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»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 1

по курсу «Методы оптимизации»

«Поиск минимума унимодальной функции»

Студент группы ИУ9-82Б Виленский С. Д.

Преподаватель Посевин Д. П.

Москва 2024

1 Задание

Определить интервал, на котором функция является унимодальной, алгоритм определения унимодальности должен принимать на вход левую и правую точку отрезка и возвращать false — если функция на этом отрезке не унимодальная, в противном случае true. Реализовать поиск минимума унимодальной функции на полученном интервале методом прямого перебора, дихотомии (деление отрезка пополам), золотого сечения и Фибоначчи с заданной точностью по вариантам. Результат должен быть представлен на графике, точки минимизирующей последовательности должны быть выделены красным цветом, интервалы деления синим. Точность вычисления точки минимума должна варьироваться.

2 Результаты

Исходный код программы представлен в листингах 1- 4.

Листинг 1 — Нахождение минимумов функции

```
using Plots

function derivativeAtPoint(f, x)
    delta = 1e-3
    return (f(x + delta) - f(x)) / delta
end

function secondDerivativeAtPoint(f, x)
    delta = 1e-3
    return (derivativeAtPoint(f, x + delta) - derivativeAtPoint(f, x)) /
     delta
end

function findUnimodalIntervals(f, a, b, step)
    intervals = []
    actual_interval_start = a

    for x in a:step:b
        if secondDerivativeAtPoint(f, x) <= 0
            if abs(actual_interval_start - x) > step * 2
                push!(intervals, (actual_interval_start, x))
            end
            actual_interval_start = x
        end
    end
    if abs(actual_interval_start - b) > step * 2
        push!(intervals, (actual_interval_start, b))
    end
```

Листинг 2 — Нахождение минимумов функции

```
    return intervals
end

function findExtrBySegments(f, a, b, step, eps)
    iters = []

    while abs(a - b) > eps
        x1 = a + (b - a) / 3
        x2 = a + (b - a) * 2 / 3

        push!(iters, (a, x1, x2, b))

        if f(x1) > f(x2)
            a = x1
        else
            b = x2
        end
    end

    return (a + b) / 2, iters
end

function findExtrByGoldRatio(f, a, b, step, eps)
    iters = []

    goldRatio = (5^.5 - 1) / 2
    x1 = a + (1 - goldRatio) * (b - a)
    x2 = a + goldRatio * (b - a)
    x = (a + b) / 2

    while abs(a - b) > eps
        push!(iters, (a, x1, x2, b))

        if f(x1) > f(x2)
            x = x2
            a = x1
            x1 = x2
            x2 = a + b - x2
        else
            x = x1
            b = x2
            x2 = x1
            x1 = a + b - x1
        end
    end

    return x, iters
end

function findExtrByFibbonachi(f, a, b, step, eps)
    iters = []

    fib1, fib2, fib3 = 0, 1, 1
    for i in 1:16
```

Листинг 3 — Нахождение минимумов функции

```

        fib1 = fib2
        fib2 = fib3
        fib3 = fib1 + fib2
    end
    x1 = a + (fib1 / fib3) * (b - a)
    x2 = a + b - x1
    x = (a + b) / 2

    while abs(a - b) > eps
        push!(iters, (a, x1, x2, b))

        if f(x1) > f(x2)
            x = x2
            a = x1
            x1 = x2
            x2 = a + b - x2
        else
            x = x1
            b = x2
            x2 = x1
            x1 = a + b - x1
        end
    end

    return x, iters
end

f = x -> x ^ 4 - 2 * x ^ 2 + 3

intervals = findUnimodalIntervals(f, -10, 10, 2e-6)

interval_index = 1
a = intervals[interval_index][1]
b = intervals[interval_index][2]
alg_step = 2e-3
alg_eps = 1e-3

X = range(a, b, step=alg_step)
Y = [f(x) for x in X]
Y_min = minimum(Y)
Y_max = maximum(Y)
prop_Y = c -> Y_max - (Y_max - Y_min) * c

plot(X, Y, legend=false, title="ExtrBySegments")

x0, iters = findExtrBySegments(f, a, b, alg_step, alg_eps)
println(length(iters))
for (i, iter) in enumerate(iters)
    y_val = prop_Y(i / length(iters))

    hline!([y_val], color="green")
    scatter!([(iter[2], y_val), (iter[3], y_val)], color="blue")
    scatter!([(iter[1], y_val), (iter[4], y_val)], color="red")
end
```

