Алексеев В.В.

Основы научно-технических и инженерных расчетов в Scilab
Лабораторный практикум
по курсу ИНФОРМАТИКА



СарФТИ НИЯУ МИФИ

**Саровский физико-технический институт-
филиал Национального исследовательского
ядерного университета «МИФИ»**

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ЭЛЕКТРОНИКИ
Кафедра вычислительной и информационной техники

Алексеев В.В.

Основы научно-технических и инженерных расчетов в Scilab

***Лабораторный практикум
по курсу ИНФОРМАТИКА***

Саров

УДК 621.
ББК 22.176
А47

Алексеев В.В. – Основы научно-технических и инженерных расчетов в Scilab. Лабораторный практикум по курсу Информатика.

Учебно-методическое пособие. СарФТИ НИЯУ МИФИ, 2026г.

Данное пособие лабораторный практикум по курсу «Информатика» и предназначено для студентов младших курсов бакалавриата, обучающихся по направлениям подготовки «09.00.00-Информатика и вычислительная техника».

В пособии изложен необходимый теоретический материал, необходимый для выполнения требуемых расчетов в интерактивной среде Scilab. Предложены варианты для самостоятельного выполнения заданий. Приведены контрольные вопросы по изучаемой теме. Приведены основные требования по оформлению отчета о каждой выполненной работе.

Данный лабораторный практикум позволяет освоить основы научно-технических и инженерных расчетов с использованием компьютерных технологий.

Данное пособие может быть также полезно студентам и других направлений подготовки для выполнения различных расчетных работ.

© Издательство СарФТИ 2026

Содержание

Наименование	Стр.
Лабораторная работа № 1 Элементы рабочей среды Scilab и простейшие вычисления	3
Лабораторная работа № 2 Задание функций пользователя и их реализация в среде Scilab	17
Лабораторная работа № 3 Решение простейших научно-технических и инженерных задач с использованием базовых конструкций языка Scilab	26
Лабораторная работа № 4 Построение графиков в Scilab	36
Лабораторная работа № 5 Численное дифференцирование интегрирование в Scikab	75
Лабораторная работа № 6 Решение системы линейных алгебраических уравнений в Scilab	85
Лабораторная работа № 7 Решение нелинейных и трансцендентных уравнений в Scilab	90
Лабораторная работа № 8 Аппроксимация функций в Scilab	101



Лабораторная работа № 1.

«Элементы рабочей среды Scilab и простейшие вычисления»

Цель работы: Изучение и освоение возможностей рабочей среды пакета математического моделирования **Scilab** для практического использования в учебно-научной и инженерной деятельности. Приобретение практических навыков проведения простейших научно-технических вычислений в командном окне Scilab.

1.1.Теоретическая часть.

1.1.1 Рабочая среда Scilab

После запуска пакета Scilab на экране дисплея появляется его ***Рабочая среда*** (интерфейс пользователя) в стандартной конфигурации (рис. 1.1). При этом система готова к проведению вычислений по введенным командам в ***командном окне.***

Все команды отображаются на панели в виде пиктограмм с соответствующим изображением, подсказывающим их назначение.

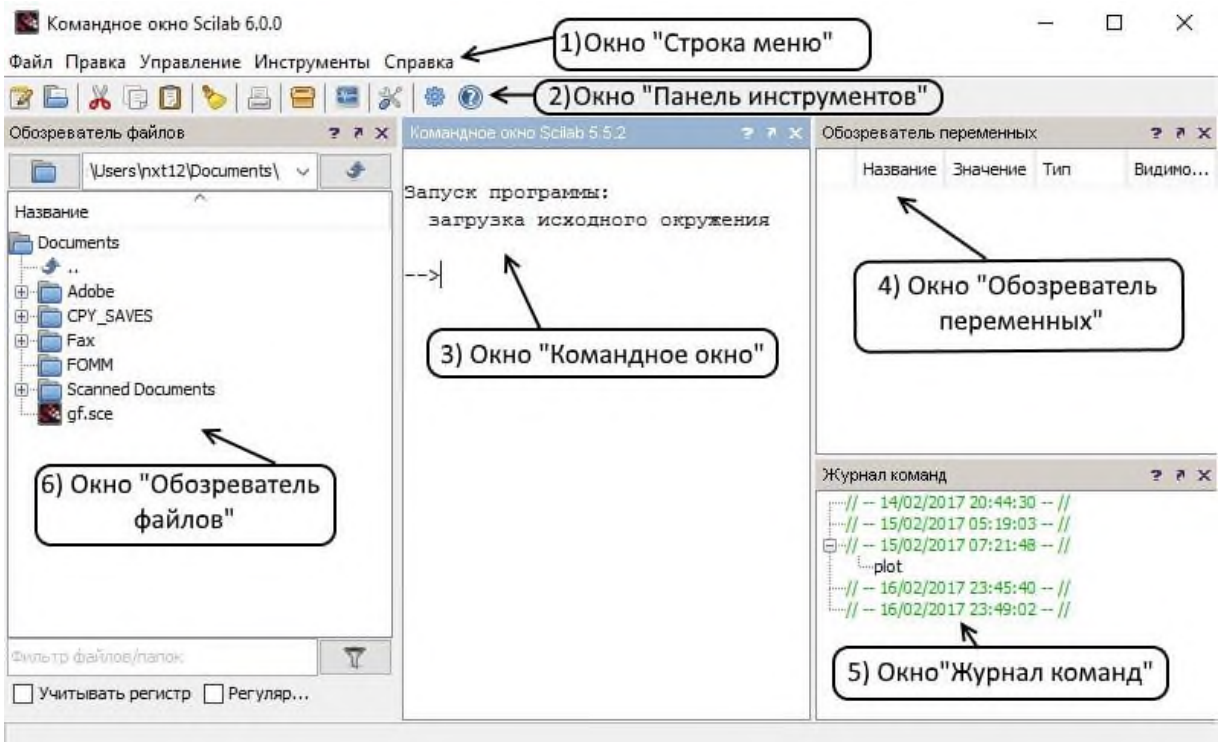


Рис. 1.1

Командное окно предназначено для отображения вводимых команд, результатов их выполнения, а также сообщений об ошибках, то есть для работы пользователя с пакетом Scilab в интерактивном режиме.

Окно Обозреватель переменных отображает содержимое рабочего пространства объектов Scilab, и позволяет выполнять определенные действия с объектами этого пространства (скалярными переменными, векторами, матрицами, функциями и др.).

Окно Журнал команд предназначено для хранения, просмотра и повторного вызова ранее введенных командных строк.

Окно Обозреватель файлов предназначено для просмотра и установки путей доступа к файловой системе и выбора текущей папки, в которой хранится нужный файл, поскольку перед работой с конкретным файлом (чтение или запись файла), к нему требуется установить путь доступа. Внизу окна отображается фильтр файлов/папок.

В стандартной конфигурации четыре основных окна, вписанные в **Рабочую среду** Scilab, могут передвигаться вместе с основным окном и вместе с ним изменять свои размеры. Если в процессе сеанса работы конфигурация окна нарушена, то ее можно восстановить. Для этого на панели инструментов следует нажать кнопку ***Настройки Scilab***, и в появившемся меню в левой части одноименного окна (рис. 1.2) выбрать команду ***Общие***, и затем выбрать ***Конфигурация рабочего стола***.

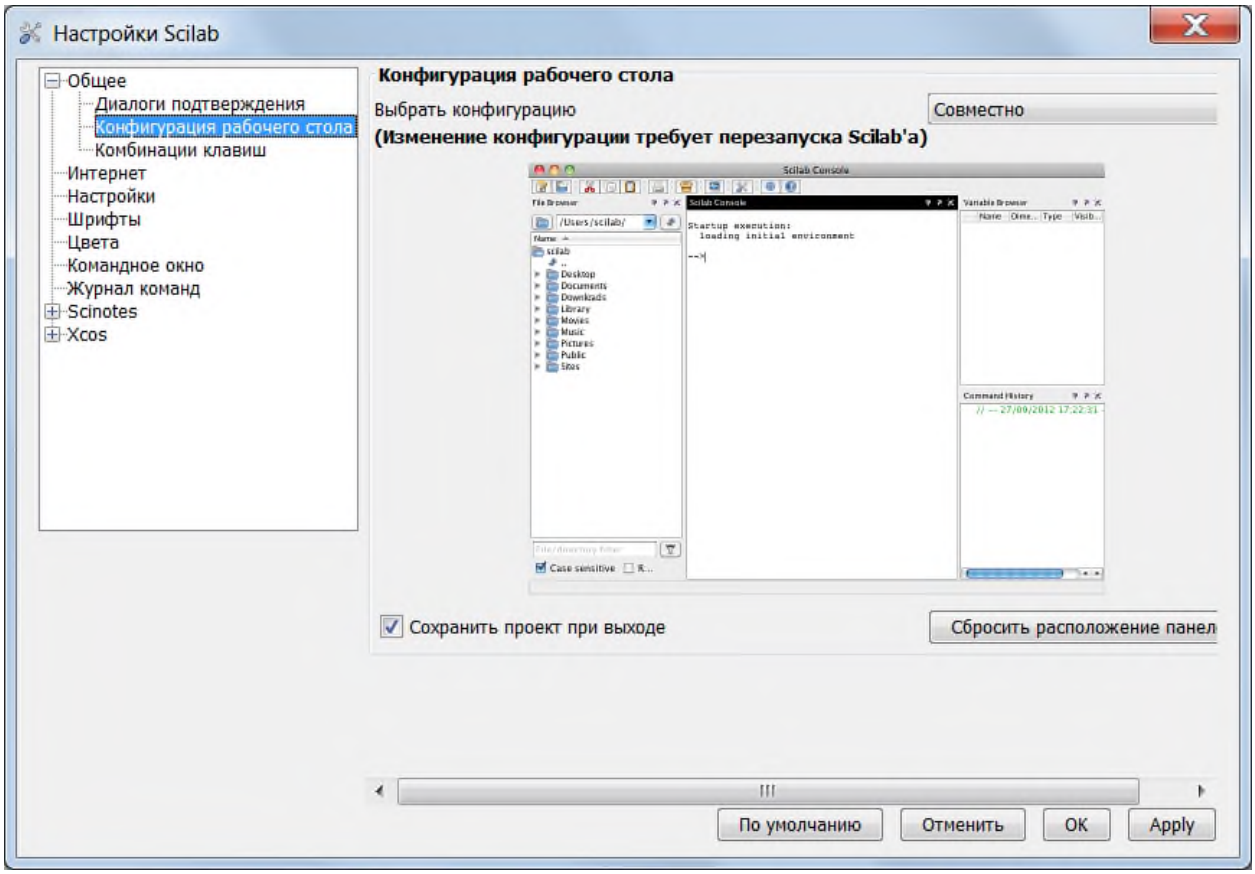


Рис. 1.2

Далее надо последовательно щелкнуть по кнопкам ***Сбросить расположение панелей***, ***Apple***, (или Применить) и ***OK***.

Следует помнить, что при изменении конфигурации рабочего стола требуется перезагрузка Scilab.

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. В этой строке можно набирать формулы или команды, удовлетворяющие синтаксису языка SciLab и завершающиеся нажатием клавиши Enter. В ней действуют элементарные приемы редактирования:

- ✓ → перемещение курсора вправо на один символ;
- ✓ ← перемещение курсора влево на один символ;
- ✓ **Home** - перемещение курсора в начало строки;
- ✓ **End** - перемещение курсора в конец строки;
- ✓ **Del** - удаление символа после курсора;
- ✓ **Backspace** - удаление символа перед курсором.

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.

На выбор имен накладываются следующие ограничения:

- ✓ первый символ имени – это либо латинская буква, либо один из символов ‘%’, ‘_’, ‘#’, ‘!’, ‘$’, ‘?’;
- ✓ последующие символами могут быть латинские буквы, цифры или символы ‘_’, ‘#’, ‘!’, ‘$’, ‘?’;
- ✓ большие и малые буквы в именах различаются;
- ✓ длина может быть любой, но учитываются только первые 24 символа, которые должны обеспечивать уникальность имени.

Для установки формата представления чисел для печати и вывода на экран используется команда:

-->**format**([type],[long]),

где type - символьная строка (“v” – для переменного формата по умолчанию; “e” - для e-формата); значение long определяет максимальное число знаков (по умолчанию 10).

Для просмотра рабочей области экрана можно воспользоваться стандартными средствами, как, например, полосами прокрутки, или клавишами перемещения курсора Page Up, Page Down.

Клавиши ↑ и ↓ также управляют курсором, однако в Scilab они имеют и 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 ↑, то появится последняя вводимая команда, повторное нажатие вызовет предпоследнюю и так далее. Клавиша ↓ выводит команды в обратном порядке. Таким образом, можно сказать, что вся информация в рабочей области находится или в зоне просмотра, или в зоне редактирования.

Следует отметить, 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.

Имена и значения переменных рабочего пространства можно сохранить в бинарный файл либо с помощь команды главного меню **Файл→ Сохранить окружение**, либо набрать команду в текущей строке:

-->**save**(filename [,x1,x2,...,xn]),

где filename–это строка, которая указывает имя файла и путь к нему. Файлы в Scilab имеют расширение .sod. (В более старых версиях расширение .dat). Если не указывать переменные x1,x2,...,xn, то будут сохранены все переменные файлового пространства.

Для загрузки переменных в начале следующей сессии достаточно выполнить команду l**oad:**

-->**load**(filename [,x1,...,xn])

Если команда **load** будет использована в середине сессии, то текущие значения переменных изменятся на загружаемые.

Для того чтобы очистить все рабочее пространство используется команда **clc** без параметров, но если указать список переменных, то она удалит только заданные переменные.

Команда **whos**() позволяет получить подробную информацию о переменных рабочего стола. Также может быть использована команда **who**, которая выводит только список имен переменных. Для просмотра значения любой переменной достаточно просто набрать ее имя и нажать клавишу Enter.

Стандартная команда **save**(filename [,x1,x2,...,xn]) не позволяет сохранить на диск содержание сессии, поэтому если существует такая

необходимость, то следует воспользоваться специальной командой для ведения так называемого дневника сессии:

-->**diary**(filename),

или—> [id,filename] = **diary**(filename, ['new'|'append']).

Например,

-->**diary**('D:/work/test402.txt')

Эта команда ведет запись на диск всех команд в строках ввода и полученных результатов в виде текстового файла с указанным именем filename ('D:/work/test402.txt'); id – это идентификатор (число) дневника.

Для окончания записи в файл используется команда:

-->**diary**(filename, close),

или

-->**diary**(id, close).

Приостановить запись можно с помощью команды:

-->**diary**(filename, pause|'off')

или

-->**diary**(id, 'pause'|'off')

Возобновляется запись в дневник командами:

-->**diary**(filename, 'resume'|'on')

или

-->**diary**(id, 'resume'|'on')

Дневник сессии сохраняется в формате .txt и просмотреть его можно с помощью Блокнота.

Закончить работу с программой можно с помощью команды

-->**exit**

1.1.2 Элементарные математические функции

Пакет Scilab снабжен достаточным количеством всевозможных встроенных функций. Приведем часто используемые элементарные математические функции:

Таблица 1.1 Тригонометрические функции

Функция	Описание функции
sin(x)	Синус числа x
cos(x)	Косинус числа x
tan(x)	Тангенс числа x
cotg(x)	Котангенс числа x
asin(x)	Арксинус числа x

acos(x)	Арккосинус числа x
atan(x)	Арктангенс числа x
exp(x)	Экспонента числа x
log(x)	Натуральный логарифм от числа x
sqrt(x)	Корень квадратный из числа x
abs(x)	Модуль числа x
log10(x)	Десятичный логарифм от числа x
log2(x)	Логарифм по основанию 2 от числа x

Пример вычисления $Z = \sqrt{|\sin(\frac{a}{b})|} \cdot e^{a^b}$:

```
--> a=1; b=0.25;
--> Z=sqrt(abs(sin(a/b)))*exp(a^b)
 Z  =
  2.3647529
```

Запись комплексных величин, используемых в формулах, напоминает общепринятый математический стандарт. Мнимая часть записывается как произведение действительного числа и мнимой единицы (%i). Когда полные комплексные числа используются в операциях умножения, деления или возведения в степень, то для устранения неоднозначности их заключают в круглые скобки. В таблице 1.2 приведены функции для работы с комплексными числами.

Таблица 1.2 Функции для работы с комплексными числами

Наименование функции	**Выполняемое действие**
a=real(x)	Выделение вещественной части комплексного числа x
b=imag(x)	Выделение мнимой части комплексного числа x
x=complex(a,b)	Построение комплексного числа x по паре вещественных чисел, где a – действительная часть, b – мнимая часть числа x
y=conj(x)	Вычисление y – сопряженного комплексного числа для числа x
y=imult(x)	Умножение числа на мнимую единицу, которое позволяет избежать проблем, возникающих при записи y=%i*x, например, когда x содержит «особые» числа с плавающей запятой, такие как

	%inf и %nan.
t=isreal(x)	Проверка, имеет ли переменная вещественные или комплексные элементы. Если x хранится как вещественная переменная, то возвращает значение «истина»; если x имеет мнимую часть (даже нулевую), то возвращает переменную «ложь».

В таблицах 1.3 и 1.4 приведены базовые арифметические и логические операции, а также элементарные алгебраические функции.

Таблица 1.3. Арифметические и логические операции

Символ	**Выполняемое действие**
+	покомпонентное сложение числовых массивов одинаковой размерности; сложение скалярной величины для каждого элемента массива;
-	покомпонентное вычитание числовых массивов одинаковой размерности; вычитание скалярной величины из каждого элемента массива;
*	умножение матриц в соответствии с правилами линейной алгебры; умножение всех компонент массива на скаляр;
.*	покомпонентное умножение элементов массива одинаковой размерности;
/	деление скаляра на скаляр; покомпонентное деление всех элементов массива на скаляр; A/B=A*B-1=A*inv(B) (A и B – квадратные матрицы одного порядка)
./	покомпонентное деление элементов массива одинаковой размерности;
^	возведение скаляра в любую степень; вычисление целой степени квадратной матрицы;
‘	вычисление сопряженной матрицы;
.’	транспонирование матрицы;
	Логические операции
&	логическое умножение скаляров; логическое покомпонентное умножение массивов одинаковой размерности; логическое умножение массива на скаляр;
|	логическое сложение скаляров; логическое покомпонентное

	сложение массивов одинаковой размерности; логическое сложение массива со скаляром;
~	логическое отрицание скаляра или всех элементов массива;
= =	проверка на равенство;
~= **или** <>	проверка на неравенство;
>	проверка на «больше»
>=	проверка на «больше или равно»
<	проверка на «меньше»
<=	проверка на «меньше или равно»

Таблица 1.4. Элементарные алгебраические функции

Категория функций	**Наименования функций и их действия**
Тригонометрические	Для аргумента в радианах: cos()–косинус, cotg() - котангенс, sin() - синус, tan() - тангенс; Для аргумента в градусах: cosd() - косинус, cotd() - котангенс, sind() – синус, tand() – тангенс.
Обратные тригонометрические	Для результата в радианах: acos() - арккосинус, acot() - арккотангенс, asin() - арксинус, atan() - арктангенс; Для результата в градусах: acosd() - арккосинус, acotd() - арккотангенс, asind() - арксинус, atand() – арктангенс.
Экспонента, логарифмы, корень	exp() - показательная функция (экспонента), log() – натуральный логарифм, log2() - логарифм по основанию 2, log10() - десятичный логарифм, sqrt() - квадратный корень
Округления	ceil() - округляет до ближайшего большего числа, fix() или int() - просто отбрасывает дробную часть, floor() - до ближайшего меньшего числа, round() - округление до ближайшего целого
Дискретная математика	gcd([p1 p2 …])– наибольший общий делитель для элементов вектора-строки, lcm([p1 p2 …]) – наименьшее общее кратное, factor(x) – разложение на множители числа x, factorial(n) – факториал n, primes(x) возвращает все простые числа в интервале между 1 и x включительно.

Модуль числа	abs()
Знак числа	sign()
Остаток от деления	k=modulo(n,m) - симметричный арифметический остаток от деления n на m (для целых чисел): k=n-m.*int(n./m), k=pmodulo(n,m) – положительный арифметический остаток от деления n на m (для целых чисел): k=n- |m|.*floor(n./|m|)

1.2. Лабораторное задание.

В соответствии с заданным вариантом, приведенным в таблице 1.2.1, выполнить лабораторное задание:

1.2.1. Изучите теоретический материал к лабораторной работе. Настройте командное окно (рабочую область, шрифты, размеры, цвет шрифта.

1.2.2. Установите путь к папке, находящейся на Вашей флешь-карте, для сохранения содержимого командного окна.

1.2.3. В соответствии с заданным вариантом, приведенным в таблице 1.2.1, выполнить лабораторное задание:

a) Вычислить значения a и *b*;
b) С помощью команды **format**() вывести значение выражения *f* на экран в возможных форматах с разным количеством знаков;
c) Выведите значение *A* с *n* знаками после запятой;
d) Задать значения комплексных чисел *A* (с помощью i) и *B* (с помощью функции complex()). Найти сумму, разность, произведение заданных чисел. Выделить действительную и мнимую части заданных чисел. Определить их модуль и аргумент;
e) Подсчитать значения косинуса, синуса, тангенса и котангенса для α и для β;
f) Округлить числа *N*1, *N*2, *N*3 всеми возможными способами;
g) Вычислить остаток от деления двумя способами для чисел: *m* и *n*;
h) Вычислить НОД и НОК для чисел: *a, b, c*. Разложите указанные числа на множители. Найдите все простые числа, их не превышающие;

1.2.4. С помощью оператора **rand**() сгенерируйте два целочисленных вектора *V*1 и *V*2 размерности 10 со значениями $|a_i| < 100$;

a) Выполните покомпонентное сложение, умножение, вычитание и деление векторов;
b) Вычислите значение десятичного логарифма каждой компоненты вектора *V*1 и *V*2;
c) Выполните покомпонентное сложение полученных векторов с округлением результата до 2-го (3-го) знака после запятой;
d) Для заданных значений i, j, k, являющихся номерами элементов заданных векторов, выполните действия:

$$V1(i) \cdot V2(k)/(V1(j) \cdot \sqrt[5]{V2(j)} + \log_k(V1(i) + V2(k)));$$

1.2.5. Полученный результат выведите в экспоненциальной форме.
1.2.6. Выполните отчет по выполненной работе.
1.2.7. Ответьте на контрольные вопросы.

Варианты индивидуальных заданий Таблица 1.2.1.

№ В.	Значения заданных параметров для пунктов задания						
	a)					b)	c)
1	a) $4.5 \cdot \left(0.64^{\frac{1}{3}} - \left(2\frac{5}{7}\right)^{-1}\right) : \left(-3\frac{6}{11}\right)$; b) $\sqrt{0.09} - \left(\sqrt{11} - 2\sqrt{5}\right) \cdot \left(\sqrt{20} - \sqrt{11}\right)$.					$f = 2 \cdot \pi$	*A*=22/7; *n*=5.
	d)	e)	f)	g)	h)	1.2.4. *d*)	

	A=3.5+6.4i;
	B=2.4-3.2i.
		α=11π/5;
		β=245°.
			*N*1=1.123;
			*N*2=1.345;
			*N*3=1.456.
				m=12;
				n=8.
					a=12;
					b=27;
					c=36.
						$i = 4$;
						$j = 7$;
						$k = 2$.	

	a)					b)	c)
2	a) $7.5 \cdot \left(0.45^{\frac{1}{3}} - \left(2\frac{5}{7}\right)^{-1}\right) : \left(-2\frac{9}{11}\right)$; b) $\sqrt{0.09} - \left(\sqrt{12} - 2\sqrt{5}\right) \cdot \left(\sqrt{20} - \sqrt{12}\right)$.					$f = 2 \cdot e$	*A*=11/7; *n*=6.
	d)	e)	f)	g)	h)	1.2.4. *d*)	

	A=2.5+6.4i;
	B=4.4-3.2i.
		α=2π/5;
		β=222°.
			*N*1=2.324;
			*N*2=2.245;
			*N*3=2.126.
				m=18;
				n=5.
					a=6;
					b=12;
					c=36.
						$i = 5$;
						$j = 8$;
						$k = 1$.	

	a)					b)	c)
3	a) $3.3 \cdot \left(0.85^{\frac{1}{3}} - \left(2\frac{5}{7}\right)^{-1}\right) : \left(-3\frac{6}{17}\right)$; *b*) $\sqrt{0.225} - \left(\sqrt{15} - 2\sqrt{5}\right) \cdot \left(\sqrt{20} - \sqrt{15}\right)$.					$f = \frac{\pi}{3}$	*A*=33/7; *n*=4.
	d)	e)	f)	g)	h)	1.2.4. *d*)	

	A=3.5+5.4i;
	B=2.4-2.2i.
		α=33π/5;
		β=345°.
			*N*1=3.133;
			*N*2=3.335;
			*N*3=3.436.
				m=35;
				n=8.
					a=15;
					b=20;
					c=25.
						$i = 9$;
						$j = 5$;
						$k = 4$.	

