МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учреждение образования
«Витебский государственный технологический университет»

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Лабораторный практикум для студентов
специальности 1-36 01 01 «Технология машиностроения»
дневной формы обучения

Витебск
2022

УДК 004.43/075

Составители:

Т. П. Стасеня, О. Г. Мандрик

Рекомендовано к изданию редакционно-издательским
советом УО «ВГТУ», протокол № 9 от 30.05.2022.

Информационные технологии : лабораторный практикум / сост. Т. П. Стасеня, О. Г. Мандрик. – Витебск : УО «ВГТУ», 2022. – 50 с.

Методические указания составлены в соответствии с типовой программой курса «Информационные технологии». В методических указаниях представлен теоретический и практический материал для решения прикладных задач с использованием информационных технологий. Материал содержит определения используемых терминов и включает вопросы, связанные со следующими темами: «Математическое и программное обеспечение обработки информации»; «Построение математических моделей»; «Пакеты прикладных программ для моделирования и численного решения оптимизационных задач»; «Компьютерное моделирование прикладных задач. Система компьютерной математики Maple»; «Графические возможности Maple».

УДК 004.43/075

СОДЕРЖАНИЕ

1 МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ОБРАБОТКИ ИНФОРМАЦИИ

Погрешности численного решения задач

Причины и классификация погрешностей

В качестве причин погрешностей решения задачи могут служить:

– неточно заданные числовые исходные данные;

– применяемый метод часто не является точным;

– при записи исходных данных, при выполнении арифметических операций и при записи результата производятся округления.

Прибегая к приближенному методу решения задачи, мы фактически решаем другую, аппроксимирующую задачу. Точное решение требует неограниченного или значительно большего числа операций, поэтому вместо точного мы довольствуемся приближенным решением.

Классификация погрешностей согласно названным причинам:

1) неустранимая погрешность;
2) погрешность метода;
3) вычислительная погрешность.

Погрешность и ее характеристики

Определение

Если a – точное значение некоторой величины, а a* – известное приближение к нему $(a^* \approx a)$, то погрешностью для a* называют величину $\Delta = a - a^*$.

Погрешность может быть представлена в двух формах.

1. Абсолютная погрешность для a* есть величина Δ(a*), про которую известно, что $|a - a^*| = |\Delta \le \Delta(a^*)|$.

2. Относительная погрешность – есть некоторая величина $\delta(a)$, про которую известно, что

$$\left|\frac{a - a^*}{a^*}\right| = \left|\frac{\Delta}{a^*}\right| \le \delta(a).$$

Эту величину часто выражают в процентах.

Для обоих форм существуют т.н. предельные погрешности, т. е. наименьшие значения из верхних границ.

Понятие значащих цифр: Это все цифры в записи числа, начиная с первой ненулевой слева.

Пример

0,04207 – четыре значащих цифры; 0,0405000 – шесть значащих цифр.

Правила записи погрешностей: обычно их пишут с одной или двумя значащими цифрами, начиная с первой ненулевой слева.

Пример

Какова относительная погрешность числа 3,14? Здесь будет a1 =3, m=3.

$$\delta(a^*) = \frac{\frac{1}{2}}{3 \cdot 10^2} = \frac{1}{600} = 0{,}00167.$$

Отсюда

Погрешности арифметических операций

Для анализа погрешностей используют формулы погрешностей для непрерывно дифференцируемых функций

$$\Delta(y^*) \approx \sum_{i=1}^{n} \left| f'_{x_i}(x^*) \right| \cdot \Delta(x_i^*);$$

$$\delta(y^*) \approx \sum_{i=1}^{n} \left| x_i^* \left\{ \ln f(x^*) \right\}'_{x_i} \right| \cdot \delta(x_i^*).$$

1 Действие сложения

$$\Delta(y^*) \approx \sum_{i=1}^{n} \left| f'_{x_i}(x^*) \right| \cdot \Delta(x_i^*);$$

$$\delta(y^*) \approx \sum_{i=1}^{n} \left| x_i^* \left\{ \ln f(x^*) \right\}'_{x_i} \right| \cdot \delta(x_i^*).$$

Одно неточное число испортит всю сумму.

2 Действие вычитания

$$\Delta(y^*) \approx \Delta(x_1^*) + \Delta(x_2^*);$$

$$\delta(y^*) \approx \frac{x_1^* \cdot \delta(x_1^*) + x_2^* \cdot \delta(x_2^*)}{x_1^* - x_2^*}.$$

Избегать вычитания близких чисел.

3 Действие умножения

$$\Delta(y^*) \approx \sum_{i=1}^{n} \frac{y^*}{x_i^*} \Delta(x_i^*);$$

$$\delta(y^*) \approx \delta(x_1^*) + \delta(x_2^*) + \ldots + \delta(x_n^*).$$

4 Действие деления

$$\Delta(y^*) \approx \frac{x_2^* \cdot \Delta(x_1^*) + x_1^* \cdot \Delta(x_2^*)}{(x_2^*)^2};$$

$$\delta(y^*) \approx \delta(x_1^*) + \delta(x_2^*).$$

5 Погрешности некоторых элементарных функций

5.1 Степенная функция $y = x^a$

$$\Delta(y^*) = a(x^*)^{a-1} \cdot \Delta(x^*);$$

$$\delta(y^*) \approx |a| \delta(x^*).$$

5.2 Показательная функция $y = a^x \quad (a > 0)$

$$\Delta(y^*) = a \cdot \ln a \cdot \Delta(x^*);$$

$$\delta(y^*) \approx \Delta(x^*) \ln a \, .$$

5.3 Логарифмическая функция $y = \ln x$

$$\Delta(y^*) \approx \frac{1}{x^*} \Delta(x^*);$$

$$\delta(x^*) \approx \frac{1}{x^*} \Delta(x^*).$$

5.4 Тригонометрические функции. Для $\sin x$ *и* $\cos x$ значения Δ не превосходят значения Δ аргумента

$\Delta(\sin x^*) \le \Delta(x * 0;$

$\Delta(\cos x^*) \le \Delta(x^*).$

Для тангенса $\Delta(tgx^*) \ge \Delta(x^*)$, особенно для углов, близких к $\pi/2$ *и* $k\pi$.

Погрешность метода. Вычислительная погрешность.
Понятие об устойчивости метода

Обобщенная запись большинства задач вычислительной математики имеет вид

$$y = A(x). \qquad (1)$$

Здесь x и y принадлежат функциональным пространствам R1 и R2, A – некоторый заданный оператор. Задача состоит в отыскании у по заданному x.

Основным методом является замена пространств R1 и R2 и оператора A некоторыми другими пространствами $\overline{R}_1, \overline{R}_2$ и оператором $\overline{A}$, которые более удобны для вычислительных целей. Иногда заменяют одно из пространств: (R1 или R2) или оператор A. Задача $\overline{y} = \overline{A}(\overline{x})$ в каком-то смысле должна быть близка к точному решению исходной задачи (1).

В результате замены задачи (1) задачей

$$\overline{y} = \overline{A}(\overline{x}) \qquad (2)$$

появляется разность $y - \overline{y}$, которая называется погрешностью метода. Ее варьируют в широких пределах, используя теорию вопроса (способ аппроксимации пространств и функций).

Для решения задачи (2) необходимо указать алгоритм, то есть совокупность правил и расчетных формул, определяющих процесс применения и приводящих к искомому результату. Вот здесь в манипулировании данными и возникает вычислительная погрешность. Практика вычислений требует, чтобы процесс счета был устойчивым. Суть дела в следующем. Часто метод предписывает выполнение ряда шагов вычислений. Обычно ряд шагов тем длиннее, чем выше заданная точность вычислений. Некоторая ожидаемая погрешность одного из этапов оказывает влияние на результаты последующих шагов счета.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ (4 ЗАДАНИЯ)

Все задания выполнить средствами MS Excel, Maple и запрограммировать расчеты в среде Delphi. Результаты оформить в виде отчета, расписав порядок действий.

Действия с приближенными числами

Пользуясь рассмотренным теоретическим материалом, выполнить указанные ниже задания.

Задание 1. Определить число верных знаков в каждом из чисел x_i $(i = 1, 2, \ldots, 5)$, если известны относительные погрешности этих чисел: $\delta(x_1) = 1\%$, $\delta(x_2) = 0{,}5\%$, $\delta(x_3) = 2\%$, $\delta(x_4) = 0{,}1\%$, $\delta(x_5) = 3\%$. Параметр Ω выберем равным 1/2, т. е. абсолютная погрешность не должна превосходить пяти единиц разряда, следующего за этой цифрой.

Таблица 1 – Варианты к заданию 1

№ вариант	Числа x_i				
	x_1	x_2	x_3	x_4	x_5
1	2,310546	-0,280012	12,60405	201,3985	0,008109
2	123,4872	38,67981	961,2434	20,00352	-0,004801
3	32,68,902	-0,004702	26,04100	99,9998	3,214562
4	21,3405	0,0273181	1,204060	102,3452	3,1415426
5	3,140209	2,880012	34,56010	0,00211816	2,7182818
6	0,01326103	400,0026	10000	-18,46230	718,281828
7	1,326124	0,004002	1002,1	3462,32	-1,60082
8	2134,563	3,025020	-1,209401	0,0014856	2000,00
9	0,0203521	82,90526	0,00016092	-12,36457	102,005
10	-1,008409	-28163,0	9,899987	1,00040	82,95103
11	106,2052	1,04352	2048,21	-6,21853	0,0084532
12	0,00312054	-1,204081	0,00114856	201,02	5,342109
13	20,3150	-2,88016	0,10680	0,0014650	1020,0
14	0,096104	1,03254	-64,1666	1029,0	0,0001234
15	10,096104	-1,03254	764,1656	61029,0	-0,0001234
16	-0,18465	1,21412	2001,01	1,116200	312,502
17	0,001819	-12,64512	100,00	211,6273	0,1819162
18	1,14241	-0,02561	-10,0421	10,00	10400
19	-0,008612	12,3401	-0,09699	1,00813	212,212
20	21,3602	0,11112	920000	-1,21211	-0,006215

Пример решения

Пусть $x = 1{,}4142;\ \delta(x) = 0{,}1\% = 0{,}001.$ Число верных знаков числа x определяется m – наибольшим целочисленным решением неравенства

$$\delta(x) =\leq \frac{\Omega}{(a_1 + 1)\cdot 10^{m-1}},$$

где a_1 – первая значащая цифра числа x. Имеем

$$0{,}001 =\leq \frac{1/2}{2\cdot 10^{m-1}} = \frac{1}{4\cdot 10^{m-1}}$$

$$0{,}004\cdot 10^{m-1} \leq 1.$$

Наибольшим целочисленным решением неравенства является $m = 3$. Следовательно, число x имеет по крайней мере три верных цифры.

Задание 2. Найти величины $y_1 + 2 \cdot y_2;\ y_3 - 4 \cdot y_4 + y_1;\ y_5 - y_2 + 3 \cdot y_3 + 2 \cdot y_4$ и указать абсолютную и относительную погрешности результата, предполагая, что все цифры исходных данных верные.

Задание 3. Найти величины $y_1 \cdot y_4;\ y_2 / y_4;\ \dfrac{y_1(y_3 + y_4)y_2}{y_5 y_3}$ и указать абсолютную и относительную погрешности результата, предполагая, что все цифры исходных данных верные.

Задание 4. Вычислить $u = u(a_1, a_2, a_3)$ для заданной функции u. Указать Δ(u) и δ(u), предполагая, что a_1, a_2, a_3 заданы со всеми верными цифрами.

Таблица 2 – Варианты к заданиям 2, 3, 4

№ варианта	Числа y_i					Числа a_i		
	y_1	y_2	y_3	y_4	y_5	a_1	a_2	a_3
1	96,104	1,0325	1029	-6,41667	1,02	0,184	1,017	1,634
2	0,18465	-1,214	20,012	306,4	1,116	1,3041	0,8641	34,68
3	0,8135	84,201	0,0064	9,999	36,72	0,341	1,26	1,343
4	31,67	6,800	1,554	0,0062	1,0031	21,62	0,3564	1,26
5	9,998	3,0482	0,6437	0,00372	10,230	0,087	0,6481	1,542
6	0,00113	9,805	55,55	6,242	0,1234	0,162	1,0423	16,24
7	21,314	8,05	97,81	0,0120	3,1625	2,007	-1,24	0,0846
8	0,2008	86,53	3,6452	10,204	0,0084	0,131	1,214	0,8672
9	5,042	0,1034	25,046	0,0398	3,14156	0,1644	1,241	1,30
10	25,6	0,03408	1,46	305,6	1,10123	2,601	0,481	0,216
11	1207,48	0,00574	3,6062	1,0023	0,146	2,341	15,6	0,3124
12	2,603	1,82	0,04631	1,234	0,21	0,481	0,206	2,341
13	125,48	4,547	-861,24	0,00842	0,311	1,586	1,64	0,08426
14	2,243	1,2087	0,03027	101,2	3,1415	0,367	0,018	2,3
15	0,2718	34,21	-1,2604	0,0013	201,0	0,057	1,183	21,0
16	0,098	3,2401	234,00	100	1,62	0,312	1,3412	0,0982
17	1,981	0,13	124,6	1,3256	0,0038	10,231	14,2	0,0072
18	98,6	1,2148	0,00101	3,25	40	1,14	0,4621	1,123
19	0,00912	1213	3,21	62541	102,0	1,214	0,0482	11,6
20	1,268	0,081	11,2451	1011	8,64	0,912	1,9514	1,007

Аналитическое выражение для функции $u = u(a_1, a_2\ a_3)$ к задаче 4

Варианты	$u = u(a_1, a_2\ a_3)$
1–5	$u = tg(3a_1 + a_2^2 - a_3)$
6–10	$u = \dfrac{1}{3}(\pi \cdot a_1 + a_2) \cdot \sin(3a_3 + 2a_1)$
11–15	$u = \sqrt[3]{a_2 - a_1^2} \cdot \lg(a_3 + 2a_1)$
16–20	$u = \sqrt{a_3 + a_1 a_2^3} + \exp(a_1^2 + 2a_3)$

2 ПОСТРОЕНИЕ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ

2.1 Метод наименьших квадратов

Постановка задачи о приближении функций

Пусть на некотором множестве задана система функций

$\phi_0(x), \phi_1(x), \ldots, \phi_m(x), \ldots,$

которые в дальнейшем будем считать достаточно гладкими (например, непрерывно дифференцируемыми) функциями. Назовем эту систему основной.

