z,[theta,alpha,leg,flag,ebox][keyn=valuen]),

plot3d1(x,y,z,[theta,alpha,leg,flag,ebox][keyn=valuen]),

здесь x - вектор -столбец значений абсцисс;

y - вектор -столбец значений ординат;

z - матрица значений функции;

theta, alpha - действительные числа , которые определяют в градусах сферические координаты угла зрения на график. Попросту говоря, это угол, под которым наблюдатель видит отображаемую поверхность;

leg - подписи координатных осей графика;

flag – массив, состоящий из 3 целочисленных параметров [mode,type,box]. Здесь mode - устанавливает цвет поверхности (см . табл . 5.1).

ebox - определяет границы области , в которую будет выводиться поверхность , как вектор [xmin,xmax,ymin,ymax,zmin,zmax]. Этот параметр может использоваться только при значении параметра type=1;

keyn=valuen - последовательность значений свойств графика key1=value1, key2=value2,...,keyn=valuen, таких как толщина линии , ее цвет , цвет заливки фона графического окна, наличие маркера и др.

Таким образом, функции plot3d (plot3d1) в качестве параметров необходимо передать прямоугольную сетку и матрицу значений в узлах сетки.

Задание 1. Постройте графики функций**:**

$$f(x) = \frac{3}{x^3} + \frac{2}{x^2} + \frac{1}{x};$$

$$z(x, y) = \sin\left(\frac{x}{y}\right)\cos\left(\frac{x}{y}\right);$$

Лабораторная работа №7
Дифференциальное и интегральное исчисление

Цель работы: получение навыков нахождения производных функций и решения интегралов с использованием математических пакетов.

Дифференциальное и интегральное исчисление в Scilab
Интегрирование в Scilab

Вычислить определенный интеграл в Scilab можно при помощи функции int (a, b, f), где a и b —нижний и верхний пределы интегрирования соответственно, f — имя подынтегральной функции [2].

Нахождение интеграла будет выглядеть следующим образом [2]:

*-->function y=f(x),y=3*cos(x/2) endfunction*

-->intg(0,%pi,f)

ans =6.

Вычисление производной в Scilab

В Scilab можно вычислять производную функции в заданной точке. Вычисление происходит при помощи команды numdiff(f,x0), где f — имя дифференцируемой функции переменной x, x0 — координата точки в которой необходимо вычислить производную [2].

Задание 1. Найти производную первого порядка в точке x=1:

$y = \sin^3(x)$.

Задание 2. Найти производные 1го, 2го и 3го порядков в указанных точках:

$f(x) = \frac{x}{2x-1}$, $f'(0), f''(1), f'''(2)$.

Задание 3. Вычислить определенный интеграл:

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

Лабораторная работа №8
Программирование в Smath Studio и Scilab

Цель работы: получение навыков программирования с использованием математических пакетов.

Функции программирования
Программирование линейного вычислительного процесса

Линейным называется вычислительный процесс, в котором операторы выполняются последовательно, один за другим.

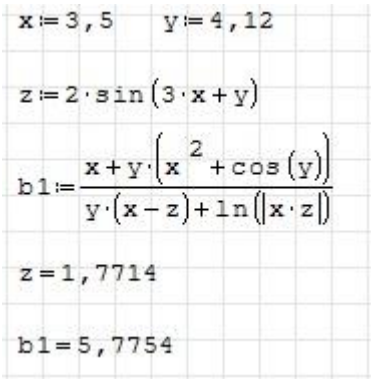
Рассмотрим на примере процесс программирования линейного алгоритма в Smath Studio.