<table>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
<tr><td rowspan="3">4</td><td colspan="5">a) $4.5\cdot\left(0.44^{\frac{1}{3}}-\left(2\frac{5}{7}\right)^{-2}\right):\left(-4\frac{6}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{11}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{11}\right)$.</td><td colspan="2">$f=4\cdot\pi$</td><td>$A$=44/7;
n=7.</td></tr>
<tr><td>d)</td><td>e)</td><td>f)</td><td>g)</td><td>h)</td><td colspan="3">1.2.4. d)</td></tr>
<tr><td>A=4.5+5.4i;
B=4.4-3.2i.</td><td>α=11π/4;
β=135°.</td><td>$N1$=4.43;
$N2$=4.445;
$N3$=4.458.</td><td>m=21;
n=4.</td><td>a=14;
b=21;
c=28.</td><td colspan="3">$i=3$;
$j=2$;
$k=9$.</td></tr>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
<tr><td rowspan="3">5</td><td colspan="5">a) $5.5\cdot\left(0.64^{\frac{1}{3}}-\left(5\frac{5}{7}\right)^{-1}\right):\left(-3\frac{6}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{11}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{11}\right)$.</td><td colspan="2">$f=\frac{5}{e}$</td><td>$A$=5/7;
n=7.</td></tr>
<tr><td>d)</td><td>e)</td><td>f)</td><td>g)</td><td>h)</td><td colspan="3">1.2.4. d)</td></tr>
<tr><td>A=5.5+6.4i;
B=2.4-5.2i.</td><td>α=22π/5;
β=275°.</td><td>$N1$=4.123;
$N2$=5.345;
$N3$=6.456.</td><td>m=10;
n=8.</td><td>a=10;
b=15;
c=20.</td><td colspan="3">$i=7$;
$j=1$;
$k=4$.</td></tr>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
<tr><td rowspan="3">6</td><td colspan="5">a) $6.5\cdot\left(0.64^{\frac{1}{3}}-\left(6\frac{5}{7}\right)^{-1}\right):\left(-3\frac{6}{11}\right)$;
$b)\ \sqrt{0.36}-\left(\sqrt{13}-2\sqrt{3}\right)\cdot\left(\sqrt{12}-\sqrt{13}\right)$.</td><td colspan="2">$f=6\cdot e$</td><td>$A$=26/7;
n=6.</td></tr>
<tr><td>d)</td><td>e)</td><td>f)</td><td>g)</td><td>h)</td><td colspan="3">1.2.4. d)</td></tr>
<tr><td>A=6.5+5.4i;
B=5.4-3.2i.</td><td>α=11π/6;
β=244°.</td><td>$N1$=6.163;
$N2$=6.365;
$N3$=6.466.</td><td>m=12;
n=8.</td><td>a=12;
b=18;
c=36.</td><td colspan="3">$i=6$;
$j=9$;
$k=2$.</td></tr>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
<tr><td rowspan="3">7</td><td colspan="5">a) $7.5\cdot\left(0.64^{\frac{1}{3}}-\left(7\frac{5}{7}\right)^{-1}\right):\left(3\frac{7}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{7}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{7}\right)$.</td><td colspan="2">$f=7\cdot e$</td><td>$A$=44/7;
n=5.</td></tr>
<tr><td>d)</td><td>e)</td><td>f)</td><td>g)</td><td>h)</td><td colspan="3">1.2.4. d)</td></tr>
<tr><td>A=7.5-6.4i;
B=2.4-3.2i.</td><td>α=11π/7;
β=135°.</td><td>$N1$=7.773;
$N2$=7.775;
$N3$=7.777.</td><td>m=14;
n=8.</td><td>a=14;
b=21;
c=28.</td><td colspan="3">$i=4$;
$j=4$;
$k=8$.</td></tr>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
<tr><td rowspan="3">8</td><td colspan="5">a) $8.5\cdot\left(0.84^{\frac{1}{3}}+\left(2\frac{5}{7}\right)^{-1}\right):\left(-3\frac{8}{11}\right)$;
$b)\ \sqrt{0.64}-\left(\sqrt{11}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{11}\right)$.</td><td colspan="2">$f=8\cdot\pi$</td><td>$A$=88/7;
n=8.</td></tr>
<tr><td>d)</td><td>e)</td><td>f)</td><td>g)</td><td>h)</td><td colspan="3">1.2.4. d)</td></tr>
<tr><td>A=8.5+6.4i;
B=2.4-3.2i.</td><td>α=11π/6;
β=244°.</td><td>$N1$=1.823;
$N2$=1.385;
$N3$=1.458.</td><td>m=21;
n=8.</td><td>a=21;
b=35;
c=56.</td><td colspan="3">$i=5$;
$j=9$;
$k=2$.</td></tr>
<tr><td></td><td colspan="5">a)</td><td colspan="2">b)</td><td>c)</td></tr>
</table>

<table>
<tr><td rowspan="3">9</td><td colspan="4">a) $5.9\cdot\left(0.81^{\frac{1}{3}}-\left(9\frac{5}{7}\right)^{-1}\right):\left(3\frac{9}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{10}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{10}\right)$.</td><td>$f=9\cdot\pi$</td><td>$A$=22/14;
n=7.</td></tr>
<tr><td>d)
A=9.5+6.4i;
B=2.4-9.2i.</td><td>e)
α=9π/4;
β=249°.</td><td>f)
$N1$=1.923;
$N2$=1.595;
$N3$=1.459.</td><td>g)
m=45;
n=4.</td><td>h)
a=18;
b=24;
c=36.</td><td>1.2.4. d)
$i=2$;
$j=6$;
$k=7$.</td></tr>
<tr><td colspan="4">a)</td><td>b)</td><td>c)</td></tr>
<tr><td rowspan="3">10</td><td colspan="4">a) $4.5\cdot\left(0.125^{\frac{1}{3}}+\left(10\frac{5}{7}\right)^{-1}\right):\left(-3\frac{10}{13}\right)$;
$b)\ \sqrt{0.123}-\left(\sqrt{10}-5\sqrt{2}\right)\cdot\left(\sqrt{50}-\sqrt{10}\right)$.</td><td>$f=2\cdot\pi\cdot e$</td><td>$A$=66/14;
n=8.</td></tr>
<tr><td>d)
A=3.5+10.4i;
B=2.4-7.2i.</td><td>e)
α=11π/7;
β=124°.</td><td>f)
$N1$=1.321;
$N2$=1.325;
$N3$=1.346.</td><td>g)
m=48;
n=5.</td><td>h)
a=20;
b=24;
c=50.</td><td>1.2.4. d)
$i=7$;
$j=8$;
$k=5$.</td></tr>
<tr><td colspan="4">a)</td><td>b)</td><td>c)</td></tr>
<tr><td rowspan="3">11</td><td colspan="4">a) $4.5\cdot\left(0.64^{\frac{1}{3}}-\left(1\frac{5}{7}\right)^{-2}\right):\left(-3\frac{11}{13}\right)$;
$b)\ \sqrt{0.314}-\left(\sqrt{13}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{13}\right)$.</td><td>$f=11+\pi$</td><td>$A$=33/7;
n=6.</td></tr>
<tr><td>d)
A=3.5+7.4i;
B=7.4-3.2i.</td><td>e)
α=11π/4;
β=345°.</td><td>f)
$N1$=1.127;
$N2$=1.345;
$N3$=1.452.</td><td>g)
m=12;
n=8.</td><td>h)
a=12;
b=27;
c=36.</td><td>1.2.4. d)
$i=8$;
$j=4$;
$k=3$.</td></tr>
<tr><td colspan="4">a)</td><td>b)</td><td>c)</td></tr>
<tr><td rowspan="3">12</td><td colspan="4">a) $4.5\cdot\left(0.64^{\frac{1}{3}}-\left(2\frac{5}{7}\right)^{-1}\right):\left(-3\frac{6}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{11}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{11}\right)$.</td><td>$f=2\cdot\pi$</td><td>$A$=22/7;
n=5.</td></tr>
<tr><td>d)
A=3.5+6.4i;
B=2.4-3.2i.</td><td>e)
α=11π/5;
β=245°.</td><td>f)
$N1$=1.123;
$N2$=1.345;
$N3$=1.456.</td><td>g)
m=36;
n=16.</td><td>h)
a=11;
b=33;
c=44.</td><td>1.2.4. d)
$i=5$;
$j=3$
$k=10$.</td></tr>
<tr><td colspan="4">a)</td><td>b)</td><td>c)</td></tr>
<tr><td rowspan="3">13</td><td colspan="4">a) $13.5\cdot\left(0.64^{\frac{1}{3}}-\left(2\frac{5}{7}\right)^{-1}\right):\left(-3\frac{6}{11}\right)$;
$b)\ \sqrt{0.09}-\left(\sqrt{11}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{11}\right)$.</td><td>$f=13\cdot\pi$</td><td>$A$=55/7;
n=7.</td></tr>
<tr><td>d)
A=9.5-6.4i;
B=2.4+3.2i.</td><td>e)
α=13π/5;
β=145°.</td><td>f)
$N1$=1.321;
$N2$=1.325;
$N3$=1.326.</td><td>g)
m=26;
n=16.</td><td>h)
a=26;
b=39;
c=78.</td><td>1.2.4. d)
$i=9$;
$j=5$;
$k=10$.</td></tr>
<tr><td colspan="4">a)</td><td>b)</td><td>c)</td></tr>
</table>

14	a) $14.5\cdot\left(0.64^{\frac{1}{3}}-\left(2\frac{5}{7}\right)^{-1}\right):\left(-3\frac{6}{11}\right)$; $b)\ \sqrt{0.576}-\left(\sqrt{14}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{14}\right)$.				$f=14\cdot\pi$	A=π/7; n=5.
	d)	e)	f)	g)	h)	1.2.4. d)
	A=2.5-5.4i; B=2.4+6.2i.	α=14π/5; β=144°.	$N1$=1.423; $N2$=1.445; $N3$=1.456.	m=81; n=16.	a=12; b=28; c=36.	$i=8$; $j=3$; $k=8$.
	a)				b)	c)
15	a) $15.5\cdot\left(0.64^{\frac{1}{3}}-\left(2\frac{5}{7}\right)^{-1}\right):\left(3\frac{6}{15}\right)$; $b)\ \sqrt{3.64}+\left(\sqrt{15}-2\sqrt{5}\right)\cdot\left(\sqrt{20}-\sqrt{15}\right)$.				$f=15\cdot e$	A=88/7; n=7.
	d)	e)	f)	g)	h)	1.2.4. d)
	A=8.5+6.4i; B=2.4-8.2i.	α=11π/8; β=325°.	$N1$=1.153; $N2$=1.155; $N3$=1.156.	m=12; n=8.	a=12; b=27; c=36.	$i=6$; $j=5$; $k=3$.

1.3. Методические указания по выполнению лабораторной работы.

1.3.1. С целью подготовки качественного отчета по выполненной работе создайте на своем флешь-носителе рабочий файл, куда Вы будете помещать скопированные фрагменты командного окна;

1.3.2. Каждый выполняемый пункт отделяйте строкой комментария;

1.3.3. Скопированные фрагменты командного окна рекомендуется форматировать средствами соответствующего редактора текста с целью их “компактного” представления в отчетном документе.

1.4. Требования к отчету.

1.4.1. Отчет должен быть представлен в виде печатного документа, созданного средствами офисного пакета и включать в себя:

- ✓ Титульный лист;
- ✓ Цель лабораторной работы;
- ✓ Заданный вариант технического задания и его содержимое;
- ✓ Результаты выполнения каждого пункта задания с соответствующими копиями фрагментов командного окна, включающими заданные значения;
- ✓ Вывод.

1.5. Контрольные вопросы.

1.5.1. Назначение окна Журнал команд.

1.5.2. Окно отображения информации о данных.

1.5.3. Назначение команд who, who_user, who(a).

1.5.4. Каким образом установить формат для вывода числовых значений на экран?

1.5.5. Каким образом перенести командную строку из окна Журнал команд в окно Командное окно?
1.5.6. Каким образом установить формат для вывода числовых значений на экран?
1.5.7. Форматы числовых данных в системе Scilab.
1.5.8. Что такое системные константы?
1.5.9. Что такое символьные константы?
1.5.10. Команда объявления символьных переменных.
1.5.11. Какой символ используется для определения комментария?
1.5.12. Какая из операций .* или * является операцией поэлементного умножения?
1.5.13. Какой символ предназначен для запрета вывода результата выполнения команды на экран?

Лабораторная работа № 2.
«Задание функций пользователя и их реализация в среде Scilab»

Цель работы: Изучение основных способов задания функций пользователя и их реализация в среде Scilab. Приобретение практических навыков проведения инженерно-технических и научных расчетов с использованием пакета Scilab.

2.1 Теоретическая часть

Имеется несколько способов создания функций в Scilab.

Первый способ - это применение оператора **deff**, который в общем виде можно записать так:

deff(‘[v1, v2, … , vn]=**f_name**(w1, w2, … wm)’,’v1=выражение 1; v2=выражение 2;…. vn=выражение n’), где

f_name - имя функции, по которому происходит её вызов;

v1, v2, … ,vn - выходные параметры, т.е. то, что функция возвращает;

w1, w2, … ,wm - входные параметры , т.е. то, что функция принимает;

выражение 1, выражение 2,… выражение n – функции, вычисляющие значения выходных параметров v1, v2, … ,vn соответственно.

Например, необходимо задать функцию вычисления периметра заданного правильного *n*-угольника, вписанного в окружность с заданным радиусом. В этом случае периметр правильного *n*-угольника, вписанного в окружность радиусом *R*, может быть вычислен по формуле:

$$P = 2 \cdot R \cdot n \cdot \sin\left(\frac{\pi}{n}\right),$$

где P - искомый периметр, R – радиус окружности, n- число сторон правильного *n*-угольника.

Соответствующая функция, вычисляющая значение периметра с использованием функции deff, может быть представлена в виде:

--> deff('Perimetr=P(n,R)','Perimetr=2*R*n*sin(%pi/n)'), где Perimetr- значение вычисляемого периметра, P – имя функции.

Обращение к функции имеет “стандартный” вид.

Пусть необходимо вычислить периметр правильного 20-угольника, вписанного в окружность радиусом 10. Обозначим количество сторон 20-угольника переменной N, а радиус окружности переменной Rad. И далее обращаемся к построенной функции с именем P с передачей ей соответствующих значений n и R:

--> deff('Perimetr=P(n,R)','Perimetr=2*R*n*sin(%pi/n)');

```
--> N=20;Rad=10;
--> P(N,Rad)
 ans  =
   62.573786
```

Конечно, для последующего использования вычисленного значения периметра целесообразно его сразу обозначить каким-то именем, например, PN. Тогда в этом случае можно сразу указать имя переменной для возвращаемого значения функции, т.е.:

```
--> PN=P(N,Rad);
```

При необходимости можно посмотреть это вычисленное функцией P значение:

```
--> PN
 PN  =
   62.573786
```

Второй способ создания функции - это применение конструкции вида:

function [v1, v2, … , vn] = **f_name**(w1, w2, … wm) **операторы** (тело функции)
endfunction,

где

f_name - имя функции, по которому происходит её вызов;
v1, v2, … ,vn - выходные параметры, т.е. то, что функция возвращает;
w1, w2, … ,wm - входные параметры , т.е. то, что функция принимает.

Для вышеприведенного примера вычисление значения периметра правильного n-угольника в командном окне это будет выглядеть так:

```
--> function Perimetr=P(n,R)
  > Perimetr=2*R*n*sin(%pi/n)
  > endfunction

--> P(N,Rad)
 ans  =
  62.573786
```

В случае задания более сложных функций, включающих большое количество промежуточных вычислений, рекомендуется создавать файл-сценарии, позволяющие сохранять все команды на диске в виде отдельного файла.

Для подготовки, редактирования и отладки файлов-сценариев служит специальный редактор **SciNotes**, который можно вызвать, выполнив команду

главного меню «Инструменты». В результате работы этой команды будет создан новый файл-сценарий. По умолчанию он имеет имя **Untitled1.sce**.

Для сохранения введенной информации необходимо выполнить команду ***Файл – Сохранить*** из меню редактора. Если информация сохраняется впервые, то появится окно ***Сохранить как…*** Ввод имени в поле ***Имя файла*** и щелчок по кнопке ***Сохранить*** приведет к сохранению информации, находящейся в окне редактора. Файлы-сценарии сохраняют с расширением **.sce**. Открывает ранее созданный файл команда главного меню ***Открыть файл ….***

Выполнить операторы файла-сценария можно несколькими способами:

- из меню редактора SciNotes вызвать команду ***Сохранить и выполнить***;
- из главного меню Scilab вызвать команду ***Выполнить*** и указать имя файла-сценария.

Например, для вышеприведенного примера вычисления периметра правильного n-угольника файл-сценарий будет иметь вид:

```
function Perimetr=P(n, R)
   Perimetr=2*n*R*sin(%pi/n);
endfunction
```

После сохранения файла на диске с именем, например, Alex1 в командном окне появится сообщение:

```
--> exec('D:\work\Alex1.sce', -1).
```

Вызов функции осуществляется из командного окна. Если, например, нужно вычислить периметр 20-угольника, вписанного в окружность радиусом 10, то в командном окне задаем команды:

```
--> N=20;  Radius=10;
```

Обращаясь к функции, получаем результат:

```
--> PN=P(N,Radius)
 PN  =
   62.573786
```

Добавим вычисление площади вписанного правильного *n*-угольника (значение S) и вычисление отношения периметра к диаметру (PD). В этом случае функция будет возвращать три значения, которые будут представлены в виде вектора-строки. Соответствующий файл-сценарий будет иметь вид:

```
function [Perimetr, S, PD]=P(n, R)
   Perimetr=2*n*R*sin(%pi/n);
   S=n*(R^2)*sin(2*%pi/n)/2;
   PD=Perimetr/(2*R)
endfunction
```

Вычислим значения периметра (PN), площади (PL) и отношение периметра к диаметру (PD) для n=10 с радиусом равным 1:

```
--> [PN,PL,PI]=P(10,1)
 PN  =
  6.1803399
 PL  =
  2.9389263
 PI  =
  3.0901699
```

Следует заметить, что при значительном увеличении n вычисленное значение PI будет близко к значению числа π.

2.2 Лабораторное задание.

2.2.1 Изучите теоретическую часть по построению и реализации функций пользователя;

2.2.2 В соответствии с заданным преподавателем вариантом, приведенным в таблице 2.2.1, выполните построение заданной функции $f(x, y)$ с использованием функции **deff**, а затем с использованием **function**;

2.2.3 Проведите тестовые расчеты и сравните полученные результаты;

2.2.4 Все команды и результаты их исполнения скопируйте в рабочий файл (для оформления отчета);

2.2.5 Разработайте алгоритм вычисления требуемых параметров 2-го задания своего варианта. Составьте блок-схему алгоритма. Напишите соответствующий псевдокод;

2.2.6 Откройте редактор SciNotes и напишите программный код (файл-сценарий, скрипт) реализации алгоритма;

2.2.7 Сохраните полученный скрипт на диске;

2.2.8 Составьте тестовую задачу и проверьте правильность работы алгоритма;

2.2.9 Скопируйте созданный скрипт в рабочий файл;

2.2.10 Выполните отчет по выполненной работе;

2.2.11 Ответьте на контрольные вопросы.

Варианты индивидуальных заданий. Таблица 2.2.1

Вар.	Задание
1	$f(x,y)=\dfrac{2\cos\left(x-\frac{\pi}{6}\right)}{\frac{1}{2}+sin^2y}+cos^2x^3\cdot e^x;$
	Постройте функцию, вычисляющую значения периметра, площади и углов ромба, заданного значениями своих диагоналей d_1 и d_2.
2	$f(x,y)=\dfrac{\left\|x^{x/y}\right\|-\sqrt[3]{y/x}}{\pi^x+\log_{10}\left\|x\cdot y\right\|};$
	Постройте функцию, вычисляющую радиусы вписанной и описанной окружностей для треугольника с заданными сторонами a, b и c.
3	$f(x,y)=\dfrac{\sqrt[3]{x^2}\cdot y^3+\sqrt[3]{y^2}\cdot x^3}{x\cdot e^y+y\cdot e^x};$
	Постройте функцию, вычисляющую вес сферической оболочки толщиной h, если диаметр внутренней границы сферической оболочки равен r, а плотность материала равна ρ.
4	$f(x,y)=\dfrac{x\cdot e^y+y\cdot e^x}{y\cdot e^y+x\cdot e^x};$
	Постройте функцию, вычисляющую площадь сечения правильной треугольной призмы, проходящего через сторону верхнего основания и противоположную вершину нижнего основания, объемы тел, разделенных сечением, по заданным значениям стороны основания a и высоты призмы H.
5	$f(x,y)=\dfrac{(sin\,x+\cos y)\cdot x^y}{(\sin y-\cos x)\cdot y^x};$
	Постройте функцию, вычисляющую объем, периметр основания и боковую поверхность правильной шестиугольной пирамиды, если задана высота пирамиды H и площадь основания S.
6	$f(x,y)=\dfrac{2^{\ln x}+y^e}{\log_2(x+y)+\sqrt[3]{x\cdot y}};$

	Постройте функцию, вычисляющую полную поверхность усеченного конуса, его объем и длину его образующей по заданным значениям радиусов оснований R, r и высоте h.
7	$f(x,y) = \frac{(x/y)^2 + (y/x)^2}{ln^2 x + ln^2 y}$;
	Постройте функцию, вычисляющую площадь сечения конуса, проходящего через вершину конуса и хорду основания, стягивающую дугу в 60^o, объемы тел, полученных в результате сечения конуса плоскостью, если задана высота конуса H и угол наклона плоскости сечения к плоскости основания α.
8	$f(x,y) = \frac{3 \cdot x \cdot \ln y - 5 \cdot y \cdot \sqrt{x}}{x \cdot \sqrt{\pi} + y\sqrt[3]{x^2}}$;
	Постройте функцию, вычисляющую боковую поверхность, объем и длину ребра правильной пирамиды по заданным значения длин сторон оснований A, a и высоте пирамиды H.
9	$f(x,y) = \frac{\sin x - \log_2 y}{\cos x + \lvert \ln(x+y) \rvert}$;
	Постройте функцию, вычисляющую площадь основания, площадь боковой поверхности и объем прямого параллелепипеда, основанием которого является ромб, если заданы длины диагоналей ромба, а высота параллелепипеда равна удвоенной длине стороны ромба.
10	$f(x,y) = \frac{\sin x \cdot \ln y - \cos y \cdot \ln x}{\cos(x-y) + \sin(x+y)}$;
	Постройте функцию, вычисляющую объем, площадь боковой поверхности, площадь основания цилиндра по заданной величине диагонали его осевого сечения d и углу между диагональю и его образующей α.
11	$f(x,y) = \frac{y \cdot e^x - x^2 \cdot \pi}{e^{x+y} + e^{x-y}}$;
	Постройте функцию, вычисляющую площадь сечения,

	параллельного оси цилиндра и отстоящего от центра на заданную величину d, а также объем цилиндра, если задана высота цилиндра H и площадь осевого сечения S.
12	$f(x,y) = \frac{x \cdot \ln\lvert x-y\rvert + y \cdot \ln\lvert x+y\rvert}{\lvert x^3+y^2\rvert \cdot \ln(x+y)^3}$;
	Постройте функцию, вычисляющую объем тела, ограниченного поверхность куба и вписанного в него шара, а также разность полных поверхностей между ними.
13	$f(x,y) = \frac{ln^2\lvert x\rvert + ln^2\lvert y\rvert}{\sqrt{\pi \cdot \lvert x\rvert} + \sqrt[3]{e \cdot y}}$;
	Постройте функцию, вычисляющую полную поверхность усеченного конуса высотой h, в основание которого вписан треугольник, у которого одна сторона равна a, а противолежащий ей угол равен β. Образующая конуса наклонена к плоскости основания под углом φ.
14	$f(x,y) = \frac{y \cdot \ln x - x \cdot e^y}{\lvert (x+y) \cdot \ln(x^2+y^2)\rvert}$;
	Постройте функцию, вычисляющую объем призмы, площадь боковой поверхности, если в основании призмы лежит параллелограмм с стороной равной a. Через данную сторону и противоположную ей сторону верхнего основания проведено сечение, составляющее угол β с основанием. Заданная площадь сечения равна S.
15	$f(x,y) = \frac{\cos(x \cdot y) - \sin(x-y)}{tg^2 x + \log_2(ctg\ y)}$;
	Постройте функцию, вычисляющую площадь и периметр фигуры «звезда», построенной на вписанном в окружность радиусом R правильном пятиугольнике.
16	$f(x,y) = \frac{x^3 \cdot y^2 + 12x - 4(x + \sqrt[3]{y^2})}{\log_4(x+y) - \log_2(x-y)}$;
	Постройте функцию, вычисляющую площадь основания, боковую поверхность и объем прямой призмы, если основанием призмы

	является «звезда Давида». Радиус описанной окружности вокруг основания равен *R*, а высота призмы равна *H*.
17	$f(x,y)\dfrac{5x^2\cdot\sqrt{x\cdot y}+2y\cdot\sqrt[3]{(x-y)}}{\lg(x+y)+\ln(x+y)}$;
	Постройте функцию, вычисляющую площадь боковой поверхности, объем цилиндра, вписанного в сферу, а также отношение полных поверхностей сферы и цилиндра, если задан высота цилиндра, равная диаметру основания, равного *D*.
18	$f(x,y)=\dfrac{x^2\cdot\cos y-y^2\cdot\sin x}{\lg(\ln(x+y^2)+x^2)}$;
	Постройте функцию, вычисляющую площадь, объем прямого параллелепипеда, а также площадь его диагонального сечения, если заданы: стороны основания равны *a* и *b*, и угол между ними равен *γ*, площадь меньшего диагонального сечения равна *S*.
19	$f(x,y)=\dfrac{e^x\cdot\cos y+e^y\cdot\sin x}{e^{(x+y)}+\pi\cdot 2\ln(x+y}$;
	Постройте функцию, вычисляющую объем, поверхность шара, вписанного в прямую треугольную призму (т.е. шар касается верхнего и нижнего основания), а также площадь основания и площадь боковой поверхности призмы, если заданы две стороны основания *a* и *b*, и задана высота *h* к стороне *a* из противоположного к ней угла.
20	$f(x,y)=\dfrac{x\cdot tg(x+y)-y\cdot ctg\,(x-y)}{x^2+y^2-\ln\lvert x\cdot y\rvert}$;
	Постройте функцию, вычисляющую площадь поверхности, объем сферы, вписанной в прямой цилиндр (т.е. сфера касается оснований и каждой ее образующей), а также отношение полных поверхностей сферы и цилиндра, если задан диаметр сферы *D*.
21	$f(x,y)=\dfrac{\sqrt[3]{x^2\cdot y^2}+12x-4(x+\sqrt[3]{y^3})}{\log_8(x+y)-\log_4\lvert y-2x\rvert}$;
	Постройте функцию, вычисляющую площади боковых поверхностей прямой шестиугольной призмы и вписанной в нее

	правильной треугольной пирамиды, а также объемы соответствующих тел, высота призмы равна h, а радиус основания описанной окружности равен R.

2.3 Контрольные вопросы

2.3.1 Способы задания функций в Scilab;
2.3.2 Как осуществляется вызов функции в Scilab?
2.3.3 Как указываются возвращаемые значения функции?
2.3.4 Для чего служит функция **printf**? Как она записывается в Scilab? Чем она отличается от функции **disp**?
2.3.5 Что означает следующая форма записи числа: 1.234E-5?
2.3.6 Что такое блок-схема алгоритма? Как она строится?
2.3.7 Приведите условные обозначения элементов блок-схемы алгоритма и их назначение.
2.3.8 Для чего используется редактор **SciNotes**?
2.3.9 Как создать файл-сценарий и сохранить его на внешнем носителе?

Лабораторная работа № 3
«Решение простейших научно-технических и инженерных задач с использованием базовых конструкций языка Scilab»

Цель работы: Изучение и освоение базовых элементов структурного программирования в среде Scilab. Приобретение практических навыков построения и реализации алгоритмов численного моделирования научно-технических и инженерных задач.

3.1 Теоретическая часть.

В Scilab имеется встроенный язык программирования, который необходимо знать для эффективной работы в среде. Данный язык является структурным с поддержкой объектов.

В структурном программировании сценарий представляет последовательность инструкций, которые выполняются последовательно друг за другом, словно на конвейере. (Работа с командной строкой в каком-то смысле является написанием сценария, в котором инструкции генерируются вами «по ходу»).

Как известно, в структурном программировании существует ряд особых конструкций, существенно повышающие эффективность написания кода. К ним относятся:

Ветвление - однократное выполнение одной из двух или более операций в зависимости от заданного условия;

Цикл - многократное выполнение одной или нескольких инструкций до тех пор, пока не выполнится некоторое условие. Один проход по всем инструкциям внутри цикла называется итерацией.

Ветвления

В языке Scilab имеется две конструкции, организующие ветвление:

- Конструкция **if ... else;**
- Конструкция **select ... case**.

Общий синтаксис конструкции **if ... else** имеет следующий вид

if <условие 1> **then**
<операторы 1>
elseif <условие 2> **then**
<операторы 2>
...
elseif <условие n> **then**

<операторы n>
else
<операторы>
end

На позициях <условие> записываются логические выражения, которые возвращают логический или числовой тип данных.

Общий синтаксис второй конструкции ветвления, которую называют также ***оператором альтернативного выбора***, имеет вид:

select <значение>
case <значение 1>
<операторы 1>
case <значение 2>
<операторы 2>
...
case <значение n>
<операторы n>
else
<операторы>
end

В отличии от первой конструкции, вторая требует не логическое выражение, а конкретное значение, которое затем сравнивается по порядку со значением каждой ветви, начинающейся с ключевого слова **case**. При первом же совпадении выполняются операторы этой ветви. Если ни одного совпадения не оказалось, то выполняются операторы ветви **else.**

Циклы.

В Scilab имеется всего два оператора циклов:

- оператор цикла **for**;
- оператор цикла с предусловием **while** .

Оператор цикла **for** имеет вид:

for x=xn:hx:xk
<операторы>
end

Где x- имя скалярной переменной параметра цикла, xn - начальное значение параметра цикла, xk -конечное значение параметра цикла, hx -шаг

цикла. Если шаг цикла равен то 1, hx можно опустить и в этом случае оператор for будет иметь вид

for x=xn:xk
<операторы>
end

Оператор цикла **while** имеет вид:

while <условие>
<операторы>
end

Здесь <условие> -логическое выражение. <операторы> будут выполняться циклически, пока логическое условие истинно.

Для организации простейшего ввода в Scilab можно воспользоваться функциями x=**input**('title').

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

Например, введем команду:

--> X=input('Введите значение X ')

После нажатия клавиши «Enter» на экране появится сообщение:

“Введите значение X”

Набрав, например, 1234 и нажав «Enter» на экране появится вывод введенного значения X:

X =
1234.

Для вывода в текстовом режиме можно использовать функцию **disp** следующей структуры **disp**(*S*), где *S* имя переменной или заключенный в кавычки текст.

Приведенные выше базовые структуры программирования в среде Scilab позволяют реализовать практические любые алгоритмы проведения научно-технических и инженерных расчетов.

3.2. Лабораторное задание.

3.2.1 Изучите теоретический материал к лабораторной работе;

3.2.2 В соответствии с заданным вариантом (таблица 3.2.1) постройте таблицу значений функции $f(x)$ при изменении x на отрезке [$x0$, xk] с шагом xh и с заданными коэффициентами. Постройте блок-схему и

псевдокод алгоритма табулирования функции для заданных диапазонов ее аргументов;

3.2.3 Напишите файл-сценарий вычисления заданной функции с использованием Scinotes и сохраните соответствующий файл на рабочем диске;

3.2.4 Составьте тестовую задачу и проверьте правильность работы написанного кода. Входные данные должны передаваться из командного окна. В случае наличий ошибок проведите отладку кода;

3.2.5 Скопируйте текст файл-сценария и результаты работы алгоритма в рабочий файл;

3.2.6 В соответствии с заданным вариантом (таблица 3.2.2), постройте блок-схему и псевдокод алгоритма вычисления значения заданной функции по соответствующей ей формуле;

3.2.7 Напишите файл-сценарий вычисления функции с использованием SciNotes и сохраните соответствующий файл на рабочем диске. Проведите тестирование и отладку написанного кода;

3.2.8 Обращение к функции осуществляется из командного окна;

3.2.9 Оформите отчет о выполненной работе;

3.2.10 Ответьте на контрольные вопросы.