Функции вида $Q_m(x) = c_0\phi_0(x), + c_1\phi_1(x) + \ldots + c_m\,\phi_m(x)$, где $c_0, c_1, \ldots, c_m$ – постоянные коэффициенты, называются обобщенными полиномами порядка m. В частности, если основная система состоит из целых неотрицательных степеней переменной x, т. е. $\phi_0(x) = 1,\ \phi_1(x) = x, \ldots, \phi_m(x) = x^m$, то $Q_m(x) = c_0 + c_1x + \ldots + c_m\,x^m$ есть обычный алгебраический полином степени m.

Задача о приближении функции ставится следующим образом: данную функцию f(x) требуется заменить обобщенным полиномом $Q_m(x)$ заданного порядка m так, чтобы отклонение (в известном смысле) функции f(x) от обобщенного полинома $Q_m(x)$ на указанном множестве $X = \{x\}$ было наименьшим. При этом полином $Q_m(x)$ в общем случае называется **аппроксимирующим.**

Если множество X состоит из отдельных точек $x_1, x_2, \ldots, x_n$, то приближение называется **точечным.** Если же X есть отрезок $a \le x \le b$, то приближение называется **интервальным.**

Используя термин отклонения двух функций, который в зависимости от обстоятельств понимается по-разному, мы получаем различные типы задач теории приближений:

- интерполирование,
- среднеквадратичное приближение,
- равномерное приближение и т. п.

Суть метода наименьших квадратов

На практике часто бывает, что заданный порядок приближающего полинома $Q_m(x)$ значительно меньше числа узлов n. В этом случае интерполирование становится невозможным и приходится прибегать к построению аппроксимирующего приближающего полинома для данной функции. Обычно здесь используют точечный способ наименьших квадратов.

Согласно этому методу за меру отклонения полинома

$Q_m(x) = a_0 + a_1x + \ldots + a_m\,x^m$

от данной функции f(x) на множестве точек $x_1, x_2, \ldots, x_n$ принимает величину

$$S_m = \sum_{i=0}^{n} [Q_m(x_i) - f(x_i)]^2 \ ,$$

равную сумме квадратов отклонения полинома $Q_m(x)$ от функции f(x) на заданной системе точек.

Пример. Рассмотрим численный пример для построения алгебраического аппроксимирующего полинома второй степени $y = a_0 + a_1x + a_2x^2$ для таблично заданной функции (рис. 1):

	,78	,56	,34	,12	,81
y	2,50	1,20	1,12	2,25	4,28

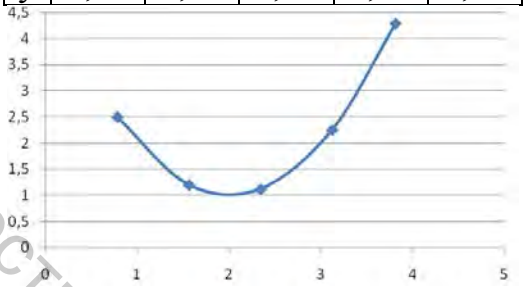


Рисунок 1 – График функции на листе MS Excel

Построение полинома по данным графика.

Решение. Вычисления для $m = 2$, $n = 4$ сведем в таблицу:

x^0	x^1	x^2	x^3	x^4	y	xy	x^2y
1	0,78	0,608	0,475	0,370	2,50	1,950	1,520
1	1,56	2,434	3,796	5,922	1,20	1,872	2,921
1	2,34	5,476	12,813	29,982	1,12	2,621	6,133
1	3,12	9,734	30,371	94,759	2,25	7,020	21,902
1	3,81	14,516	55,306	210,717	4,28	16,307	62,128
5	11,61	32,768	102,761	341,750	11,35	29,770	94,604

Система уравнений для определения коэффициентов a_0, a_1, a_2 имеет вид

$$\begin{cases} 5a_0 + 11{,}61a_1 + 32{,}768a_2 = 11{,}350, \\ 11{,}61a_0 + 32{,}768a_1 + 102{,}761a_2 = 29{,}770, \\ 32{,}768a_0 + 102{,}761a_1 + 341{,}750a_2 = 94{,}604, \end{cases}$$

Решив эту систему, получаем $a_0 = 5{,}045;\ a_1 = -4{,}043;\ a_2 = 1{,}009$.

Следовательно, искомый полином есть $y_1 = 5{,}045 - 4{,}043 \cdot x + 1{,}009 \cdot x^2$.

Пересчитав исходную таблицу по полученной формуле, видим, что вычисленные значения y_1 близки к значениям y, как это видно из таблицы:

x	0,78	1,56	2,34	3,12	3,81
y	2,5	1,2	1,12	2,25	4,28
y_1	2,505	1,193	1,109	2,252	4,288
y_1 - y	0,005	-0,007	-0,011	0,002	0,008

Модель выбрана удачно, а построенные графики накладываются друг на друга.

2.2 Задача построения обобщенного алгебраического полинома

В приведенном выше примере использован алгебраический полином второго порядка. Рассмотрим программу построения алгебраического полинома в общем виде в консольном режиме Delphi.

Такой полином имеет вид $Q_m(x) = a_0 + a_1 x + \ldots + a_{m-1} x^{m-1} + a_m x^m$,

где $a_0, a_1, \ldots ; a_m$ – коэффициенты полинома; m – порядок полинома.

Для построения модели полинома нужно задавать таблицу значений x и y исходной функции $y_i = f(x_i)$, *где* $i = 0, 1, \ldots, n$ (всего n+1 значений) и порядок полинома m.

Чтобы понятнее был алгоритм задачи, построим в общем виде таблицу, по которой будем вычислять коэффициенты s_i и t_i для системы линейных уравнений с неизвестными коэффициентами $a_0, a_1, \ldots , a_m$. Обобщенная таблица будет иметь вид:

x^0	x^1	x^2		x^m	x^{m+1}		x^{2m}	x^0y	x^1y	x^2y		x^my
1	x_0	x_0^2	...	x_0^m	x_0^{m+1}	...	x_0^{2m}	y_0	x_0y_0	$x_0^2y_0$	...	x_0^my
1	x_1	x_1^2	...	x_1^m	x_1^{m+1}	...	x_1^{2m}	y_1	x_1y_1	x_1^2y1	...	x_1^my
1	x_2	x_2^2	...	x_2^m	x_2^{m+1}	...	x_2^{2m}	y_2	x_2y_2	$x_2^2y_2$	...	x_2^my
...	...	...	...	...	...	...	...	...	...	...	...	..
1	x_{n-1}	x_{n-1}^2	...	x_{n-1}^m	x_{n-1}^{m+1}	...	x_{n-1}^{2m}	y_{n-1}	$x_{n-1}y_{n-1}$	$x_{n-1}^2y_{n-1}$	...	x_{n-1}^my
1	x_n	x_n^2	...	x_n^m	x_n^{m+1}	...	x_n^{2m}	y_n	x_ny_n	$x_n^2y_n$	...	x_n^my
s_0	s_1	s_2	...	s_m	s_{m+1}	...	s_{2m}	t_0	t_1	t_2	...	t_m

В таблице должны быть m+1 значений сумм t и 2m+1 значений сумм s и в общем случае должно быть: $n > m$.

Согласно обобщенной таблице в консольном режиме Delphi разработана программа, по которой реализуется построение алгебраического полинома по заданным таблице функции и порядку полинома m.

Программа в консольном режиме Delphi

```
program MNK_konsol;
{$APPTYPE CONSOLE}
{ Построение аппроксимирующего полинома
  методом наименьших квадратов }
uses
  SysUtils, Math;
Type Dannye=array[0..30] of extended;
   TablX=array[0..11] of extended;
   TablY=array[0..5] of extended;
   Matrica=array[0..5,0..6] of extended;
```

```
Var  i,i1,j,k,L,m,n,n1:integer;
   A1,D,G,X1:extended;
   S:TablX;T:TablY;
   X,Y:Dannye; A,C:Matrica;
BEGIN
   Write('Zadajte kolichestvo tochek isxodnoj tablicy: ');
   Readln(n1); n:=n1-1;
   Write('Vvedite porjadok approksimirujychego polinoma: ');
   Readln(m);  k:=2*m;
   Writeln('Vvedite tablicu ishodnych dannych');
   Writeln('Posle kajdoj pary chisel najimaite <Enter>');
   for i:=0 to n do    begin
    Write(' ':16,'X(',i:2,'), Y(',i:2,'): ');
    Readln(X[i],Y[i]) end;
   { Фрмирование итогов расчетной таблицы   }
   { Столбцы X^0 ... X^2m  }
   S[0]:=n1;    // столбец 0
   for L:=1 to k do begin S[L]:=0; // столбцы 1 . . . 2m
     for i:=0 to n do begin X1:=X[i];
       S[L]:=S[L]+IntPower(X1,L);
                    end;  end;
   { Столбцы YX^0 ... YX^m }
   for L:= 0 to m do begin T[L]:=0;
     for i:=0 to n do
       T[L]:=T[L]+Y[i]*IntPower(X[i],L);   end;
   { Формирование матрицы системы линейных уравнений
         для нахождения коэффициентов полинома  }
   for i:=0 to m do   begin
     for j:= 0 to m do A[i,j]:=S[i+j];
       A[i,m+1]:=T[i] end;
   { Обнуление матрицы C: }
   n:=m; m:=m+1;
   for i:=0 to n do
     for j:=0 to m do C[i,j]:=0;
   { Решение системы линейных уравнений, метод Гаусса  }
   for L:=0  to n do  begin A1:=A[L,L]; i1:=L;
     for i:=L+1 to n do  begin
       if Abs(A1)<Abs(A[i,L])
         then begin A1:=A[i,L]; i1:=i  end  end;
   for j:=0 to m+1 do begin  D:=A[i1,j];
      A[i1,j]:=A[L,j]; A[L,j]:=D  end;
   G:=A[L,L]; if G=0 then begin
      Writeln('Zadacha imeet mnojestvo reshenij'); Halt end;
```

```
      for j:=L to m do C[L,j]:=A[L,j]/G;
      for i:=L+1 to n do
         for j:=L+1 to m do
         A[i,j]:=A[i,j]-A[i,L]*C[L,j];
      end { of L };
      X[n]:=C[n,m];
      for i:=m-2 downto 0 do  begin x1:=C[i,m];
        for j:=i+1 to M-1 do X1:=X1-X[j]*C[i,j];
        X[i]:=X1;    end;  Writeln;
      { Вывод коэффициентов искомого полинома }
      Writeln(' ':11,'Koefficienty approksimiruychego polinoma:');
      for i:=0 to n do
       Writeln(' ':16,'A(',i:2,')= ',X[i]:14:7);
      Writeln; Writeln('Dlja vychoda najmite "Enter" '); Readln;
End.
```

```
Zadajte kolichestvo tochek isxodnoj tablicy: 5
Vvedite porjadok approksimirujychego polinoma: 2
Vvedite tablicu ishodnych dannych
Posle kajdoj pary chisel najimaite <ET>
               X( 0), Y( 0): 0.78 2.50
               X( 1), Y( 1): 1.56 1.20
               X( 2), Y( 2): 2.34 1.12
               X( 3), Y( 3): 3.12 2.25
               X( 4), Y( 4): 3.81 4.28

          Koefficienty approksimiruychego polinoma:
               A( 0)=      5.0221476
               A( 1)=     -4.0142602
               A( 2)=      1.0023414

Dlja vychoda najmite "Enter"
```

Рисунок 2 – Рабочее окно программы с решением задачи

Построенный полином будет иметь вид (рис. 2): $Q_2(x) = 5{,}022 - 4{,}014x + 1{,}002x^2$.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Построить аппроксимирующий алгебраический полином третьего порядка для функции, заданной таблицей. Варианты таблиц даны в конце лабораторной работы.

Порядок выполнения работы

1. Реализовать и отладить по контрольному примеру в среде Delphi программу `MNK_konsol` построения аппроксимирующего полинома методом

наименьших квадратов.

2. Записать вид модели аппроксимирующего полинома и построить в среде табличного процессора MS Excel график функции по заданным точкам. Сделать предварительный вывод о том, подходит ли заданный вид модели исходным табличным данным функции.

3. Построить таблицу с суммами для коэффициентов системы уравнений.

4. Построить систему линейных уравнений для расчета коэффициентов искомого аппроксимирующего полинома и решить задачу построения полинома, используя отлаженную программу `MNK_konsol`.

5. Дополнить исходную таблицу значениями функции, вычисленными по аппроксимирующему полиному, использовав MS Excel. Сделать вывод о точности построенного полинома.

Содержание отчета:

– название и цель работы;

– номер варианта и исходные данные (таблица, количество точек и порядок полинома);

– результаты выполнения работы по пунктам 1–5;

– выводы к работе.

