Федеральное агентство по образованию РФ

Государственное образовательное учреждение высшего профессионального образования

Белгородский Государственный Технологический Университет

им. В.Г. Шухова

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №6**

**по дисциплине «Вычислительная математика»**

***Одномерная минимизация функции***

Выполнил: студент группы ПВ-21

Браткова Ирина Олеговна

Проверил: Бондаренко Т.В.

Белгород 2017г.

**Цель работы**: изучить методы нахождения приближенного решения задачи одномерной минимизации функциии получить практические навыки их применения.

**Задания к работе:**

* + Найти область определения заданной функции *у* =*f*(*x*) и построить её график, используя равномерную сетку значений *хi* (шаг сетки выбрать самостоятельно).
  + Найти промежутки унимодальности функции *у* = *f*(*x*), используя график.
  + Найти первую *y’=f’*(*x*) и вторую*y’’=f’’*(*x*) производные заданной функции *у*=*f*(*x*).
  + Найти точное решение задачи одномерной минимизации ― минимум функции *у*=*f*(*x*), .
  + Найти приближенное решение задачи одномерной минимизации ― вручную, используя численные методы одномерной минимизации:
  + метод оптимального поиска;
  + метод, основанный на использовании чисел Фибоначчи;
  + метод деления отрезка пополам.

С точностью е=0,01.

Необходимые параметры методов выбрать самостоятельно.

Подробно достаточно выполнить только первый шаг численного метода решения, окончательный результат может быть с помощью приложения MSExcel или составленной программы.

* Определить абсолютную Δ и относительную *δ* погрешность решения задачи одномерной минимизации для каждого из используемых численных методов. Представить полученные результаты в виде таблицы (см. табл. 1).

Таблица 1

Оценка погрешности численных методов одномерной минимизации

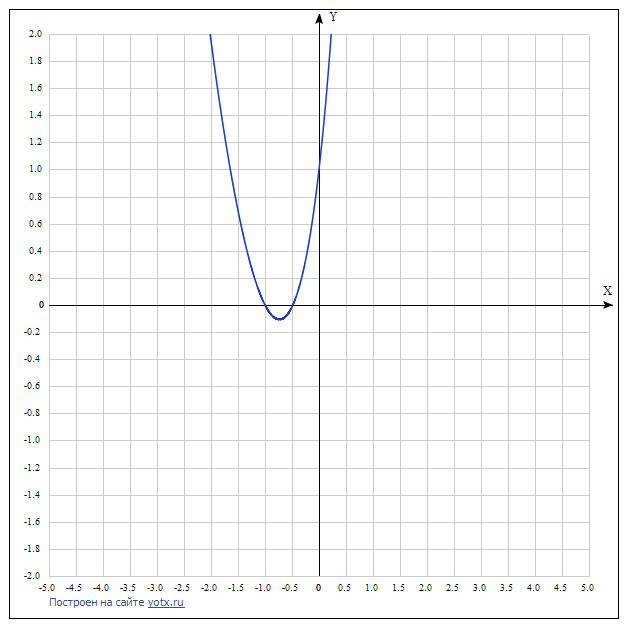
|  |  |  |  |
| --- | --- | --- | --- |
|  | Задача одномерной минимизации | | |
| Погрешность | метод оптимального поиска | метод чисел Фибоначчи | метод деления отрезка пополам |
| Δ |  |  |  |
| δ |  |  |  |

* Описать в модуле функции, которые возвращают приближенные значения минимума функции *у* =*f*(*x)* для заданного промежутка унимодальности *x*R с точностью *ε* каждым из рассмотренных численных методов.
* Составить программу для вычисления приближенного решения задачи одномерной минимизации с использованием функций, описанных в модуле.

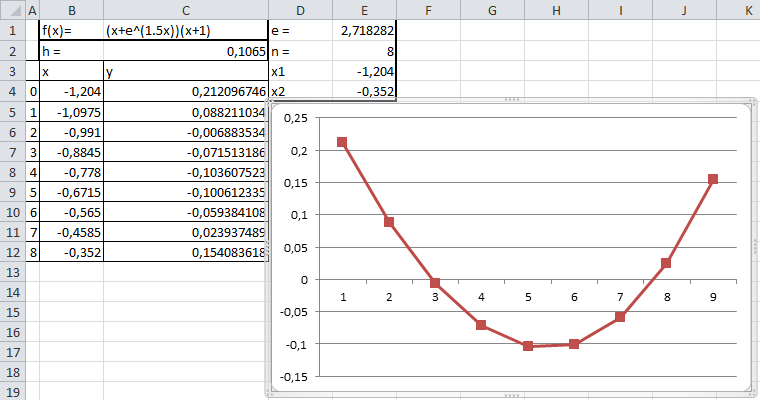
Функция: 

1.Область определения функции:

График функции:



2. Промежуток унимодальной функции: .