Варианты индивидуальных заданий. Таблица 3.2.1

Вар.	**Заданная функция**	**Заданные значения**
1.	$f(x)=\begin{cases} x^2-1 & \text{если } x\le 0;\\ a\cdot\cos x & \text{если } 0<x\le 1.5;\\ b\cdot\sin x & \text{если } x>2. \end{cases}$	$x0=-2; xk=4;$ $xh=0.5;$ a=1.25; b=2.5
2.	$f(x)=\begin{cases} e^{-ax}-1 & \text{если } x>2.5;\\ 5\cdot sin\,ax & \text{если } -1<x\le 2;\\ \cos x\cdot sin\,bx & \text{если } x<-2.5. \end{cases}$	$x0=-5; xk=5;$ $xh=0.5;$ a=0.5; b=π.
3.	$f(x)=\begin{cases} e^{-\alpha x} & \text{если } \lvert x\rvert<2;\\ x^2+2 & \text{если } 2\le x<4;\\ x+b & \text{если } -4\le x\le -2. \end{cases}$	$x0=-4; xk=4;$ $xh=0.25;$ α=0.1; b=-2.
4.	$f(x)=\begin{cases} tg\,(x+\alpha) & \text{если } 0\le x\le \frac{\pi}{4};\\ \frac{1}{b+\cos x} & \text{если } \frac{\pi}{4}<x<\frac{3\pi}{4};\\ \sin(x+2\alpha) & \text{если } \frac{3\pi}{4}\le x<2\pi. \end{cases}$	$x0=0; xk=2\pi;$ $xh=\frac{\pi}{10};$ $\alpha=\frac{\pi}{6};$ b=1.

5.	$f(x)=\begin{cases}-x^2+5x+6 & \text{если } -3\le x\le -2;\\ x^2-5x+6 & \text{если } 3\le x\le 4;\\ ax+b & \text{противном случае.}\end{cases}$	$x0=-3; xk=5;$ $xh=0.2;$ a=2.5; b=-2.5.
6.	$f(x)=\begin{cases}e^x & \text{если } x>2;\\ ax+b & \text{если } \lvert x\rvert\le 2;\\ x^a & \text{если } x<-2\,.\end{cases}$	$x0=-3; xk=4;$ $xh=0.2;$ a=1.5 ; b=2.5.
7.	$f(x)=\begin{cases}e^{-x} & \text{если } \lvert x\rvert>3;\\ ax+b & \text{если } \lvert x\rvert\le 2;\\ x^a & \text{если } 2<x\le 3\,.\end{cases}$	$x0=-4; xk=4;$ $xh=0.2;$ a=2.5 ; b=3.
8.	$f(x)=\begin{cases}x^{\frac{2}{3}} & \text{если } x>6;\\ a\cdot\sin x & \text{если } x<5\,;\\ \sqrt{\sqrt{x}+b} & \text{если } 5\le x\le 6.\end{cases}$	$x0=2; xk=8;$ $xh=0.5;$ a=8 ; b=4.
9.	$f(x)=\begin{cases}ae^{bx} & \text{если } \frac{\pi}{4}\le x\le\frac{3\pi}{4};\\ a\cdot\sin bx & \text{если } 0\le x<\frac{\pi}{4};\\ \sqrt{\lvert a\cdot\cos bx\rvert+4} & \text{если } \frac{3\pi}{4}<x<\pi.\end{cases}$	$x0=0; xk=\pi;$ $xh=0.1\pi;$ a=5; b=-0.2.
10.	$f(x)=\begin{cases}a\cdot\lg x+b & \text{если } 1\le x<4;\\ a\cdot\ln x+b & \text{если } 4\le x\le 7;\\ a\cdot\log_2 x+b & \text{если } x>7;\end{cases}$	$x0=1; xk=10;$ $xh=1;$ a=5; b=10.
11.	$f(x)=\begin{cases}\log_a x & \text{если } x\ge 6;\\ \lg\sqrt[3]{(bx+11)} & \text{если } 3\le x<6\\ a\cdot x+b & \text{если } -x0\le x<3\end{cases}$	$x0=-4; xk=10;$ $xh=1;$ a=3; b=11.
12.	$f(x)=\begin{cases}x^a & \text{если } x0\le x<2;\\ x^b & \text{если } 2\le x\le 3;\\ \left(\frac{a}{b}\right)^x & \text{если } x>3.\end{cases}$	$x0=-2; xk=4;$ $xh=0.2;$ a=1.2; b=2.4.
13.	$f(x)=\begin{cases}x+\frac{a}{x} & \text{если } x0\le x<3;\\ x^2+\frac{b}{x^2} & \text{если } 3\le x<5;\\ x^3+\sqrt[3]{x} & \text{если } 5\le x\le xk.\end{cases}$	$x0=1.5; xk=7;$ $xh=0.5;$ a=13; b=8.
14.	$f(x)=\begin{cases}a\cdot e^{-x}+b & \text{если } x0\le x\le 0.2;\\ b\cdot e^{-x}+a & \text{если } 0.2<x\le 0.4;\\ e^{-ax}+e^{-bx} & \text{если } 0.4<x\le xk.\end{cases}$	$x0=0; xk=0.8;$ $xh=0.1;$ a=5; b=10.

15.	$f(x)=\begin{cases} a\cdot\sin x+b\cdot\cos x & \text{если } x0\le x<\pi;\\ b\cdot\sin x+a\cdot\cos x & \text{если } \pi\le x\le\frac{3\pi}{2};\\ a\cdot b\cdot(\sin x+\cos x) & \text{если } x>\frac{3\pi}{2}. \end{cases}$	$x0=0; xk=2\pi;$ $xh=\frac{\pi}{6};$ a=2; b=4.
16.	$f(x)=\begin{cases} e^{ax}\cdot\sin x & \text{если } 0\le x\le\frac{\pi}{2};\\ e^{bx}\cdot\cos x & \text{если } \frac{\pi}{2}<x<\pi;\\ \lg\lvert a\cdot x+b\rvert & \text{если } x\ge\pi. \end{cases}$	$x0=0; xk=2\pi;$ $xh=\frac{\pi}{6};$ a=-2.5 ; b=-4.
17.	$f(x)=\begin{cases} \lg(x+a) & \text{если } 8\le x<12;\\ \lg(x+b) & \text{если } 12\le x\le 15;\\ \lg(ax+bx) & \text{если } x>15. \end{cases}$	$x0=8; xk=18;$ $xh=1;$ a=5; b=10.
18.	$f(x)=\begin{cases} a\cdot\lg x+b\cdot\ln x & \text{если } 0.5\le x\le 1;\\ b\cdot\lg x+a\cdot\ln x & \text{если } 1<x<2.5;\\ a\cdot b\cdot\lg x\cdot(\ln x) & \text{если } x\ge 2.5 \end{cases}$	$x0=0.5; xk=4;$ $xh=0.25;$ a=1; b=3.68
19.	$f(x)=\begin{cases} a\cdot x\cdot\sin(b\cdot x) & \text{если } x>\frac{3\pi}{2}\\ b\cdot x\cdot\cos(a\cdot x) & \text{если } \frac{2\pi}{3}<x\le\frac{3\pi}{2}\\ (a+b)\cdot\sin(2\cdot x) & \text{если } x0\le x\le\frac{2\pi}{3} \end{cases}$	$x0=0; xk=2\pi;$ $xh=\frac{\pi}{8};$ a=2; b=2.
20.	$f(x)=\begin{cases} 2^x\cdot\log_2 x+a & \text{если } 4\le x\le 7;\\ e^x\cdot\ln x+b & \text{если } 7<x\le 10;\\ 10^x\cdot\lg x-a\cdot b & \text{если } 10<x\le 12. \end{cases}$	$x0=4; xk=12;$ $xh=1;$ a=2; b=0.

Таблица 3.2.2

Вар.	**Задание**
1.	Вычислите с точностью $\varepsilon=10^{-5}$ константу Эйлера (основание натурального алгоритма), воспользовавшись разложением в ряд: $e^x=1+\frac{x}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+\cdots+\frac{x^n}{n!}+\cdots$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
2.	Вычислите с точностью $\varepsilon=10^{-5}$ значение функции $y=\sqrt{x}$, воспользовавшись рекуррентной формулой: $y_{i+1}=0.5\cdot\left(y_i+\frac{x}{y_i}\right);\ i=0,1,2,\ldots;\ y_0=\frac{x}{2}.$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
3.	Вычислите с точностью $\varepsilon=10^{-5}$ значение функции arctg x,

	воспользовавшись разложением в ряд: $\operatorname{arctg} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots + (-1)^n \frac{x^{2n+1}}{2n+1} + \cdots$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
4.	Вычислите с точностью $\varepsilon=10^{-5}$ значение функции $y = \frac{1}{\sqrt{x}}$, Воспользовавшись формулой: $y_{i+1} = 1.5y_i - 0.5xy_i^3;\ \ i = 0,1,2,\cdots;\ \ y_0 = 1.$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
5.	Вычислите с точностью $\varepsilon=10^{-5}$ значение функции $\sin x$, воспользовавшись разложением в ряд: $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \cdots$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
6.	Вычислите с точностью $\varepsilon=10^{-5}$ значение функции $y = \cos x$, воспользовавшись разложением в ряд: $\cos x = x - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!} + \cdots$ Сравните результат со значением, полученным с помощью соответствующей встроенной функции.
7.	Вычислите корень уравнения $x - 0.5(\sin x^2 - 1) = 0$ с точностью $\varepsilon=10^{-3}$, воспользовавшись итерационной формулой: $x_{i+1} = 0.5(\sin x_i^2 - 1);\ \ i = 0,1,2,\cdots:\ \ x_0 = 0.25.$ Проверьте правильность решения подстановкой коня в уравнение.
8.	Вычислите $sh\, x$ с точностью $\varepsilon=5\cdot10^{-5}$, воспользовавшись разложением в ряд: $sh\, x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + \frac{x^{2n+1}}{(2n+1)!} + \cdots$ Сравните результат со значением, полученным с помощью встроенной функции для вычисления e^x, используя соотношение: $sh\, x = \frac{e^x - e^{-x}}{2}$.
9.	Вычислите с точностью $\varepsilon=10^{-4}$ значение функции $y = \ln(1+x)$, воспользовавшись разложением в ряд: $\ln(1+x) = \frac{x}{1} - \frac{x^2}{2} + \frac{x^3}{3} + \cdots + (-1)^{n+1} \frac{x^n}{n} + \cdots$ Сравните результат со значением, полученным с помощью

	соответствующей встроенной функции.
10.	Вычислите $ch\,x$ с точностью $\varepsilon=5\cdot10^{-5}$, воспользовавшись разложением в ряд: $ch\,x = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \cdots + \frac{x^{2n}}{(2n)!} + \cdots$ Сравните результат со значением, полученным с помощью встроенной функции для вычисления e^x, используя соотношение: $sh\,x = \frac{e^x + e^{-x}}{2}$.
11.	Вычислите корень уравнения $x - \cos x$ с точностью $\varepsilon=10^{-3}$, воспользовавшись итерационной формулой: $x_{i+1} = \cos x_i;\ \ i = 0,1,2,\cdots:\ x_0 = 0.$ Проверьте правильность решения подстановкой найденного корня в уравнение.
12.	Вычислите значение числа π как отношение периметра вписанного в окружность правильного n-угольника к диаметру окружности с точностью $\varepsilon=10^{-5}$. Увеличение сторон n-угольника осуществляйте методом удвоения, т.е. $n_{i+1} = 2n_i.\ n_0 = 3.$
13.	Вычислите корень уравнения $f(x) = x^4 + 2x^2 - x - 1 = 0$ с точностью $\varepsilon=10^{-4}$, воспользовавшись итерационной формулой: $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)};\ \ i = 0,1,2,\cdots;\ \ x_0 = 0.$ Проверьте правильность решения подстановкой найденного корня в уравнение.
14.	Вычислите значение числа π как отношение периметра описанного около окружности правильного n-угольника к диаметру окружности с точностью $\varepsilon=10^{-5}$. Увеличение сторон n-угольника осуществляйте методом удвоения, т.е. $n_{i+1} = 2n_i.\ n_0 = 4.$
15.	Вычислите значение числа π используя формулу Мадхава (индийский ученый, XV век) с точностью до 5-го знака после запятой. $\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \cdots + \cdots\right).$
16.	Вычислите корень уравнения $f(x) = tg\,x - x = 0$ с точностью $\varepsilon=10^{-4}$, воспользовавшись итерационной формулой: $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)};\ \ i = 0,1,2,\cdots;\ \ x_0 = 4.6.$ Проверьте правильность решения подстановкой найденного корня в уравнение.
17.	Вычислите значение бесконечной суммы с точностью $\varepsilon=10^{-5}$, если

	$S=\frac{1}{2\cdot 3}+\frac{1}{2\cdot 3\cdot 4}+\frac{1}{3\cdot 4\cdot 5}+\cdots\frac{1}{(n+1)\cdot(n+2)\cdot(n+3)}+\cdots$
18.	Вычислите значение бесконечной суммы с точностью $\varepsilon=10^{-4}$ при заданном действительном $x\neq 0$, если $S=\sum_{k=0}^{\infty}\frac{x^{2k}}{2^k k!}$.
19.	Дано действительное число $b<0$. Последовательность $a_1, a_2, \cdots a_k, \cdots$ образована по закону: $a_1=b;\ a_k=\frac{a_{k-1}+1}{1-sin^2 k},\quad k=2,3,\cdots.$ Найдите номер первого неотрицательного члена последовательности и его числовое значение.
20.	Дано действительное число $x\neq 0$. Последовательность $a_1, a_2, \cdots a_n, \cdots$ образована по закону: $a_n=\frac{(-1)^n x^{2k}}{n(n+1)(n+2)}$. Найдите сумму первых k элементов данной последовательности, если k-наименьшее целое число, удовлетворяющее двум условиям: $k>10$ и $\lvert a_{k+1}\rvert<10^{-5}$.

3.3. Методические рекомендации

3.3.1. Перед выводом таблицы значений заданной функции рекомендуется обозначить колонки выводимых значений;

3.3.2. Для вывода таблицы значений целесообразно использовать формат **printf**() с указанием формата выводимых значений. Колонки значений должны быть разделены промежутком. Ниже приведен пример простой программы и результат ее работы:

```
4.  function F=Atest4(x1, x2, h)
5.     printf('  i     s1     Y      Z      W\n')
6.     Y=0; Z=1; W=0;
7.     for i=x1:h:x2
8.        s1=i+2
9.        Y=Y+i
10.       Z=Z*i
11.       W=Y+Z
12.       printf(' %4.2f  %4.2f  %6.2f  %6.2f  %6.2f\n',i,s1,Y,Z,W)
13.    end
14.    F='Конец таблицы'
15. endfunction
```

Результат выполнения программы:

```
--> Atest4(3,6,1)
   i        s1         Y         Z         W
 3.00     5.00      3.00      3.00      6.00
 4.00     6.00      7.00     12.00     19.00
 5.00     7.00     12.00     60.00     72.00
 6.00     8.00     18.00    360.00    378.00
 ans  =
  "Конец таблицы"
```

Здесь следует отметить, что в данном примере функция возвращает строковую переменную "Конец таблицы", информируя о завершении вычислений. Если необходимо вывести все полученные расчетные значения для дальнейшего их использования в вычислениях, то F должна быть оформлена в виде соответствующей матрицы.

3.4 Контрольные вопросы

3.4.1 Приведите формат оператора ветвления **if ... else.**
3.4.2 Приведите формат оператора альтернативного выбора **select ... case**.
3.4.3 В каком случае удобней использовать оператор выбора варианта **select** вместо условного оператора **if**?
3.4.4 Приведите формат оператора цикла **for**.
3.4.5 В каких случаях целесообразно использовать оператор **for**?
3.4.6 Приведите формат оператора цикла **while.**
3.4.7 В каких случаях целесообразно использовать оператор **while**?
3.4.8 Что такое **зацикливание** программы? Когда оно возникает?
3.4.9 Как можно передать управление в теле цикла следующей итерации?
3.4.10 В каких случаях используется оператор **break**?

Лабораторная работа № 4
Построение графиков в Scilab

Цель работы: изучение и освоение основных методов построения графиков в системе Scilab. Приобретение практических навыков использования пакета компьютерной графики для визуализации инженерно – технических и научных расчетов.

4.1 Теоретическая часть.

В среде Scilab все графики строятся по точкам, при этом каждые две соседние точки соединяются друг с другом отрезком. Чем меньше расстояние между точками, тем меньше искажение графика за счет замены его реального образа кусочно-линейной функцией.

Внешний вид графика определяют следующие факторы:

- Математическая функция и система координат:
  - ***двухмерный график*** - график функции с одним аргументом, либо, если функция представлена неявно, одномерной функции;
    - в прямоугольной системе координат;
    - в полярной системе координат;
    - в комплексной плоскости (например, годографы);
    - в векторной плоскости;
  - ***трехмерный график*** - график функции с двумя аргументами, либо, если функция представлена неявно, двухмерной функции;
    - в пространственной прямоугольной системе координат;
    - в сферической системе координат;
    - в цилиндрической системе координат;
    - в векторном пространстве.
- Способ представления:
  - точечный график;
  - линейный график (с учетом всевозможных представлений линии);
  - полигональный график;
  - градиентный график;
  - контурный график
  - гистограмма.

4.2 Построение графиков

Любой график имеет следующие основные элементы:

- собственно сам график функции (или графики), включая параметры их отображения;

- оси системы координат, включая внешний вид, масштаб отсчета и подписи осей;
- координатная сетка;
- подпись графика;
- легенда;
- надписи на самом графике.

4.2.1 Построение графиков одномерных функций

Для построения одномерных функций в прямоугольной системе координат в Scilab служат следующие функции:

- семейство функций **plot2d*()**, выполняющих построение по точечной функции, где символ звездочки кодирует заранее определенный шаблон:
  - **plot2d** - строит график по форматам, определяемым пользователем;
  - **plot2d2** - строит график в виде ступенчатой функции;
  - **plot2d3** - строит график в виде вертикальных полосок;
  - **plot2d4** - строит график с указанием направления;
- Функция **fplot2d()**, выполняющая построение по заранее определенной функции;
- Функция **plot()** - являющаяся аналогом семейства **plot2d** и служит для совместимости с пакетом MATLAB.

4.2.2 Функция plot()

Для построения графиков простейших функций вида y = f(x), в Scilab можно использовать функцию **plot**, обращение к которой имеет вид:

plot(x,y,[s]),

где x – вектор или матрица действительных чисел (массив абсцисс), y - массив ординат; s - необязательный параметр, отвечающий за цвет, толщину и тип линии и др.

Синтаксис данной функции может иметь вид:

- plot(y);
- plot(x, y);
- plot(x, fun);
- plot(x1, y1, LineSpec1, x2,y2,LineSpec2, ...xN, yN,LineSpecN, GlobalProperty1,.. GlobalPropertyM);
- plot(x1,fun1,LineSpec1, x2,y2,LineSpec2,...xN,funN,LineSpecN, GlobalProperty1, ..GlobalPropertyM).

Т.е. для построения графика вида y = f(x) в Scilab необходимо предварительное формирование двух векторов одинаковой длины: вектора

значений аргументов x и вектора соответствующих им значений функции y. Последующее выполнение команды **plot(**x, y**)** открывает графическое окно и отображает в нем график функции y = f(x). Создать новое графическое окно можно и путем выполнения команды **scf**(n), где n – номер графического окна.

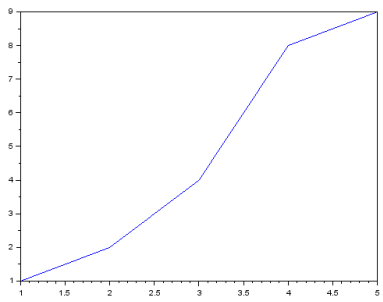
По умолчанию график строится в текущем графическом окне, то есть в последнем графическом окне, которое было открыто до этого. Если никакого окна открыто не было, то по умолчанию текущее окно – окно с номером 0.

Для очистки текущего графического окна используется ко манда clf(), для очистки графического окна с номером n – команда clf(n).

Если аргумент опущен, то предполагается, что x принимает значения от 1 до n, где n – число точек кривой, задаваемых параметром y;

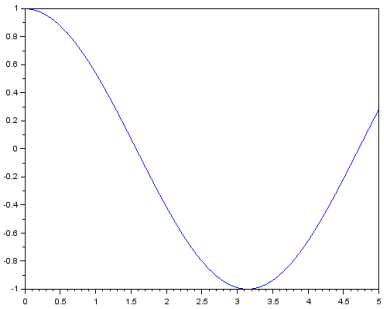
Пример:

```
--> clf(); y=[1 2 4 8 9]; plot(y)
```



Пример:

```
--> clf();x=0:0.1:5;plot(x,cos)
```



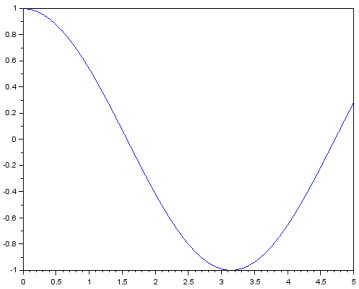
LineSpec - это необязательный параметр, который используется для спецификации строящихся линий графика. Можно задать стиль линии (сплошная, пунктирная, штрихпунктирная и т.д.), ее цвет, маркеры и т.д.

Например, если необходимо построить график пунктирной линией красного цвета с маркерами в виде ромба, можно записать в функции plot аргумент 'r--d' или '--dire' или '--reddiam' или 'diamondred--' (см. справку Scilab по LineSpec). Кроме пунктирной, тип линии может быть: сплошным (обозначается как -), штрихом, чередующемся с двумя точками (:), штрихом, чередующемся с одной точкой (-.). По умолчанию кривая графика имеет сплошной тип линии толщиной 1. Чтобы изменить толщину кривой нужно указать требуемое значение в функции plot() аргумента 'thickness'

Пример: а) - значения аргументов по умолчанию; б) – цвет линии красный, тип линии - пунктирная линия с маркерами в виде крестика, толщина линии 2.

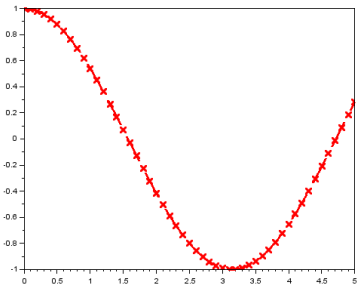
а)

--> clf();x=0:0.1:5;plot(x,cos)



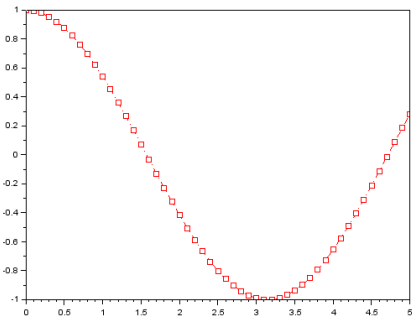
б)

-->clf();x=0:0.1:5;plot(x,cos,'r--x','thickness',3)



GlobalProperty - это необязательный параметр, представляющий собой совокупность пар вида {Свойство,Значение}, которые определяют глобальные свойства, присущие всем кривым, создаваемым функцией plot. Название свойства - символьная переменная. Значение может быть действительным числом, целым числом или строкой (скаляром или матрицей) в зависимости от типа используемого свойства. Например, спецификация: красный цвет, стиль линии длинное тире-точка, маркер в виде ромба может быть записана так: 'Color', 'red', 'LineSt', '-.', 'Marker', 'diam'. Полный набор возможных свойств можно посмотреть в справке по GlobalProperty.

--> clf();x=0:0.1:5;plot(x,cos,'Color','red','Linest','-.','Marker','s')



Пример:

Построить график функции y = sin(cos(x)) с помощью функции plot.

Пусть x изменяется на интервале [−2π; 2π] с шагом 0,1. Сформируем массив x. Вычисляя значение функции y = sin(cos(x)), для каждого значения массива x создадим соответствующее ему значение массива Y. Толщину линии графика выберем равной 3. Затем, воспользовавшись функцией **plot**(x,y) в открытом графическом окне получаем соответствующую кривую. Для построение соответствующего графика можно использовать следующие способы:

```
--> x=-2*%pi:0.1:2*%pi; plot(x,sin(cos(x)), 'thickness',3);
--> x=[-2*%pi:0.1:2*%pi]; plot(x,sin(cos(x)), 'thickness',3);
--> x=[-2*%pi:0.1:2*%pi]; Y=sin(cos(x)); plot(x,Y, 'thickness',3);
--> x=linspace(-2*%pi,2*%pi,123); Y=sin(cos(x)); plot(x,Y, 'thickness',3);
--> clf(); x=-2*%pi:0.1:2*%pi; function y=f(x),f=sin(cos(x)),endfuction,
        plot(x,f(x), 'thickness',3)
```

При выполнении любой из этих команд в графическом окне появится изображение кривой (рис. 4.1.

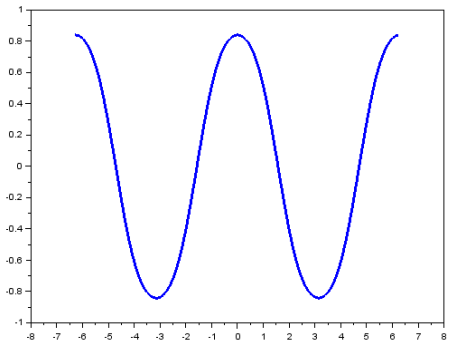


Рис. 4.1.

Для оформления вида графика используются следующие основные функции:

- ***xgrid()*** -добавление сетки на график*;*

Синтаксис данной функции может иметь вид:

- ✓ xgrid;
- ✓ xgrid(color);
- ✓ xgrid(color, thickness);
- ✓ xgrid(color, thickness, style),

где color - целое число, задающее цвет сетки; thickness действительное число, задающее толщину сетки, style - целое число, указывающее на стиль изображения сетки.

- ***xtitle****("Название графика", "Название оси абсцисс", "Название оси ординат")* - добавление подписей;
- ***legend(.., , ..)*** - создание легенды с перечнем всех отображенных графиков в системе координат;
- ***subplot(mnk)*** - функция разделения окна на матрицу, содержащую *m* строк, *n* столбцов, а *k*- это номер ячейки, в которой будет отображен график.

- **bar**(x, y) - команда для построения гистограммы, которая отображает значения y в зависимости от значений x в виде столбцов на координатной плоскости.

  Введем, например, далее в командном окне команды:

  --> xgrid(1,2,1)// сетка черного цвета (1), с толщиной линий 2, линии сетки сплошные (тип 1).

  --> xtitle('График функции y=sin(cos(x)','Ось x','Ось y')

  Получим такой вид графика (рис 4.2):

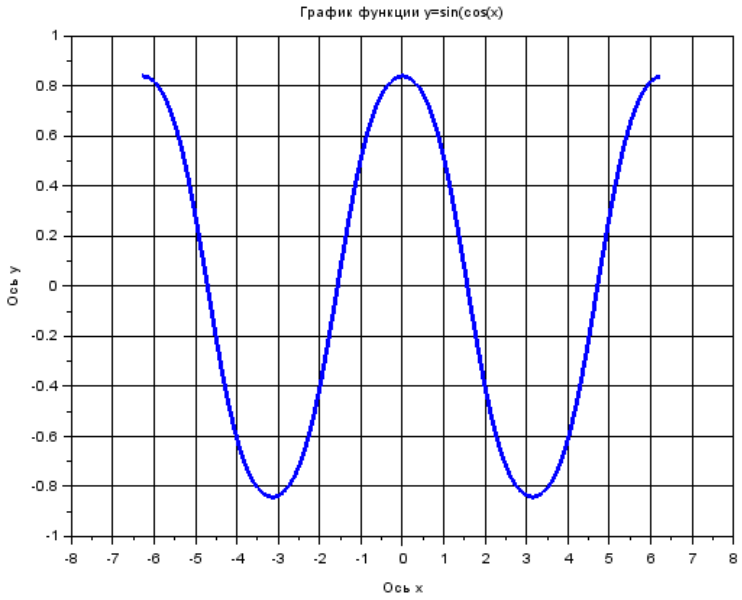


Рис. 4.2

Для задания цвета вместо его номера можно использовать его RGB код. Например: xgrid(color(128,48,0),1,3) – цвет сетки коричневый, толщина линий 1, пунктирная.

Установить желаемый вид и цвет графика можно, указав строковый параметр s, который может состоять из одного, двух или трёх символов, определяющие соответственно: ***цвет линии, тип маркера, тип линии графика***. Возможные значения представлены в таблицах:

Символы, определяющие цвет линии графика.

Символ	Цвет
y	желтый
m	розовый
c	голубой
r	красный
g	зеленый

b	синий
w	белый
k	черный

Символы, определяющие тип линии графика.