Таблица 3 – Варианты к лабораторной работе

x_i	Варианты $y_i(x)$													
	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14
0,1	**2,05**	**2,15**	**0,10**	**0,17**	**0,00**	**2,09**	**2,02**	**1,99**	**2,23**	**2,07**	**2,18**	**-0,10**	**-0,16**	**2,09**
0,2	**1,94**	**2,41**	**-0,01**	**0,07**	**0,29**	**2,05**	**1,98**	**2,03**	**2,29**	**2,17**	**2,43**	**-0,21**	**0,01**	**2,31**
0,3	**1,92**	**2,58**	**-0,19**	**0,17**	**0,52**	**2,19**	**1,67**	**2,20**	**2,27**	**2,21**	**2,40**	**0,01**	**0,10**	**2,72**
0,4	**1,87**	**2,84**	**-0,11**	**0,05**	**0,77**	**2,18**	**165**	**2,39**	**2,62**	**2,31**	**2,43**	**0,05**	**0,16**	**2,77**
0,5	**1,77**	**3,28**	**-0,31**	**0,12**	**0,93**	**2,17**	**1,57**	**2,19**	**2,72**	**2,10**	**2,65**	**-0,13**	**0,05**	**2,78**
0,6	**1,88**	**3,46**	**-0,78**	**0,00**	**1,20**	**2,27**	**1,42**	**2,61**	**2,82**	**2,09**	**2,75**	**-0,23**	**0,35**	**2,97**
0,7	**1,71**	**4,02**	**-0,64**	**0,01**	**1,20**	**2,58**	**1,37**	**2,35**	**3,13**	**2,12**	**2,67**	**-0,21**	**0,19**	**3,00**
0,8	**1,60**	**4,11**	**-0,85**	**-0,05**	**1,35**	**2,73**	**1,07**	**2,60**	**3,49**	**1,63**	**2,66**	**-0,43**	**0,50**	**3,51**
0,9	**1,56**	**4,61**	**-1,18**	**-0,21**	**1,39**	**2,82**	**0,85**	**2,55**	**3,82**	**1,78**	**2,63**	**-0,57**	**0,74**	**3,43**
1,0	**1,40**	**5,03**	**-1,39**	**-0,50**	**1,48**	**3,04**	**0,48**	**2,49**	**3,95**	**1,52**	**2,75**	**-0,44**	**1,03**	**3,58**
1,1	**1,50**	**5,34**	**-1,79**	**-0,50**	**1,52**	**3,03**	**0,35**	**2,50**	**4,22**	**1,16**	**2,41**	**-0,44**	**1,06**	**3,58**
1,2	**1,26**	**5,86**	**-2,02**	**-0,86**	**1,71**	**3,45**	**-0,30**	**2,52**	**4,48**	**1,07**	**2,24**	**-0,83**	**1,49**	**3,51**
1,3	**0,99**	**6,33**	**-2,48**	**-1,24**	**1,72**	**3,62**	**-0,61**	**2,44**	**5,06**	**0,85**	**2,12**	**-0,78**	**1,79**	**3,82**

Окончание таблицы 3

x_i	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14
1,4	**0,97**	**6,81**	**-2,93**	**-1,47**	**1,87**	**3,85**	**-1,20**	**2,35**	**5,50**	**0,56**	**1,74**	**-0,81**	**2,03**	**3,90**
1,5	**0,91**	**7,21**	**-3,26**	**-1,79**	**1,86**	**4,19**	**-1,39**	**2,26**	**5,68**	**0,10**	**1,57**	**-1,06**	**2,22**	**3,77**
1,6	**0,71**	**7,67**	**-3,91**	**-2,25**	**1,89**	**4,45**	**-1,76**	**2,19**	**6,19**	**-0,25**	**1,17**	**-1,41**	**2,50**	**3,81**
1,7	**0,43**	**8,23**	**-4,41**	**-2,55**	**2,04**	**4,89**	**-2,28**	**2,24**	**6,42**	**-0,65**	**0,96**	**-1,40**	**2,88**	**4,00**
1,8	**0,54**	**8,68**	**-4,91**	**-3,18**	**1,73**	**5,06**	**-2,81**	**2,34**	**7,04**	**-1,06**	**0,63**	**-1,70**	**3,21**	**3,97**
1,9	**0,19**	**9,35**	**-5,30**	**-3,60**	**2,04**	**5,63**	**-3,57**	**1,96**	**7,57**	**-1,65**	**0,25**	**-1,96**	**3,63**	**4,08**
2,0	**0,01**	**9,93**	**-6,00**	**-3,93**	**2,03**	**5,91**	**-4,06**	**2,19**	**8,10**	**-2,01**	**-0,01**	**-1,91**	**3,90**	**4,08**

2.3 Постановка и реализация задачи интерполирования по Лагранжу

Постановка задачи

Задача интерполирования функции f(x) состоит в том, чтобы найти значение f(x) при $x \neq x_i$ $(i = 0,1,2,\ldots, n)$ через некоторый многочлен $\tau(x)$, если заданы узлы интерполирования $x_0, x_1, x_2, \ldots, x_n$ и значения функции в этих узлах.

Задача интерполирования решается следующим образом:

– выбирается система действительных функций $\varphi_i(x)$ на отрезке [a,b];

– строится обобщенный многочлен $\tau(x) = \sum_{i=0}^{n} c_i \varphi_i(x)$.

Здесь коэффициенты c_i задают таким образом, чтобы значения $\tau(x)$ в узлах интерполирования совпадали бы с f(x): $\tau(x_i) = f(x_i)$, $k = 0, 1, 2, \ldots, n$. Тогда $\tau(x)$ будет для f(x) обобщенным интерполяционным многочленом, и его значения берут в качестве приближенных для f(x).

Требования к выбору $\tau(x)$.

1. Функция $\tau(x)$ должна существовать в области [a,b].

2. $\tau(x)$ должна быть единственным обобщенным многочленом $\tau(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \ldots + c_n\varphi_n(x).$

Пример 1. Пусть функция f(x) задана таблично как результат наблюдений на отрезке [a,b], т.е. $y_i = f(x_i)$, $i = 0,1,2,...,n$ и пусть нужно узнать f(x) при $x \neq x_0, x_1, x_2, ..., x_n$.

Тогда после построения $\tau(x)$ имеем

$\tau(x) = f(x)$ *при* $x = x_i$ *и* $\tau(x) \approx f(x)$ *при* $x \neq x_i$.

Пример 2. Функция имеет сложную аналитическую зависимость. Упростим ее, построив $\tau(x)$.

Возникает вопрос: какой должна быть система функций $\tau(x)$? На практике используют следующие системы функций:

– $1, x, x^2, ..., x^n$ – алгебраическое интерполирование;

– 1, sinx, cosx, sin2x, cos2x, …, sin nx, cos nx – тригонометрическое интерполирование;

– $e^{\alpha_0 x}$, $e^{\varepsilon_1 x}$, $e^{\alpha_2 x}$, ..., $e^{\alpha_n x}$ – экспоненциальное интерполирование.

Алгебраическое интерполирование. Формула Лагранжа

Формула Лагранжа для построения интерполяционного алгебраического полинома имеет вид

$$L_n(x) = \sum_{k=0}^{n} y_k \cdot Q_n^k(x) .$$

Здесь n – степень полинома с заданными в точках x_1, x_2, …, x_n значениями y_0, y_1, y_2, …, y_n; Q_n^k – коэффициенты Лагранжа; $Q_n^k = \dfrac{\omega(x)}{(x - x_k) \cdot \omega'(x_k)}$;

$$\omega(x) = (x - x_0)(x - x_1)...(x - x_n);$$

$$\omega'(x_k) = (x_k - x_0)(x_k - x_1)...(x_k - x_{r-1})(x_k - x_{k+1})...(x_k - x_n)$$

В ω' выпадает одна скобка: $(x_k - x_k)$, в ω присутствуют все скобки.

Характеристика полинома Лагранжа:

1. Узлы могут быть неравноотстоящими, т. е. между точками может быть переменный шаг.
2. Вид ИПЛ зависит от количества узлов.
3. Поэтому его вычисляют заново при изменении значения n.
4. Степень ИПЛ всегда меньше количества точек на единицу.
5. Погрешность интерполирования $r_n(x) = f(x) - \tau(x)$, то есть отлична от нуля. В узлах $r_n(x)$ превращается в нуль.
6. Сходимость процесса интерполяции: $L_n(x) = f(x)$ *при* $n \to \infty$.

Пример 3. Построение ИПЛ. Построить ИПЛ для таблично заданной функции:

k	x_k	y_k
0	-3	8
1	-1	6
2	1	4
3	2	18

Согласно данным таблицы порядок полинома – 3.

Формула Лагранжа принимает вид

$$L_3(x) = y_0 \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x-x_0)(x_0-x_1)(x_0-x_2)(x_0-x_3)} +$$
$$+ y_1 \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x-x_1)(x_1-x_0)(x_1-x_2)(x_1-x_3)} +$$
$$+ y_2 \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x-x_2)(x_2-x_0)(x_2-x_1)(x_2-x_3)} +$$
$$+ y_3 \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x-x_3)(x_3-x_0)(x_3-x_1)(x_3-x_2)}.$$

Подставляем xk и yk . После преобразования получаем

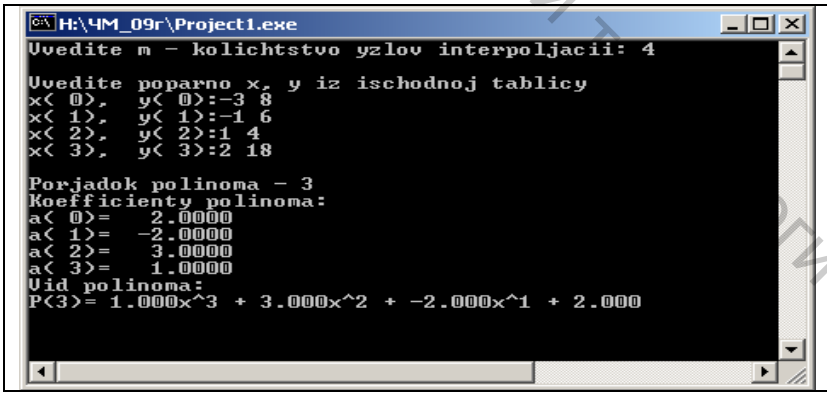
$$L_3(x) = x^3 + 3x^2 - 2x + 2.$$

Данный пример может быть использован для отладки программы.

Программа в консольном режиме Delphi

```
program IPLagran;
{$APPTYPE CONSOLE}
{Построение интерполяционного полинома Лагранжа}
uses  SysUtils;
Type V=array[0..12] of Extended;
Var m, n, i, i1, j, k : Integer;
   A, B, X, X1, Y : V;  A1,B1 : Extended;
Begin
{1. Ввод исходной матрицы}
Write('Vvedite m - kolichtstvo yzlov interpoljacii: ');
readln(m);
n:=m-1;  Writeln;
Writeln('Vvedite poparno x, y iz ischodnoj tablicy');
For i:= 0 To n Do
begin
Write('x(', i:2, '),  y(', i:2, '):');
Readln(x[i], y[i]);
end;
For i:=0 To n Do B[i]:=0;   {Обнуление массива коэффициентов}
{2. Главный цикл k. }
For k:=0 To n Do  begin  B1:=1;
  {2.1. Исключение из массива  xk }
    For i1:=0 To n Do  begin
      If i1 < k Then X1[i1]:=X[i1] Else
      If i1 > k Then X1[i1-1]:=X[i1];   end;
  {2.2. Раскрытие скобок полинома по Грунду}
    A[0]:=-X1[0]; A[1]:=1;
    For i:=1 To n-1 Do begin {of i}  A[i+1]:=1;
```

```
      For j:=i Downto 1 Do
          A[j]:= A[j-1] - A[j]*X1[i];
          A[0]:=-A[0]*X1[i];
   end {of i} ;
{2.3. Сборка в коэффициенты полинома числителей, знаменателей и дробей}
   {Подсчет знаменателей}
      For i:=0 To n-1 Do B1:=B1*(x[k]-X1[i]);
   {Получение дроби} For i:=0 To n Do
        begin A1:=A[i]*y[k]/B1;
   {Суммирование дробей} B[i]:=B[i]+A1 end {of i} ;
end {of k - конец главного цикла};
{3. Вывод параметров искомого полинома}
Writeln; Writeln('Porjadok polinoma - ', n);
Writeln('Koefficienty polinoma:');
For i:= 0 To n Do Writeln('a(', i:2, ')= ', B[i]:8:4);
Writeln('Vid polinoma:');
Write('P(', n, ')= ');
For i:=n downto 1 Do Write(B[i]:2:3,'x^',i,' + ');
Writeln(b[0]:2:3);
Readln;
End.
```



Исходная таблица к примеру

k	x_k	y_k
0	-3	8
1	-1	6
2	1	4
3	2	18

Рисунок 3 – Рабочее окно программы и исходные данные

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Построить аппроксимирующий алгебраический полином четвертого порядка для функции, заданной таблицей.

Содержание отчета:

- название и цель работы;
- номер варианта и исходные данные (таблица, количество точек и порядок полинома);
- результаты выполнения работы по пунктам 1–5;
- выводы к работе.

Порядок выполнения работы

1. Изучить теорию вопроса и проанализировать пример и программу построения интерполяционного полинома Лагранжа.

2. Реализовать текст примера программы, отладить программу по **примеру 3** и построить ИПЛ для заданного варианта.

3. Оформить отчет по работе, включив в него формулу ИПЛ, вариант задания, текст программы, рабочее окно программы с примером отладки и рабочее окно с построением ИПЛ для своего варианта.

Таблица 4 – Варианты заданий к построению ИПЛ

Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k
1	0	2	8	2	0	1	9	3	0	-4	1	4	0	-3	8
	1	3	5		1	3	4		1	-2	5		1	-2	4
	2	5	4		2	4	5		2	-1	6		2	-1	3
	3	7	6		3	6	3		3	0	7		3	1	5
	4	9	9		4	8	12		4	3	18		4	3	15
Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k
5	0	2	12	6	0	1	2	7	0	-5	22	8	0	-8	-16
	1	3	6		1	2	7		1	-3	14		1	-6	+1
	2	4	4		2	5	9		2	-1	10		2	-5	3
	3	6	8		3	6	12		3	1	11		3	-3	5
	4	8	20		4	7	16		4	2	17		4	0	13
Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k
9	0	2	-6	10	0	0	14	11	0	-6	6	12	0	-5	8
	1	3	+2		1	2	3		1	-4	4		1	-3	6
	2	5	12		2	3	-2		2	-5	-3		2	0	5
	3	7	11		3	5	1		3	-3	-1		3	2	7
	4	9	5		4	6	7		4	0	8		4	7	16
Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k
13	0	1	13	14	0	1	8	15	0	-4	9	16	0	-5	-2
	1	3	15		1	2	7		1	-3	7,4		1	-3	4
	2	5	16		2	3	6,5		2	0	2		2	0	7
	3	7	11		3	5	7,5		3	2	3		3	1	5,5
	4	10	20		4	7	8		4	5	8		4	3	2
Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k	Вар	k	x_k	y_k
17	0	2	1	18	0	1	2	19	0	-4	-5	20	0	-3	6
	1	4,5	4		1	3	5		1	-2	2		1	-2	5
	2	6	6		2	4	6,3		2	-1	2,6		2	-1	4,7
	3	8	7		3	6	7		3	1	1		3	3	6
	4	9	5		4	7	4		4	2	-4		4	6	14

2.4 Интерполирование сплайнами

Задача построения интерполирующего полинома в ряде случаев усложняется или становится невыполнимой. Например, при большом количестве точек или для функций с особенностями. В таких случаях используют т. н. сплайны.

Сплайном считают функцию Sm(x), непрерывную на всем отрезке [a; b] вместе со своими производными, которая является набором некоторых алгебраических многочленов, каждый из которых используется отдельно на своем частичном отрезке $[x_{i-1}, x_i]$, входящем в [a; b].