Пример. Составить программу для вычисления функций:

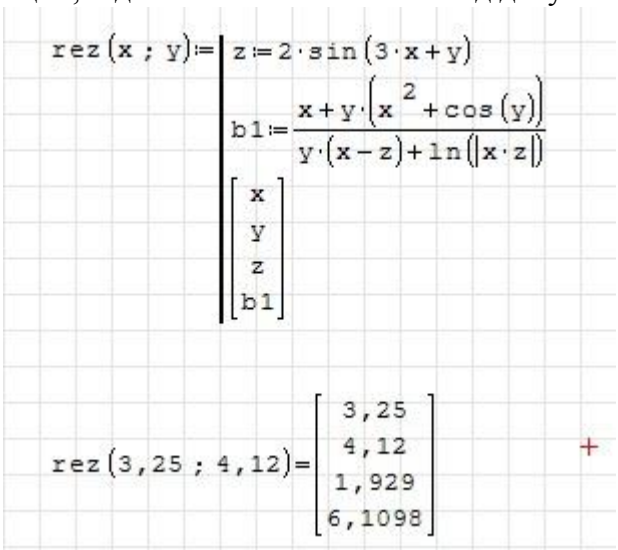
$$b = \frac{x + y(x^2 + \cos y)}{y(x - z) + \ln|xz|},$$

где $z = 2\sin(3x + 1); x = 3{,}25; y = 4{,}12.$

Решение. При решении данной задачи в SMath Studio, получим следующий документ:



Таким образом, выполняется расчет, в котором исходные данные не изменяются. Если же необходимо неоднократно выполнить один и тот же расчет, но для различных исходных данных, рациональнее средствами программирования SMath Studio записать вычисления в виде функции, заданной пользователем. Вид документа SMath Studio:



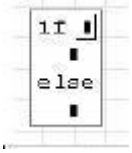
Здесь имя функции rez() выбирается пользователем, в скобках указываются имена вводимых переменных, после знака ":=" на ПИ «Программирование» выбирают команду line. Появится линия и два «квадратика» для ввода операторов. Чтобы таких «квадратиков» стало 3, как в рассматриваемом примере, нужно мышью потянуть окно шаблона вниз за правый нижний угол. В 1-й и 2-й строке записываем операторы для вычисления z и b1. Для организации вывода четырех переменных в 3-й строке с помощью ПИ «Матрица» записываем вектор.

Теперь, записав имя функции и значения x и y, можно получить результат для любого набора исходных данных.

Программирование разветвляющегося вычислительного процесса

Разветвления в программе возникают при необходимости выбора одного из нескольких возможных путей в решении задачи. Для организации разветвлений в программах используется оператор перехода.

Оператор условного перехода if выбирается на ПИ «Программирование» и позволяет изменить порядок выполнения операторов в программе в зависимости от определенных условий. Общий вид оператора:



Если условие, заданное в операторе if, истинно, то выполняется оператор (простой или составной), стоящий во второй строке. В противном случае выполняется оператор, стоящий после else. После выполнения одной из ветвей, работа программы продолжается с оператора, следующего за if.

Если в какой-то ветви требуется выполнить более одного оператора, из них необходимо образовать составной оператор, т. е. заключить эти операторы в операторные скобки – оператор line на ПИ «Программирование».

Программирование циклического вычислительного процесса

Циклическим называется вычислительный процесс, содержащий многократные вычисления по одним и тем же математическим зависимостям, но для различных значений входящих в него переменных. Количество повторений может задаваться заранее или зависеть от выполнения определенного условия, как в операторе if.

В SMath Studio используют 3 оператора цикла: while, for(3) и for(4).

С помощью оператора while можно реализовать циклический процесс, состоящий из ряда операторов, который выполняется до тех пор, пока выполняется определенное условие:

while (условие)

оператор

Если в цикле необходимо выполнить более одного оператора, то их следует заключить в операторные скобки (line), т. е. образовать из них составной оператор:

while (условие)

оператор 1

оператор 1

До тех пор, пока соблюдается условие, последовательно выполняется тело цикла. Если условие не соблюдается, то выполнение программы продолжается, начиная с оператора, следующего за циклом.