Символ	Тип линии
-	сплошная (по умолчанию)
:	штрих, чередующийся двумя точками
-.	штрих, чередующийся одной точкой
--	штриховая

Символы, определяющие тип маркера

Символ	Тип 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

В общем случае, число аргументов у функции **plot()** не ограничивается двумя. Эта функция имеет следующий формат:

plot(x1,y1,x2,y2,...).

Пример.

Построить графики функций y1=sin(x), y2=sin(2x), y3=sin(4x) на интервале x=0 ÷ 2π с шагом 0.1.

В командном окне введем команды:

```
-->x=0:0.1:2*%pi;
 -->y1=sin(x);
 -->y2=sin(2*x);
```

```
-->y3=sin(4*x);
-->clf()
-->plot(x,y1,'k',x,y2,'r',x,y3,'b','thickness',4)
-->xgrid()
```

В графическом окне получим графики заданных функций (рис. 4.3):

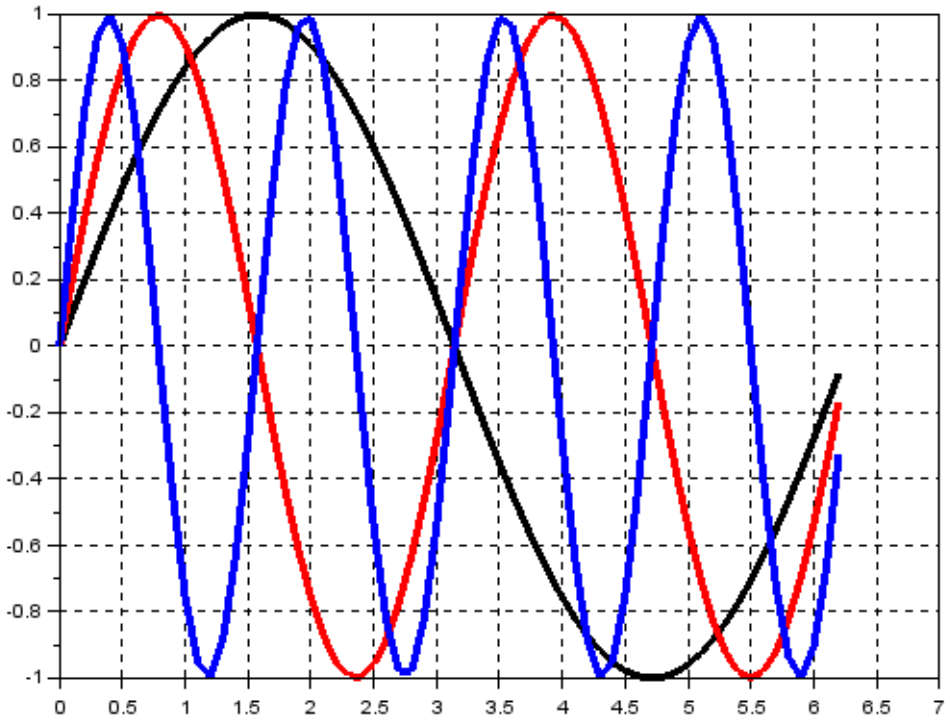


Рис. 4.3.

Имена созданных кривых можно вывести с помощью команды **legend**: **legend**(leg1, leg2, ..., legn, [pos]), где:

leg1 - имя первого графика;

leg2 - имя второго графика;

legn - имя n-го графика;

pos - числовой параметр, определяющий место расположения обозначений в графическом окне.

Значение pos	Размещение легенды
-1	В правом верхнем углу над областью графика
0	Место выбирается автоматически, чтобы не перекрывать область кривых
1	В правом верхнем углу (и по умолчанию)

2	В левом верхнем углу области графика
3	В левом нижнем углу области графика
4	В правом нижнем углу области графика

В нашем примере добавив команды title и legend получаем изображение графика с требуемыми пояснениями (рис. 4.4):

```
-->xtitle("ГРАФИКИ Y1-sin(x); Y2=sin(2x); Y3=sin(4x)","ОСЬ X","ОСЬ Y");
-->legend('Y1=sin(x)','Y2=sin(3x)','Y3=sin(4x)');
```

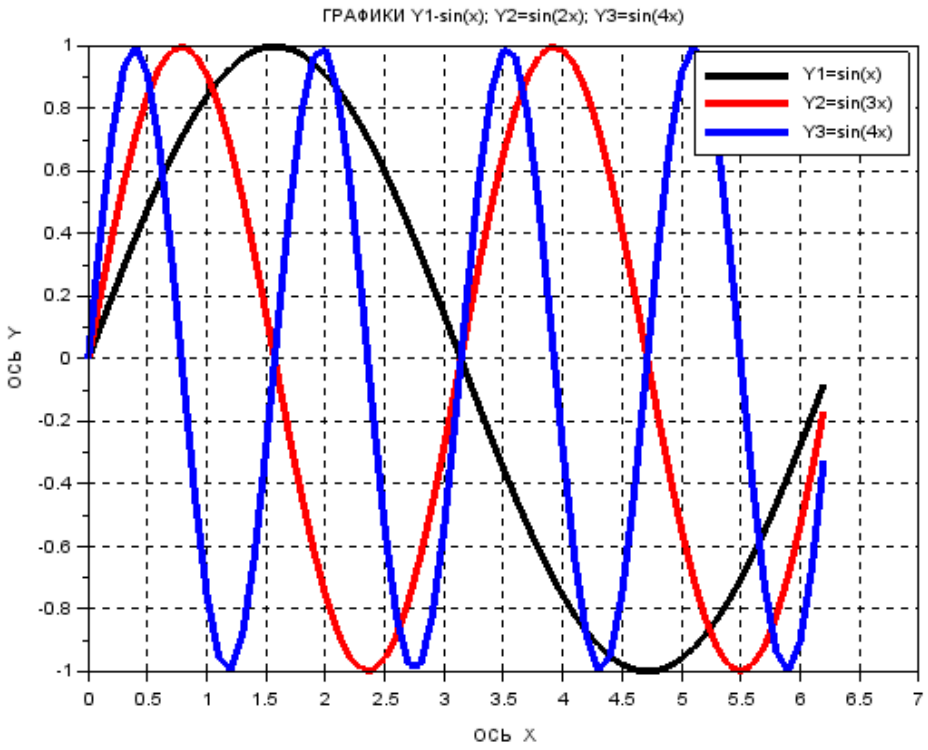


Рис. 4.4.

В Scilab можно выводить несколько графиков в одном окне, не совмещая их в одних координатных осях. Например, если графическое окно должно содержать 4 самостоятельных графика, все окно разбивается на 4 области, а затем в каждую из них выводится график функции.

Для формирования области в графическом окне служит команда **plotframe**:

plotframe(rect, tics [,grid, title, x-leg, y-leg, x, y, w, h]), где

- **rect** - вектор [xmin, ymin, xmax, ymax], который определяет границы изменения x и y-координат области;

- **tics** - вектор [nx, mx, ny, my], который определяет количество линий сетки по оси X (mx) и Y (my), величины nx и ny должны определять число подинтервалов по осям X и Y;
- **grid** - логическая переменная, которая определяет наличие (%t) либо отсутствие координатной сетки (%f). Этот параметр следует указывать для обеих осей, например, [%t,%t];
- **bound** - логическая переменная, которая при значении ‘true’ позволяет игнорировать параметры tics(2) и tics(4);
- **title** - заголовок, который будет выводится над графической областью;
- **x-leg**, **y-leg** - подписи осей графика X и Y;
- x, y - координаты верхнего левого угла области в графическом окне, w - ширина, h - высота окна. Значения x, y, w, h измеряются в относительных единицах и лежат в диапазоне [0, 1]. После определения области в нее можно вывести график функции с помощью команды plot.

4.2.3 Функция plot2d()

Следующей функцией, которая может быть использована для построения двумерных графиков, является функция **plot2d**. В общем виде обращение к функции имеет вид:

plot2d([logflag,][x,],y[,style[,strf[,leg[,rect[,nax]]]]])

- **logflag** – определяет тип шкалы (линейная или логарифмическая) по каждой из осей. Может принимать значения: "nn", "nl", "ln" и "ll". По умолчанию значение параметра установлено "nn" (линейные по обеим осям);
- строка из двух символов, каждый из которых определяет тип осей (n - нормальная ось, l - логарифмическая ось), по умолчанию - «n»;
- x - массив абсцисс;
- y - массив ординат или матрица, каждый столбец которых содержит массив ординат очередного графика - в случае, если необходимо построить графики нескольких функций y1, y2, ..., yn, когда все они зависят от одной и той же переменной x. При этом количество элементов в массиве x и y должно быть одинаковым. Если x и y - матрицы одного размера, то каждый столбец матрицы y отображается относительно соответствующего столбца матрицы x;
- **style** - устанавливает цвет для каждой кривой. Значением параметра style должен быть вектор той же размерности, что и число строящихся графиков. Может состоять либо из действительных чисел (например,

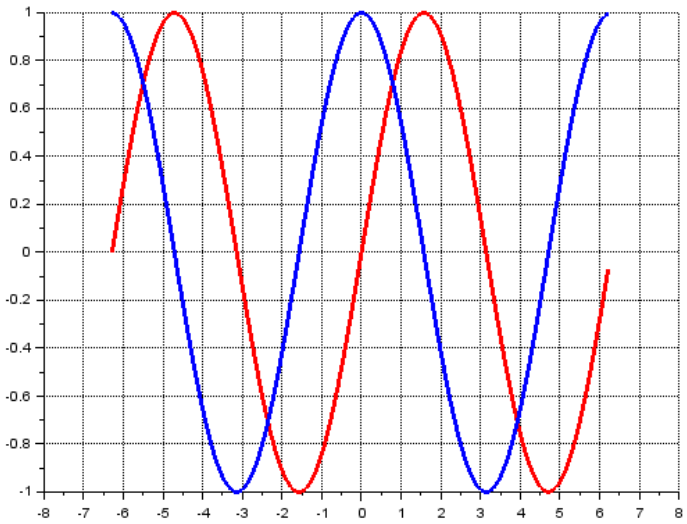
style=[3 5]), либо из строковых констант (style=[color("red"), color('blue')]). В первом случае, если параметр – положительное число, то кривая строится цветом, номер которого задан. Если число отрицательное, то кривая состоит из маркеров.

- **strf** -управление отображением некоторых элементов; строка из трех цифр-символов;
- **leg** -задает надписи легенды;
- **rect** -устанавливает видимыми границами графика;
- **nax** –устанавливает метки осей и засечки между ними. Значением должен быть вектор с действительными координатами: [nx,Nx, ny,Ny];
- управляет видимыми на оси делениями;
- **frameflag** - показывает построителю как размещать графики относительно осей;
- **axesflag** - настраивает стиль осей.

Следует отметить, что вовсе не обязательно использовать полную форму записи функции plot2d со всеми ее параметрами. В простейшем случае к ней можно обратиться кратко, как и к функции **plot**.

Пример построения 2-х графиков различного цвета.

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



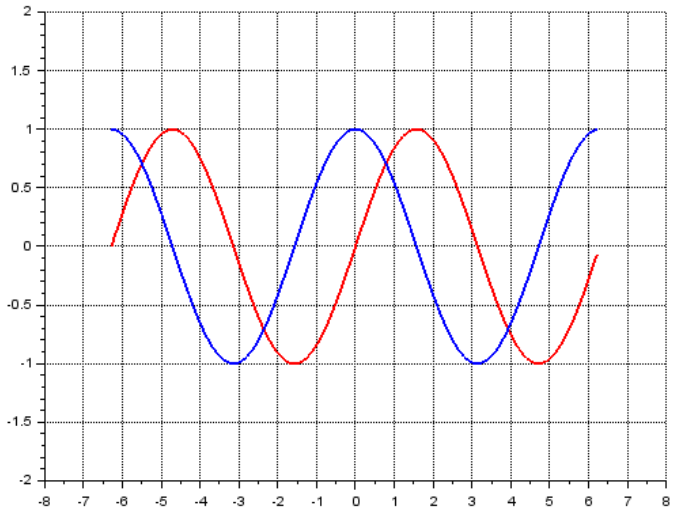
Аналогичный результат получается, если для построения использовать команду plot2d(x,y',[color("red"), color("blue")]) или plot2d(x,y',[2,5]);

Цвет может задаваться и RGB кодом, то есть для нашего примера можно использовать и такую конструкцию:

```
-->  plot2d(x,y',style=[color(255,0,245),color('blue')]);
```

- **rect** - этот вектор [xmin, ymin, xmax, ymax] определяет размер окна вокруг графика. Пример использования этого параметра:

```
--> clf()
--> x=-2*%pi:0.1:2*%pi;
--> plot2d(x,y', color("red"),color("blue")],rect=[-8,-2,8,2]);
--> xgrid()
```

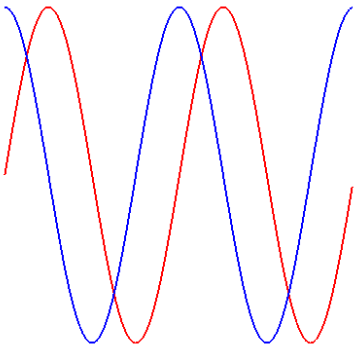


- **frameflag** - параметр определяет окно в котором, будет изображаться график, он может принимать следующие значения: 0 - не вычислять размеры окна, использовать значения по умолчанию или значения из предыдущего графика, 1 - размер окна определяется параметром rect, 2 - размер окна определяется из 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.

Примеры, позволяющие выделить следующие значения этого параметра:

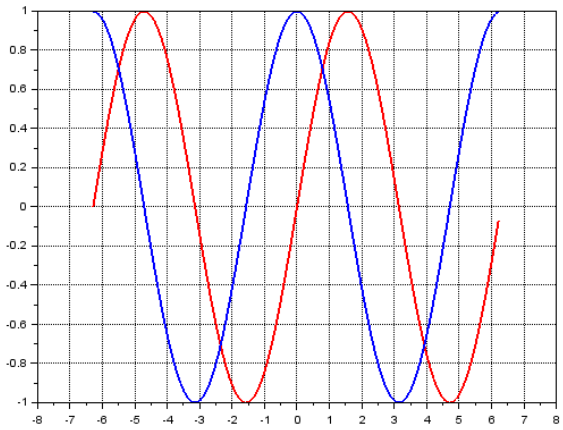
0 – нет рамки вокруг графика;

```
--> clf()
--> x=-2*%pi:0.1:2*%pi;
--> plot2d(x,y', [color("red"), color("blue")], axesflag=0);
```



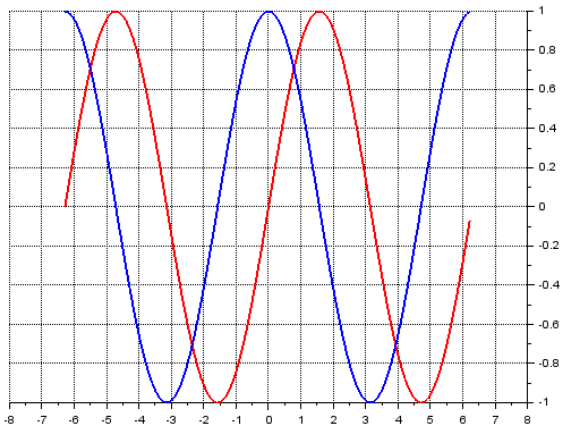
1 – изображение рамки, ось у слева;

```
--> x=-2*%pi:0.1:2*%pi;
--> plot2d(x,y', [color("red"), color("blue")], axesflag=1);
--> xgrid()
```



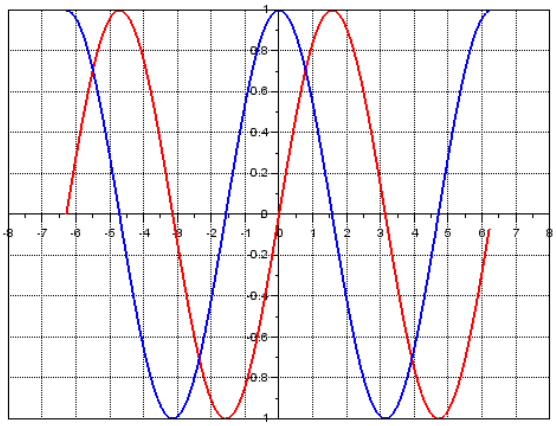
3 – изображение рамки, ось у справа;

```
--> x=-2*%pi:0.1:2*%pi;
--> plot2d(x,y',[color("red"), color("blue")], axesflag=3);
```



5 – изображение осей проходящих через точку (0,0);

```
--> plot2d(x,y',style=[color("red"), color("blue")], axesflag=5);
```

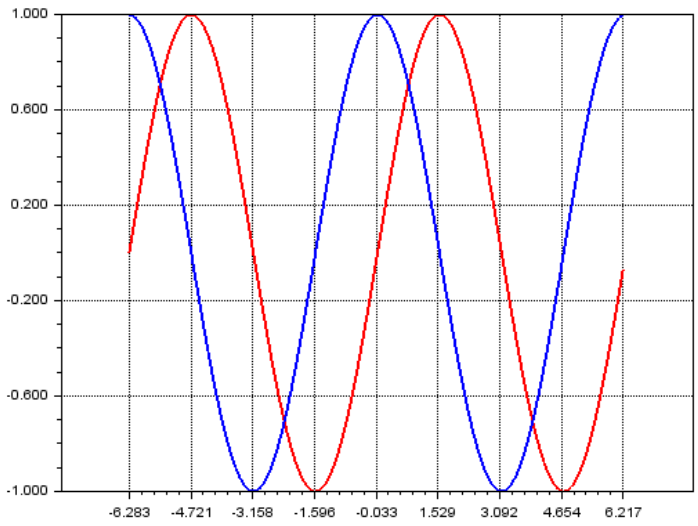


- **nax** - этот параметр используют, если параметр axesflag равен 1, nax представляет массив из четырех значений [nx, Nx, ny, Ny] – где Nx (Ny) – число основных делений с подписями под осью X (Y), nx (ny) – число промежуточных делений;

  Например, результатом выполнения следующей команды

  ```
  -->plot2d(x,y',style=[color("red"), color("blue")], axesflag=1,nax=[4,9,3,6])
  ```

  представление графика будет иметь следующий вид:

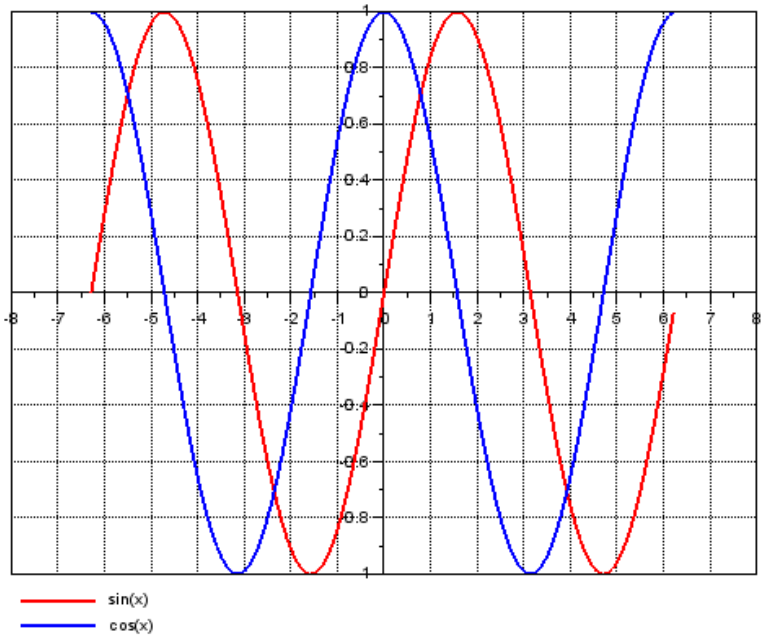


- **leg** – строка, определяющая легенды для каждого графика, структура строки такая: "leg1@leg2@leg3@...@legn", где leg1 – легенда первого графика, ..., legn – легенда n-го графика.

  Пример использования параметра legend.

```
--> clf()
```

```
--> x=-2*%pi:0.1:2*%pi;
-->plot2d(x,y',[ color("red"), color("blue")], axesflag=5, leg='sin(x)@cos(x)');
--> xgrid()
```

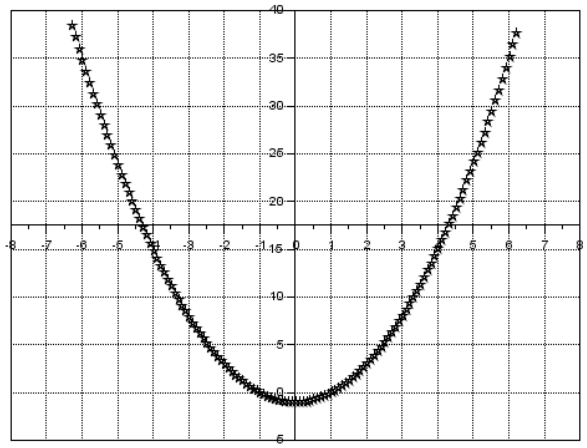


Для построения точеных графиков обращение к функции должно иметь вид: **plot2d**(x,y,m), где m-значение, определяющее вид маркера (m-отрицательное число)

Символ m	Тип маркера
-0	точка
-1	знак “плюс”
-2	крестик
-3	знак “плюс” вписанный в окружность
-4	закрашенный ромб
-5	не закрашенный ромб
-6	треугольник вершиной вверх
-7	треугольник вершиной вниз
-8	знак “плюс”, вписанный в ромб
-9	кружок
-10	звездочка
-11	квадрат
-12	треугольник вершиной вправо
-13	треугольник вершиной влево
-14	пятиконечная звезда

Пример:

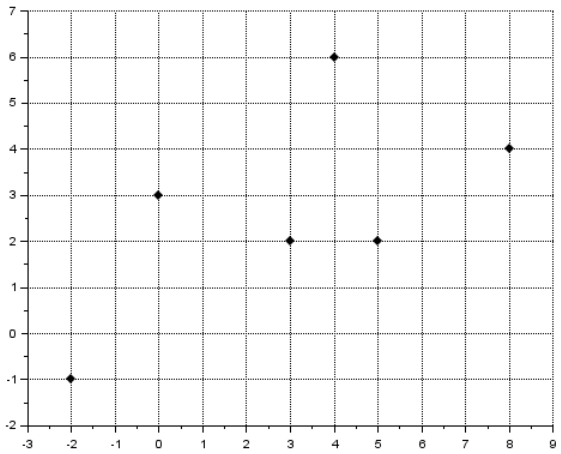
```
--> x=-2*%pi:0.1:2*%pi;
--> ym=x^2-1;
--> clf()
--> plot2d(x,ym,-14,axesflag=5);
--> xgrid()
```



Функцию plot2d можно использовать для того, чтобы просто нанести исходные данные на график, не соединяя их отрезками прямых. Для этого вместо номера цвета ставится целое отрицательное число. В зависимости от его абсолютного значения выбирается определенный тип маркера. Например, в результате работы программы

```
--> clf(); x=[3,4,-2,5,8,0]; y=[2 6 -1 2 4 3];plot2d(x,y,-4,rect=[-3,-2,9,7]); xgrid()
```

на график будут нанесены 6 точек с заданными координатами маркерами в виде закрашенного ромба:



4.2.4. Функции plot2d2, plot2d3, plot2d4

Scilab располагает и другими средствами построения двумерных графиков, имеющие другое графическое представление исходных данных. Для это используют разновидности функции plot2d: **plot2d2, plot2d3, plot2d4**.

Синтаксис функции plot2d2:

plot2d2([x],y) plot2d2([logflag],x,y,[style,strf,leg,rect,nax]).

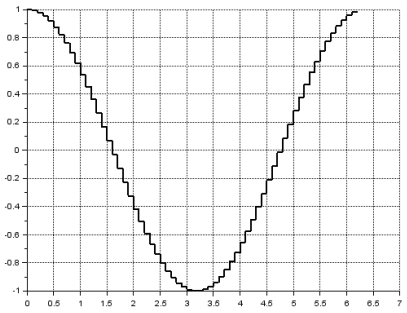
Аргументы функции – те же, что и у функции plot2d.

Функция **plot2d2(**x,y) используется для построение ступенчатых линий.

Например, в результате работы программы

--> clf();x=[0:0.1:2*%pi]'; clf();plot2d2(x,cos(x)); xgrid()

получим ступенчатый график:



Основное отличие состоит в том, что массивы x и y могут быть независимыми друг от друга функциями, но при этом они должны быть разбиты на одинаковое количество интервалов.

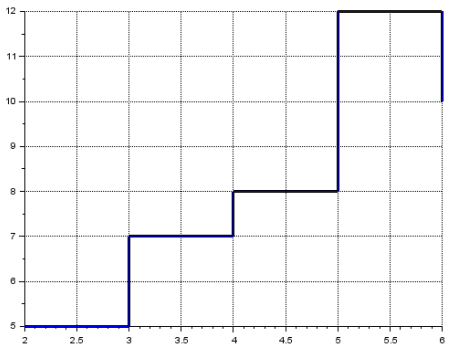
Пример:

Для заданных значений x=2, 3, 4, 5, 6 построить кусочно-ступенчатую функцию F, принимающую значения 5, 7, 8, 12, 10 соответственно.

Программа выполнения данного примера может иметь вид:

--> x=[2 3 4 5 6]; F=[5 7 8 12 10]; plot2d2(x,F); xgrid()

Результат выполнения:



Для построения функций с “заполнением” используется функция **plot2d3**, которая имеет следующие форматы представления:

- **plot2d3**([logflags,] x,y,[style,strf,leg,rect,nax]);
- **plot2d3**(y).

График будет представлен в виде вертикальных полос, ограниченных значениями функции от соответствующего аргумента x. Задаваемые аргументы, определяющие вид графика, аналогичны функции plot2d.

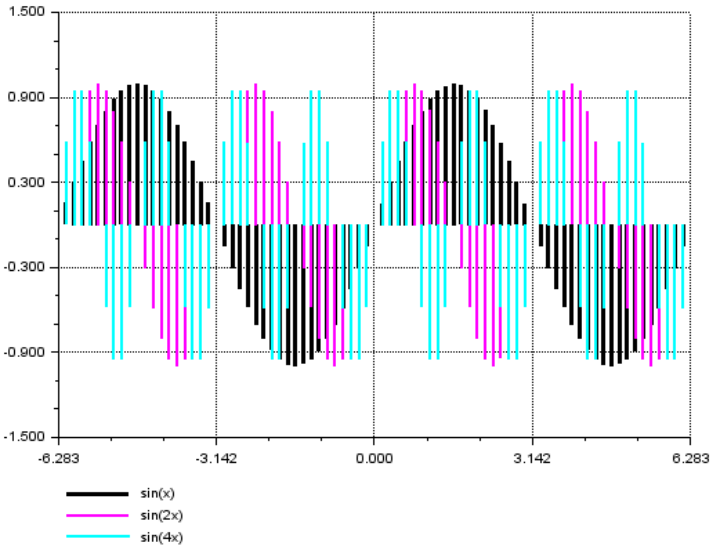
Пример.

Построить три графика в одном графическом окне с помощью функции plot2d3 с легендой и с заданными параметрами области рамки для функций: y1=sin(x), y2=sin(2x); y3=sin(3x) для x изменяющегося от -2π до 2π с шагом π/20.

В рабочем поле введем команды:

-->clf(); x=[-2*%pi:%pi/20:2*%pi]'; plot2d3(x,[sin(x) sin(2*x) sin(4*x)], [1,6,4], leg="sin(x)@sin(2x)@sin(4x)",nax=[2,5,2,6],rect=[-2*%pi,-1.5,2*%pi, 1.5])
--> xgrid()

Получим график



Для построения графиков, у которых соседние точки соединяются векторами, используется функция **plot2d4**, формат которой имеет вид:

- plot2d4([logflag,] x,y,[style,strf,leg,rect,nax]);
- plot2d4(y);
- plot2d4(x,y).

На графике будут видны маркеры в виде стрелок. Задаваемые аргументы, определяющие вид графика, аналогичны функции plot2d.

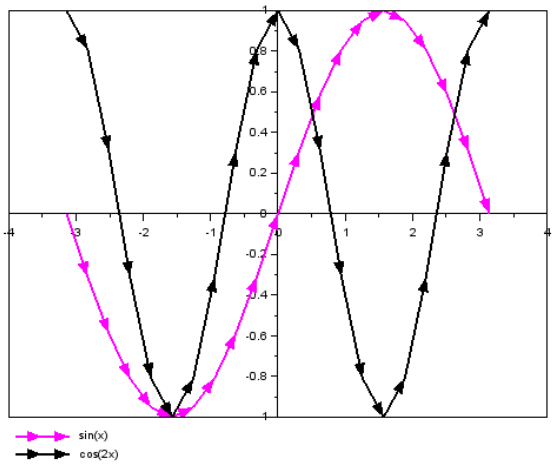
Пример.