Запишем постановку задачи интерполирования сплайном. Пусть на отрезке [a;b] оси X задана сетка $a = x_0<x_1<...<x_i<...<x_n = b$, в узлах которой x_i заданы значения $Y_i = f(x_i)$, $i = 0,1,2,...,n$ достаточно плавной функции f(x). Требуется по заданным значениям $f(x_i)$ найти функцию g(x), интерполирующую искомую функцию Y(x) следующим образом

$$\begin{cases} g_1(x),\ x \in [x_0, x_1] \\ g_2(x),\ x \in [x_1, x_{21}] \\ \cdots\cdots\cdots\cdots \\ g_n(x),\ x \in [x_{n-1}, x_n] \end{cases} \qquad (3)$$

Программа на языке Delphi построения кубического сплайна

Для задачи моделирования кубическим сплайном создадим приложение Delphi с полноценным графическим интерфейсом под Windows. Delphi-приложение имеет вид:

```
unit Unit1; // Модуль формы
interface

uses  Windows, Messages, SysUtils, Variants, Classes, Graphics,
      Controls,  Forms,  Dialogs,  Buttons,  TeEngine,  Series,
ExtCtrls,
      TeeProcs, Chart, Grids, StdCtrls;
type
  TForm1 = class(TForm)
    Label1: TLabel;
    Edit1: TEdit;
    Button1: TButton;
    Label2: TLabel;
    StringGrid1: TStringGrid;
    Button2: TButton;
    Chart1: TChart;
    Label3: TLabel;
    Series1: TLineSeries;
    Button3: TButton;
    Button4: TButton;
    BitBtn1: TBitBtn;
    procedure FormCreate(Sender: TObject);
```

```
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
Type V=array[1..20] of real;  W=array[0..20] of real;
var Form1: TForm1; i,j,k,m,n:integer; Bl:Boolean;
    a1,a3,b,h,X,F,F1,z:v; a2:W; x1,y1:Extended;

implementation
{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
m:=5;
Edit1.Text:= IntToStr(m);
StringGrid1.RowCount:=m+1;
end;

procedure TForm1.Button1Click(Sender: TObject);
// Кнопка "Прочитать n"
begin
m:=StrToInt(Edit1.Text);  n:=m-1;
StringGrid1.ColCount:=3;
StringGrid1.Cells[0,0]:='i';
StringGrid1.Cells[1,0]:='x';
StringGrid1.Cells[2,0]:='y';
StringGrid1.RowCount:=m+1;
For j:=1 to 3 do for i:=1 to m do StringGrid1.Cells[j,i]:='';
for i:=0 to n do StringGrid1.Cells[0,i+1]:=IntToStr(i);
StringGrid1.RowCount:=m+1;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin  { Ввод значений исходной таблицы: }
For i:=1 to m do
  begin x[i]:=StrToFloat(StringGrid1.Cells[1,i]);
        F[i]:=StrToFloat(StringGrid1.Cells[2,i]);
  end;
end;

procedure TForm1.Button3Click(Sender: TObject);
// Расчет модели сплайна
begin
{ Вычисление значений шага интерполирования hi }
for i:=1 to n do h[i]:=X[i+1]-X[i];
{ Вычисление свободных членов Fi  к системе уравнений }
```

```
for i:=1 to m-2 do
F1[i]:=((F[i]-F[i+1])/h[i]-(F[i+1]-F[i+2])/h[i+1])*3;
```
{Решение системы уравнений с трехдиагональной матрицей методом прогонки}

{ 4.1 Прямой ход – вычисление коэф-тов b и z }

{ 4.1.1 Первый шаг вычислений }

```
b[1]:=-h[2]/(2*(h[1]+h[2]));
z[1]:=F1[1]/(2*(h[1]+h[2]));
```

{ 4.1.2 Цикл для шагов 2 ... (n-1) }

```
for i:=2 to m-2 do  begin
  b[i]:=-h[i+1]/(2*(h[i]+h[i+1])+h[i]*b[i-1]);
  z[i]:=(F1[i]-h[i]*z[i-1])/(2*(h[i]+h[i+1])+h[i]*b[i-1])
                          end;
```

{ 4.2 Обратный ход – вычисление корней СЛУ }

{ 4.2.1 Значения из начальных условий }

```
a2[m-1]:=0; a2[m-2]:=z[m-2]; a2[0]:=0;
```

{ 4.2.2 Цикл для шагов (n-2) ... 1 }

```
for i:=m-3 downto 1 do  a2[i]:=b[i]*a2[i+1]+z[i];
```

{ Найдены коэффициенты сплайна a2(i) }

{ Расчет коэф-тов сплайна a1(i) и a3(i) через a2(i) }

```
for i:=1 to n do  begin
  a1[i]:=(-h[i]/3)*(a2[i-1]+2*a2[i])+(F[i]-F[i+1])/h[i];
  a3[i]:=(a2[i-1]-a2[i])/(3*h[i]);
                    end;
StringGrid1.ColCount:=5;
StringGrid1.RowCount:=m;
StringGrid1.Cells[0,0]:='g(k)';
StringGrid1.Cells[1,0]:='a0';
StringGrid1.Cells[2,0]:='a1';
StringGrid1.Cells[3,0]:='a2';
StringGrid1.Cells[4,0]:='a3';
for j:=0 to 4 do for i:=1 to m do
  StringGrid1.Cells[j,i]:='';
Label2.Caption:='';
```

// Выставить данные расчета модели сплайна в таблицу

```
Label2.Caption:='Коэффициенты модели';
for k:=1 to n do StringGrid1.Cells[0,k]:='g('+IntToStr(k)+')';
for i:=1 to n do StringGrid1.Cells[1,i]:=FloatToStr(F[i+1]);
for i:=1 to n do StringGrid1.Cells[2,i]:=FloatToStr(a1[i]);
for i:=1 to n do StringGrid1.Cells[3,i]:=FloatToStr(a2[i]);
for i:=1 to n do StringGrid1.Cells[4,i]:=FloatToStr(a3[i]);

end;

Procedure FotX; //Расчет значения функции по построенной модели
Begin  BL:=False; k:=1;
```

{ Нахождение интервала, содержащего заданное знач. X }

```
for i:=1 to n do begin If (x1 > X[i]) and (x1 <= X[i+1])
  Then begin k:=i; BL:=True;
    If BL and (x1=X[i]) Then y1:=F[i]; end; end;
```

```
y1:=F[k+1]+(X[k+1]-x1)*a1[k];
y1:=y1+Sqr(X[k+1]-x1)*a2[k];
y1:=y1+Sqr(X[k+1]-x1)*(X[k+1]-x1)*a3[k];
End;

procedure TForm1.Button4Click(Sender: TObject);
// Построение графика по рассчитанной модели сплайна
var ngr:extended;
begin
  ngr:=(x[m]-x[1])/64;
  x1:=x[1];
  Series1.Clear;
  While x1<=x[m] do
    begin
      Fotx;
      Series1.AddXY(x1,y1,'',clRed);
      x1:=x1+ngr;
    end;

end;
end.
```

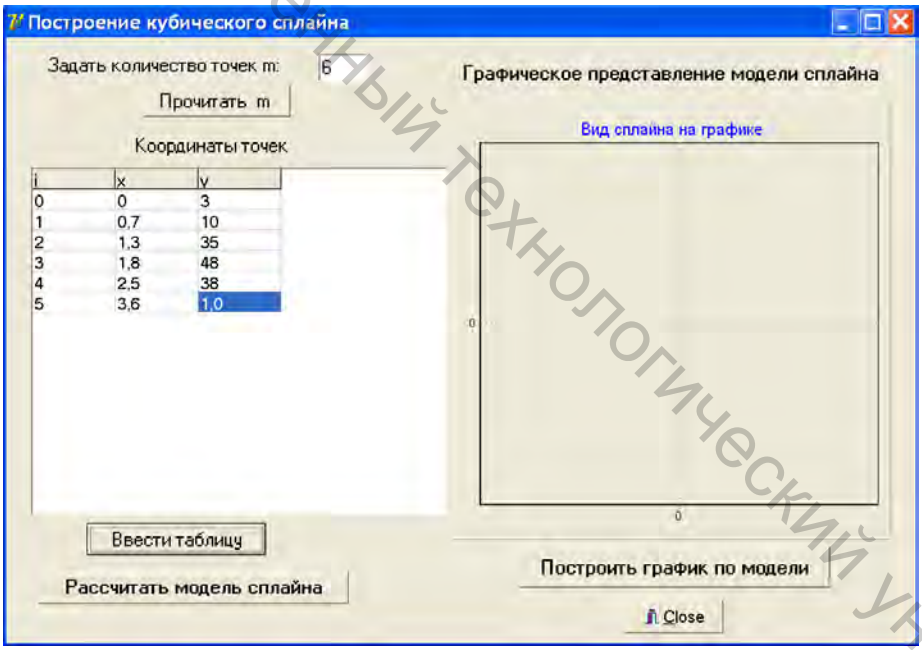


Рисунок 4 – Рабочее окно программы примера после ввода исходных данных

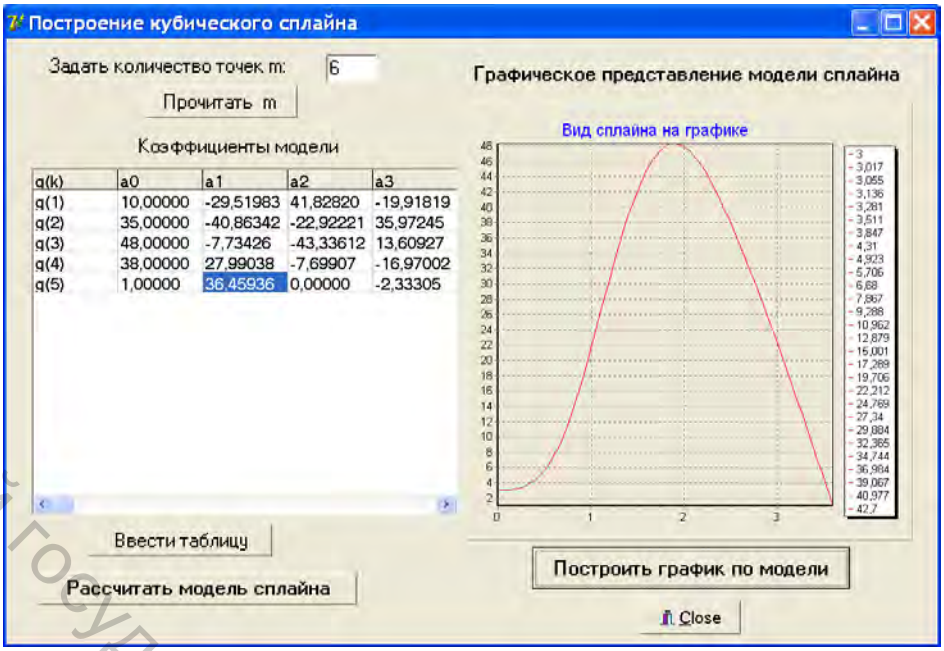


Рисунок 5 – Рабочее окно программы для примера с построенной моделью

ЗАДАНИЕ

1. Построить приложение с интерфейсом под Windows для решения поставленной задачи.
2. Отобразить решение примеров средствами Microsoft Excel.
3. Оформить отчет с решением примеров в среде MS Excel.
4. Построить другие модели кубических сплайнов.
5. Оформить отчет, поместив в него название темы, алгоритм и программу построения кубического сплайна и примеры рабочего окна программы с исходными данными и решениями.

3 ПАКЕТЫ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ МОДЕЛИРОВАНИЯ И ЧИСЛЕННОГО РЕШЕНИЯ ОПТИМИЗАЦИОННЫХ ЗАДАЧ

Математические модели в задачах оптимального проектирования

Задача параметрической оптимизации связана с улучшением зависимых параметров $\overline{\varphi}(\overline{x})$ путем изменения управляемых независимых параметров $\overline{x}$.

Задача оптимального проектирования связана с числом n параметров $\overline{x}$, которые принадлежат допустимой области решений D и которые обеспечивают критерий оптимальности $Q(\overline{x})$ и приводят к оптимальному решению $\overline{x}^*$.

Одномерная минимизация унимодальных функций.

Задача связана с поиском x* $Q(x^*) = \min Q(x)$

$a \le x \le b$

Главное здесь – методы сокращения интервала неопределенности [a;b]. Исходный интервал – априорный, интервал после сокращения – апостериорный. На каждом шаге поиска производятся испытания x_1^k и x_2^k. Стремятся, чтобы текущая апостериорная длина интервала была как можно меньше, это принцип минимакса $\min\limits_{(x_1^k,\, x_2^k)} \max\{x_2^k - a_k,\, b_k - x_1^k\}$

3.1 Метод дихотомического поиска, т.е. метод деления отрезка пополам

Выбор точек испытаний здесь упрощен

$$x_1^k = \frac{a_k + b_k - \delta}{2};\quad x_2^k = \frac{a_k + b_k + \delta}{2}.$$

δ – минимально допустимое различие между точками испытаний x_1^k и x_2^k. Считаем δ заданной погрешностью. Ищем максимум.

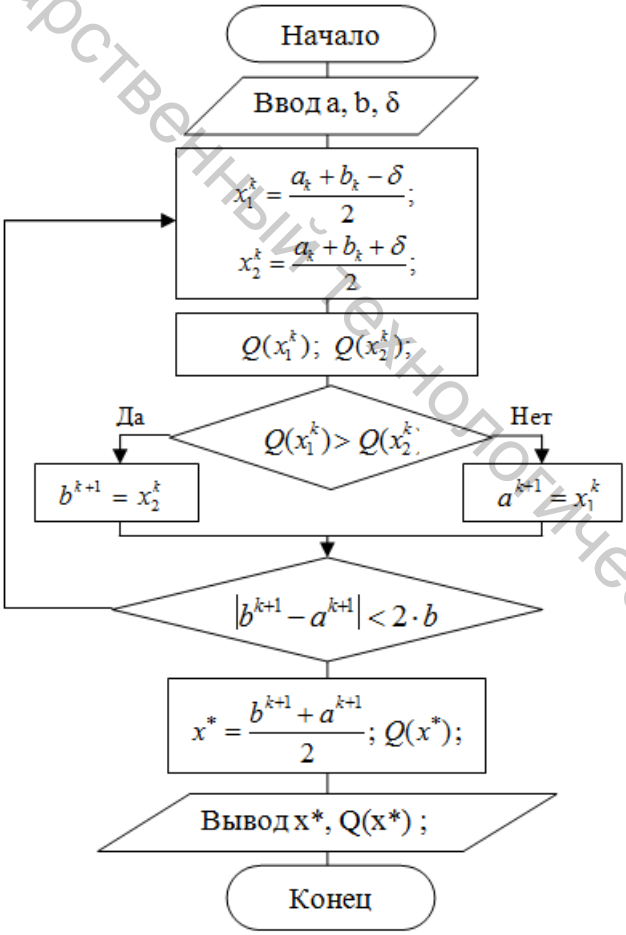


Рисунок 6 – Схема алгоритма поиска максимума

Из схемы алгоритма метода видно, что метод реализуется в цикле.

Начало цикла.