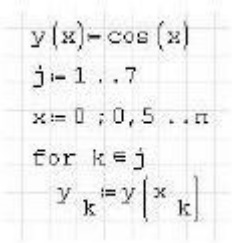
for(4) – это цикл со счетчиком, для которого нужно задать количество повторений:

$for\ k := 1; k \le n; k := k + h$

оператор

где k – переменная-счетчик, n – количество повторений, h – шаг изменения переменной k.

for(3) – это цикл, аналогичный for(4), но в нем счетчик цикла x меняется сам, и принадлежит некоторому заданному диапазону:



Основные операторы sci-языка. Функции ввода-вывода в Scilab

Для организации простейшего ввода в Scilab можно воспользоваться функциями

x=input('title');

или

x=x_dialog('title', 'stroka').

Функция *input* выводит в командной строке Scilab подсказку title и ждет пока пользователь введет значение, которое в качестве результата возвращается в переменную x.

Функция *x_dialog* выводит на экран диалоговое окно с именем title, после чего пользователь может щелкнуть OK и тогда stroka вернется в качестве результата в

переменную x, либо ввести новое значение вместо stroka, которое и вернется в качестве результата в переменную x [5].

Функция *input* преобразовывает введенное значение к числовому типу данных, а функция *x_dialog* возвращает строковое значение. Поэтому при использовании функции *x_dialog* для ввода числовых значений, возвращаемую ею строку следует преобразовать в число с помощью функции *evstr*. Поэтому можно предложить следующую форму использования функции *x_dialog* для ввода числовых значений:

x=evstr(x_dialog('title', 'stroka')).

Для вывода в текстовом режиме можно использовать функцию *disp* следующей структуры: *disp(b)*, где

b - имя переменной или заключенный в кавычки текст [5].

Оператор присваивания

Оператор присваивания имеет следующую структуру: a=b, здесь a - имя переменной или элемента массива, b - значение или выражение. В результате выполнения оператора присваивания переменной a присваивается значение выражения b.

Условный оператор

Одним из основных операторов, реализующим ветвление в большинстве языков программирования, является условный оператор *if*. Существует обычная и расширенная формы оператора *if* в Scilab. Обычный *if* имеет вид:

if условие
оператор1
else
оператор2
end

Здесь условие - логическое выражение; оператор1, оператор2 - операторы языка Scilab или встроенные функции.

Оператор *if* работает по следующем алгоритму: если условие истинно, то выполняется оператор1, если ложно то оператор2.

В Scilab для построения логических выражений могут использоваться условные операторы: &, and (логическое и), |, or (логическое или), ~, not (логическое отрицание) и операторы отношения: < (меньше), > (больше),== (равно), ~=, <> (не равно), <= (меньше или равно), >= (больше или равно) [5].

Зачастую при решении практических задач недостаточно выбора выполнения или невыполнения одного условия. В этом случае можно, конечно, по ветке *else* написать новый оператор *if*, но лучше воспользоваться расширенной формой оператора *if:*

if условие1оператор1
elseif условие2
оператор2
elseif условие3
оператор3
… elseif условие n
операторn
else
оператор
end

В этом случае оператор *if* работает так: если условие1 истинно, то выполняется оператор1, иначе проверяется условие2, если оно истинно, то выполняется оператор2, иначе проверяется условие3 и т.д. Если ни одно из условий по веткам *else* и *elseif* не выполняется, то выполняются операторы по ветке *else* [5].

Оператор альтернативного выбора

Еще одним способом организации разветвлений является оператор *select* альтернативного выбора следующей структуры:

select параметр
case значение1 then оператор1
case значение2 then оператор2
… else оператор
end

Оператор *select* работает следующим образом: если значение параметра равно значениию1, то выполняется оператор1, иначе если параметр равен значениию2, то выполняется оператор2, в противном случае, если значение параметра совпадает со значением3, то выполняется оператор3 и т.д. Если значение параметра не совпадает ни с одним из значений в группах case, то выполняются операторы, которые идут после служебного слова *else* [5].

Оператор while

Оператор цикла while имеет вид:

while условие
операторы
end

Здесь условие - логическое выражение.