Построить два графика в одном графическом окне с помощью функции plot2d4 с легендой и с заданными параметрами области рамки для функций: y1=sin(x), y2=cos(2x) для x изменяющегося от - π до π с шагом π/10.

Для введенных команд

```
--> clf(); x = [-%pi:%pi/10:%pi]'; plot2d4(x,[sin(x) cos(2*x)], style=[6,1],
leg="sin(x)@cos(2x)",axesflag=5);
```

график функций будет иметь вид:



4.2.5. Построение графиков в полярной системе координат

Полярная система координат полезна в случаях, когда отношения между точками проще изобразить в виде радиусов и углов. В декартовой, или прямоугольной, системе координат, такие отношения можно установить только путём применения тригонометрических уравнений.

В Scilab для формирования графика в полярной системе координат необходимо сформировать массивы значений полярного угла φ и полярного радиуса r, а затем обратиться к функции **polarplot**:

polarplot(fi,ro,[key1=value1,key2=value2,...,keyn=valuen]), где

- fi — полярный угол;
- ro — полярный радиус;
- keyn=valuen - последовательность значений свойств графика, определяющих его оформление.

Пример.

Построить математическую кривую, заданную уравнением

$r(\varphi) = a\cos(k\varphi + \theta)$ (полярная роза), где

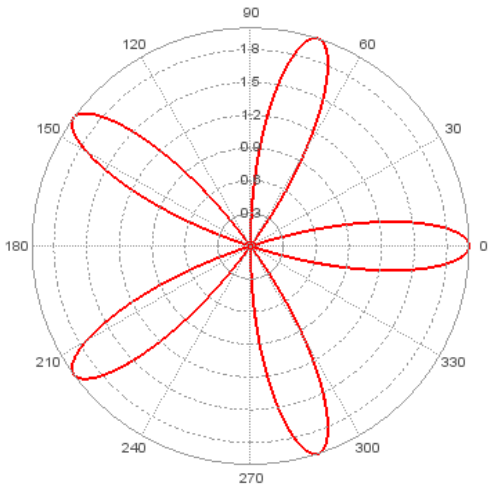
a – параметр, определяющий длину лепестков;

k- параметр, определяющий число лепестков; *k* лепестков, если *k* нечетное, и 2*k* – если *k* четное;

θ – произвольное число, определяющее угловой сдвиг.

Построение заданной кривой может быть реализовано следующим образом:

```
--> fi=0:0.01:4*%pi;
--> ro=2*(cos(5*fi));
--> polarplot(fi,ro,style=color("red"));
```



4.2.6 Построение графиков функций, заданных в параметрической форме

Заданная в декартовой системе координат кривая уравнениями

$$\begin{cases} x = x(t) \\ y = y(t) \end{cases}, \qquad (4.1)$$

где *t* – параметр кривой, называется параметрической кривой.

Область изменения параметра определяется как пересечение максимально возможных областей определения функций *x = x(t), y = y(t).* Исключение параметра *t* из системы (если оно возможно) приводит к уравнению, связывающему *x* и *y*, т.е. к уравнению вида f(*x*,*y*) = 0. Для приближенного построения графика кривой, заданной параметрически, достаточно построить таблицу значений *x* и *y* в зависимости от возможных значений параметра *t*. При этом надо учитывать $t_1 < t_2 < ... < t_n$. Затем на плоскости построить декартову систему координат и отметить точки с координатами (x_1, y_1), (x_2, y_2), ...,(x_n, y_n), соединить эти точки в порядке увеличения параметра *t*.

Таким образом, для построения графика функции, заданной параметрически в Scilab необходимо определить массив параметра *t*, определить массивы *x* = f(*t*), *y* = *g*(*t*) и построить график функции *y*(*x*), используя функции **plot**(*x*,*y*) либо **plot2d**(*x*,*y*).

Пример.

Построить график астроиды, которая задана уравнением $x^{\frac{2}{3}} + y^{\frac{2}{3}} = a^{\frac{2}{3}}$.

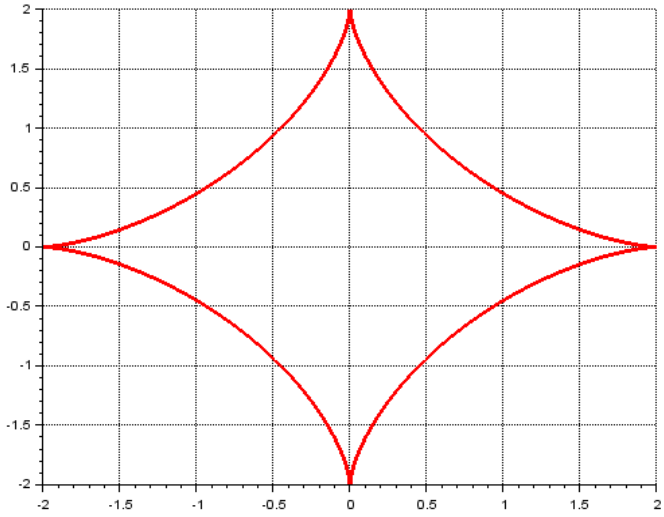
Параметрически астроида задается уравнениями

$$\begin{cases} x = a\cos 3t \\ y = asin3t, \quad t \in [0,2\pi] \end{cases} \qquad (4.2)$$

Для построения выполним следующие команды:

```
--> clf()
--> t=0:0.01:2*%pi;
--> x=2*(cos(t))^3;
--> y=2*(sin(t))^3;
--> plot2d(x,y,style=color('red'));
--> xgrid()
```

Получаем график:



4.2.7. Построение гистограмм

Для построения гистограмм в Scilab используют функции **bar, histc, histplot, hist3d.**

Формат функции **bar** имеет вид:

bar(x, y [,width] [,colors] [,style]).

Если x и y - векторы, то строятся прямоугольники стилем 6 (polyline_style type 6), причем размерности векторов x и y должны совпадать. Среди значений x и y могут быть отрицательные.

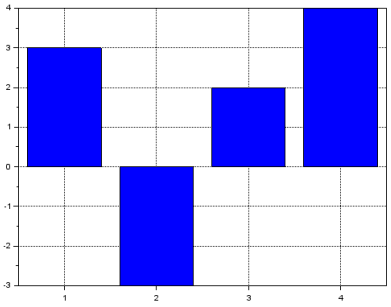
Необязательный параметр width определяет величину зазора между прямоугольниками и имеет значения 0< width ≤1. По умолчанию значение width=0.8. Зазор можно убрать совсем, если принять width=1.

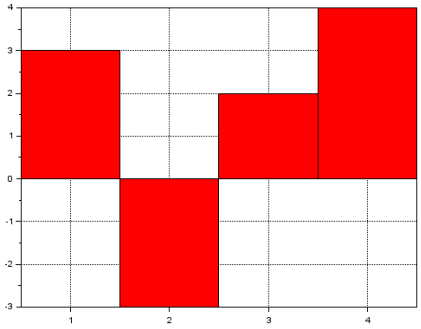
Параметр colors определяет цвет прямоугольников. По умолчанию выбирается синий.

Примеры :

а. clf(); x=[1 2 3 4];y=[3 -3 2 4];bar(x,y);xgrid().

б. clf(); x=[1 2 3 4];y=[3 -3 2 4];bar(x,y,1);xgrid()

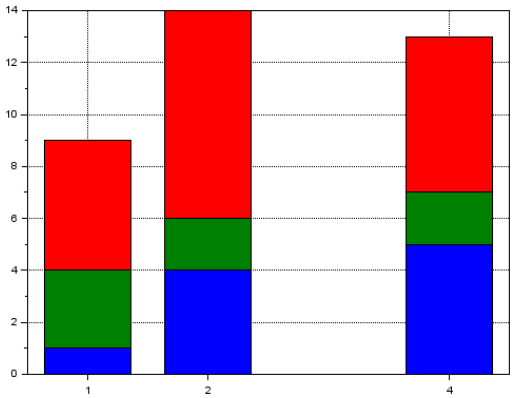
а. 

б. 

Символьная константа style принимает значения по умолчанию 'grouped' (сгруппированный) или 'stacked' (стековый – гистограмма с накоплением).

Пример:

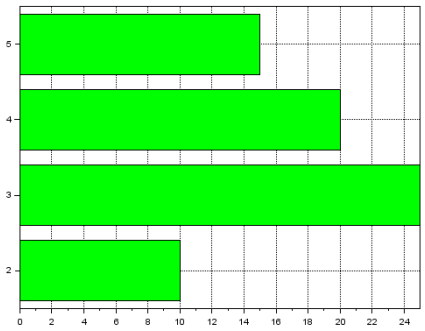
--> clf();A=[1 2 4]; B=[1,3,5;4,2,8;5,2,6]; bar(A,B,'stacked');xgrid()



Если прямоугольники необходимо расположить горизонтально, то в этом случае используется функция **barh**. При этом значения x и y следует поменять местами.

Пример:

clf(); x=[2 3 4 5]; y=[10 25 20 15]; barh(x,y,'g'); xgrid()



Для вычисления параметров гистограммы в Scilab используется функция **histc**, имеющая следующий формат:

- p = histc(data,n);
- p = histc(data,t),

где data – вектор исходных данных, n – число интервалов (целое положительное число), t вектор, элементы которого расположены в возрастающем порядке. Элементы вектора задают концы интервалов.

Примеры:

Пусть задан массив исходных данных: x=10, 12, 23, 24, 13, 16, 40, 21, 12, 14, 24, 11, 25, 23, 34, 23, 21; Необходимо разбить массив на 5 интервалов одинаковой длины (n=5) и подсчитать сколько значений попало в каждый интервал. p – это число значений, попавших в каждый из пяти интервалов [10-16], (16-22], (22-28], (28-34], (34-40] (шаг вычисляется автоматически: (40-10)/5=6).

Решением является следующая программа:

```
--> x=[10 12 23 24 13 16 40 21 12 14 24 11 25 23 34 23 21];
--> p=histc(x,5)
p  =
  7.  2.  6.  1.  1.
```

Если необходимо определить число значений входящих в указанные границы исходного массива, то это можно сделать указав вектор границ:

```
--> p=histc(x,[10 15 25 35 40 70])
p  =
  6.  9.  1.  1.  0.
```

Т.е. в интервал [10 15] попало 6 значений, в интервал (15 25] – 9, и т.д.

Если необходимо подсчитать относительную частоту попавших в каждый интервал значений, то есть поделить число значений, попавших в

каждый интервал, на объем выборки (общее число всех значений – 17 в нашем примере), то используется такой вариант функции:

```
--> p=histc(x,5,"countsNorm")
p  =
          0.4117647   0.1176471   0.3529412   0.0588235   0.0588235
```

Т.е. – это числа 7/17 2/17 6/17 1/17 1/17. Сумма всех чисел равна 1.

Если необходимо поделить число значений, попавших в каждый интервал, на ширину интервала (в примере это константа равная 6), то можно использовать такой вариант функции:

```
--> p=histc(x,5,"density")
 p  =
   1.1666667   0.3333333   1.   0.1666667   0.1666667
```

Если необходимо поделить число значений, попавших в каждый интервал, и на ширину интервала, и на объем выборки, то функция можно представить в виде:

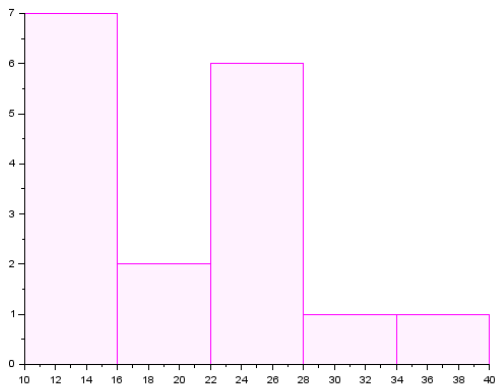
```
--> p=histc(x,5,"densityNorm")
 p  =
   0.0686275   0.0196078   0.0588235   0.0098039   0.0098039
```

Последний вариант используется для построений графиков плотности распределения вероятностей.

Если необходимо подсчитать количество значений, попавший в каждый интервал, и построить гистограмму, то можно использовать функцию p=histplot(n, x, normalization=%f, [style]). Для текущего примера получаем:

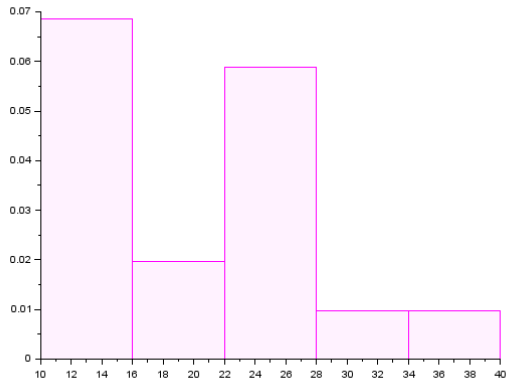
```
--> clc(); p=histplot(5, x, normalization=%f, style=6)
p  =
   7.   2.   6.   1.   1.
```

и построена гистограмма (ненормализованный вид)



Если необходимо, чтобы сумма площадей построенных прямоугольников равнялась единице, то нужно использовать нормализацию:

```
--> p=histplot(5, x, normalization=%t,style=6)
 p  =
          0.0686275   0.0196078   0.0588235   0.0098039   0.0098039
```



Еще одна форма записи функции histplot:

[cf, ind] = histplot(n, data [,normalization] [,polygon],)

polygon – необязательный параметр. Если он задан, то вместе с гистограммой строится полигон частот: середины верхних граней прямоугольников соединяются отрезками прямых.

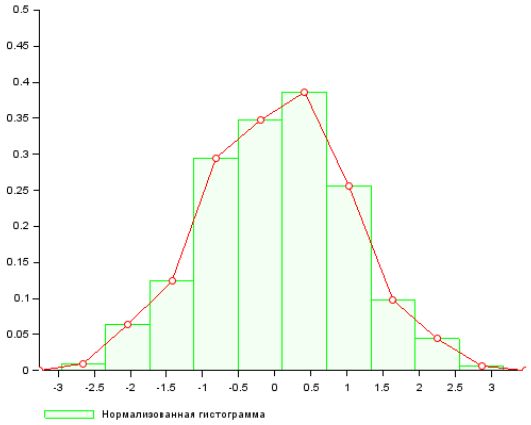
Пример.

Сгенерировать 1000 случайных чисел, распределенных по нормальному закону N(0,1). Диапазон полученных значений разбить на 10 интервалов одинаковой длины. Подсчитать количество значений, попавших в каждый интервал. По полученным данным по строить нормализованную гистограмму и полигон частот.

Решение можно представить в виде:

```
--> data = rand(1,1000,"normal");
--> clf(); histplot(10,data, polygon=%t,leg='Нормализованная гистограмма',
     style=3, rect=[-3,0,3,0.5])
 ans  =
0.0097848   0.0636015   0.1239414   0.2951762   0.3473621   0.3865015
0.2560368   0.0978485   0.0440318   0.0065232
```

Строится гистограмма зеленого цвета (style=3). Пределы изменения оси x от -3 до 3 и y от 0 до 0.5.



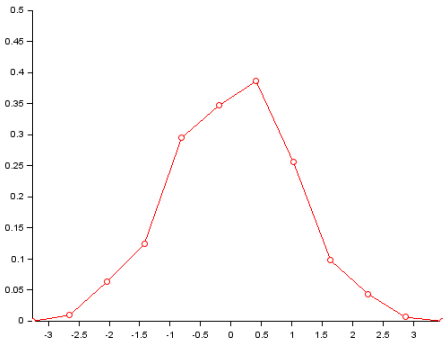
Если нужен только полигон частот, то можно задать style=8 и убрать текст легенды. При том же сгенерированном векторе данных data при выполнении команд

--> clf(); histplot(10, data, polygon=%t,style=8, rect=[-3,0,3,0.5])

получаем

ans =

0.0097848 0.0636015 0.1239414 0.2951762 0.3473621 0.3865015 0.2560368 0.0978485 0.0440318 0.0065232



Для построение круговых диаграмм служит оператор **pie**, который имеет следующие форматы:

- **pie**(x);
- **pie**(x[,sp[,txt]]),

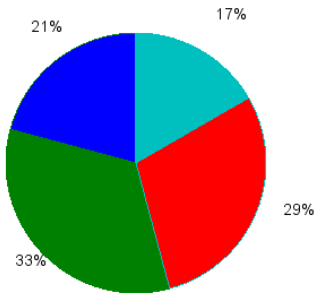
где x - скаляр или вектор, состоящий из положительных чисел x_i, txt – символьный массив.

Если размерность вектора x равна N, то круговая диаграмма x состоит из N частей, размер каждой части равен $\frac{x_i}{\sum_{i=1}^{n} x_i}$.

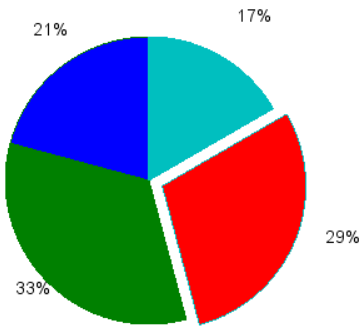
Пример.

Построить круговую диаграмму для массива M=25, 40, 35, 12.

После ввода команды --> clf(); pie([25 40 35 20]) получаем диаграмму:

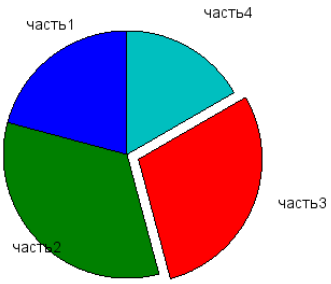


Если необходимо из круговой диаграммы вырезать один или несколько секторов, то в этом случае нужно использовать конструкцию **pie**(x[,sp[,txt]]), где sp - вектор размерностью N, определяющий номер выпадающих секторов. Если значение i-того элемента вектора sp не равна 0, то соответствующий сектор отделяется от других. Например, выполнение команды --> clf(); pie([25 40 35 20],[0 0 1 0]) позволяет получить следующий вид круговой диаграммы:



Вектор txt позволяет подписывать соответствующую i-ю часть диаграммы. По умолчанию надпись соответствующей части – ее процент от общей. Размерность вектора также равна N. Это можно проиллюстрировать следующим примером:

--> clf(); pie([25 40 35 20],[0 0 1 0],['часть1','часть2','часть3','часть4'])



Как уже отмечалось, для расположения нескольких графиков в одном окне без наложения их друг на друга используется функция **subplot**(m,n,k), где m – количество окон по горизонтали, n - по вертикали, k - номер окна, в которое будет выводиться текущий график (нумерация ведется по строкам слева направо). Функция должна стоять до оператора построения графика.

Пример.

Разбить графическое окно на 4 подокна и в каждом из них построить свой график.

Чтобы можно корректировать построение графика лучше написать соответствующий скрипт в редакторе и далее запустить на выполнение. В данном примере взяты различные графики, каждый из которых описывается в рамках своего окна.

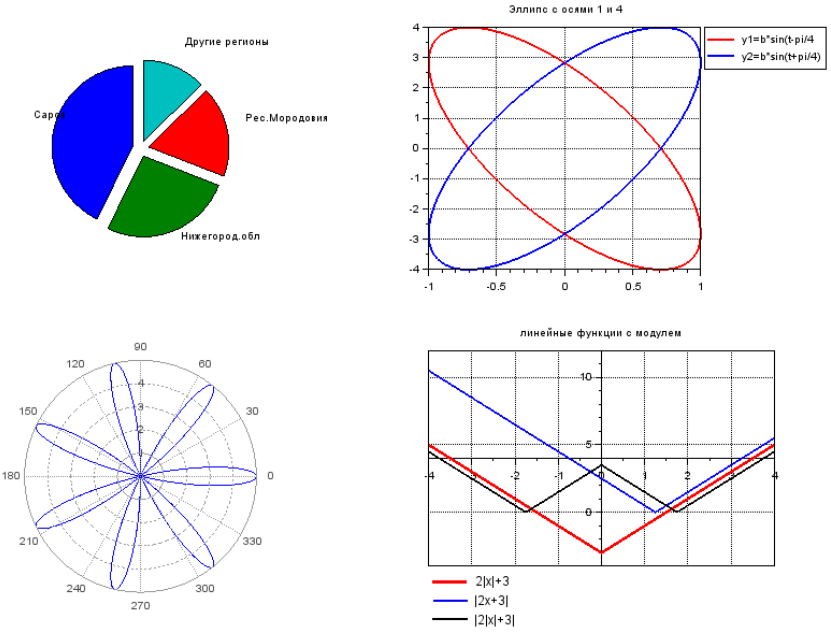
```
function GR()
  clf()
  subplot(2,2,1)
  st=[47,29,20,14];

  subplot(2,2,2)
  a=1; b=4; t=0:%pi/50:2*%pi; x=a*cos(t); y1=b*sin(t-%pi/4);
  y2=b*sin(t+%pi/4);
  plot(x,y1,'r',x,y2,'b','thickness',2);
  xgrid();  xtitle('Эллипс с осями 1 и 4'); legend('y1=b*sin(t-
  pi/4','y2=b*sin(t+pi/4)',-1);

  subplot(2,2,3)
  fi=0:0.01:4*%pi; r=5*(cos(7*fi));  polarplot(fi,r,style=color("blue"));

subplot(2,2,4)
```

```
  x=-4:0.05:4; y1=2*abs(x)-3; y2=abs(2*x-3); y3=abs(2*(abs(x))-3);
  Y=[y1;y2;y3];
  plot2d(x,Y',[ color("red"), color("blue"), color("black")],axesflag=5,
  leg='2|x|+3@|2x+3|@|2|x|+3|');
  xgrid()
endfunction
```



4.3. Форматирование графиков

При построении графиков в форматах соответствующих команд указывались и элементы оформления графика: цвет линий, толщина линий, названия графика и осей, расположение осей и т.п. Но Scilab позволяет выполнить оформление графика и после его построения, что порой облегчает задачу его форматирования.

Рассмотрим основные команды и их форматы наиболее часто используемые при оформлении графиков.

4.3.1. Команда title

Команда **title** используется для добавления на график заголовка (титульной надписи), а также нанесения текста в пустом графическом окне.

Синтаксис команды:

- title(my_title);
- title(my_title, <Property>),

где my_title - символьная константа, т.е. надпись, которая устанавливается над графиком. <Property>- представляет собой последовательность пар за писей вида {Свойство, Значение}, которые определяют набор свойств

созданного заголовка и их можно изменить. Названия всех свойств – символьные константы.

Для заголовка можно задавать следующие свойства:

- backgroundcolor – добавление заголовка в рамку, которая заливается определенным цветом, например: 'backgroundcolor', 'red'; color - цвет заголовка, например,'color', color(128,48,0);
- edgecolor – цвет границы рамки. Значения цветов - символьные константы, например, 'edgecolor', 'green';
- fontname – задает шрифт создаваемой надписи. Можно задать 10 шрифтов:
  - "courrier" или 0,
  - "symbol" или 1,
  - "times" или 2,
  - "times italic" или 3,
  - "times bold" или 4,
  - "times bold italic" или 5,
  - "helvetica" или 6,
  - "helvetica italic" или 7,
  - "helvetica bold" или 8,
  - "helvetica bold italic" или 9.

Значение шрифта можно задавать либо цифрой, либо символьной константой – названием шрифта, например, 'fontname', 3;

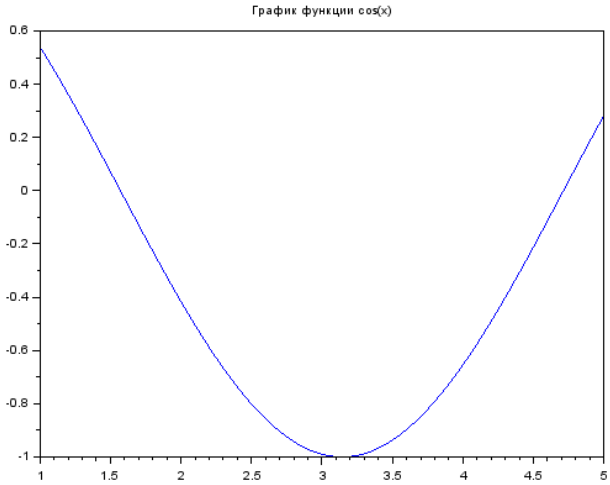
- fontsize - размер кегля. Целое число, большее или равное 0;
- position – вектор размерности 2: [x, y]. Задает позицию начала надписи. По умолчанию заголовок помещается в центре над верхней рамкой графика. Значения компонентов вектора могут принимать значения: x из отрезка [xmin, xmax], y – из отрезка [ymin, ymax]. При значениях, выходящих за пределы графического окна, надпись не видна;
- rotation – позволяет повернуть заголовок. Поворот производится против часовой стрелки на заданный угол;
- visible – позволяет скрыть текст надписи. В этом случае надо установить значение параметра равным "off": 'visible', 'off'. По умолчанию установлено значение "on".

Пример использования команды **title**:

```
--> clf();x=1:.05:5;plot(x,cos(x));title('График функции cos(x)')
```

Выполнение команд дает результат:

```
--> clf();x=1:.05:5;plot(x,cos(x));title('График функции cos(x)')
```



Пример.

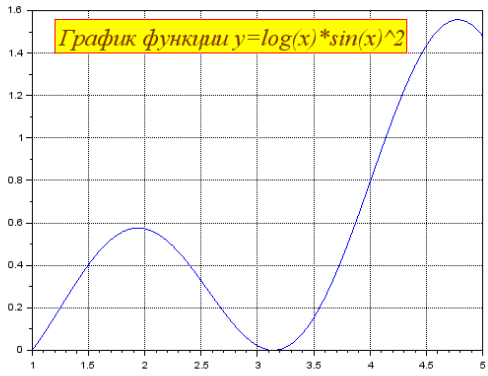
Построить график функции $y=\ln(x)\cdot\sin^2(x)$, x принимает значения от 1 до 5 с шагом 0.05. Заголовок графика разместить внутри области построения начиная с точки с координатами (1.2, 1.4). Цвет заголовка: «scilabbrown4» (RGB код (128, 48, 0)). Заголовок заключить в рамку красного цвета, которую залить желтым цветом. Размер заголовка установить равным 5, использовать шрифт "times italic".

Вполне очевидно, что записать все требуемые значения параметров title одной командой можно, но это будет слишком длинная строка, что вызовет некоторые неудобства в ее написании. Поэтому здесь целесообразно разбить форматирование на несколько коротких команд.

Для решения поставленной задачи напишем скрипт:

```
function grafic()
   clf()
   x=1.:0.05:5;
   y=log(x).*(sin(x))^2
   plot(x,y);
   xgrid
   r='График функции y=log(x)*sin(x)^2';
   title(r, 'backgroundcolor',  'yellow', 'color', color(128,48,0))
   title(r,'edgecolor', 'red', 'fontsize', 5)
   title(r,'fontname', 3, 'position', [1.2 1.4])
endfunction
```

И далее после его сохранения и выполнения командой обращения к скрипту --> grafic получаем:



4.3.2. Команда xtitle

Команда xtitle служит для написания заголовка графика и названия его осей. Формат команды имеет вид:

- xtitle(title,[x_label,[y_label,[z_label]]],),

где title – название графика, x_label, y_label, z_label – названия осей X, Y и Z соответственно.

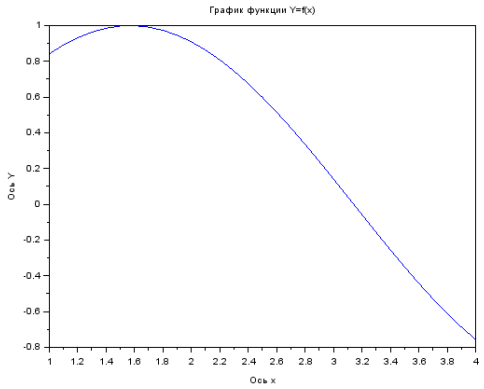
Пример

--> clf();x=1:.1:4;plot(x,sin(x))

Далее вводим команду форматирования заголовка и осей X и Y:

--> xtitle("График функции Y=f(x)", "Ось x ","Ось Y ")

Получаем:



Конечно, команда не длинная и ее можно в данном случае записать одной строкой:

clf();x=1:.1:4;plot(x,sin(x),xtitle("График функции Y=f(x)", "Ось x ","Ось Y "))

Результат будет, конечно, таким же.

4.3.3. Команда label

Команда label используется для установки надписей возле осей X, Y и Z.

Формат команды:

xlabel([axes_handle], label, [property_name, property_value,...]);

ylabel([axes_handle], label, [property_name, property_value,...]);
zlabel([axes_handle], label, [property_name, property_value,...]) , где

- label – символьная переменная, содержащая название оси;
- axes_handle – необязательный параметр, предназначенный для точного определения оси, для которой устанавливается надпись;
- [property_name, property_value,...] - необязательные параметры, содержащие пары{Свойство, Значение}.