– вычисляем x_1^k и x_2^k ;

– для x_1^k и x_2^k вычисляем $Q(x_1^k)$ *и* $Q(x_2^k)$;

– проверяем отношение $Q(x_1^k) > Q(x_2^k)$. Если «да», то $b^{k+1} = x_2^k$, иначе $a^{k+1} = x_1^k$;

– проверяем условие $\left|b^{k+1} - a^{k+1}\right| < 2 \cdot b$ на выход из цикла.

Конец цикла.

Уточняем точку максимума $x^* = \frac{b^{k+1} + a^{k+1}}{2}$; вычисляем $Q(x^*)$.

ЗАДАНИЕ

Разработать приложение на основе алгоритма (рис. 6) для поиска минимума. Поиск минимума заложен в условии варианта задачи. Таблица вариантов дана в конце темы.

Рекомендация: вычисление значений функции f(x) выполнять через обращение к подпрограмме-функции.

3.2 Минимизация функции одной переменной методом «золотого сечения»

В методе «золотого сечения » мы вычисляем одну точку на каждом шаге, но усложняется деление интервала.

Предположим, что функция f(x) имеет только один минимум на интервале [a,b] в точке x*. Численная задача поиска экстремума состоит в последовательном сужении интервала неопределенности путем его деления. Можно по-разному делить интервал на части. Наилучшей стратегией будет та, что быстрее приведет к результату. Наилучшей считают стратегию Фибоначчи. Ее применимость усложняется необходимостью использовать числа Фибоначчи. Проще воспользоваться методом «золотого сечения».

«Золотое сечение» – это деление заданного отрезка ab так, чтобы ab/Lb =Lb/Lm, где ab – весь отрезок, Lb – большая часть отрезка, Lm – меньшая часть отрезка.

Координаты Хлев и Хпр соответственно равны: Хлев = b – tay (b - a); Хпр = a + tay (b - a).

Из схемы (рис. 7) видны этапы поиска минимума.

1. Делим [a, b] точками Хлев и Хпр, (слева и справа). Вычисляем f (Хлев), f (Хпр), чтобы отбросить отрезок a – Хлев или Хпр – b. На оставшемся отрезке уже есть одна точка «золотого сечения». Строим вторую точку.
2. Процесс 1 повторяется. При этом интервал неопределенности уменьшается в 0,6180339 раз. Повторение будет, пока интервал неопределенности не станет меньше заданной погрешности delta.
3. Исходными данными для задачи являются f(x), a, b, delta, tay.

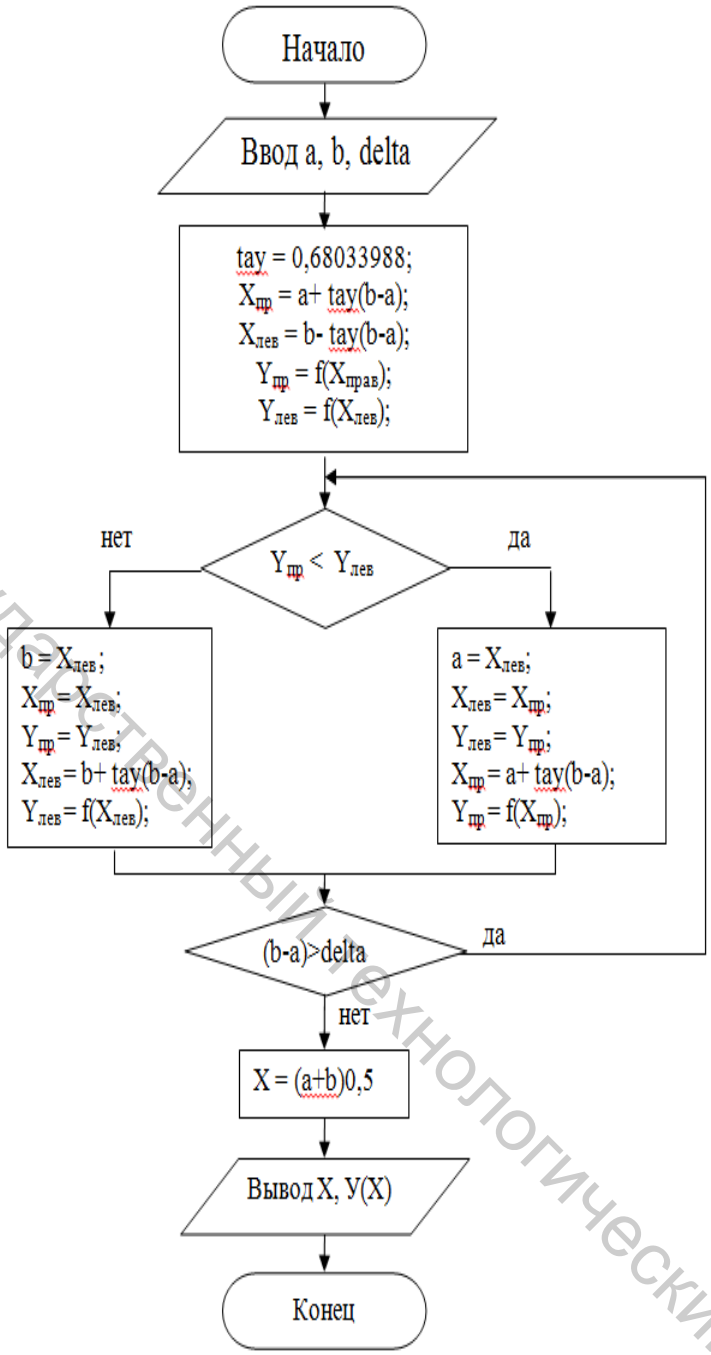


Рисунок 7 – Схема алгоритма метода «золотого сечения»

ЗАДАНИЕ

1. Разработать программу на языке Delphi в консольном режиме минимизации унимодальной функции на заданном отрезке согласно выданному варианту.
2. Вариант задания брать в конце темы «Методы оптимизации».
3. Вычисление значений функции f(x) выполнять через обращение к подпрограмме-функции.

3.3 Минимизация функции одной переменной методом Фибоначчи

Суть метода Фибоначчи. Считают, что стратегия поиска минимума в методе Фибоначчи самая эффективная среди методов оптимизации унимодальных функций с ограничениями.

Стратегия метода использует свойства ряда чисел Фибоначчи, который генерируется по правилу: $fi_0 = 1;\ fi_1 = 1;$ далее:

$$fi_2 = fi_0 + fi_1;\ \ldots;\ fi_i = fi_{i-1} + fi_{i-2};\ \ldots;$$
$$fi_n = fi_{n-1} + fi_{n-2},$$

где $i = 2, 3, 4, \ldots, n$.

Тогда $fi_2 = 2;\ fi_3 = 3;\ fi_4 = 5;\ fi_5 = 8;\ fi_6 = 13;\ fi_7 = 21\ \ldots$

Алгоритм метода Фибоначчи

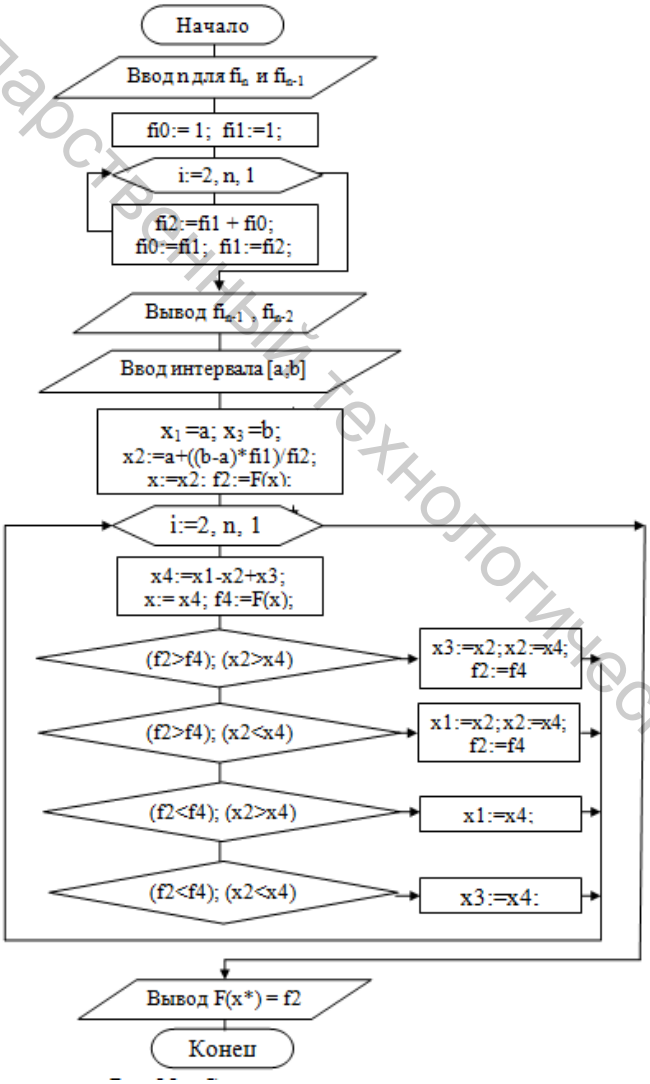


Рисунок 8 – Схема хода решения задачи

Пример. Минимизировать методом Фибоначчи функцию $y = x^4 - 14x^3 + 60x^2 - 70x$ на интервале поиска $x \in [-2; 3]$.

Предлагаем к методу Фибоначчи программу на языке Delphi в консольном режиме, разработанную согласно приведенным выше схеме и примеру.

```
program FiboMinimize;
{$APPTYPE CONSOLE}
uses  SysUtils, Math;
var x,x1,x2,x3,x4,a,b,f2,f4:extended; fi0,fi1,fi2,i,n:Integer;

Function F(x:Extended):Extended;
begin F:=x*x*x*x - 14*x*x*x+60*x*x-70*x; end;

begin
  {Этап 1. Генерация числа Фибоначчи Fi(n) для заданного n}
    Write('Vvedite n: ');  Readln(n);
   fi0:= 1;  fi1:=1;   {Подготовка к циклу }
   For i:=2 To n Do
     begin
      fi2:=fi1 + fi0; {Вычисление (i+2) -го числа fi. }
      fi0:=fi1;  fi1:=fi2;{Обновление значений для следующего цикла }
     end;
   Write('fin = ', fi1); Writeln('  fin-1 = ',fi0);
   Fi2:=Fi1; Fi1:=Fi0; {Вернул числа на место}
   Writeln('Nazmite Vvod');
   Readln;
   Write('Zadajte interval poiska [a,b]: '); Readln(a,b);

   x1:=a; x3:=b; x2:=a+((b-a)*fi1)/fi2; x:=x2; f2:=F(x);
   //Writeln('Znacyenija tekuszhego intervala:');
   For i:=2 To n Do
      begin
         x4:=x1-x2+x3; x:= x4; f4:=F(x);
        If (f2>f4) and (x2>x4) Then
                   begin x3:=x2; x2:=x4; f2:=f4 end;
         If (f2>f4) and (x2<x4) Then
                   begin x1:=x2; x2:=x4; f2:=f4 end;
         If (f4>f2) and (x2>x4) Then x1:=x4;
         If (f4>f2) and (x2<x4) Then x3:=x4;
         //Writeln(x1:12:8,'  ',x3:12:8);
      end;
   Writeln('Znachenie F(x*)= ', f2:12:9);
   WRiteln('x= ',x2:10:7,'  F(x)= ',f(x2):12:8);
   Readln;
end.
```

```
Vvedite n: 12
fin = 233  fin-1 = 144
Nazmite Vvod

Zadajte interval poiska [a,b]: -2 3
Znachenie F(x*)= -24.367210658
x=  0.7896996  F(x)= -24.36721066
```

Рисунок 9 – Рабочее окно программы с исходными данными и результатами

ЗАДАНИЕ

1. **Разработать программу с использованием** метода Фибоначчи на основе примера для своего варианта, сравнить результаты с предыдущими по данной теме.
2. Отобразить решение примеров средствами Microsoft Excel.
3. Оформить отчет с решением примеров в среде MS Excel.

Таблица 5 – Варианты заданий

№	Функция	Ответ	№	Функция	Ответ
1.	$y = x^2 + e^{-0,85x}$	x=0,3229 y=0,864	16.	$y = x^4 - 0,3 \cdot arctg(3,5x$	x=0,43109 y=-0,2611
2	$y = x^2 + 2 \cdot e^{-0,65x}$	x=0,4768 y=1,6943	17.	$y = x^4 - 0,1 \cdot arctg(4,0x$	x=0,33113 y=-0,0803
3.	$y = x^2 + 3 \cdot e^{-0,45x}$	x=0,5314 y=2,644	18.	$y = x^4 + 0,2 \cdot arctg(4,5x$	x=-0,3837 y=-0,1875
4.	$y = x^2 + 4 \cdot e^{-0,25x}$	x=0,4471 y=3,777	19.	$y = x^4 + 0,4 \cdot arctg(5x)$	x=-0,4405 y=-0,4202
5.	$y = x^2 + 5 \cdot e^{-0,05x}$	x=0,1242 y=4,9845	20.	$y = x^4 + 0,8 \cdot arctg(5,5x$	x=-0,5028 y=-0,9151
6.	$y = x^2 + 6 \cdot e^{+0,15x}$	x=-0,4224 y=5,8101	21.	$y = -4,0 \cdot x + e^{\lvert x-0,2 \rvert}$	x=1,5863 y=-2,3452
7.	$y = x^2 + 7 \cdot e^{0,35x}$	x=-0,8954 y=5,9185	22.	$y = -3,4 \cdot x + e^{\lvert x-0,4 \rvert}$	x=1,62338 y=-2,1208
8.	$y = x^2 + 8 \cdot e^{0,55x}$	x=-1,1614 y=5,5723	23.	$y = -2,8 \cdot x + e^{\lvert x-0,6 \rvert}$	x=1,6296 y=-1,7629
9.	$y = x^2 + 9 \cdot e^{0,75x}$	x=-1,2862 y=5,0844	24.	$y = -2,2 \cdot x + e^{\lvert x-0,8 \rvert}$	x=1,5885 y=-1,2946
10.	$y = x^2 + 10 \cdot e^{0,95x}$	x=-1,3356 y=4,5955	25.	$y = -1,6 \cdot x + e^{\lvert x-0,9 \rvert}$	x=1,3700 y=-0,5920
11.	$y = x^4 - 1,5 \cdot arctg(x)$	x=0,6426 y=-0,6862	26.	$y = -1,0 \cdot x + e^{\lvert x-1,2 \rvert}$	x=1,20000 y=-0,2000
12.	$y = x^4 - 1,3 \cdot arctg(1,5x$	x=0,63472 y=-0,82681	27.	$y = -0,4 \cdot x + e^{\lvert x-1,4 \rvert}$	x=1,4000 y=0,44000
13.	$y = x^4 - 1,1 \cdot arctg(2,0$	x=0,60614 y=-0,83418	28.	$y = -0,2 \cdot x + e^{\lvert x-1,6 \rvert}$	x=1,6000 y=0,68000
14.	$y = x^4 - 0,9 \cdot arctg(2,5$	x=0,57029 y=-0,75744	29.	$y = 0,8 \cdot x + e^{\lvert x-1,8 \rvert}$	x=1,8000 y=2,4400
15.	$y = x^4 - 0,7 \cdot arctg(3,0$	x=0,52994 y=-0,62766	30.	$y = 1,4 \cdot x + e^{\lvert x-2,0 \rvert}$	x=1,6635 y=3,7289

4 КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ПРИКЛАДНЫХ ЗАДАЧ

4.1 Численное интегрирование. Приближенное вычисление определенного интеграла методом Симпсона

Процесс интегрирования – важная составная часть многих научно-технических задач. Теоретически вычисление **определенного интеграла** выполняется по известной формуле Ньютона – Лейбница

$$\int_a^b f(x)dx = F(x)\Big|_a^b = F(b) - F(a)\,, \qquad (4)$$

где $F'(x) = f(x)$, если f(x) на отрезке [a,b] непрерывна и имеет первообразную F(x).