Производные и точное значение функции:

Первая производная:



Вторая производная:



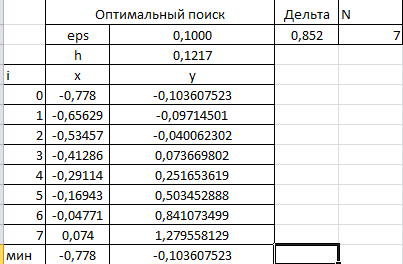
Точное значение:



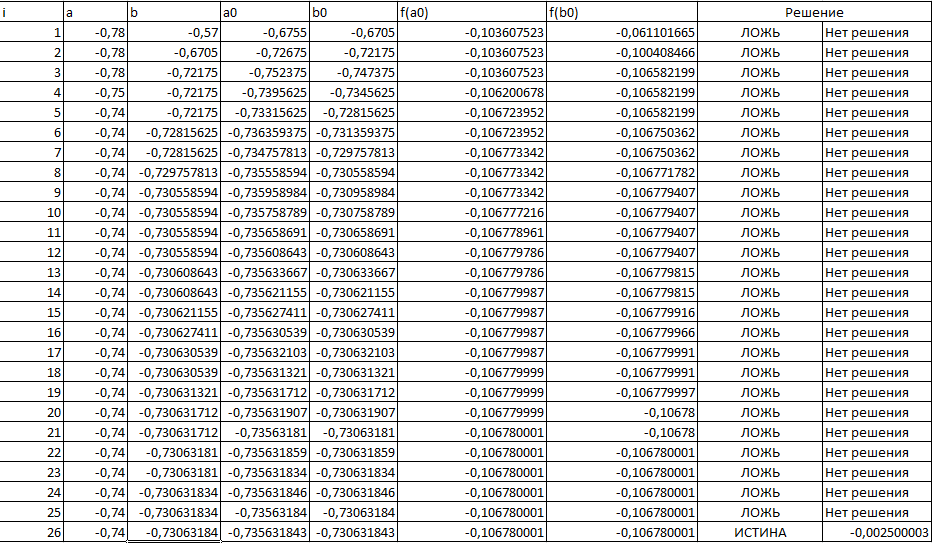
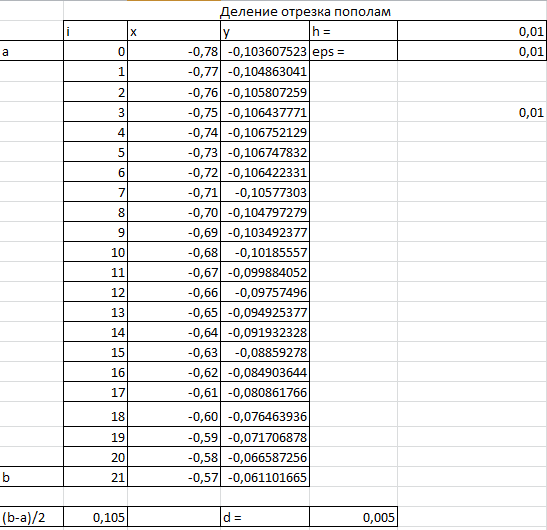
X Y

|  |  |
| --- | --- |
| -0,778 | -0,103607523 |

**Метод оптимального поиска**



**Метод деления отрезка пополам**



|  |  |  |  |
| --- | --- | --- | --- |
|  | Задача одномерной минимизации | | |
| Погрешность | метод оптимального поиска | метод чисел Фибоначчи | метод деления отрезка пополам |
| Δ | 0 | 0,004023437 | 0,061904762 |
| δ | 0% | 0,002409244% | 0,03706872% |

Текст программы:

#include "stdafx.h"

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

#include <locale.h>

typedef float(\*function)(float x);

float mf(float x){

return (((x\*x)-4)\*(2\*(x\*x)-3));

}

float OptiSearch(function f, float A, float B, int N){

float xmin = A;

float fmin = f(A);

float h = (B - A) / (N - 1);

for (int i = 1; i < N; i++){

float xi = A + i\*h;

float fi = f(xi);

if (fi < fmin){

fmin = fi;

xmin = xi;

}

}

return xmin;

}

float Dichotomy(function f, float A, float B, float e){

float lb = e / 10;

float a = A, b = B;

while ((b - a) / 2 > e){

float x1 = (a + b) / 2 - lb;

float x2 = (a + b) / 2 + lb;

f(x1) <= f(x2) ? b = x2 : a = x1;

}

return (a + b) / 2;

}

//Вычисление N-го члена ряда Фибоначчи

int F(int n){

if (n == 0 || n == 1)

return n;

else

return F(n - 1) + F(n - 2);

}

float Fibonacci(function f, float A, float B, float e){

float lb = e / 10;

int N = 0;

float a = A, b = B;

while (F(N) < (B - A) / (2 \* e))

N++;

float x1 = a + (F(N - 2) / F(N))\*(b - a);

float x2 = a + (F(N - 1) / F(N))\*(b - a);

for (int k = 0; k < N - 3; k++){

if (f(x1) <= f(x2)){

b = x2;

x2 = x1;

x1 = a + (F(N - k - 3) / F(N - k - 1))\*(b - a);

}

else{

a = x1;

x1 = x2;

x2 = a + (F(N - k - 2) / F(N - k - 1))\*(b - a);

}

}

x2 = x1 + lb;

f(x1) <= f(x2) ? b = x2 : a = x1;

return(a + b) / 2;

}

int main(){

float A = 1.67, B = 1.87, Xmin1, Xmin2, Xmin3;

float e;

int N;

setlocale(LC\_ALL, "Russian");

printf("1)Метод оптимального поиска\n");

printf("Введите N - количество отрезков ");

scanf\_s("%i", &N);

Xmin1 = OptiSearch(mf, A, B, N);

printf("min(f(x)) примерно равно %f\n", Xmin1);

printf("\n2)Метод отрезка пополам\n");

printf("Введите e - критерий точности ");

scanf\_s("%f", &e);

Xmin2 = Dichotomy(mf, A, B, e);

printf("min(f(x)) примерно равно %f\n", Xmin2);

printf("\n3)Метод чисел Фибоначчи\n");

printf("Введите e - критерий точности ");

scanf\_s("%f", &e);

Xmin3 = Fibonacci(mf, A, B, e);

printf("min(f(x)) примерно равно %f\n", Xmin3);

system("PAUSE");

return 0;

}