Доступными для всех осей являются:

- "visible"- видимость оси со значениями "on" (ось видна) или "off" (ось не видна);
- "fontsize" или "font_size"- размер кегля (константа);
- "fontname" или "font_style"- название шрифта.

Доступно 10 шрифтов. Можно задавать либо имя шрифта, либо цифру, соответствующую его имени:

  - "courrier" или 0,
  - "symbol" или 1,
  - "times" или 2,
  - "times italic" или 3,
  - "times bold" или 4,
  - "times bold italic" или 5,
  - "helvetica" или 6,
  - "helvetica italic" или 7,
  - "helvetica bold" или 8,
  - "helvetica bold italic" или 9;
- "rotation", "font_angle"– угол поворота в градусах;
- "position"- позиция метки (вектор размерности 2);
- "color", "font_foreground"- цвет шрифта – символьная переменная, принимающая одно из возможных значений: "red", "green", "blue", "cyan", "magenta", "yellow", "black", "white". Вместо название цвета можно задать его RGB код. Так, вместо "red" можно написать color(255,0,0). В случае использования RGB кода список возможных цветов значительно расширяется.
- "edgecolor", "foreground"-цвет рамки вокруг текста. Возможные значения: "red", "green", "blue", "cyan", "magenta", "yellow", "black", "white". Вместо название цвета можно задать его RGB код.

- "backgroundcolor", "background" - цвет фона заливки надписи помещенной в рамку. Возможные цвета: "red", "green", "blue", "cyan", "magenta", "yellow", "black", "white". Вместо название цвета можно задать его RGB код.

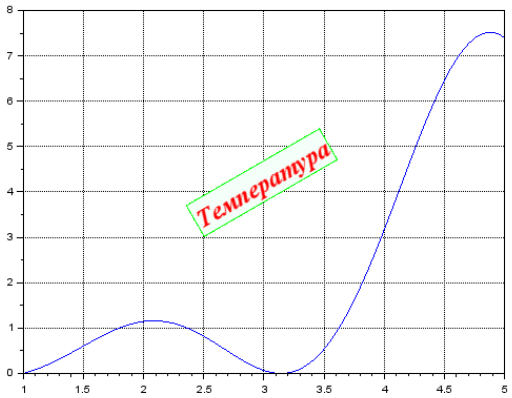
Пример.

Построить график функции $y=x\cdot\ln(x)\cdot\sin^2(x)$ на отрезке [1, 5] с шагом 0.05. Название метки оси X «Температура». Начало надписи установить в точке с координатами (2.5, 3). Для метки по оси X установить шрифт «times bold italic» красного цвета с размером кегля, равным 5, задать угол поворота надписи 30^o, вокруг метки создать рамку зеленого цвета. Внутреннюю часть рамки залить цветом «mint cream». Добавить к графику сетку.

После написания скрипта и его выполнения

```
function grafic2()
clf()
x=1:.05:5;
y=log(x).*(x.*sin(x)^2)
plot(x,y)
xlabel("Температура","fontsize",5,"fontname","times bold italic",
"rotation",30, "position",[2.5 3], "color", "red","edgecolor",
"green","backgroundcolor",color(245,255,250))
xgrid()
endfunction
```

получаем следующий график со смещенной надписью по осиX:



4.3.4. Команда legend

Используется для написания легенды и при этом могут использоваться различные варианты команды legend. Наиболее используемый вариант команды имеет вид:

legend(["string1","string2","string3",….],a=pos, [,boxed]),

где "stringn"– строковая константа, pos – число, значение которого определяет расположение легенды на графике:

Значение pos	Расположение легенды на графике
1	в верхнем правом углу
2	в верхнем левом углу
3	в нижнем левом углу
4	в нижнем правом углу
5	легенда перетаскивается мышью
-1	в правом верхнем углу справа от рамки графика
-2	в левом верхнем углу слева от рамки графика
-3	в левом нижнем углу слева от рамки графика
-4	в правом нижнем углу справа от рамки графика
-5	в левом верхнем углу над рамкой графика
-6	в левом нижнем углу под рамкой графика

Параметр boxed определяет наличие рамки вокруг легенды и по умолчанию принимает значение %t («истина»).

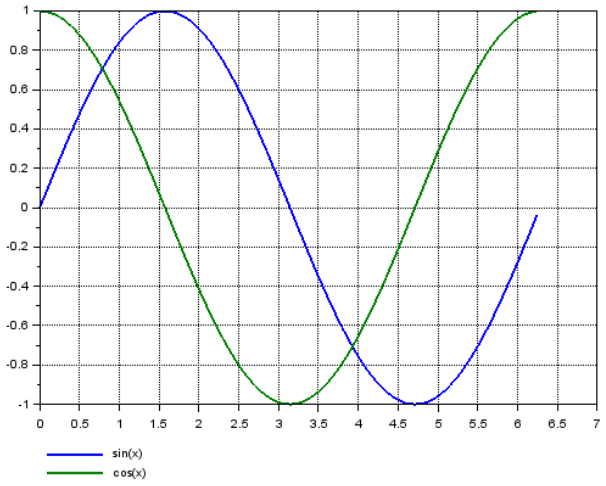
Пример

Построить в одном графическом окне два графика y1=sinx и y2=cosx линиями разного цвета толщиной 2. x меняется от 0 до 2π с шагом 0.05. Легенду разместить в левом нижнем углу под графиком. Рамку вокруг легенды убрать. Добавить сетку графического окна.

Выполнив команды:

clf(); x=0:0.05:2*%pi; y1=sin(x); y2=cos(x); plot(x,y1,x,y2,'thickness',2), legend("sin(x)","cos(x)",-6, boxed = %f), xgrid()

получаем:



4.4 Лабораторное задание.

4.4.1 С использованием функции **plot**() в интервале [-10;10] постройте графики линейных функций:

$y1 = ax + b;\ \ y2 = (a + b)x + a;\ \ y3 = (a - 2)x + c;$

Значения коэффициентов a, b и c и последующие требуемые значения выбираются из таблицы 4.1 в соответствии с заданным вариантом.

Графики должны отличаться по цвету. Установите толщину графиков размером 3 (4).

Задайте координатную сетку. Приведите название графиков. Обозначьте оси координат.

Приведите легенду графиков.

4.4.2 Разбейте графическое окно на 4 области и постройте соответственно графики функций:

$y_1 = ax + b;\ y_2 = |ax + b|;\ \ y_3 = a|x| + b;\ \ y_4 = |a|x|+b|.$

4.4.3 Постройте касательную к параболе $ax^2 + bx + c$ в точках x_0;

4.4.4 Постройте прямую, перпендикулярную к прямой, заданной уравнением $y=ax+b$, в точке $x=e$.

4.4.5 Постройте прямую, параллельную прямой $y=ax+b$ и проходящую через заданную точку M_1;

4.4.6 Постройте прямую, перпендикулярную касательной к кривой, заданной уравнением $y = ae^{-0.1bx}$ в точке $x=t$ и проходящей через точку M_2;

4.4.7 Постройте кривую окружности радиусом R и с центром в точке $(a;b)$;

4.4.8 Постройте эллипс с осями x и y, если $x=|a \cdot c|$, $y=|b \cdot c|$, и имеющий наклон к оси $0x$ 45^o.

4.4.9 Постройте логарифмическую спираль и спираль Архимеда. ($R_{ls} = ae^{0.1*b*\varphi}$; $R_A = a\varphi$ для $0 \le \varphi \le 4\pi$;

4.4.10 В соответствии с заданным вариантом выполните индивидуальное задание приведенной в таблице 4.2.

4.4.11 Ответьте на контрольные вопросы.

4.4.12 Оформите отчет о выполненной работе

Таблица 4.1

№ вар.	Заданные значения коэффициентов, точек, величин										
	a	b	c	x_0	x_0	x_0	e	M_1	t	M_2	R
1	1	-4	-3	-3	-2	0	3	(5; 0)	-1	(2; 4)	5
2	2	-1	-4	-3	2	4	2	(1; -2)	-2	(1; 2)	6
3	-2	-6	2	-4	-1.5	0	3	(1; 10)	-2	(8; -14)	7
4	3	-3	1	-4	0.5	3	2	(-1; 0)	-3	(0; 8)	8
5	2	-1	2	-3	-1	4	3	(-2; 2)	-1	(-2; 2)	12
6	4	-2	1	-2	0	2	3	(1; 20)	0	(1; 5)	9
7	5	-3	4	1	3	5	-3	(1; -15)	3	(4; 6)	10
8	-4	3	3	-2	2	4	3	(-1; 20)	2	(2; -6)	8
9	4	-2	1	-4	3	3	-1	(1; -5)	-2	(0; 6)	9
10	-2	1	-1	-4	0	4	2	(1; -10)	0	(-1; -2)	5
11	4	-3	-1	-4	-1	2	1	(-1; 10)	3	(3; -2)	3
12	2	-5	2	-3	1	4	0	(2; 5)	2	(1; 6)	4
13	-1	5	-3	-4	-2	2.5	-2	(0; -5)	-1.5	(0; -6)	3
14	-3	2	-1	0.5	2	4	2	(1; 10)	0	(1; 4)	5
15	2	-2	2	-4	-2	5	-1	(0; 0)	-2	(-1; 4)	2.5
16	4	-6	4	-4	2	4	2	(1; -15)	-2	(1; 20)	5
17	5	-5	3	-4	-1	3	2	(2; -5)	-2	(0; 20)	3
18	-3	3	5	-3	-1	0.5	-1	(1; 5)	-1	(-1; -6)	3
19	4	-2	-2	-3	1	4	2	(1; -10)	-0.5	(1; 6)	3
20	2	-1	2	-2	0	1	-1.5	(0; -6)	-1	(-2; 2)	2.5
21	-4	3	2	-0.5	1	4	-2	(1; 10)	-3	(-2; 12)	4
22	-5	1	-1	-2	0	2	-1	(-4; 0)	2	(3; -6)	3
23	6	-3	1	-3.5	-1.5	2	-2	(-1; -20)	-3	(-2; 5)	5
24	3	-1	2	-3	0.2	4	1.5	(1; 5)	0	(0; 3)	3
25	4	1	-3	-4	-2	3	-3	(-2; -5)	-1	(0; -5)	6

Таблица 4.2

№ Вар.	Задание
1	Постройте ромб по 3-м заданным координатам его вершин и впишите в него окружность.
2	Постройте ромб по 3-м заданным координатам его вершин и определите его площадь и периметр.
3	Постройте шестиконечную звезду (звезда Давида) на основе правильного шестиугольника и определите его площадь.
4	Постройте пятиконечную звезду на основе правильного пятиугольника с и определите его площадь.

5	Построить параллелограмм по заданным координатам 2-х его смежных вершин и точки пересечения диагоналей. Определить площадь и периметр построенного параллелограмма.
6	Построить треугольник по заданным координатам его вершин. Определить координаты вписанной и описанной окружностей. Построить вписанную и описанную окружности.
7	Построить треугольник по заданным координатам двух его вершин и заданным значениям 2-х его сторон, инцидентных 3-ей вершине.
8	Постройте процесс вычисления значения числа π как отношение длины периметра вписанного правильного n-угольника к диаметру описанной окружности.
9	Постройте процесс вычисления значения числа π как отношение длины периметра описанного правильного n-угольника к диаметру вписанной окружности.
10	Постройте квадрат по заданным координатам 2-х его противоположных вершин.
11	Постройте параллелограмм по заданным координатам 3-х его вершин.
12	Построить ромб по заданным координатам 2-х его противоположных вершин и длине второй диагонали.
13	Построить прямоугольник по заданным координатам вершин одной стороны и значению длины другой.
14	Построить равнобедренную трапецию по заданным координатам вершин основания и координате вершины боковой стороны.
15	Построить шестиугольник по заданным координатам его вершин. Определить его площадь и периметр.
16	Построить параллелограмм по заданным координатам 2-х его смежных вершин и точки пересечения диагоналей. Определить площадь и периметр построенного параллелограмма.
17	Построить пятиугольник по заданным координатам его вершин. Определить площадь и периметр полученного пятиугольника.
18	Построить эллипс с заданными полуосями a и b и заданной точкой их пересечения $M(x,y)$, наклоненного относительно оси 0y на 45 градусов, и в который вписана окружность, центр которой совпадает с точкой пересечения осей эллипса. Определить площадь части эллипса без учета площади вписанной окружности.

19	Построить фигуру, получающуюся из двух треугольников, имеющих только одну совпадающую вершину, координаты которых заданы. Определить площади каждого треугольника и периметр полученной фигуры.
20	По заданным 3-м координатам прямоугольника построить сам прямоугольник с диагоналями. В треугольники, полученные пересечением диагоналей, вписаны окружности. Определить радиусы вписанных окружностей о соответствующие их центры.
21	Квадрат задан координатами трех его вершин вершинами. В квадрат вписана окружность, в которую в свою очередь тоже вписан квадрат. Построить полученную фигуру и определить разность площадей большого и малого квадратов.
22	В окружность с заданным центром в точке $M(x,y)$ и заданного радиуса R вписан правильный шестиугольник, в который вписана окружность и в которую в свою очередь вписан квадрат. Построить соответствующую фигуру и определить разность площадей шестиугольника и квадрата.

Лабораторная работа № 5
Численное дифференцирование и интегрирование в Scilab

Цель работы: изучение и освоение методов численного дифференцирования и интегрирования в системе Scilab. Приобретение практических навыков научно – технических и инженерных расчетов с использованием пакета математического моделирования Scilab.

5.1 Теоретическая часть

Задачи численного дифференцирования и интегрирования возникают, когда необходимо вычислить производные или определённые интегралы от функций, заданных таблично или, когда непосредственное дифференцирование и интегрирование затруднено из-за сложного аналитического вида функций. Основные формулы численного дифференцирования вытекают непосредственно из определения производной функции *f(x)*. Но, как хорошо известно, численное дифференцирование функций, исходя из определения производной, может привести к большим погрешностям результатов, в свою очередь, для производных высших порядков возможны ещё большие погрешности. Более общий подход к задаче численного дифференцирования табличной функции основан на использовании интерполяционных полиномов.

Не все версии Scilab, к сожалению, позволяют получить аналитические выражения производных от функции ***f(x)***. В Scilab имеется функция **numderivative()**, предназначенная для вычисления значений производных 1-го или 2-го порядка в заданных точках от аналитических функций. Эта функция имеет следующие форматы:

numderivative(*f*, *x*);

J=**numderivative**(*f*, *x*);

J=**numderivative**(*f*, *x*, *h*);

h=**numderivative**(*f*, *x*, *h*, *order*);

[*J*,*H*]=**numderivative**(*f*, *x* ,*h*, *order*).

Входные параметры:

f – функция или список функций, от которых берутся производные;

x – вектор значений независимых переменных, в которых вычисляются производные;

h– шаг, используемый в конечно-разностных приближениях. Если *h* не указан, то шаг по умолчанию вычисляется в зависимости от *x* и порядка. Если *h*-матрица 1x1, то она расширяется до того же размера, что и *x*;

order– порядок конечных разностей (порядок по умолчанию равен 2), доступные значения *order*: 1, 2 или 4.

Выходные параметры:

J – матрица (*необязательный параметр),* для функции от одной переменной размером 1×1, для функции от двух переменных – 2×2 (значения частных производных выводятся на ее диагонали);

H– матрица (*необязательный параметр),* в которую выводятся частные производные второго порядка.

Рассмотрим примеры использования функции **numderivative()** для вычисления производных.

Пример:

Вычислить значения производной заданной функции $f(x) = 3^x + 3x$ в точке $x = 2$

Решение этой задачи в командном окне будет иметь вид:

```
--> deff('y=f(x)','y=3^x+2*x');
--> numderivative(f,2)
 ans  =
   11.887511
```

Пример:

Следующий пример иллюстрирует вычисление производных заданной выше функции в точках, заданных вектором значений x. Например, в точках x=2, 3, 4, 5.

```
--> x=[2 3 4 5];
--> numderivative(f,x)
 ans  =
   11.887511   0.         0.         0.
   0.         31.662532   0.         0.
   0.         0.         90.987595   0.
   0.         0.         0.         268.96279
```

Для получения выражения производной от полинома, созданного в аналитическом виде используется функция **derivat().**

Пример:

```
-> p = poly([2, 2, 5, 4], "x", "c")
 p  =
  2 +2x +5x² +4x³ // заданный полином;
--> p1 = derivat(p) //получение выражения производной полинома;
 p1  =
  2 +10x +12x²//полученное выражение производной полинома.
```

Получить значения производных полинома в заданных точках можно, конечно, также с использованием функции **numderivative().**

--> c// заданный полином;

--> xp=[2 3 4 6]// вектор значений x, в которых вычисляются производные;

--> numderivative(fp,xp)//вычисление производных в заданных точках.

```
 ans  =
   70.   0.    0.    0.
   0.    140.  0.    0.
   0.    0.    234.  0.
   0.    0.    0.    494.
```

Функция **numderivative()** позволяет вычислить и значения частных производных. Например, вычисление частных производных функции $f(x_1, x_2, x_3) = {x_1}^2 x_2 + {x_2}^2 x_3 + {x_3}^2 x_1$ для $x_1 = 1;\ x_2 = 3;\ x_3 = 6$:

--> deff('fc=f(x)','fc=x(1)^2*x(2)+x(2)^2*x(3)+x(3)^2*x(1)');

--> x=[1 3 6];

--> numderivative(f,x)

```
 ans  =
   42.   37.   21.
```

Для дифференцирования функций, заданных в виде массива, в Scilab используется метод численного дифференцирования, основанный на том, что функцию *y(x),* заданную в равноотстоящих точках x_i $(i = 0, 1, 2 \cdots, n,\ n \in N)$ отрезка $[a, b]$ с помощью значений $y_i = f(x_i)$, приближенно заменяют интерполяционным полиномом Ньютона, построенным для множества узлов $x_0, x_1, \cdots, x_k$ $(k \le n)$, и вычисляют производные $y' = f'(x)$, $y'' = f''(x)$ и т.д

$$y'(x_0) = \frac{1}{h}\left(\Delta y_0 - \frac{\Delta^2(y_0)}{2!} + \frac{\Delta^3(y_0)}{3!} - \frac{\Delta^4(y_0)}{4!} + \frac{\Delta^5(y_0)}{5!} - \cdots\right)$$

Для численного дифференцирования в Scilab используется команда vd=**diff**(*y* [,*n*]), где *y* –значения функции *y(x)* в виде вектора вещественных чисел, *n* –порядок дифференцирования. В результате функция выводит вектор вещественных чисел vd, представляющий собой разности порядка *n* интерполяционного полинома Ньютона $\Delta y, \Delta_2 y, \cdots, \Delta_k y$.

Пример:

```
y  = 5   20   45   80   125   180  245
```

--> d1=diff(y)

```
 d1  =
   15.   25.   35.   45.   55.   65.
```

```
--> d2=diff(y,2)
 d2  =
   10.   10.   10.   10.   10.
--> d3=diff(y,3)
 d3  =
   0.   0.   0.   0.
```

Вычисление интегралов

При вычислении определенных интегралов первообразную функцию $F(x)$ не всегда удается выразить аналитически, а кроме того иногда подынтегральная функция $f(x)$ задана в виде таблицы (x_i и y_i, где i= 1, 2, …, n). Это приводит к необходимости использования численных методов интегрирования. Существует ряд методов численного интегрирования. Во всех этих методах вычисление осуществляется по приближенным формулам, называемым **квадратурами.**

Что касается вычисления значений определённых интегралов, то в основе большинства приближенных методов вычисления лежит разбиение интервала интегрирования $[a,b]$ на n элементарных отрезков и замена на каждом частичном отрезке интегрирования подынтегральной функции $f(x)$ интерполяционным полиномом определенной степени или сплайнами разных типов. Точки разбиения $a = x_0, x_1, \cdots, x_n = b$ называются узлами интегрирования, а расстояния между узлами $h_i = x_i - x_{i-1}$ шагами интегрирования. Шаг интегрирования может быть как постоянным, так и переменным. Приближенное значение интеграла определяется суммой частичных интегралов от функций $\varphi_i(x)$, взятых в пределах от x_{i-1}до x_i:

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} \int_{x-1}^{x_i} \varphi_i(x)dx.$$

Так, например, для получения формул прямоугольников, трапеций и Симпсона используют полиномы соответственно нулевой, первой и второй степени.

Формулы прямоугольников:

$$\int_a^b f(x)dx = h\sum_{i=0}^{n-1} y_i = h\sum_{i=1}^{n} y_i,$$

где h-шаг интегрирования, y_i-значение подинтегральной функции от аргумента x_i, i=0, 1, 2, …, n, n-число разбиений интервала интегрирования $[a,b]$.

Формула трапеций:

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

где h-шаг интегрирования, y_0-значение подинтегральной функции при $x=a$, y_n- значение подинтегральной функции при $x=b$.

Формула Симпсона:

$$\int_a^b f(x) = \frac{h}{3}(y_0 + 4y_1 + 2y_3 + \cdots + 2y_{n-2} + 4y_{n-1} + y_n).$$

Для вычисление интеграла функции, которая описана набором значений x_i и y_i, используется функция **inttrap**([*x*],*y*).

Вычислить значение интеграла *y(x)* функции, значения которой заданы в виде массива векторов *x* и *y*.

```
--> x=0:0.25:5; //массив x
--> y=0:0.5:10;//массив значений функции y(x).
--> inttrap(x,y) // значение полученного интеграла функции
 ans  =
   25.
```

Пример:

Вычислить значение интеграла функции $f(x) = \int_3^{12} \sqrt{5x^2+4}\,dx$ по методу трапеций с шагом 0.2 и с шагом 0.01.

С шагом 0.2:

```
--> x=3:0.2:13;
--> y=sqrt(5*x^2+4);
--> Int=inttrap(x,y)
 Int  =
   180.18807
```

С шагом 0.01:

```
--> x=3:0.01:13;
--> y=sqrt(5*x^2+4);
--> Int=inttrap(x,y)
 Int  =
   180.18777
```

Для вычисления интеграла по формуле Симпсона в Scilab применяется функция **integrate(),** формат которой имеет вид:

integrate(*fun***,** *x*, *a*, *b*, [,*er1* [,*er2*]]),

где *fun*- функция, задающая подынтегральное выражение в символьном виде;

x- переменная интегрирования, так же задается в виде символа;

a, *b*- пределы интегрирования, действительные числа;

er1 и *er2*– переменные, задающие абсолютную или относительную точность вычислений (действительные числа).

При обращении к этой функции шаг интегрирования не задается, а используется параметр - требуемая точность вычисления интеграла. Если точность вычисления не задана, то вычисление проводится с погрешностью, установленной по умолчанию.

Для предыдущей функции, например, получаем:

```
--> integrate('sqrt(5*x^2+4)','x',3,13,0.0001)
 ans  =
  180.18777
```

5.2 Лабораторное задание

5.2.1 Выберите из таблицы 5.1 вариант индивидуального задания.

5.2.2 В соответствии с вариантом найдите уравнение касательной и уравнение нормали в указанной точке x_k заданной кривой.

5.2.3 Найдите уравнение прямых, параллельных касательной и нормали, определенных в п.2 и проведенных через указанную точку x_p.

5.2.4 Постройте графики заданной кривой и найденных прямых в одном графическом окне. Все линии должны быть различимы по цвету. Выполните форматирование графика с указанием его названия (указания функции), описания осей, легенды.

5.2.5 Для заданного вектора V_x определите значения соответствующих производных.

5.2.6 Для заданной функции y_2 найдите значения частных производных для указанных значений ее аргументов x_i.

5.2.7 Оформите отчет по данному разделу. Из таблицы 5.2 выберите вариант индивидуального задания.

5.2.8 Вычислите определенный интеграл 1 с использованием функции, реализующей формулу трапеций inttrap(x, y), предварительно выбрав пределы интегрирования и получив таблицу значений подынтегральной функции с шагом h, задав значения аргумента и функции в виде векторов.

5.2.9 Вычислите значение определенного интеграла с использованием функции, реализующей формулу Симпсона integrate ('f',a,b), где f– имя функции. Сравните полученный результат с результатом, полученном с использованием функции inttrap(x, y)

5.2.10 Постройте график подинтегральной функции. Выполните интегрирование функции, заданной в виде массива данных (интеграл 2). Постройте график функции с использованием функции plot2d3 и ее огибающую.

5.2.11 Оформите отчет по данному разделу.

5.2.12 Ответьте на контрольные вопросы.

Таблица 5.1

№ Вар	Заданные функции и значения	
1	$y1_1 = 3\sin 4x \cdot e^{2x} - 4x^2 - 3.5x$ $y_2 = x_1^2 x_2 + x_1 x_2^2 + x_3^2 x_1$	$x_k = 2;\ x_p = 4;\ V_x$ =[1, 2.5, 3, 5] $x_1 = 2;\quad x_2 = 1;\ x_3 = 4.$
2	$y_1 = 2\cos^2 x \cdot e^{0.5x} + 3x$ $y_2 = 2x_3 x_2^2 + 3x_1^2 x_1 - x_3^2 x_1$	$x_k = 3;\ x_p = 2;\ V_x$ =[2, 4, 5, 6] $x_1 = 3;\quad x_2 = 1;\ x_3 = 5.$
3	$y_1 = 3x^2 - 2\ln x + 2$ $y_2 = x_1^2 x_2^2 + x_1 x_2^2 + x_1^2 x_3^2$	$x_k = 1;\ x_p = 2;\ V_x$ =[1, 2, 3, 4] $x_1 = 2;\quad x_2 = 1;\ x_3 = -1.$
4	$y_1 = 4\cos 3x \cdot \log_4 5\,x + 2$ $y_2 = e^{x_1^2} x_2^3 + \ln x_2^2 + 4x_1$	$x_k = 2;\ x_p = 3;\ V_x$ =[1.5, 2, 3.5, 4] $x_1 = 2;\quad x_2 = 1;\ x_3 = 4.$
5	$y_1 = 3\,x^2 \sin 4x + 2x - 4$ $y_2 = x_1 \ln x_2 + x_2 \ln x_3 - x_1 x_2 x_3$	$x_k = 0;\ x_p = \pi;\ V_x$ =[-1, 1, 2, 4] $x_1 = 4;\quad x_2 = 2;\ x_3 = 6.$
6	$y_1 = 7x^2 - e^{0.2x} - 1$ $y_2 = x_1 \sin x_2 + x_2 \sin x_3 + x_3 \sin x_1$	$x_k = -2;\ x_p = 2;\ V_x$ =[-1, 0, 1, 2] $x_1 = -\pi;\quad x_2 = 0;\ x_3 = \pi.$
7	$y_1 = 0.5x^2 + e^{0.1x} + x$ $y_2 = x_1 e^{x_2} + x_2 e^{x_3} + x_3 e^{x_1}$	$x_k = -1;\ x_p = 1;\ V_x$ =[-2, 0, 2, 5] $x_1 = -0.5;\quad x_2 = 0.5;\ x_3 = 1.$
8	$y_1 = 8x^2 \cos 2x + 10x \sin x + x$ $y_2 = x_1 \sin x_1 + x_2 \sin x_2 + x_3 \sin x_3$	$x_k = -2;\ x_p = 2;\ V_x$ =[-1, 0, 1, 2] $x_1 = 0\quad x_2 = \pi;\ x_3 = 2\pi.$
9	$y_1 = -2.5x^2 + 7x + 3$ $y_2 = x_1 \ln x_2 + x_2 \ln x_3 + x_3 \ln x_1 x_2$	$x_k = 2.5;\ x_p = 1;\ V_x$ =[-2, 1, 2, 3] $x_1 = 3;\quad x_2 = 4;\ x_3 = 5.$
10	$y_1 = x \sin x - \log_3 x^2$ $y_2 = x_1^2 x_3 + x_1 x_2^2 - x_1 x_2 x_3$	$x_k = 0.8;\ x_p = \pi;\ V_x$ =[1, 1.5, 2, 4] $x_1 = -4;\quad x_2 = -2;\ x_3 = 4.$
11	$y_1 = x e^{\cos x} + 2$ $y_2 = x_1 \cos x_2 + x_2 \sin x_3 + x_3 \cos x_1$	$x_k = -2;\ x_p = 1;\ V_x$ =[-1, 0, 1, 2.7]

		$x_1 =$ -π/2; $x_2 = 0$; $x_3 = \pi/2$.
12	$y_1 = -2.5x^2 + 7x + 3$ $y_2 = x_1 \ln x_2 + x_2 \ln x_3 + x_3 \ln x_1 x_2$	$x_k = 2.5$; $x_p = 1$; V_x =[-2, 1, 2, 3] $x_1 =$ 3; $x_2 =$ 4; $x_3 = 5$.
13	$y_1 = x \sin x - \log_3 x^2$ $y_2 = x_1^2 x_3 + x_1 x_2^2 - x_1 x_2 x_3$	$x_k = 0.8$; x_p =π; V_x =[1, 1.5, 2, 4] $x_1 =$ -4; $x_2 = -2$; $x_3 = 4$.
14	$y_1 = xe^{\cos x} + 2$ $y_2 = x_1 \cos x_2 + x_2 \sin x_3 + x_3 \cos x_1$	$x_k =$ -2; $x_p = 1$; V_x =[-1, 0, 1, 2.7] $x_1 =$ -π/2; $x_2 = 0$; $x_3 = \pi/2$.
15	$y_1 = x^2 \cos(x + \pi) - 8 * \ln x^2 - 2$ $y_2 = \sin x_1 \sin x_2 \sin x_3$	$x_k = 4$; $x_p = 2$; V_x =[1, π, 2, 2π] $x_1 =$ -π/4; $x_2 = \pi/2$; $x_3 = \pi/4$.
16	$y_1 = x \sin^2 x + \log_5 x^2 - x$ $y_2 = \cos x_1 \cos x_2 \cos x_3$	$x_k =$ -2; x_p =-0.5; V_x =[-1, 0, 1, 2] $x_1 =$ -π/3; $x_2 = \pi/3$; $x_3 = \pi/6$.
17	$y_1 = 2e^x - 0.5x\pi^x$ $y_2 = \log_2(x_1 + x_2) + x_1 x_2 \log_2 x_3$	$x_k = 1.4$; x_p =2;V_x =[1,1.3,2, 2.5] $x_1 =$ 2; $x_2 =$ 4; $x_3 = 8$.
18	$y_1 = x^2 + 5 \sin x + \ln \cos^2 x$ $y_2 = x_1 e^{x_2 + x_3} + x_2 + e^{x_1}$	$x_k =$ -2; x_p =2;V_x =[-1,1.3, 2, 2.5] $x_1 = e$; $x_2 =$ 1; $x_3 = \pi$.
19	$y_1 = 2\pi^{\cos 2x} - x$ $y_2 = x_1 \ln x_2 x_3 + x_1 \ln(x_2 + x_3)$	$x_k = 3$; $x_p = -2$; V_x =[-1,0, 1, 4] $x_1 =$ 3; $x_2 = e/2$; $x_3 = e$. .
20	$y_1 = x^2 \cos x + 3x \sin x + \sin 2x$ $y_2 = x_1^3 x_2 + x_2^3 x_3 + x_3^3 x_1$	$x_k =$ -2; x_p =2; V_x =[-1,0, 1, 2] $x_1 =$ 5; $x_2 =$ 8; $x_3 = 12$.
21	$y_1 = \cos x \ln(3x + 2) - \sin x \ln 2x^2$ $y_2 = \sqrt{x_1 x_2 + x_2 x_3 + x_1 x_3}$	$x_k = 1$; x_p =4; V_x =[2, 4, 5, 5.2] $x_1 =$ 4; $x_2 =$ 8; $x_3 = 16$.
22	$y_1 = 3e^{\cos x} + 5e^{\sin x}$ $y_2 = x_1 \cos x_2 + x_2 \cos x_3 + x_3 \cos x_1$	x_k =1.4; x_p =2; V_x =[0.7, 1, 2, 3] $x_1 =$ π/4; $x_2 = \pi/2$; $x_3 = \pi/4$.
23	$y_1 = 3^{x+1} - \ln 2^{x+2} - (-2)^x$ $y_2 = x_1 2^{x_2} + x_2 2^{x_3} + x_3 2^{x_1}$	$x_k = 0.5$; $x_p = -2$; V_x =[-1,0,1, 2] $x_1 =$ 2 ; $x_2 = 4$; x_3 =6.
24	$y_1 = \ln^2(x^3 + 2) - 5 \ln(x^2 + 2) - 2x$ $y_2 = x_1^3 x_2^3 + x_2^2 x_3^2 + x_1 x_3$	x_k =2.8; x_p =-1; V_x =[2, 3, 4, 5] $x_1 =$ 2; $x_2 =$ 4; $x_3 = 5$.
25	$y_1 = x^2 \sin^2 x + x \cos x - x$ $y_2 = x_1^2 + x_2 x_3^3 + x_1 x_2 x_3$	$x_k = 2$; x_p =1; V_x =[-0.5,0, 0.5, 1] $x_1 = 4$; $x_2 =$ 5; $x_3 = 6$.