Однако первообразная не всегда может быть найдена или f(x) может быть задана таблично. Тогда важное значение приобретает задача численного анализа, использующая численные методы приближенного вычисления определенного интеграла на основании ряда значений подынтегральной функции f(x).

В реализации алгоритма на компьютере удобнее использовать простую квадратурную формулу Симпсона, повторяя ее n/2 раз в цикле на всем отрезке [a, b].

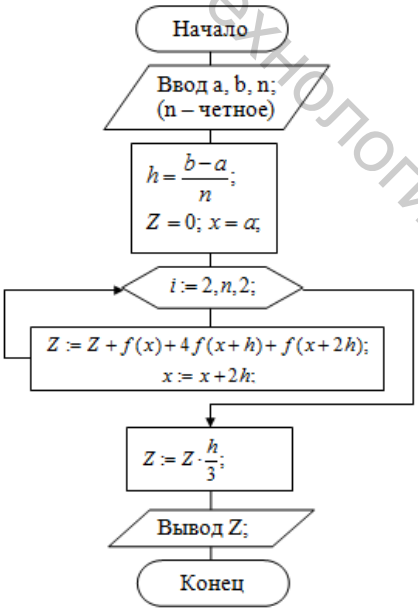


Рисунок 10 – Схема алгоритма процесса

Удобно задавать значение n равным двойке в целой положительной степени, что повышает точность. Результатом является приближенное значение Z искомого интеграла.

Предлагается дополнить алгоритм, повторив вычисление Z для удвоенного количества интервалов (для 2n интервалов). Затем, используя значения Z(n) и Z(2n) в поправке по Рунге, вычислить уточненное значение Z.

Как видно, чтобы использовать поправку по Рунге, необходимо выполнить дополнительный пересчет Z. При этом уровень точности мы не контролируем.

Метод удобно реализовать, организовав вычисления по простой формуле Симпсона в подпрограмме (рис. 10). Покажем схему метода двойного пересчета в примере программы на Dephi:

```
program Project2;
{ Вычисление определённого интеграла по простой формуле Симпсона }
{$APPTYPE CONSOLE}
uses  SysUtils, Math;
Const eps=1e-6; // заданная точность
      n0=16; // начальное разбиение интервала интегрирования
Var  a,b,r,x,Z,Z1:extended; i,n:integer;

{Подынтегральная функция}
Function fx(x:extended):extended;
  begin result:=1+Sqr(sin(x)) end;

{Функция вычисления по формуле Симпсона и суммирование}
Function itg(a1,b1:extended; n1:integer):extended;
var h,P:extended;
begin     { Метод Симпсона}
  h:=(b1-a1)/n1; x:=a; P:=0;
  For i:=1 to n div 2 Do
    begin
     P:=P+fx(x)+4*fx(x+h)+fx(x+2*h); x:=x+2*h;
    end;
  Result:=P*h/3;
end;  // of Function itg

{Основная программа}
Begin
  n:=n0;
  Writeln('Vychislenie opredelennogo integrala');
  Writeln(' ':8, 'po formule Simpsona');
  Write('Vvedite predely a,b - > '); //пределы интегрирования
  Readln(a,b);
  Z:=itg(a,b,n); // Первый вызов метода Симпсона
  Repeat   // Цикл пересчета с контролем текущей точности
    n:=n*2;
```

```
    Z1:=itg(a,b,n);
    r:=abs(Z1-Z);
    Z:=Z1;
  Until r < eps;
  Writeln(' ':11, 'Rezultat:');
  Writeln(' ':11, 'Z=', z:12:6); // Вывод результата
  Readln;
END.
```

В тексте программы записана подынтегральная функция $f(x) = 1 + \sin^2 x$, заданная точность зафиксирована в разделе констант, а решаемая задача имеет вид: вычислить $Z = \int_a^b (1 + \sin^2 x)dx$ на интервале [1; 2].

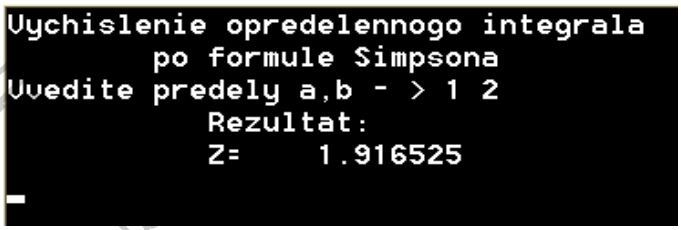


Рисунок 11 – Окно программы с решением задачи

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Разработать программу приближенного вычисления определенного интеграла (ОИ) с применением квадратурных формул Ньютона – Котеса. Применить внутренние подпрограммы в алгоритме основной задачи.

Для выданного варианта задания разработать программы в консольном режиме Delphi в виде двух заданий согласно нижеследующим пунктам.

Задание 1. Разработать программу вычисления ОИ по формуле Симпсона, используя схему алгоритма реализации простой квадратурной формулы Симпсона и поправку по Рунге для уточнения результата. Программа должна содержать два цикла: первый цикл – для вычисления Z(n), второй цикл – для вычисления Z(2n) и подпрограмму-функцию для вычисления значений подынтегральной функции f(x). Вычисленный в приложении результат проверить по заданной в варианте первообразной (колонка 5 табл. 6).

Задание 2. Реализовать программу (смотри пример программы) с методом двойного пересчета по формуле Симпсона. Метод Симпсона заключить в подпрограмму-процедуру, а вычисление f(x) – в подпрограмму-функцию. Головная программа должна содержать вызов процедуры метода Симпсона, а процедура метода – вызов подпрограммы-функции для вычисления значений f(x).

Оформить отчет по работе.

Таблица 6 – Варианты заданий

№ вар.	Пределы интегрир-я	Подынтегральная функция f(x)	Точность вычисл.	Значение первообразной функции F(x)
1	2	3	4	5
1	[1; 3,5]	$\ln x/(x\sqrt{1+\ln x})$	0,000001	$\frac{2}{3}(\ln x+1)^{3/2}-2(\ln x+1)^{1/2}+\frac{4}{3}$
2	[2;3]	1/(x lgx)	0,000001	2,3026(ln lnx-ln ln2)
3	[1;4]	$(\ln^2 x)/x$	0,000001	$\frac{1}{3}\ln^3 x$
4	[0,1; 0,9]	$\sqrt{e^x-1}$	0,000001	$2\sqrt{e^x-1}-2arctg(\sqrt{e^x-1})$
5	[0; 1]	$x\cdot e^x\cdot\sin x$	0,000001	$(x\cdot e^x(\sin x-\cos x)+e^x\cdot\cos x-1)/2$
6	[0; 2]	$x\cdot sh\,x$	0,000001	$(x(e^x+e^{-x})/2-(e^x-e^{-x})/2$
7	[0; 2]	$1/(\sqrt{9+x^2})$	0,000001	$\ln(x+\sqrt{x^2+9})-\ln 3$
8	[1; 2,5]	$\frac{1}{x^2}\sin\frac{1}{x}$	0,000001	$\cos\frac{1}{x}-\cos 1$
9	[0; 3]	$x\cdot arctgx$	0,000001	$\frac{x^2}{2}arctgx-\frac{x}{2}+\frac{arctgx}{2}$
10	[0; 3]	$\arcsin\sqrt{\frac{x}{1+x}}$	0,000001	$x\cdot\arcsin\sqrt{\frac{x}{1+x}}-\sqrt{x}+arctg\sqrt{x}$
11	[1; 3]	$x^x(1+\ln x)$	0,000001	x^x-1
12	[0; 1]	2^{3x}	0,000001	$\frac{1}{3\ln 2}(2^{3x}-1)$
13	[0; 1]	$\frac{x-arctgx}{\sqrt{1+x^2}}$	0,000001	$\sqrt{1+x^2}arctgx-\ln(x+\sqrt{1+x^2})$
14	[0; 2]	$(e^{3x}+1)/(e^x+1)$	0,000001	$(e^{2x})/2-e^x+x+0{,}5$
15	[0; π/2]	$\sin^2 x$	0,000001	$x/2-(\sin 2x)/4$
16	[0; 0,9999]	$x^2\sqrt{4-x^2}$	0,000001	$2\arcsin\frac{x}{2}-\frac{1}{2}\sin(4\arcsin\frac{x}{2})$
17	[0; π]	$e^x\cos^2 x$	0,000001	$\frac{e^x}{2}(1+\frac{2\sin 2x+\cos 2x}{5})-0{,}6$
18	[1; e]	$(x\cdot\ln x)^2$	0,000001	$\frac{x^3}{27}(9\ln^2 x-6\ln x+2)-\frac{2}{27}$
19	[0; 3	$\arcsin\sqrt{\frac{x}{1+x}}$	0,000001	$x\cdot\arcsin\sqrt{\frac{x}{1+x}}-\sqrt{x}+arctg\sqrt{x}$

Окончание таблицы 6

1	2	3	4	5
20	[0; 1]	$\frac{x^2-1}{(x^2+1)\sqrt{x^4+1}}$	0,000001	$-\frac{\sqrt{2}}{2}\arcsin(\frac{\sin 2arctgx}{\sqrt{2}}$
21	[0;1,5]	$\frac{e^x(1+\sin x)}{1+\cos x}$	0,000001	$e^x tg\frac{x}{2}$
22	[0;1]	$\frac{1}{(3\sin x+2\cos x)^2}$	0,000001	$\frac{3}{26}-\frac{3\cos x-2\sin x}{13(2\cos x+3\sin x)}$
23	[1;2]	$\frac{x}{x^4+3x^2+2}$	0,000001	$\frac{1}{2}\ln\frac{x^2+1}{x^2+2}-\frac{1}{2}\ln\frac{2}{3}$
24	[1;2]	$\frac{x^3}{3+x}$	0,000001	$9x-\frac{3x^2}{2}+\frac{x^3}{3}-27\ln(3+x)-$ $-\frac{47}{6}+27\ln 4$
25	[1;1,5]	$\sin x\cdot\ln(\operatorname{tg} x)$	0,000001	$\ln(tg\frac{x}{2})-\cos x(\ln(tgx))-$ $-\ln(tg0{,}5)+(\cos 1)\ln(tg1)$

4.2 Решение алгебраических и трансцендентных уравнений

В практических вычислениях часто приходится решать уравнение вида: f(x) = 0, где функция f(x) определена и непрерывна на некотором конечном или бесконечном интервале a < x < b.

Задача приближенного отыскания корней разделяется на два этапа.

Первый этап – отделение корней. Заключается в отыскании достаточно малых областей, в каждой из которых заключен один и только один корень уравнения. Как это рекомендуется делать, рассмотрено в последующих разделах.

Второй этап – уточнение значений отделенных корней с заданной точностью путем использования выбранного численного метода. Назовем методы, которые мы здесь будем рассматривать:

- метод простых итераций;
- метод дихотомии:
- метод Ньютона.

Методы, требующие проверки на сходимость вычислительного процесса, перед их программированием должны быть проверены на пригодность к решению конкретного уравнения.

4.2.1 Метод простых итераций решения нелинейного уравнения *f(x)=0*

Метод простых итераций (иначе метод последовательных приближений) решения уравнения $f(x)=0$ состоит в замене исходного уравнения эквивалентным ему уравнением $x=\varphi(x)$ и построении последовательности $x_{n+1}=\varphi(x_n)$, сходящейся при $n\to\infty$ к точному решению.

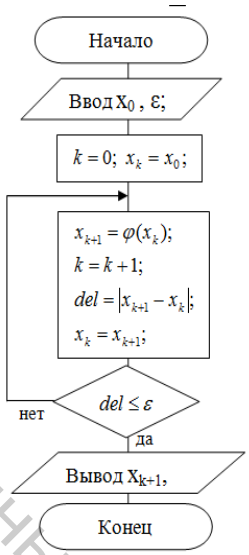


Рисунок 12 – Схема алгоритма метода простых итераций

Руководствуясь схемой, разрабатываем программу для метода простых итераций, работающую в среде Delphi в консольном режиме:

```
program MPI_1; // Метод простых итераций
{$APPTYPE CONSOLE}
uses  SysUtils, Math;
var x0, x, y, del, eps: extended; k:Integer;
Function Fi(x:extended):extended;
  begin Fi:=0.1*exp(x); end; // правая часть формулы метода
begin
    Writeln('Metod prostyh iteracij');
    Write('Vvedite xo, eps:  '); Readln(x0, eps);
    k:=0; x:=x0;
    Repeat y:=Fi(x); k:=k+1;
      del:=abs(y-x); x:=y;
      Writeln(y:12:9); // вывод текущего уточнения
      Until del <= eps;
      Writeln('    Koren  raven:  ',y:12:9);  //  вывод
результата
      Writeln('  Vypolneno ', K, ' iteracij'); Readln;
end.
```

Пример. Найти корни уравнения $e^x - 10x = 0$ с точностью $\varepsilon = 0{,}0001$.

Выполним **этап отделения корней** заданного уравнения. Теория вопроса указывает на три способа: аналитический, графический и табулирование заданной функции. Выбор зависит как от уравнения, так и от наличия соответствующего инструментария. Удобнее всего сочетание названных способов.

Можно использовать при анализе калькулятор с функциями. Мы выполним этот этап сначала в среде Excel, а затем – в среде Maple. Решить вопрос можно в консольном режиме Delphi, используя цикл табулирования функции.

