



**Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)» (МГТУ  
им. Н.Э. Баумана)**

---

**ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)**

**КАФЕДРА «Информационная безопасность» (ИУ8)**

**Отчёт**

**по лабораторной работе № 1**

**по дисциплине «Теория систем и системный анализ»**

**Тема: «Исследование методов прямого поиска экстремума унимодальной функции  
одного переменного»**

**Вариант 3**

**Выполнила: Бакаев Ф.Б.,  
студент группы ИУ8-31**

**Проверила: Коннова Н.С.,  
доцент каф. ИУ8**

**г. Москва, 2020 г.**

## Цель работы

Исследовать функционирование и провести сравнительный анализ различных алгоритмов прямого поиска экстремума (пассивный поиск, метод дихотомии, золотого сечения, Фибоначчи) на примере унимодальной функции одного переменного.

## Условие задачи

На интервале  $[0,3]$  задана унимодальная функция одного переменного  $f(x) = -\sqrt{x} \cdot \sin(x)$ . Используя метод Фибоначчи, найти интервал нахождения минимума  $f(x)$  при заданном числе точек ( $N = 29$ ). Провести сравнение с методом оптимального пассивного поиска. Результат, в зависимости от числа точек разбиения  $N$ , представить в виде таблицы.

## График заданной функции

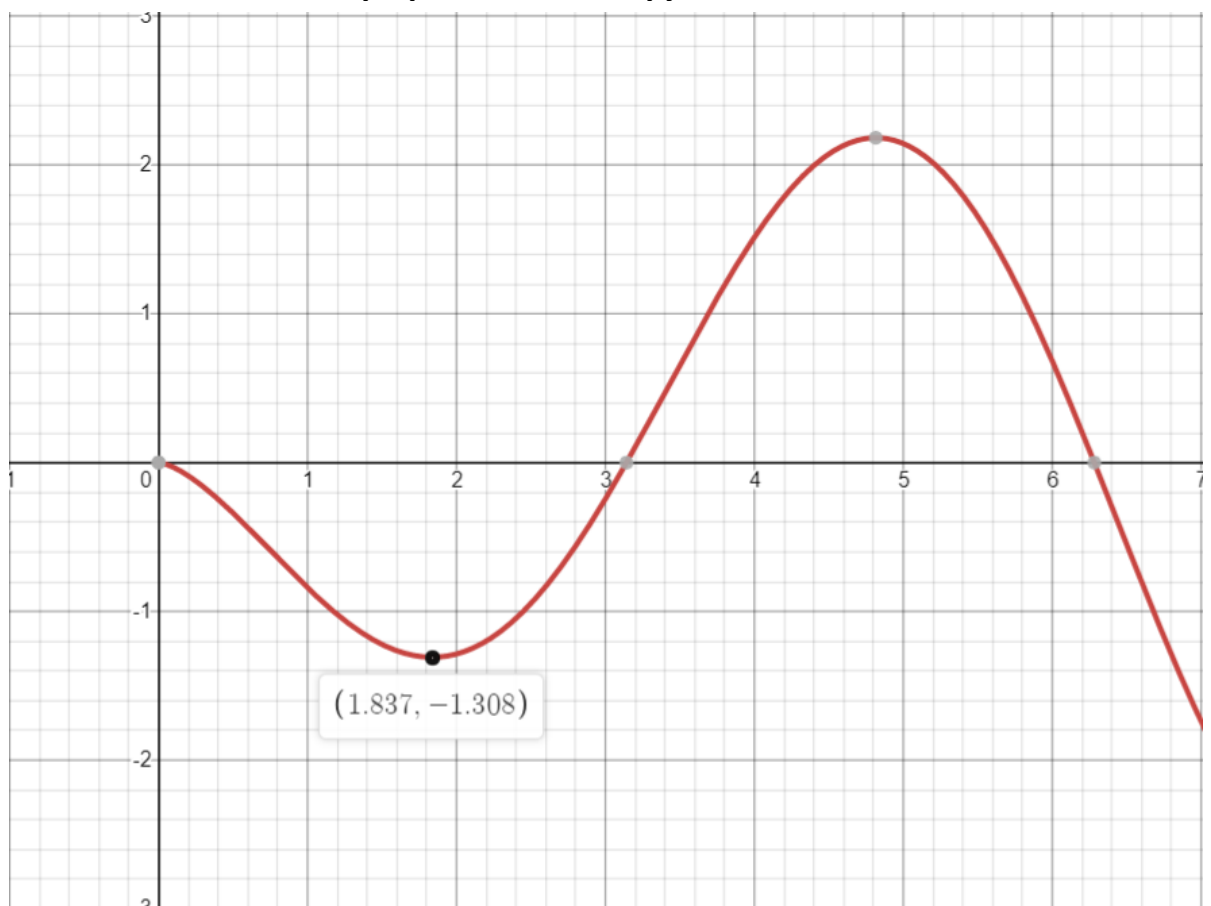


Рисунок 1 - График функции  $y = -\sqrt{x} \cdot \sin(x)$  на  $[0;3]$

Part 1. Optimal passive search:

-----					
	Количество точек (N)		Значение x в минимуме		Погрешность
-----					
	1		1.5		1.5
	2		2		1
	3		1.5		0.75
	4		1.8		0.6

	5		2		0.5	
	6		1.7142857143		0.429	
	7		1.875		0.375	
	8		2		0.333	
	9		1.8		0.3	
	10		1.9090909091		0.273	
	11		1.75		0.25	
	12		1.8461538462		0.231	
	13		1.9285714286		0.214	
	14		1.8		0.2	
	15		1.875		0.188	
	16		1.7647058824		0.176	
	17		1.8333333333		0.167	
	18		1.8947368421		0.158	
	19		1.8		0.15	
	20		1.8571428571		0.143	
	21		1.7727272727		0.136	
	22		1.8260869565		0.13	
	23		1.875		0.125	
	24		1.8		0.12	
	25		1.8461538462		0.115	
	26		1.8888888889		0.111	
	27		1.8214285714		0.107	
	28		1.8620689655		0.103	
	29		1.8		0.1	

-----

1.8 +- 0.1

Part 2. Fibonacci search :

-----					
	Количество точек (N)		Значение x в точке		
-----					
	1		2.2917960675		
	2		1.583592135		
	3		2.0212862363		
	4		1.750776405		
	5		1.917960675		
	6		1.8146351138		

	7		1.8784938226	
	8		1.8390269701	
	9		1.8297101099	
	10		1.8447851061	
	11		1.8354682459	
	12		1.8332688341	
	13		1.8368275582	
	14		1.8376676578	
	15		1.8363083455	
	16		1.8371484451	
	17		1.8366292323	
	18		1.8365066714	
	19		1.8367049973	
	20		1.8365824363	
	21		1.8365534674	
	22		1.8366002634	
	23		1.8366114053	
	24		1.8365935782	
	25		1.8366047201	
	26		1.836598035	
	27		1.8365958067	
	28		1.8365980351	
	29		1.8365958068	

-----

## График зависимостей погрешности от числа точек N

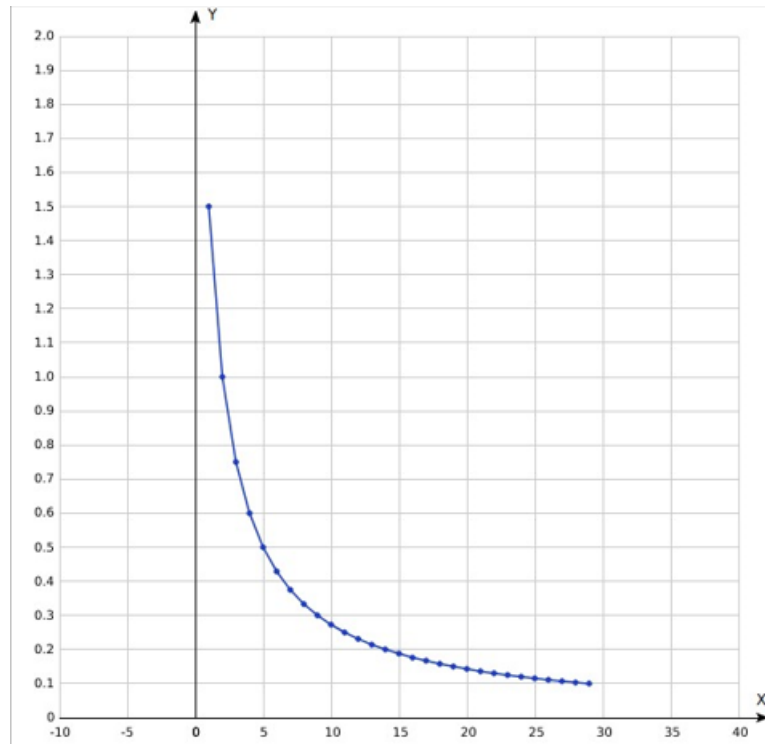


Рисунок 2 - График зависимости погрешности от числа точек  $N$  для оптимального пассивного поиска

## Выводы

Из полученных таблиц и графиков видно, что метод Фибоначчи значительно эффективнее метода пассивного поиска при отыскании экстремума унимодальной функции одного переменного.