Таблица 5.2

№ вар.	Интеграл 1	Интеграл 2
1	$\int_a^b (\cos x)^2 (\sin x)^3 \, dx$	x=[2, 3, 5, 7, 8, 10, 12, 13, 14, 16]; y=[7, 6, 6, 6, 5, 3, 3, 2, 1,0].

2	$\int_a^b \frac{\sqrt{4-x^2}}{x^2}dx$	x=[1, 3, 4, 6, 7, 8, 10, 11, 12, 14]; y=[13.5, 7.5, 7.5, 7.5, 7.5, 6, 10.5,10.5, 13.5 12].
3	$\int_a^b \frac{x^2}{\sqrt{3x^2+3x+1}}dx$	x=[0 2 4 6 8 10 12 14 16 18]; y=[2, 20, 12, 8, 20, 16, 8, 4, 24, 12].
4	$\int_a^b \frac{1}{3(\sin 2x)^2}dx$	x=[5 6 7 8 9 10 11 12 14 16]; y=[12, 6, 9, 6, 10, 10, 14, 16, 9, 11].
5	$\int_a^b \frac{x^3\cos x}{e^{0.5x}}dx$	x=[1, 2, 3, 4, 6, 8, 10, 13, 16, 20]; y=[7, 10, 9, 11, 8, 12, 12, 15, 16, 16].
6	$\int_a^b \frac{e^{0.5x}}{\sqrt{1+3e^{-0.5x}}}dx$	x=[2, 3, 5, 6, 8, 9, 10, 12, 14, 15]; y=[12, 12, 11, 14, 14, 16, 16, 18, 20, 20].
7	$\int_a^b (\cos 2x)^2+(\sin 3x)^3\,dx$	x=[1, 2, 3, 4, 6, 8, 10, 12, 14, 16]; y=[20, 20, 22, 22, 22, 26, 26, 30, 36, 40].
8	$\int_a^b (2x+3)e^x dx$	x=[10, 12, 14, 16, 18, 20, 21, 22, 23, 24]; y=[6, 8, 8, 12, 14, 18, 24, 30, 35, 40].
14,9	$\int_a^b \frac{e^x+x}{e^{3x}-x^2}dx$	x=[5, 6, 7, 8, 10, 12, 14, 17, 18, 19]; y=[45, 44, 42, 40, 36, 36, 40, 46, 50, 58].
10	$\int_a^b \frac{x^2\ln(x-1)}{x+2}dx$	x=[10, 11, 12, 13, 14, 16, 18, 20, 22, 24]; y=[14, 12, 15, 16, 18, 22, 24, 26, 28, 30].
11	$\int_a^b \frac{x^2+1}{x^2(1+x)}dx$	x=[4, 5, 7, 8, 10, 11, 13, 14, 16, 18]; y=[54, 52, 50, 50, 56, 60, 66, 72, 80,90].
12	$\int_a^b \frac{3e^x}{3+x^3}dx$	x=[12, 14, 16, 18, 20, 21, 22, 23, 24, 25]; y=[17, 20, 14, 25, 17, 15, 15, 15, 20, 25].
13	$\int_a^b \frac{2\lvert\sin 2x\rvert}{x}dx$	x=[11, 14, 17, 20, 23, 24, 25, 26, 27, 28]; y=[3, 5, 7, 6, 8, 10, 12, 14, 18, 22].
14	$\int_a^b (e^{\sin x}+e^{\cos x})dx$	x=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]; y=[-4, -2, 0, 0, 2, 6, 10, 14, 18, 22].
15	$\int_a^b x(1+e^{-0.5x})dx$	x=[3, 4, 6, 7, 9, 10, 12, 13, 14, 15]; y=[-12, -10, -5, -1, 4, 8, 12, 16, 20, 24].
16	$\int_a^b x^2(\cos 2x+\lvert\sin 2x\rvert)dx$	x=[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]; y=[4, 6, 8, 8, 7, 7, 9, 12, 16, 22].
17	$\int_a^b \frac{\log_5(x^2+2)}{\ln(2x+x^{2)}}dx$	x=[17, 18, 19, 20, 22, 24, 26, 28, 29, 30]; y=[5, 3, 0, -3, -5, -5, -4, -3, -2, -1].
18	$\int_a^b \frac{\sin^2 x}{\sin x+\cos 2x}dx$	x=[-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5]; y=[0, 2, 4, 5, 6, 6, 6, 5, 4, 2, 0].
19	$\int_a^b (x^{0.75}+3x^{0.3})dx$	x=[-2.5, -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5]; y=[6, 5, 4, 3, 2, 0, -2, -3, -4, -5, -6].
20	$\int_a^b \frac{\sin 3x}{\cos^2 5x}dx$	x=[-6, -5, -3, -1, 0, 1, 2, 3, 4, 5, 6]; y=[12, 10, 8, 6, 4, 4, 4, 6, 8, 10, 12].
21	$\int_a^b (3x+\sin^2 3x)dx$	x=[-2.5, -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5]; y=[0.1, 0.2, 0.4, 0.8, 1.2, 2, 1.2, 0.8, 0.4, 0.2,0.1].
22	$\int_a^b \log_4(e^{2x}+x^2)\,dx$	x=[10, 12, 14, 16, 17, 19, 21, 22, 24, 26, 28]; y=[0.2, 1, 1.5, 2, 4, 8, 4, 2, 1.5, 1, 0.2].

23	$\int_a^b \log_5(x^2 e^{-0.5x} + 0.5)\,dx$	x=[-2, -1.5, -1, 0, 1, 2, 3, 4, 6, 8]; y=[0.2, 0.6, 1, 1.8, 3, 3, 2, 1.1, 0.5, 0.1].
24	$\int_a^b (e^x \cos x - \sin 0.5x)\,dx$	x=[-4, -3.5, -3, -2.5 -2, 0, 2, 2.5, 3, 3.5, 4]; y=[0.2, 0.8, 2, 4, 8, 12, 8, 4, 2, 1, 0].
25	$\int_a^b \frac{3x^3 + 2x^2 + x + 4}{\sqrt{x^4 + 2x^3 + x^2 + 2}}\,dx$	x=[5, 7, 9, 10, 11, 12, 13, 14, 16, 18, 20]; y=[0.25, 0.5, 1, 2, 4, 5, 4, 2, 1, 0.5, 0.25].

5.3 Методические указания:

5.3.1 Создайте файл.doc, для копирования результатов выполняемой работы с целью последующего оформления отчета.

5.3.2 При построении графиков исследуемых функций обязательно установите границы окна с использованием команды rect(…), значения которых должны быть по модулю больше максимальных значений аргумента и функции.

5.3.3 При выполнении пункта 9 пределы интегрирования необходимо выбирать с учетом ‘критических’ точек функции.

5.3.4 Для построения огибающей в п.12 можно использовать функцию plot(x,y). Толщина огибающей должна быть не менее 2 и иметь цвет, отличимый от других линий графика. Все графики должны иметь сетку.

5.3.5 В отчете должны быть приведены все пояснения по каждому выполняемому пункту.

5.4 Контрольные вопросы

5.4.1 Какой функцией в **Scilab** определяется символьное значение определенного интеграла?

5.4.2 Назначение функции **inttrap(x,y)?**

5.4.3 Что возвращает функция **inttrap(x,y),** если **y(x)** – матрица?

5.4.4 Можно ли использовать функцию **inttrap(x,y),** если узлы по оси **x** - не равноотстоящие?

5.4.5 Какая функция Scilab позволяет вычислить определенный интеграл с заданной точностью?

5.4.6 Способы задания подынтегральной функции при вычислении определенного интеграла с использованием функции **integrate().**

5.4.7 Как задать точность вычисления определенного интеграла?

Лабораторная работа № 6
Решение системы линейных алгебраических уравнений в Scilab

Цель работы: изучение и освоение методов решения систем линейных алгебраических уравнений в системе Scilab. Приобретение практических навыков научно – технических и инженерных расчетов с использованием пакета математического моделирования Scilab.

6.1 Теоретическая часть.

Большая часть вычислительных задач сводится к решению систем линейных уравнений вида:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + a_{n2} + \cdots + a_{nn}x_n = b_n \end{cases}, \qquad 6.1$$

где $a_{i,j}$ –коэффициенты уравнений, x_i – неизвестные, b_i - свободные члены уравнений, i, j = 1,2,….n.

Для записи системы уравнений в матричном виде представим коэффициенты при неизвестных в виде прямоугольной матрицы, а неизвестные x_i и свободные члены –в виде вектора-столбца:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \cdots a_{1n} \\ a_{21} & a_{22} & a_{23} \cdots a_{2n} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1} & a_{n2} & a_{n3} \cdots a_{nn} \end{bmatrix}; \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}; \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{bmatrix}. \qquad 6.2$$

Тогда в матричном виде система уравнений будет иметь вид:

$$A \cdot X = B. \qquad 6.3$$

Эта система линейных уравнений имеет единственное решение, если входящие в нее уравнения линейно независимы. Необходимым и достаточным условием этого является условие неравенства нулю определителя системы. Если определитель равен нулю, то система либо не имеет решений, либо имеет бесконечное множество решений.

Алгоритмы решения задач такого типа подразделяются на прямые, когда решение находится за конечное число арифметических операций, и итерационные, когда решение находится только с заданной точностью, а

точное решение теоретически можно получить за бесконечное число итераций.

Один из известных прямых методов является метод Гаусса. Модифицированный метод Гаусса носит название Гаусса-Жордана, предложенный Жорданом. В случае применения итерационных алгоритмов используются методы Якоби, Гаусса-Зейделя. Для решения систем нелинейных уравнений используют методы простых итераций, метод Ньютона и др.

В Scilab имеется достаточной большой набор различных встроенных функций, позволяющих находить решение систем уравнений: **rref(), linsolve(), inv(A), fsolve().**

Рассмотрим решение системы линейных уравнений 6.1 с использованием функции **linsolve().**

Вызов данной функции (обращение к функции) осуществляется командой

linsolve(A, b),

где A – матрица коэффициентов уравнений (5.1), а b –вектор-столбец свободных членов уравнений, значения которых берутся с обратным знаком

Пример 6.1

Решить систему уравнений:

$$\begin{cases} 3x_1 + 8x_2 - 5x_3 + 12x_4 = 18 \\ 5x_1 - 3x_2 + 7x_3 + 11x_4 = -32 \\ 6x_1 - 11x_2 + 2x_3 - 7x_4 = -41 \\ 13x_1 + 4x_2 + 2x_3 + 3x_4 = 20 \end{cases}.$$

Здесь матрица коэффициентов A и вектор-столбец b будут иметь вид:

$$A = \begin{bmatrix} 3 & 8 & -5 & 12 \\ 5 & -3 & 7 & 11 \\ 6 & -11 & 2 & -7 \\ 13 & 4 & 2 & 3 \end{bmatrix}; \qquad b = \begin{bmatrix} 18 \\ -32 \\ -41 \\ 20 \end{bmatrix}.$$

В командном окне решение системы будет выглядеть, например, в таком виде:

```
-->A=[3 8 -5 12;5 -3 7 11; 6 -11 2 -7;13 4 2 3]; //Ввод матрицы A
 -->b=[18;-32;-41;20]; //Ввод вектора-столбца b
 -->X=linsolve(A,-b) //Решение системы уравнений
 X  =
    0.3274133
    5.1647982
```

0.3567973
- 1.8763866 Т.е. получили вектор решений:

x_1= 0.3274133; x_2= 5.1647982 ; x_3=0.3567973 ; x_4=- 1.8763866.

Если необходимо сделать проверку, то для этого просто выполните команду A*X и получите значения свободных членов уравнений (правой части уравнений). Они должны совпасть.

Для нашего примера имеем:

```
-->A*X
 ans  =
   18.
 - 32.
 - 41.
   20.
```

Пример 6.2

Решить заданную систему из примера 5.1 методом Крамера.

В этом случае необходимо определить значение главного определителя матрицы A - ΔA, и значения определителей вспомогательных матриц Δx_1, Δx_2, Δx_3, Δx_4, которые получаются из матрицы A заменой столбца, состоящих из коэффициентов, стоящих при x_i. И далее получаем значения $x_i=\Delta x_i/\Delta A$.

```
function [x]=KRAMER(A, b)
A1=A;A1(:,1)=b; //Первая вспомогательная матрица detx1
A2=A;A2(:,2)=b; //Вторая вспомогательная матрица detx2
A3=A;A3(:,3)=b; //Третья вспомогательная матрица  detx3
A4=A;A4(:,4)=b; //Четвертая вспомогательная матрица detx4
D=det(A); //Главный определитель
//Определители вспомогательных матриц:
d(1)=det(A1); d(2)=det(A2); d(3)=det(A3); d(4)=det(A4);
x=d/D //Вектор решений
endfunction
-->exec('D:\WorkScilab\KRAMERAMETOD.sce', -1)
-->X=KRAMER(A,b)
 X  =
   0.3274133
   5.1647982
   0.3567973
 - 1.8763866
```

Получили, конечно, такое же решение.

Широко используется при решении СЛАУ матричный метод решения, основанный на умножении левой и правой частей уравнения

$A \cdot X = B$ на обратную матрицу A^{-1} слева, т.е. $(A^{-1}A)X = A^{-1}B$. Т.к. $(A^{-1}A)=E$ – (единичная матрица), то $(A^{-1}A)X=EX=X$.
Отсюда и получается, что $X=A^{-1}B$.

В Scilab обратная матрица для заданной матрицы A находится командой **inv**(A), т.е. A^{-1} = inv(A).

Тогда решение примера 5.1 в Scilab матричным методом будет иметь вид:

```
-->X=inv(A)*b
 X  =
    0.3274133
    5.1647982
    0.3567973
  - 1.8763866
```

В Scilab СЛАУ может быть также решена с использованием функции **rref**(Ab), которая приводит матрицу Ab к треугольной форме с использованием при этом метод исключения Гаусса. Последний столбец полученной матрицы решения будет содержать значения искомых решений.

Для нашего примера 5.1. решение системы с использованием функции rrev() будет иметь вид:

```
-->Ab=[A b] //получаем матрицу Ab добавлением к матрице A вектора-столбца b
 Ab  =
    3.    8.   - 5.    12.    18.
    5.   - 3.    7.    11.  - 32.
    6.   - 11.   2.   - 7.   - 41.
    13.   4.    2.    3.     20.

-->X=rref(AA) //-решение системы. Последний столбец - значения неизвестных, т.е. результат решения системы.
 X  =
    1.   0.   0.   0.   0.3274133
    0.   1.   0.   0.   5.1647982
    0.   0.   1.   0.   0.3567973
    0.   0.   0.   1.  - 1.8763866
```

Как видно, решение системы линейных уравнений разными методами дает идентичный результат.

6.2 Лабораторное задание

6.2.1 Сгенерируйте с использованием команды **rand()** целочисленную квадратную матрицу A(6,6), которая будет матрицей коэффициентов при неизвестных X=[x_1, x_2, x_3, x_4, x_5, x_6], а также

вектор-столбец b(6), представляющий правую часть уравнения AX=b;

6.2.2 Решите сформированную систему алгебраических линейных уравнений с использованием функции **linsolve(A,-b)**. Выполните проверку правильности решения системы;

6.2.3 Решите это же уравнение матричным методом: $X=A^{-1}\cdot b$. Выполните проверку. Сравните результаты решения с предыдущим методом;

6.2.4 Решите систему уравнений методом Крамера, написав соответствующий скрипт. Формирование матриц коэффициентов уравнения следует формировать в командном окне с последующей их передачей в скрипт для решения;

6.2.5 Решите систему уравнений с использованием функции **rref**(Ab). Сравните результаты решения с результатами решения предыдущими методами.

6.2.6 Оформите отчет о выполненной работе.

Лабораторная работа № 7

Решение нелинейных и трансцендентных уравнений в Scilab

Цель работы: изучение и освоение методов решения нелинейных и трансцендентных уравнений в системе Scilab. Приобретение практических навыков научно – технических и инженерных расчетов с использованием пакета математического моделирования Scilab.

7.1 Теоретическая часть.

7.1.1 Алгебраические уравнения

Любое уравнение P(*x*)=0, где P(*x*) - многочлен отличный от нулевого, называется алгебраическим уравнением или полиномом. Всякое алгебраическое уравнение относительно *x* можно записать в виде

$$a_0x^n + a_1x^{n-1} + a_2x^{n-1} + \cdots + a_{n-1}x^1 + a_n = 0,$$

где $a_0 \neq 0$, n≥1 и a_i- коэффициенты алгебраического уравнения n-ой степени. Например, линейное уравнение - это алгебраическое уравнение первой степени, квадратное – второй, кубическое – третьей, и так далее.

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.**

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. По умолчанию fl=r.

Следующий пример отражает создание полинома p, имеющего в качестве корня тройку и полинома f с коэффициентом три

```
-->p=poly(3,'x','r');
 -->f=poly(3,'x','c');
 -->p
 p  =
   - 3 + x
 -->f
```

```
 f  =
    3
```

Примеры создания более сложных полиномов:

```
-->//Полином с корнями 1, 0 и 2
 --> poly([1 0 2],'x')
 ans  =
       2x -3x² +x³
```

Т.е. получен полином $2x - 3x^2 + x^3$, корнями которого являются указанные числа [1 0 2].

Полином с коэффициентами 1, 0 и 2:

```
--> poly([1 0 2],'x','c')
 ans  =
  1 +2x²
```

Рассмотрим примеры символьных операций с полиномами:

```
--> p1=poly([-1 2],'x','c')
 p1  =
  -1 +2x
--> p2=poly([3 -7 2],'x','c')
 p2  =
  3 -7x +2x²
--> p1+p2 //Сложение
 ans  =
  2 -5x +2x²
--> p1-p2 //Вычитание
 ans  =
  -4 +9x -2x²
--> p1*p2 //Умножение
 ans  =
  -3 +13x -16x² +4x³
--> p1/p2 //Деление
 ans  =
    1
   -----
   -3 +x
--> p2^2 //Возведение в степень
 ans  =
```

9 -42x +61x² -28x³ +4x⁴

Функция **roots**(p) предназначена для решения алгебраического уравнения. Здесь p - это полном, созданный функцией **poly** и представляющий собой левую часть уравнения P(x)=0.

Пример:

Найти корни полинома $2x^2 - 8x^3 + 8x^2 - 1 = 0$.

Для решения этой задачи необходимо задать полином p. Делается это при помощи функции poly, предварительно определив вектор коэффициентов V. Следует заметить, что в уравнении отсутствует переменная x в первой степени, а это означает, что соответствующий коэффициент равен нулю:

```
--> V=[-1 0 8 -8 2];
--> p=poly(V,'x','c')
 p  =
  -1 +8x² -8x³ +2x⁴
```

Теперь найдем корни полинома^

```
--> X=roots(p)
 X  =
   2.306563  + 0.i
   1.5411961 + 0.i
  -0.306563  + 0.i
   0.4588039 + 0.i
```

Графическое решение задачи, показанное на рис.7.1, позволяет убедится, что корни найдены верно, т.к. на графике видно, что функция пересекает ось X в четырех точках, значения которых приблизительно равны: 2.3, 1.5, -0.3, 0.45. А точное значение корней дает решение соответствующего уравнения.

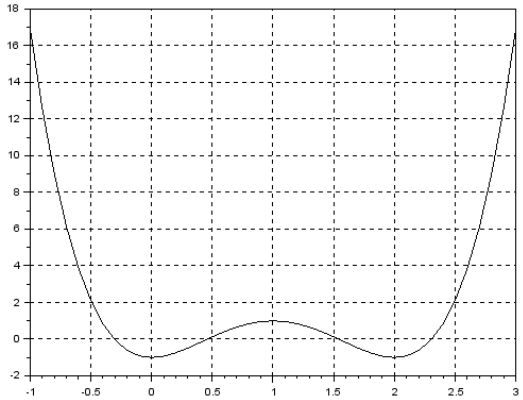


Рис. 7.1

Пример:

Найти корни полинома $x^3 + 0.4x^2 + 0.6x - 1 = 0$.

Решение этой задачи аналогично решению предыдущей, только разница заключается в способе вызова необходимых для этого функций:

```
--> roots(poly([-1 0.6 0.4 1],'x','c'))
 ans  =
  -0.5576818 + 1.0425361i
  -0.5576818 - 1.0425361i
   0.7153636 + 0.i
```

Проиллюстрируем решение графиком рис.6.2:

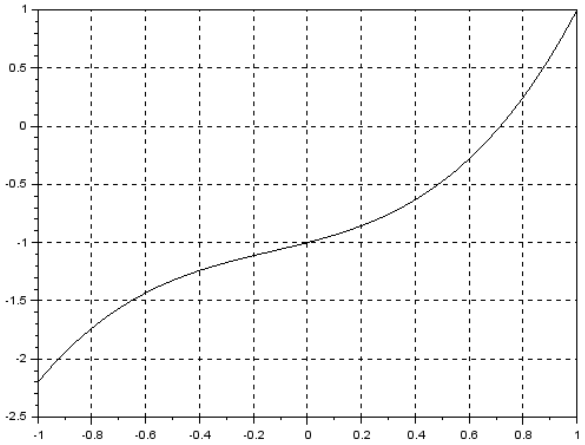


Рис.7.2

Пример:

Найти решение уравнения y(x)=0, если $y(x) = x^4 - 18x^2 + 0.6$

Решение этой задачи представлено ниже и отличается от предыдущих лишь способом определения полинома. Графическое решение представлено на рис 7.3.

```
--> x=poly(0,'x');
--> y=x^4-18*x^2+.6;
--> roots(y)
 ans  =
  -4.2387032 + 0.i
   4.2387032 + 0.i
  -0.1827438 + 0.i
   0.1827438 + 0.i
```

Конечно, если это уравнение можно решить и через roots(p):

```
-->roots(poly([0.6 0 -18 0 1],'x','c'))
```

```
 ans  =
  - 4.2387032
    4.2387032
  - 0.1827438
    0.1827438
```

Как видно – получили тот же результат.

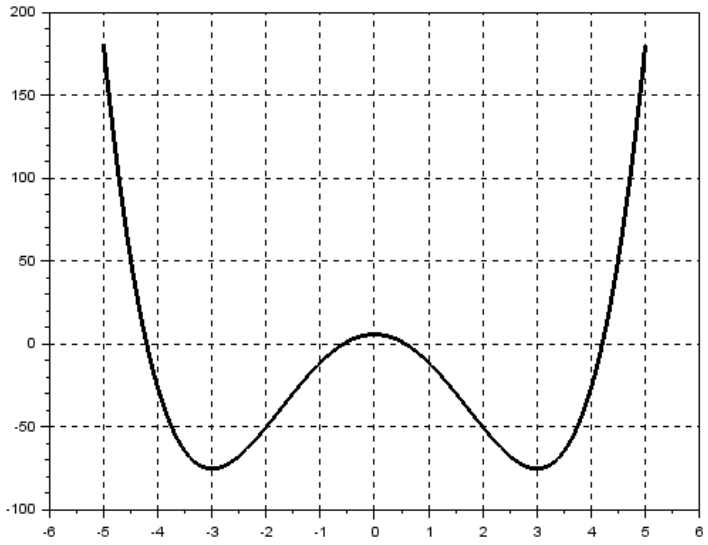


Рис. 7.3

7.1.2 Решение трансцендентных уравнений

Уравнение f(x)=0 называется трансцендентным, если неизвестное x является аргументом трансцендентной функции (показательной, логарифмической, тригонометрической и т.п.).

Например: $a^x - bx^2 + c = 0$; $xsin(x) - \log_a x = 0$ и т.п.

В общем случае аналитическое решение уравнения f(x)=0 можно найти только для узкого класса функций, поэтому трансцендентные уравнения решаются преимущественно численными методами. Обычно этот процесс заключается в выборе интервала, в который попадает по крайней мере один корень уравнения, а затем производится численное приближение к этому корню. Для определения интервала можно использовать графический способ. Далее проводят уточнение отделенных корней, или иначе говоря, находят корни с заданной точностью.

Для решения трансцендентных уравнений в Scilab определена всего одна функция

fsolve(x0,f),

где x0, начальное приближение, f – функция, описывающая левую часть уравнения y(x)=0.

Если возможности этой функции по какой-либо причине не устраивают, то обычно пишут собственную функцию, реализующую некоторый численный метод.

Рассмотрим применение этой функции на примерах.

Пример.

Найти решение уравнения $\sqrt[3]{(x-1)^2} - \sqrt[3]{x^2}=0$.

Сначала определим интервал изоляции корня заданного уравнения. Для этого воспользуемся графическим методом отделения корней. Если выражение стоящее в правой части уравнения представить в виде разности двух функций f(x) и g(x)=0, то абсцисса точки пересечения линий f(x) и g(x) и есть корень данного уравнения. В нашем случае f (x) = $\sqrt[3]{(x-1)^2}$, g(x) = $\sqrt[3]{x^2}$. Ниже на рис 6.4 видно что, корень данного уравнения лежит в интервале [0;1].