Отделение корней в среде Excel. Выберем произвольный интервал и протабулируем на нем функцию $y = e^x - 10x$ для заданного уравнения $e^x - 10x = 0$. Получим:

Таблица 7 – Табулирование функции $y = e^x - 10$

x	-10	-8	-6	-4	-2	0	2	4	6	8	10
y	100	80	60	40,018	20,135	1	-12,61	14,6	343,43	2901	21926

Видим, что отрицательных корней нет. Корни есть там, где функция меняет знак. Это происходит на отрезке [0, 4]. Построим вторую таблицу, уменьшив шаг:

x	0	0,5	1	1,5	2	2,5	3	3,5	4
y	1	-3,351	-7,28	-10,52	-12,61	-12,82	-9,914	-1,885	14,598

Теперь можем сказать, что один корень принадлежит отрезку [0; 1], а другой корень – отрезку [3,5; 4]

Покажем функцию y(x) на графике, для чего построим точечную диаграмму для второй рассчитанной таблицы. Получим график, приведенный ниже по тексту. Отметим, что пользуясь графиком, мы можем точнее назвать интервалы отделения искомых корней. Однако оставим их такими же.

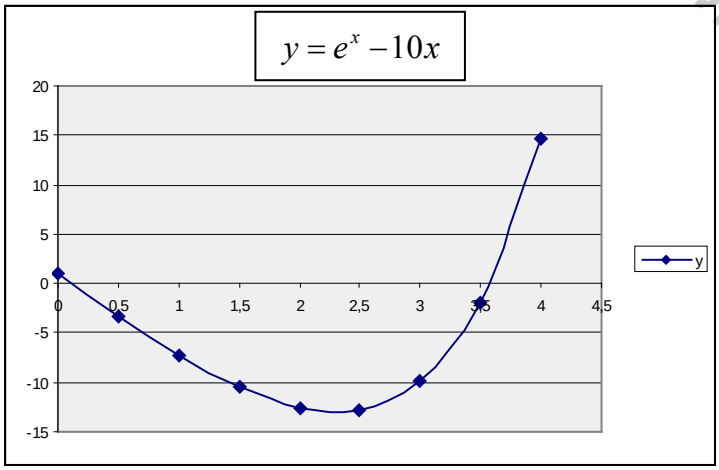


Рисунок 13 – График отделения корней в среде MS Excel

Отделение корней в среде Maple. Для работы запустим классическую версию пакета. Зададим процедуру построения графика:

```
> plot(exp(x)-10*x, x=-0.2..4);
```

Покажем рисунок графика и корни ξ_1 *и* ξ_2 на нем:

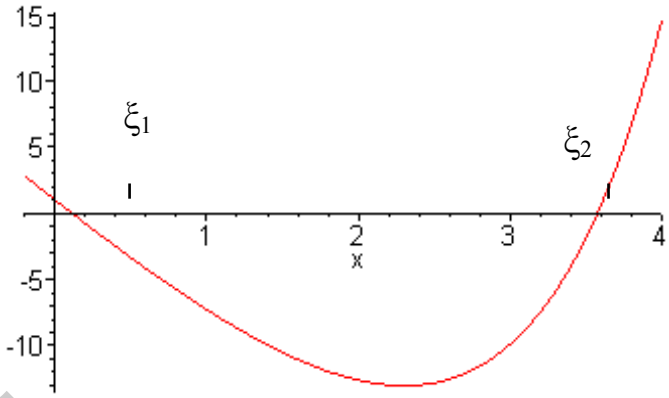


Рисунок 14 – Графика и корни ξ_1 *и* ξ_2 на нем

Обращаем внимание на то, что в Maple для построения задаем не таблицу, а саму функцию. Здесь проще варьировать пределами диапазона на оси x, повторно запуская процедуру `plot`.

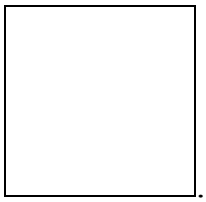
Этап уточнения корней

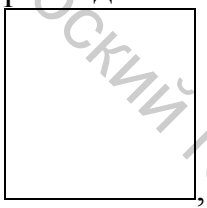
Подготовим исходное уравнение к итерационному процессу, преобразовав его к виду: $x = 0{,}1 \cdot e^x$. Здесь $\varphi(x) = 0{,}1e^x$, $\varphi'(x) = 0{,}1e^x$. На отрезке [0; 1] $0 < \varphi'(x) < 1$, то есть $q = 0{,}1$. В качестве начального приближения выбираем $x_0 = 1$, т. е. правую границу отрезка, содержащего первый корень ξ_1. Вычисления прекращаем, когда $|x_n - \varphi(x_n)| < \varepsilon$. Последовательные приближения в этом случае таковы:

0,271828 → 0,131236 → 0,114024 → 0,112078 → 0,111860 → 0,111836.

```
Metod prostyh iteracij
Vvedite xo, eps:  1 1e-5
 0.271828183
 0.131236150
 0.114023702
 0.112077869
 0.111859996
 0.111835628
 0.111832902
  Koren raven:  0.111832902
  Vypolneno 7 iteracij
```

Рисунок 15 – Рабочее окно программы с вычислением первого корня и поэтапным выводом результатов итераций

Так как $|\xi - x_6| \le \frac{0,1}{1-0,1} \cdot |x_6 - x_5| \le \frac{0,00004}{0,9} \le 0,000005$, то принимаем  . В этом результате все знаки верные.

В определении второго корня есть препятствие: $\varphi'(x_0) > 1$. Метод расходится. Поэтому преобразуем исходное уравнение к виду: $x = \ln 10x$. Здесь  , $\varphi'(x) = \frac{1}{x}$ и при x>2 производная $\varphi'(x)$ оценивается сверху: $|\varphi'(x)| = |1/x| \le 0,5$, *т.е.* $q = 0,5$ Если в качестве начального приближения корня ξ_2 взять $x_0 = 2$, то получаем следующие последовательные приближения: 2,99573 → 3,39977→3,52629→3,56283→

→ 3,57314 → 3,57603 → 3,57684 → 3,57706 → 3,57713.

Чтобы получить нужный результат, заменяем в `Function Fi` текст

`begin Fi:=0.1*exp(x); end;` на `begin Fi:=Ln(10*x); end;`

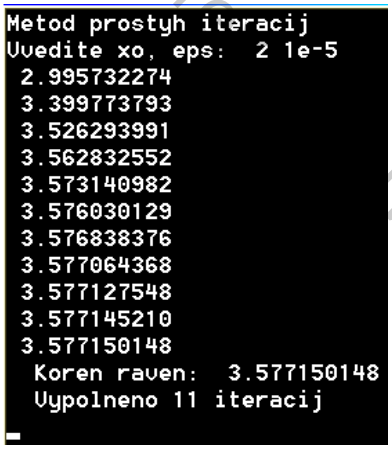


Рисунок 16 – Рабочее окно программы с вычислением второго корня

Принимаем $\xi_2 = 3,5771$ с погрешностью 0,0001, так как

$|\xi_2 - x_0| \le \frac{q}{1-q}|x_9 - x_8| \le |x_9 - x_8| \le 0,0001.$

ЗАДАНИЕ

Реализуйте программу для рассматриваемого здесь примера в консольном режиме Delphi. Получив результаты, модернизируйте программу:

1) функцию φ(x) вычисляйте в подпрограмме-функции;

2) в основной программе оставьте ввод исходных данных (как на схеме), реализацию метода и вывод результатов с форматированием;

3) решите задачу для заданного преподавателем варианта методом простых итераций.

4) отобразить решение примеров средствами Microsoft Excel и Maple.

Содержание отчета:

– название темы;

– номер варианта и условие задачи;

– демонстрация графика с отделением корней уравнения;

– программа в консольном режиме Delphi для своего варианта согласно выданному заданию;

– выводы.

Решить нелинейное уравнение f(x) = 0.

Таблица 8 – Варианты заданий

Вариант	Уравнение f(x) = 0	Вариант	Уравнение f(x) = 0
1 и	$3\sin\sqrt{x}+0{,}35x-3{,}8=0$	16 и	$1-x+\sin x-\ln(1+x)=0$
2 Н	$0{,}25x^3+x-1{,}2502=0$	17 Н	$3x-14+e^x-e^{-x}=0$
3 пд	$x+\sqrt{x}+\sqrt[3]{x}-2{,}5=0$	18 пд	$\sqrt{1-x}-tgx=0$
4 и	$x(3+\sin 3{,}6x)-1=0$	19 и	$x+\cos(x^{0{,}52}+2)=0$
5 Н	$0{,}1x^2-x\ln x=0$	20 Н	$3\ln^2 x+6\ln x-5=0$
6 пд	$tgx-\frac{1}{3}tg^3x+\frac{1}{5}tg^5x-\frac{1}{3}=0$	21 пд	$\sin x^2+\cos x^2-10x=0$
7 и	$\arccos x-\sqrt{1-0{,}3x^3}=0$	22 и	$x^2-\ln(1+x)-3=0$
8 Н	$3x-4\ln x-5=0$	23 Н	$2x\sin x-\cos x=0$
9 пд	$\cos\frac{2}{x}-2\sin\frac{1}{x}+\frac{1}{x}=0$	24 пд	$e^x+\sqrt{1+e^{2x}}-2=0$
10 и	$\sqrt{1-0{,}4x^2}-\arcsin x=0$	25 и	$\ln x-x+1{,}8=0$
11 Н	$e^x-e^{-x}-2=0$	26 Н	$xtgx-\frac{1}{3}=0$
12 пд	$\sin(\ln x)-\cos(\ln x)+2\ln x=0$	27 пд	$tg\frac{x}{2}-ctg\frac{x}{2}+x=0$
13 и	$x-2+\sin\frac{1}{x}=0$	28 и	$0{,}4+arctg\sqrt{x}-x=0$
14 Н	$e^x+\ln x-10x=0$	29 Н	$\sqrt{1-x}-\cos\sqrt{1-x}=0$
15 пд	$\cos x-e^{-\frac{x^2}{2}}+x-1=0$	30 пд	$0{,}6\cdot 3x-2{,}3x-3=0$

Примечание: и – метод простых итераций;
Н – метод Ньютона;
пд – метод половинного деления.

4.2.2 Приближенное решение уравнения f(x) = 0 методом деления пополам (метод бисекций)

Пусть задана непрерывная функция f(x) и требуется найти корень уравнения f(x) = 0. Предположим, что найдем отрезок [a,b] такой, что f(a)×f(b)<0. Тогда согласно теореме Больцано – Коши внутри отрезка [a,b] существует точка c, в которой значение функции равно нулю, т. е. $f(c) = 0,\ c \in [a,b]$. Итерационный метод бисекций состоит в построении последовательности вложенных отрезков

$$\{[a_n,b_n] \mid [a_n,b_n] \subset [a_{n-1},b_{n-1}] \subset [a,b]\},$$

на концах которых функция принимает значения разных знаков. Каждый последующий отрезок получают делением пополам предыдущего. Процесс построения последовательности отрезков позволяет найти нуль функции f(x) (корень уравнения f(x) = 0) с любой заданной точностью.

Если на отрезке [a;b] находятся несколько корней уравнения f(x) = 0, то процесс сходится к одному из них. Метод не применим к отысканию кратных корней четного порядка. В случае кратных корней нечетного порядка он менее точен.

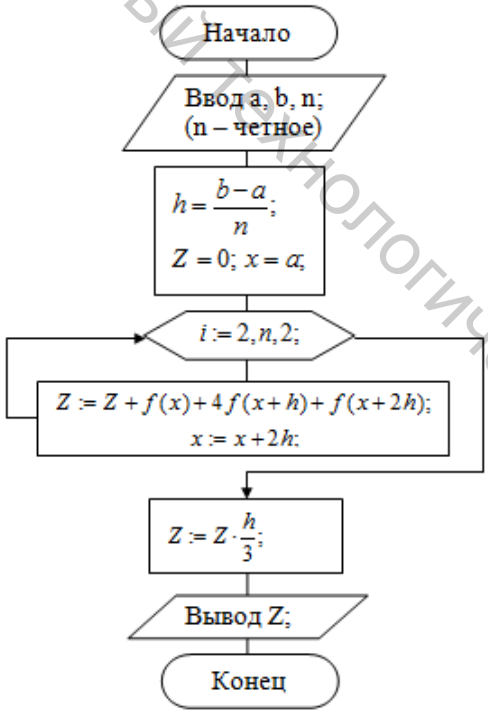


Рисунок 17 – Схема алгоритма

В библиотеке математических функций Math языка Delphi есть функция Sign(x) вычисления знака или нуля типа NValueSign (–1..+1); которая равна +1, если x>0, равна –1, если x<0 и равна нулю, если x=0. Подобная функция использована в схеме алгоритма.

Пример. Найти отличный от нуля корень уравнения $x^2 - 5 \cdot \sin x = 0$ с четырьмя верными знаками после запятой. Искомый корень легко отделяется на отрезке [1.57; 3,14]. Чтобы найти корень с тремя верными знаками после запятой, принимаем $\varepsilon = 0{,}0005$. Программа в консольном режиме Delphi может иметь вид:

```
program Dichot_Plis;
{}
{$APPTYPE CONSOLE}
uses  SysUtils, Math;
var a,b,eps,x0,y,r,c1:Extended; k:Integer;
function F(x:Extended):Extended;
  begin F:=x*x-5*sin(x); end; // задаем решаемое уравнение
procedure Dichot(a1,b1,eps1:extended; var x01:Extended);
  Label 245,542;
  begin
    k:=0; c1:=1;
    245:x01:=(a1+b1)/2; k:=k+1;
    y:=F(x01);
    if (y=0) or ((b1-a1) < eps) then goto 542
      else
        begin
          y:=sign(y)*c1;
          r:=F(a1); r:=sign(r)*c1;
          if r*y<0 then b1:=x0 else a1:=x0;
          goto 245;
        end;
    542:end;
begin // главная программа
  Write('Vvedite a,b,eps: ');
  Readln(a,b,eps);
  Dichot(a,b,eps,x0);
  writeln;
  Writeln('   Rezultaty:');
  writeln('Koren raven: ',x0:10:6);
  Readln;
end.
```

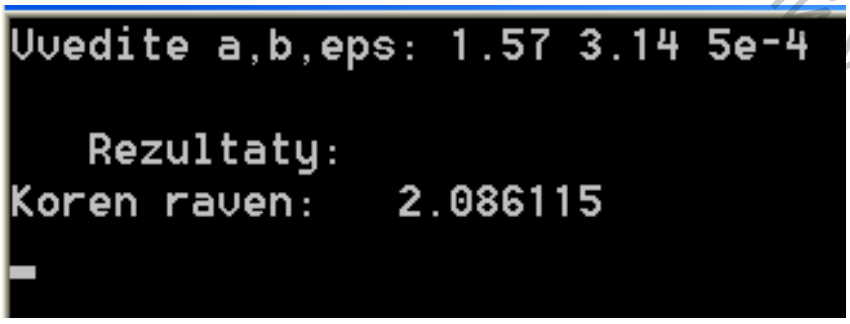


Рисунок 18 – Окно программы с вычислениями

Получить вариант задания от преподавателя. Выполнить работу согласно расписанному ниже перечню пунктов 1–4.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Порядок выполнения работы

1. Отделить корни уравнения f(x) = 0, т. е. найти для каждого корня отрезок (по возможности наименьший), на котором функция f(x) удовлетворяет условиям Больцано – Коши.
2. Разработать в консольном режиме Delphi программу, содержащую внутреннюю процедуру реализации метода и подпрограмму-функцию вычисления f(x).
3. Произвести вычисление по программе для заданного варианта.
4. Отобразить решение примеров средствами Microsoft Excel.
5. Оформить отчет с решением примеров в среде MS Excel.
6. Оформить отчет по работе.