### Приложение. Исходный код программы

```
#include <iostream>
#include <cmath>
#include <iomanip>
#include <vector>
#include <algorithm>

using std::cout;
using std::endl;

#define pogr 0.1

double f(const double& x) {
    return (-sqrt(x) * sin(x));
}

std::vector<std::pair<double, double>> opt_passive(const double& a, const double& b) {
    std::vector<std::pair<double, double>> values;
    size_t N = 1;
    double delta = (b - a) / (N + 1);
    double x_min_y;
    while (delta > pogr) {
        std::vector<double> vec_y;
        delta = (b - a) / (N + 1);
        for (size_t k = 1; k <= N; ++k) {
            vec_y.push_back(f((b - a) / (N + 1) * k + a));
        }
        size_t y_min_k = std::min_element(vec_y.begin(), vec_y.end()) - vec_y.begin() +
1;
        x_min_y = (b - a) / (N + 1) * y_min_k + a;
        values.push_back({ x_min_y, delta });
        N++;
    }
    return values;
}

unsigned int Fib(const size_t& n) {
    if (n < 1)
        return 0;
    unsigned int f1 = 0, f2 = 1, fn = 0;
    for (size_t i = 1; i < n; ++i) {
        fn = f1 + f2;
        f1 = f2;
        f2 = fn;
    }
    return fn;
}

std::vector<double> fib(size_t N, double& a, double& b) {
    std::vector<double> values;
    double x1 = a + (b - a) * Fib(N) / Fib(N + 2);
    double x2 = a + b - x1;
    double y1 = f(x1);
    double y2 = f(x2);
    while (N-- > 0) {
        if (y1 > y2) {
```

```

        a = x1;
        x1 = x2;
        x2 = b - (x1 - a);
        y1 = y2;
        y2 = f(x2);
        values.push_back(x2);
    }
    else {
        b = x2;
        x2 = x1;
        x1 = a + (b - x2);
        y2 = y1;
        y1 = f(x1);
        values.push_back(x1);
    }
}
return values;
}

void PrintValues(const std::vector<double>& values) {
    cout << std::string(45, '-') << endl;
    cout << std::setw(23) << std::left << "| Количество точек (N) " << "|";
    cout << std::setw(24) << std::left << " Значение x в точке |" << endl;
    cout << std::string(45, '-') << endl;

    for (size_t i = 0; i < values.size(); i++) {
        cout << "|";
        cout << std::setw(12) << std::right << i + 1;
        cout << std::setw(11) << "|";
        cout << std::setw(17) << std::right << std::setprecision(11) << values[i];
        cout << std::setw(4) << "|" << endl;
    }
    cout << std::string(45, '-') << endl;
    cout << values[7] << endl;
}

void PrintValues(const std::vector<std::pair<double, double>>& values) {
    cout << std::string(62, '-') << endl;
    cout << std::setw(23) << std::left << "| Количество точек (N) " << "|";
    cout << std::setw(23) << std::left << " Значение x в минимуме |";
    cout << std::setw(15) << std::left << " Погрешность |" << endl;
    cout << std::string(62, '-') << endl;

    int i = 0;
    double ans1, ans2;
    for (auto const& val : values)
    {
        cout << "|";
        cout << std::setw(12) << std::right << i + 1;
        cout << std::setw(11) << "|";
        cout << std::setw(18) << std::setprecision(11) << val.first << std::setw(6) <<
"|";
        cout << std::setw(9) << std::setprecision(3) << val.second;
        cout << std::setw(5) << "|" << endl;
        i++;
        ans1 = val.first;
        ans2 = val.second;
    }
    cout << std::string(62, '-') << endl;
    cout << ans1 << " +- " << ans2 << endl;
}

int main() {
    setlocale(LC_ALL, "Russian");

    size_t N = 29;

```

```

double a = 0.0, b = 3.0;

cout << "Вариант №3:" << endl;
cout << "Функция  $-\sqrt{x} * \sin(x)$  для интервала  $[0, 3]$ " << endl << endl;
cout << "Первый метод: метод оптимального пассивного поиска" << endl;
cout << "Для погрешности 0,1" << endl;
PrintValues(opt_passive(a, b));
cout << endl;
cout << "Второй метод: метод Фибоначчи" << endl;
cout << "N задается заранее" << endl;
cout << "При N=29 для точности 0,1 достаточно 7 итераций" << endl;
PrintValues(fib(N, a, b));
return 0;
}

```