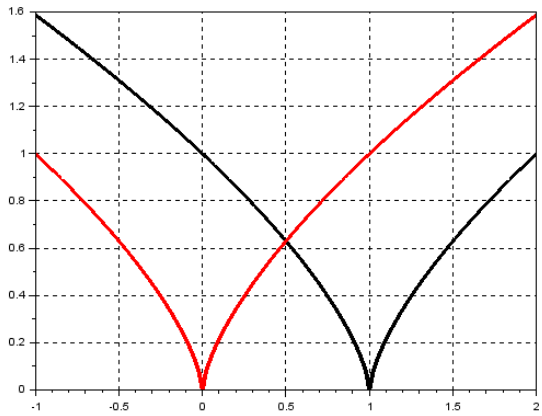


Рис. 7.4

Выберем ноль в качестве начального приближения, далее зададим функцию, описывающую уравнение, и решим его:

```
-->deff('[y]=f1(x)','y1=((x-1)^2)^(1/3),y2=(x^2)^(1/3),y=y1-y2')
 -->fsolve(0,f1)
 ans  =
    0.5
```

Конечно, это уравнение легко решается аналитически, но это – пример.

Пример: Найти корни уравнения $\frac{e^x}{5} - 2(x-1)^2 = 0$.

Если построить график функции $Y(x) = \frac{e^{\mathrm{x}}}{5} - 2(\mathrm{x}-1)^2$ (рис.6.5), то видно, что он трижды пересекает ось абсцисс, то есть уравнение имеет три корня.

Последовательно вызывая функцию **fsolve** с различными начальными приближениями получим все решения заданного уравнения:

```
-->deff('[y]=f(x)','y=exp(x)/5-2*(x-1)^2');
 -->fsolve([0;2;5],f)
ans  =
    0.5778406
    1.7638701
    5.1476865
```

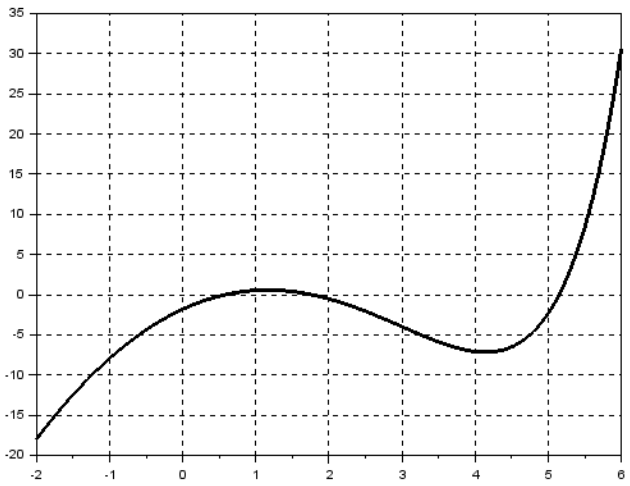


Рис. 7.5.

Пример. Найти корни уравнения:

$$\frac{e^{\mathrm{x}}}{10} - 2(\mathrm{x}-2)^2 = -\sqrt[3]{\mathrm{x}^2}.$$

Следует еще раз отметить, что для решения трансцендентных уравнений необходимо правильно выбирать приближение, для чего требуется представлять левую часть уравнения как функцию и строить график. Если начальное приближение очень далеко от корня, то нет гарантии, что функция **fsolve()** найдет решение.

В этом уравнении, взяв приближение x0=6, получаем:

```
function y=f(x)
y=exp(x)/10-2*(x-1).^2+x.^(2/3);
endfunction
--->fsolve(6,f)
```

```
 ans  =
    6.2477723
```

Если взять приближение x0=4, то получим

```
-->fsolve(5,f)
        !--error 98
```

Функция аргумента scilab вернула некорректную переменную.

Для решения системы нелинейных уравнений в Scilab так же применяют функцию fsolve(x0,f).

Пример: Решить систему уравнений: $\begin{cases} x^2 + y^2 = 1; \\ x^3 - y = 0. \end{cases}$

Графическое решение системы, приведенное на рис. 6.6, показывает, что она имеет две пары корней. Окружность и гипербола пересекаются в точках [0.8;0.6] и [-0.8;-0.6]. Эти значения приблизительны Для того чтобы уточнить их применим, функцию fsolve, предварительно определив систему с помощью файл функции:

```
function [y]=fun(x)
y(1)=x(1)^2+x(2)^2-1;
y(2)=x(1)^3-x(2);
endfunction
-->exec('D:\WorkScilab\sistema.sce', -1)
--> fsolve([0.5;0.5],fun)
 ans  =
   0.8260314
   0.5636242
--> fsolve([-0.5;-0.5],fun)
 ans  =
  -0.8260314
  -0.5636242
```

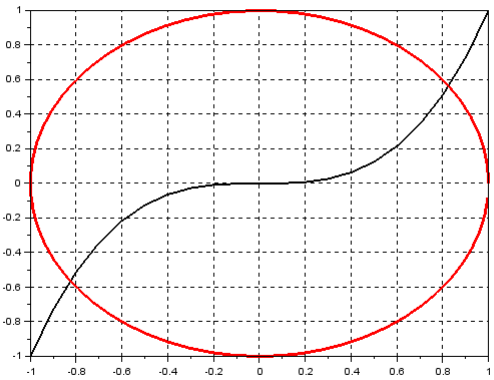


Рис. 7.6

Рассмотрим еще один пример решения системы нелинейных уравнений.

Решить систему уравнений: $\begin{cases} x_1^2 + x_2^2 + x_3^2 = 1 \\ 2x_1^2 + x_2^2 - 4x_3 = 0 \\ 3x_1^2 - 4x_2 + x_3^2 = 0 \end{cases}$.

```
function [y]=SUR(x)
y(1)=x(1)^2+x(2)^2+x(3)^2-1
y(2)=2*x(1)^2+x(2)^2-4*x(3)
y(3)=3*x(1)^2-4*x(2)+x(3)^2
endfunction
--->exec('D:\WorkScilab\sis3ur.sce', -1)
 ---> fsolve([0.5;0.5;0.5],SUR) //решение системы уравнений.
 ans  =
  0.7851969
  0.4966114
  0.3699228
```

7.2 Лабораторное задание

7.2.1 Изучите использование функции **poly**() для построения полиномов заданной степени;

7.2.2 Задайте произвольный вектор длины 6 с целочисленными значениями и при помощи функции **poly**() постройте полином, коэффициентами которого являются числа заданного вектора;

7.2.3 С использованием функции **poly**() постройте полином, корнями которого являются числа заданного вектора;

7.2.4 С использованием функции **poly**() постройте полиномы P1 и P2, корнями которого являются числа: [2, 3,-1, 1, 3] и [2, 3, -1]. Выполните операции над полученными полиномами: P1+P2; P1-P2; P2-P1; $P1 \cdot P2; P2 \cdot P1; P1/P2$. Проанализируйте полученные результаты выполненных операций;

7.2.5 В соответствии с заданным вариантом решите уравнение:

$$ax^5 + bx^4 + cx^3 + dx^2 + ex + f = 0;$$

7.2.6 Постройте график заданного полинома и оцените правильность найденного решения;

7.2.7 Постройте вектор длины 4, элементами которого являются комплексно-сопряженные числа, например, VC=[*a+bi, a-bi, c+di, c-di*]. Значения *a,b,c,d* выберите произвольно. Постройте

соответствующий ему полином, корнями которого являются элементы вектора VC;

7.2.8 Решите уравнение, заданное полученным в п.7.2.7 полиномом. проанализируйте полученный результат;

7.2.9 Для заданной функций в таблице 7.2 графическим способом оцените начальное значение корня уравнения и вычислите его значение с использованием функции **fsolve**():

$$a^x + bx = c;$$

7.2.10 С использованием функции **fsolve**() решите уравнение:

$$\sin(x)\,(e^{0.2x} - 2) - \cos(x)\ln x = 0.$$

Начальные приближения корней задайте вектором, оценив их предварительно графическим способом;

7.2.11 Напишите скрипт для решения трансцендентных уравнений методом простой итерации уравнения $e^{-x} - x^2 = 1$ с различной точностью;

7.2.12 Решите уравнение п.7.2.11 с помощью функции **fsolve**(). Проанализируйте полученные результаты.

Таблица 7.1.

Значения коэффициентов для уравнения (п.7.2.5)

$$ax^5 + bx^4 + cx^3 + dx^2 + ex + f = 0$$

№ варианта	a	b	c	d	e	f
1	3.	5.	3.	9.	9.	- 8.
2	0.	3.	- 1.	3.	- 7.	8.
3	5.	4.	1.	6.	- 6.	1.
4	6.	4.	- 9.	5.	1.	- 8.
5	9.	- 1.	4.	3.	- 5.	9.
6	1.	5.	1.	- 5.	7.	- 8.
7	9.	- 3.	8.	8.	7.	- 8.
8	6.	- 1.	4.	- 8.	0.	- 8.
9	9.	- 1.	1.	- 3.	- 2.	6.
10	0.	1.	9.	- 9.	- 9.	- 8.
11	1.	- 4.	- 1.	- 7.	- 5.	- 5.
12	- 6.	0.	- 4.	3.	0.	- 1.
13	0.	- 2.	- 8.	- 7.	- 4.	2.
14	- 8.	- 9.	7.	7.	9.	- 7.

15	1. 6. 4. 0. 1. - 6.
16	- 1. 0. 9. 0. 8. - 3.
17	3. 7. 6. - 2. 9. 0.
18	- 8. - 3. 5. 1. 5. - 8.
19	0. 2. 8. 7. - 8. 7.
20	4. - 4. - 4. 5. 5. 8.

Таблица 7.2.

Значения коэффициентов для уравнения (п.7.2.9)

$$F(x) = a^x + bx = c.$$

№ варианта	a	b	c
1	π	6	19
2	$\sqrt{\pi}$	$\ln \pi$	e^3
3	e	$\ln \pi$	5
4	6	4	3
5	4	9	12
6	3	1	-4
7	8	9	17
8	9	15	$\log_2 36$
9	12	3	π/e
10	10	e	e^π
11	4	3	17
12	e^2	π^2	π/e
13	π/e	$\ln \pi$	e^π
14	2π	$lg\pi^4$	12
15	π^e	$17e^{-0.5}$	-25
16	$\sqrt{\pi^e}$	e^π	$3\pi e^2$
17	8.1	$10\pi/e$	2π
18	5	1.5^π	3
19	$\ln 63$	-14π	2
20	e^2	$\sqrt{\pi^e}$	e/π

Лабораторная работа № 8
Аппроксимация функций в Scilab

Цель работы: изучение и освоение методов аппроксимации функций с использованием пакета Scilab. Приобретение практических навыков применения метода наименьших квадратов аппроксимации функций с использованием средств компьютерной математики.

8.1. Теоретическая часть

Как правило, при обработке экспериментальных данных требуется определить функциональную зависимость полученных значений или значения каких-либо коэффициентов, определяющих поведение конкретной функциональной зависимости. Очень часто в этих случаях используется метод наименьших квадратов (МНК).

Идея метода наименьших квадратов заключается в том, чтобы найти аппроксимирующую функцию $Y = f(x,\ a_0, a_1, \dots a_к)$, чтобы сумма квадратов отклонений измеренных значений y_i от расчетных Y_i была наименьшей:

$$S = \sum_{i=1}^{n} \left(y_i - f\left(x_{i,}\ a_0, a_1, a_2, \cdots, a_k\ \right)^2 \right) \to min. \qquad 8.1$$

S рассматривается здесь как функция параметров $a_0, a_1, a_2, \cdots, a_k$ так как величины $x_{i,}\ y_i$ известны из экспериментальных данных.

Вид функции $f(x_{i,}\ a_0, a_1, a_2, \cdots, a_k)$ может быть известен из теоретических соображений или определяться характером расположения экспериментальных точек y_i на плоскости x0y.

Задача сводится к определению коэффициентов a_i из приведенного условия.

Для реализации этой задачи в Scilab предусмотрена функция

[a,S]=**datafit**(F,z,a0),

где F - функция параметры, которой необходимо подобрать; z - матрица исходных даных; c - вектор начальных приближений; a - вектор коэффициентов; S - сумма квадратов отклонений измеренных значений от расчетных.

Пусть, например, получены экспериментальные данные, которые должны описываться линейной функцией $y(x) = kx + b$. Необходимо определить такие значения коэффициентов k и b, которые бы наилучшим образом соответствовали линейной зависимости экспериментальных данных.

x	0.9	1.9	3.1	4.2	4.9	6.3	7.1	7.8	9.2	9.6
y	1.1	0.9	2.	2.1	3.	4.2	4.8	5.1	5.5	6.3

Построив график, видим, что линейная зависимость вполне очевидна.

Теперь найдем значения *k* и *b* с использованием функции **datafit**:

```
//Апроксимацция y=kx+b
function [zr]=G(c, z)
zr=z(2)-c(1)-c(2)*z(1);
endfunction
//Исходные данные
x=[0.9 1.9 3.1 4.2 4.9 6.3 7.1 7.8 9.2 9.6];
y=[1.1 0.9 2 2.1 3 4.2 4.8 5.1 5.5 6.3];
//Построение графика экспериментальных данных
plot2d(x,y,-4);
//Вектор начальных приближений
c=[0;0];
//Формирование матрицы исходных данных
z=[x;y];
//Решение задачи
[c,err]=datafit(G,z,c);
disp(c);
// Построение графика подобранной функции
t=0:0.05:10;
Ptc=c(1)+c(2)*t;
xgrid();
plot(t,Ptc,'r','thickness',3);

-->exec('D:\DISK ALEXEEV\WORKScilab\aprtest.sce', -1)

    0.0118637
    0.6342066
```

В командном окне выведены значения коэффициентов:

c(1) = *b*=0.0118637

c(2)=*k*= 0.6342066.

Т.е. экспериментальные значения изменяются по линейному закону

y=0.6342066***x***+0.0118637.

Рис. 8.1 наглядно показывает результат аппроксимации исходных данных по линейному закону, значения коэффициентов которого и были найдены.

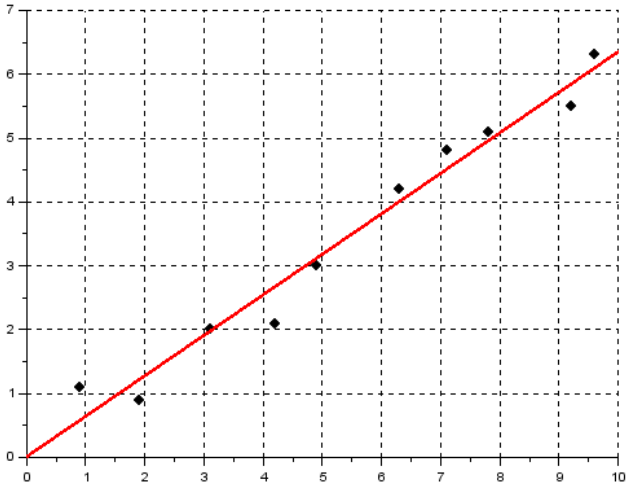


Рис. 8.1

Рассмотрим другой пример:

Имеем исходные данные, связанные квадратичной зависимостью $y=ax^2+bx+c$:

x=[1.32 1.40 1.50 1.62 1.70 1.80 1.90 2.00 2.11 2.20 2.32 2.40 2.51];
y=[3.30 3.50 3.85 4.25 4.50 4.85 5.40 6.00 6.60 7.30 9.20 10.20 13.50];

В этом случае в программе нужно изменить функцию F

Это пример квадратичной зависимости

```
// Апроксимацция y=ax^2 + bx+c
    function [zr]=G(c, z)
    zr=z(2)-c(1)-c(2)*z(1)-c(3)*z(1)^2;
    endfunction
    //Исходные данные
    x=[1.32 1.40 1.50 1.62 1.70 1.80 1.90 2.00 2.11 2.20 2.32 2.40 2.51];
    y=[3.30 3.50 3.85 4.25 4.50 4.85 5.40 6.00 6.60 7.30 9.20 10.20 13.50];
    //Построение графика экспериментальных данных
    plot2d(x,y,-4);
    //Вектор начальных приближений
    c=[0;0;0];
    //Формирование матрицы исходных данных
    z=[x;y];
    //Решение задачи
    [c,err]=datafit(G,z,c);
    disp(c);
    // Построение графика подобранной функции
    t=1.3:0.1:2.5;
    Ptc=c(1)+c(2)*t+c(3)*t.^2;
```

```
xgrid();
plot(t,Ptc,'r');
```

Получаем результат:

```
--> exec('D:\WorkScilab\gr2708.sce', -1)
  20.709515
 -23.387692
  8.0073823
```

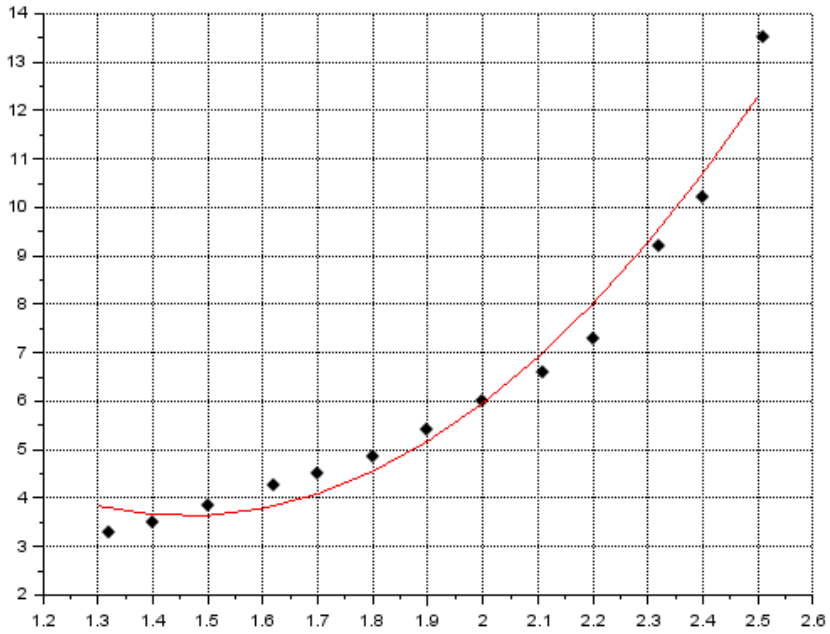


Рис. 8.2

Т.е. получаем, что значения соответствуют функциональной зависимости $y(x) = 8.0073823x^2 - 23.387692x + 20.709515$

Пусть необходимо выполнить аппроксимацию этих же исходных данных по экспоненциальной зависимости $y = ae^{bx} + c$. Тогда в этом случае получаем:

```
// Апроксимацция y=ae^bx+c
    function [zr]=G(c, z)
    zr=z(2)- c(1)*exp(c(2)*z(1));
    endfunction
    //Исходные данные
    x=[1.32 1.40 1.50 1.62 1.70 1.80 1.90 2.00 2.11 2.20 2.32 2.40 2.51];
    y=[3.30 3.50 3.85 4.25 4.50 4.85 5.40 6.00 6.60 7.30 9.20 10.20 13.50];
```

```
//Построение графика экспериментальных данных
clf()
plot2d(x,y,-4);
//Вектор начальных приближений
c=[0;0;0];
//Формирование матрицы исходных данных
z=[x;y];
//Решение задачи
[c,err]=datafit(G,z,c);
disp(c);
// Построение графика подобранной функции
t=1.3:0.1:2.5;
Ptc=c(1)*exp(c(2)*t);
xgrid();
plot(t,Ptc,'r');
```

Получаем результат:

```
--> exec('D:\WorkScilab\gr2708.sce', -1)
   0.4925781
   1.2778276
   0.
```

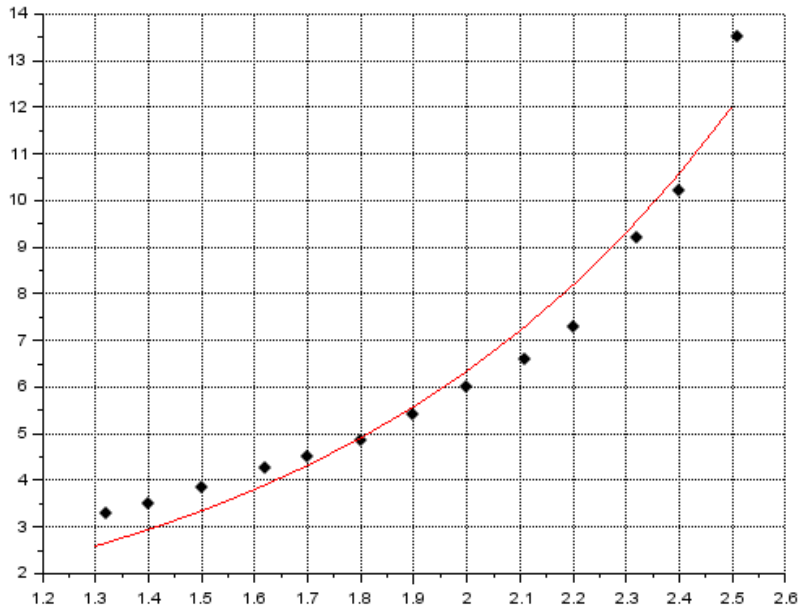


Рис. 8.3

Т.е. аппроксимирующая функция исходных данных по экспоненциальному закону имеет вид: $y(x) = 0.4925781e^{1.2778276x} + 0.$

Очень часто необходимо выполнить аппроксимацию исходных данных по заданной функциональной зависимости, для чего необходимо построить алгоритм нахождения аппроксимирующей функции в соответствии с выражением 8.1.

Рассмотрим пример составление уравнений для аппроксимации линейной зависимости типа $y = ax + b$.

Итак, известно, что принцип МНК заключается в выполнении соотношения 8.1:

$$S = \sum_{i=1}^{n} (u_i - f(z_i, a, b, \dots))^2 \to min.$$

Для линейной функции $y = ax + b$ $u_i = y_i, z_i = x_{i,}$ и тогда данное выражение будет иметь вид:

$$S = \sum_{i=1}^{n} (y_i - (ax_i + b))^2 \to min.$$

Для нахождения коэффициентов a и b необходимо решить систему уравнений:

$$\begin{cases} \frac{dS}{da} = 0 \\ \frac{dS}{db} = 0 \end{cases},$$

откуда получаем систему уравнений с неизвестными a и b

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + nb = \sum_{i=1}^{n} y_i \end{cases}$$

Вычислив суммы по заданным значениям $x_{i,}$ и y_i и подставив их в уравнения находим значения коэффициентов a и b.

Пример:

X_i	2	3	4	5	6	7	8	9	10
Y_i	7.5	9	10.5	14	17.5	20	21.5	25	28

Найденные суммы равны:

$$\sum_{i=1}^{9} x_i^2 = 384; \quad \sum_{i=1}^{9} x_i = 54; \quad \sum_{i=1}^{9} x_i y_i = 1076; \quad \sum_{i=1}^{9} y_i = 153.$$

Теперь, решая систему уравнений

$$\begin{cases} 384a + 54b = 1076 \\ 54a + 9b = 153 \end{cases}.$$

получаем: a=2.6333; b=1.2.

Графики заданных точек и прямая аппроксимации (красная) приведены на рис. 8.4

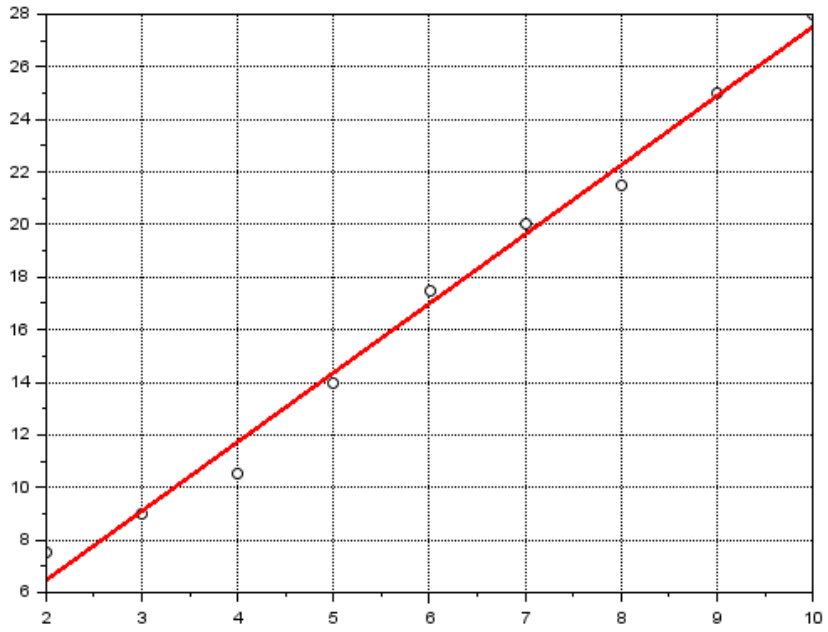


Рис.8.4

Скрипт решения задачи можно записать в виде:

```
//Линейная аппроксимация
function [X]=aproline(x, y);
   n=length(x);// длина массивов
   Sxy=sum(x.*y)// сумма произведений x(i)*y(i)
   Sx2=sum(x^2)//сумма квадратов x(i)
   Sx=sum(x) //  сумма массива x
   Sy=sum(y) // сумма массива y
   M=[Sx2 Sx;Sx n];  //полученная матрица коэффициентов уравнения
   V=[Sxy;Sy] // вектор значений правой части уравнений
   X=linsolve(A,-V) //решение системы
endfunction
clf()
```

```
plot(x,y,'k-o');
YA=X(1)*x+X(2);  //график по исходным данным
plot(x,YA,'r','thickness',4);  // график, построенный по аппроксимирующей зависимости.

//аппрксимация гиперболы
function [Z]=aprogip(x, y);
   n=length(x);// длина массивов
   z=1./y;
   Sxy=sum(x.*z)
   Sx2=sum(x.^2)
   Sx=sum(x)
   Sy=sum(z)
   M=[Sx2 Sx;Sx n];
   V=[Sxy;Sy]
   X=linsolve(M,-V) //решение системы
   A=1./X(1);
   b=X(2)./X(1);
   Z(1)=A
   Z(2)=b
plot(x,y,'k-o');
xg=min(x)-1:0.01:max(x)+1
YA=A./(b+xg);  //аппроксимирующая функция
plot(xg,YA,'r','thickness',4);//график, построенный по аппроксимирующей зависимости.
xgrid()
xtitle('аппроксимация гиперболической зависимости','Значения x','Значения y');
legend('аппроксимируемые данные','аппроксимирующая функция')
endfunction

x=[1 2 3 4 5 6 7 8 9 10];
yE=[1 0.7 0.58 0.5 0.4 0.38 0.33 0.3 0.26 0.25];
aprogip(x,yE)
"A"
  2.9895885
 "b"
  2.1216804
```

8.2 Лабораторное задание.

8.2.1 Изучите и освойте использование функции datafit() для аппроксимации функциональных зависимостей. Выполните приведенные примеры.

8.2.2 Для приведенных табличных значений выполните их аппроксимацию с использованием функции **datafit**().

8.2.3 По линейной зависимости:

X_i	1	2	3	4	5	6	7	8	9	10
Y_i	3.1	4.8	6.9	9.2	11.3	12.9	14.9	7.2	18.2	21.2

8.2.4 По квадратичной зависимости:

X_i	1	2	3	4	5	6	7	8	9	10
Y_i	4.4	14.3	30	51.5	82	114	149	203	252	313

8.2.5 Для предыдущего варианта выполните аппроксимацию по экспоненциальной зависимости вида $y = Ae^{bx}$;

8.2.6 Напишите скрипт для аппроксимации функций п.2.1 и 2.2 по методу наименьших квадратов. Сравните результаты аппроксимации.

8.2.7 Разработайте скрипты для аппроксимации указанных преподавателем функций:

$a)\ y = Ae^{bx};\ \ b)\ y = \frac{1}{ax+b};\quad c)\ y = ax^3 + bx^2 + cx + d;$

$d)\ y = \sqrt{ax+b};\quad e)\ y = \ln(ax+b);\quad f)\ y = e^{ax+b};$ g) $y = a \cdot 2^{bx};$

$h)\ y = ae^{b\sqrt{x}};\quad i)\ y = \sqrt[3]{ax^2+bx+c};\quad j)\ \sqrt{ax^3+bx^2+cx+d};$

$k)\ y = \frac{2}{\sqrt{ax+b}};\quad l)\ y = \frac{3}{ax^2+bx+c};\quad m)\ y = \frac{4}{\sqrt{ax^2+bx+c}};$

Составьте тестовые задачи и проверьте правильность работы соответствующих программ.

8.2.8 Выполните отчет о проделанной работе.