Таблица 9 – Варианты заданий

Вариант	f(x)	a	b	c	d
1	Найти наименьший положительный корень уравнения f(x) = 0, где $f(x) = tg\,ax - bx$	0,6319	0,9217	-	-
2		9,4837	13,8249	-	-
3		0,9464	1,3825	-	-
4		8,5174	12,4424	-	-
5		1,8927	2,7650	-	-
6		4,4161	6,4516	-	-
7	Найти больший корень уравнения f(x) = 0, где $f(x) = \ln(ax) - bx + c$	0,3049	0,3436	1,5	-
8		9,1461	3,3081	12,0	-
9		0,6098	0,6872	1,5	-
10		8,5366	2,6209	2,0	-
11		0,9146	1,0308	2,5	-
12		7,9268	1,9337	3,0	-
13	Найти наименьший положительный корень уравнения f(x) = 0, где $a \cdot \sin bx - cx$	0,33	2,3	0,5	-
14		10	7,375	7,75	-
15		1	2,2	1	-
16		6,3	5,189	5	-
17		1,67	2,5	1,5	-
18		8	6,18	6,25	-
19	Решить уравнение f(x) = 0, где $f(x) = a \cdot e^{-bx} - x$	0,312	0,7586	-	-
20		0,893	0,52	-	-
21		9,385	0,963	-	-
22		1,44	0,98	-	-
23		0,25	0,8	-	-
24		6,7	0,6	-	-
25		0,5	0,667	-	-
26		12	2	-	-
27		2,1	0,0143	-	-
28	Найти корни уравнения f(x) = 0, где $f(x) = ax^3 + bx^2 + cx + d$	2,113	-6,44	-3,19	15,13
29		3,91	-11,79	-1,56	18,67
30		1,203	-3,53	-1,36	7,11

4.2.3 Приближенное решение нелинейного уравнения *f(x) = 0* методом Ньютона

Если известно хорошее начальное приближение решения уравнения $f(x)=0$, то эффективным методом повышения точности является метод Ньютона (метод касательных). Метод состоит в построении итерационной последовательности

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)},$$

сходящейся к корню уравнения $f(x)=0$. Сформулируем достаточные условия сходимости метода.

Теорема. Пусть f(x) определена и дважды дифференцируема на [a,b], причем $f(a)\cdot f(b)<0$, а производные $f'(x)$ и $f''(x)$ сохраняют знак на отрезке [a,b]. Тогда, исходя из начального приближения $x_0 \in |a,b|$, удовлетворяющего неравенству $f'(x_0)\cdot f''(x_0)>0$, можно построить последовательность $x_{n+1} = x_n - f(x_n)/f'(x_n), n = 0, 1, 2, \ldots$, сходящуюся к единственному на [a,b] решению ξ уравнения $f(x)=0$.

Схема алгоритма метода Ньютона изображена ниже.

Метод Ньютона эффективен, если известно хорошее начальное приближение для корня и в окрестности корня график функции имеет большую крутизну.

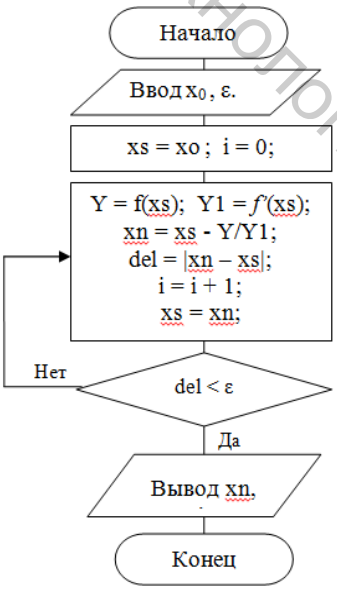


Рисунок 19 – Схема алгоритма метода Ньютона

Для приближенного вычисления корней уравнения $f(x)=0$ методом Ньютона рекомендуется разработать подпрограмму-процедуру.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Используя схему алгоритма метода Ньютона разработать программу приближенного вычисления корня уравнения $f(x)=0$ для заданной дважды дифференцируемой функции f(x) с точностью ε_0 и произвести вычисление. Программу разработать в виде Window – приложения на языке Delphi с построением формы интерфейса пользователя или в консольном режиме Delphi (по указанию преподавателя). Варианты – в конце темы.

Порядок выполнения работы

Напоминаем, что решение уравнений рассматриваемого здесь типа состоит из двух этапов. Первый этап – отделение корней. Второй этап – уточнение отделенного корня путем использования численного метода (у нас – метода Ньютона).

1. Используя табулирование функции, графику или аналитически отделить корень уравнения $f(x)=0$, то есть найти по возможности наименьший отрезок, содержащий искомый корень ($\xi \in [a,b]$).

Рекомендуется для отделения корня использовать построение графика функции $y=f(x)$ для уравнения $f(x)=0$ на произвольном интервале, а затем сузить интервал и определить отрезок, содержащий искомый корень.

2. Выбрать начальное приближение корня $x_0 \in [a,b]$ и проверить метод на сходимость (так, чтобы $f(x_0)\cdot f''(x_0)>0$).

3. Оценить снизу величину $m=\min_{[a,b]}|f'(x)|$, оценить сверху величину $M_2=\max_{[a,b[}|f''(x)|$.

4. По заданному ε выбрать значение ε для условия окончания итерационного процесса, $\varepsilon=\sqrt{2m_1\varepsilon_0/M_2}$.

5. Составить головную программу, содержащую обращение к подпрограмме, реализующей метод Ньютона (как NEWTON) и вывод результатов на экран. Предусмотреть печать признака IH.

6. Провести вычисление по программе.

Пример. Найти корни уравнения $10x - e^x = 0$ с точностью $\varepsilon_0 = 0{,}0001$.

Корни уравнения легко отделяются графически. Сначала представим уравнение в виде $e^x = 10x$. Затем построим графики функций $y=10x$ и $y=e^x$. Искомые корни лежат на пересечении графиков этих функций. Первый корень лежит на отрезке [0,1], а второй – на отрезке [3,4]. Рекомендуем построить график функции в среде Excel или Maple.

Запишем функцию f(x) и ее две производные:

$f(x)=10x-e^x;\ f'(x)=10-e^x;\ f''=-e^x.$

Первая и вторая производные сохраняют знак на отрезке [0,1]. При этом

$$m_1=\min_{[0,1]}|f'(x)|=\min_{[0,1]}(10-e^x)>7;\quad M_2=\max_{[0,1]}|f''(x)|=\max_{[0,1]}e^x=e$$

Для требуемой точности $\varepsilon_0=0{,}0001$ в условии окончания процесса

$\varepsilon = \sqrt{2m_1\varepsilon_0 / M_2} = \sqrt{2 \cdot 7 \cdot 10^{-4} / e}$.

В качестве начального приближения x0 = 0, поскольку $f(0) \cdot f''(0) = 1 > 0.$

В результате вычислений получим:

```
корень X=0.111833, количество итераций=2
```

За приближенное значение корня, вычисленное за две итерации принимаем $x = 0.1118 \pm 0.0001$

Для получения этого же результата методом простых итераций потребовалось пять итераций.

Аналогично можно найти второй корень этого же уравнения. Оценивая параметры m_1 и M_2 на отрезке [3,4] и принимая в качестве начального приближения $x_0 = 3$, за четыре итерации получаем ξ_2 = 3,577. Решая задачу методом простых итераций, получают этот результат за девять итераций.

Решить уравнение $f(x) = 0$

Таблица 10 – Варианты заданий

Вариант	f(x_	a	b	c	d
1	2	3	4	5	6
1	Найти первый положительный корень уравнения f(x) = 0, где f(x) = $tg\,ax - bx$	1,2618	1,8433	-	-
2		2,5237	3,6868	-	-
3		3,47	5,0691	-	-
4		8,8328	12,903	-	-
5		7,571	11,06	-	-
6		5,6782	8,2949	-	-
7	Найти корни уравнения f(x) = 0, где f(x) = $\ln(ax) - bx + c$	1,2195	1,3744	0,5	-
8		2,7439	3,0924	1,0	-
9		3,6585	4,1232	1,5	-
10		4,2683	4,8104	2,0	-
11		5,7927	6,5284	2,5	-
12		7,3171	8,2402	3,0	-
13	Найти первый положительный корень уравнения f(x), где f(x) = $a \cdot \sin bx - cx$	2,33	2,857	2	-
14		4	3,8125	3,25	-
15		5,33	4,59	4,25	-
16		6	4,99	4,75	-
17		7	5,5857	5,5	-
18		9,667	7,176	7,5	-
19	Найти корень уравнения f(x) = 0, где f(x) = $a \cdot e^{-bx} - x$	0,0714	0,933	-	-
20		0,3889	0,72	-	-
21		0,5476	0,6462	-	-
22		0,6304	0,6133	-	-

Окончание таблицы 10

1	2	3	4	5	6
23		0,7	0,5882	-	-
24		0,8103	0,5524	-	-
25		0,875	0,533	-	-
26		0,9118	0,5231	-	-
27		0,9595	0,5103	-	-
28	Найти корни в f(x) = 0, где f(x) = $ax^3 + bx^2 + cx + d$	2,113	-6,44	-3,19	15,13
29		3,91	-11,79	-1,56	18,67
30		1,203	-3,53	-1,36	7,11

ЛИТЕРАТУРА

1. Абчук, В. Н. Экономико-математическое моделирование / В. Н. Абчук. – СПб., 1999. – 310 с.

2. Балашевич, В. Н. Математические методы в управлении производством / В. Н. Балашевич. – Минск, 1995. – 334 с.

3. Голицына, О. Л. Информационные технологии : учебник / О. Л. Голицына [и др.] – 2-е изд., перераб. и доп. – Москва : ФОРУМ : ИНФРА-М, 2013. – 607 с.

4. Затонский, А. В. Информационные технологии. Разработка информационных моделей и систем : учеб. пособие / А. В. Затонский. – Москва : РИОР : ИНФРА-М, 2014. – 343 с.

5. Колеснёв, В. И. Экономико-математические методы и модели в материально-техническом обеспечении АПК: сборник задач : учеб. пособие / В. И. Колеснёв. – 2-е изд., исправл. – Минск : Дикта, 2012. – 208 с.

6. Коноплева, И. А. Информационные технологии : учеб. пособие / И. А. Коноплева, О. А. Хохлова, А. В. Денисов. – 2-е изд. – Москва : Проспект, 2015. – 327 с.

7. Миксюк, С. Ф. Экономико-математические методы и модели : учебно-практическое пособие / С. Ф. Миксюк, В. Н. Комкова. – Минск : БГЭУ, 2006. – 219 с.

8. Мур, Дж. Экономическое моделирование в Microsoft Excel : пер. с англ. / Дж. Мур, Л. Уэдерфорд. – 6-е изд. – Москва : Издательский дом «Вильямс», 2004. – 1024 с.

9. Орлов, А. И. Эконометрика / А. И. Орлов. – Москва : Экзамен, 2003. – 576 с.

10. Поттосина, С. А. Экономико-математические модели и методы: учеб. пособие / С. А. Поттосина, В. А. Журавлев. – Минск : БГУИР, 2003. – 94 с.

11. Похабов, В. И. Экономико-математические методы и модели : практикум / В. И. Похабов. – Минск : БНТУ, 2003. – 130 с.

12. Терентьев, В. П. «Информатика, численные методы и компьютерная графика»: методические указания. В 2 ч. Ч. 2 / В. П. Терентьев [и др.]. – Витебск, 2005.

13. Шушкевич, Г. Ч. Компьютерные технологии в математике. Система Mathcad 14 : учебное пособие : в 2 ч. / Г. Ч. Шушкевич, С. В. Шушкевич. – Минск : Издательство Гревцова, 2010. – 287 с.

14. Щербакова, Е. Н. Информационные технологии нано- и микросистемной техники : учебно-методическое пособие / Е. Н. Щербакова; Белорусский национальный технический университет, Кафедра «Микро- и нанотехника». – Минск : БНТУ, 2017.

15. Эконометрика и экономико-математические методы и модели : учеб. пособие / Г. О. Читая [и др.] ; под ред. Г. О. Читая, С. Ф. Миксюк. – Минск : БГЭУ, 2018. – 511 с.

16. Экономико-математические методы и модели. Компьютерные технологии решения : учеб. пособие / И. Л. Акулич [и др.]. – Минск : БГЭУ, 2003. – 348 с.

17. Экономико-математические методы и модели: учеб. пособие / Н. И. Холод [и др.] ; под. общ. ред. А. В. Кузнецова. – Минск : БГЭУ, 1999. – 413 с.

18. Юферева, О. Д. Экономико-математические методы / О. Д. Юферева. – Минск : БГЭУ, 2002. – 56 с.

Учебное издание

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Лабораторный практикум

Составители:
Стасеня Тамара Петровна
Мандрик Ольга Геннадьевна

Редактор *Т.А. Осипова*
Корректор *А.В. Пухальская*
Компьютерная верстка *Д.А. Королёва*

Подписано к печати 01.06.2022. Формат 60x90 $^{1}/_{8}$. Усл. печ. листов 6,3.
Уч.-изд. листов 4,0. Тираж 35 экз. Заказ № 144.

Учреждение образования «Витебский государственный технологический университет»
210038, г. Витебск, Московский пр., 72.
Отпечатано на ризографе учреждения образования
«Витебский государственный технологический университет».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 1/172 от 12 февраля 2014 г.
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 3/1497 от 30 мая 2017 г.