Операторы будут выполняться циклически, пока логическое условие истинно. Оператор цикла while обладает значительной гибкостью, но не слишком удобен для организации «строгих» циклов, которые должны быть выполнены заданное число раз [5].

Оператор for

Оператор цикла for имеет вид:

for x=xn:hx:xk
операторы
end

Здесь x - имя скалярной переменной параметра цикла, xn - начальное значение параметра цикла, xk - конечное значение параметра цикла, hx - шаг цикла . Если шаг цикла равен 1, то hx можно опустить, и в этом случае оператор for будет таким:

for x=xn:xk
операторы
end

Выполнение цикла начинается с присвоения параметру стартового значения (x=xn). Затем следует проверка, не превосходит ли параметр конечное значение (x>xk). Если x>xk, то цикл считается завершенным, и управление передается следующему за телом цикла оператору. Если же x≤xk , то выполняются операторы в цикле (тело цикла). Далее параметр цикла увеличивает свое значение на hx (x=x+hx). После чего снова производится проверка значения параметра цикла, и алгоритм повторяется [5].

Задание 1. Подготовить описание функции, вычислить значение этой функции при x_1 и x_2:

$$F=\frac{3y+x^2z}{\pi}$$

$$\begin{cases} y=\dfrac{-8x\cdot\sin x}{e\sqrt{|x|}}, & z=\dfrac{8}{-x}, & x\le 0; \\ y=\dfrac{0{,}8x}{|\sin x|}+\cos\left(x-\dfrac{\pi}{3}\right), & z=\dfrac{2x}{\sqrt{x^3-1}}, & x>0. \end{cases}$$

$x_1=-2.34; x_2=5.65.$

Задание 2. Дана последовательность из n целых чисел. Найти количество элементов последовательности, кратных числу 2 и не кратных числу 3.

Раздел 2. Указания к самостоятельной работе студентов (СРС) и контрольные вопросы для оценивания

Вид самостоятельной работы:

1. Выполнение индивидуальных заданий и подготовка отчетов по лабораторным работам.
2. Текущая проработка теоретического материала учебников и лекций, в том числе тем, вынесенных для самостоятельного изучения:

- Решение обыкновенных дифференциальных уравнений в пакетах Smath Studio и Scilab
- Решение дифференциальных уравнений в частных производных в пакетах Smath Studio и Scilab.

Контрольные вопросы

1. Основные пакеты прикладных программ и их особенности.
2. Smath Studio и Scilab как пакет прикладных программ.
3. Способы представления информации в Smath Studio и Scilab.
4. Переменные в Smath Studio и Scilab.
5. Операции над матрицами в Smath Studio и Scilab.
6. Функции в Smath Studio и Scilab: основные и дополнительные.
7. Решение линейных уравнений в Smath Studio и Scilab.
8. Решение систем линейных уравнений в Smath Studio и Scilab.
9. Решение нелинейных уравнений в Smath Studio и Scilab. Точные и численные решения.
10. Графики в Smath Studio и Scilab: назначение, типы, способы задания.
11. Графики простейших функций в Smath Studio и Scilab.
12. Оформление графиков в Smath Studio и Scilab.
13. Графики сложных, параметрических и неявно заданных функций в Smath Studio и Scilab.

Список использованной литературы

1. Троицкая О.Н. Применение пакетов прикладных программ в математике: учеб. пособие / О.Н. Троицкая, Н.Н. Конечная; Сев. (Арктич.) федер. ун-т им. М.В. Ломоносова. - Архангельск: САФУ, 2015.- 100 с.
2. Тропин И.С., Михайлова О.И., Михайлов А.В. Численные и технические расчеты в среде Scilab (ПО для решения задач численных и технических вычислений): учебное пособие. — Москва: 2008. — 65 с.
3. Алексеев Е. Р. Scilab: Решение инженерных и математических задач / Е. Р. Алексеев, О. В. Чеснокова, Е. А. Рудченко. — М. : ALT Linux ; БИНОМ. Лаборатория знаний, 2008. — 260 с.