Листинг 4 — Нахождение минимумов функции

```

plot!()

X = range(a, b, step=alg_step)
Y = [f(x) for x in X]
Y_min = minimum(Y)
Y_max = maximum(Y)
prop_Y = c -> Y_max - (Y_max - Y_min) * c

plot(X, Y, legend=false, title="ExtrByGoldRatio")

x0, iters = findExtrByGoldRatio(f, a, b, alg_step, alg_eps)
println(length(iters))
for (i, iter) in enumerate(iters)
    y_val = prop_Y(i / length(iters))

    hline!([y_val], color="green")
    scatter!([(iter[2], y_val), (iter[3], y_val)], color="blue")
    scatter!([(iter[1], y_val), (iter[4], y_val)], color="red")
end

plot!()

X = range(a, b, step=alg_step)
Y = [f(x) for x in X]
Y_min = minimum(Y)
Y_max = maximum(Y)
prop_Y = c -> Y_max - (Y_max - Y_min) * c

plot(X, Y, legend=false, title="ExtrByFibbonachi")

x0, iters = findExtrByFibbonachi(f, a, b, alg_step, alg_eps)
println(length(iters))
for (i, iter) in enumerate(iters)
    y_val = prop_Y(i / length(iters))

    hline!([y_val], color="green")
    scatter!([(iter[2], y_val), (iter[3], y_val)], color="blue")
    scatter!([(iter[1], y_val), (iter[4], y_val)], color="red")
end

plot!()
```

Результат запуска представлен в листинге 5 и рисунках 1- 3.

Листинг 5 — Нахождение минимумов функции

```
(-10, -0.57835)
(0.57635, 10)

ExtrBySegments - 23
ExtrByGoldRatio - 20
ExtrByFibbonachi - 18
```

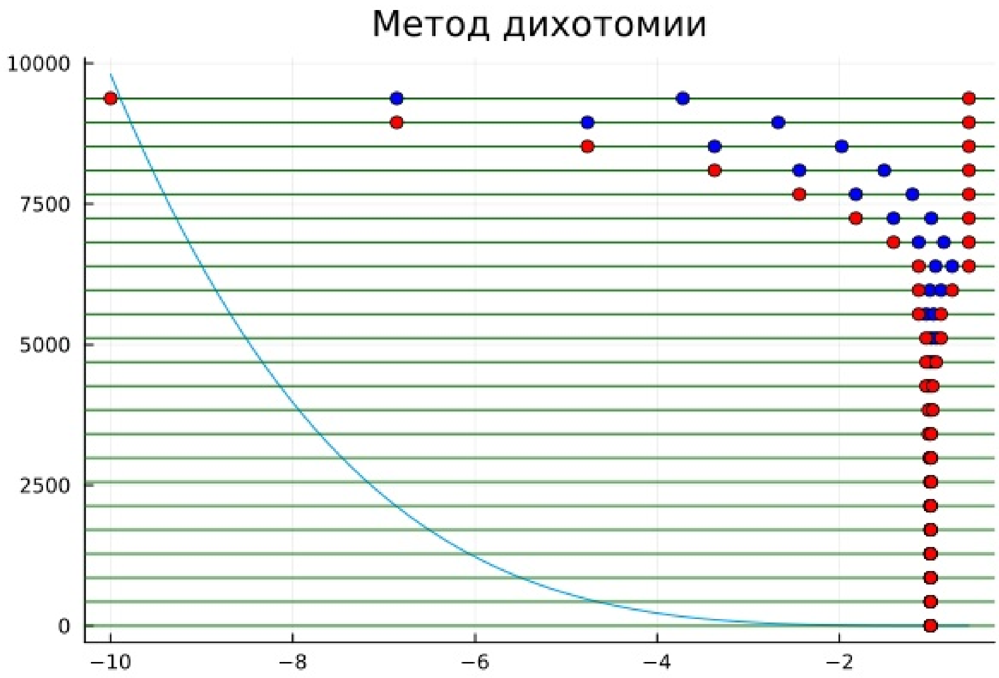

Рис. 1 — Результат работы алгоритма методом дихотомии

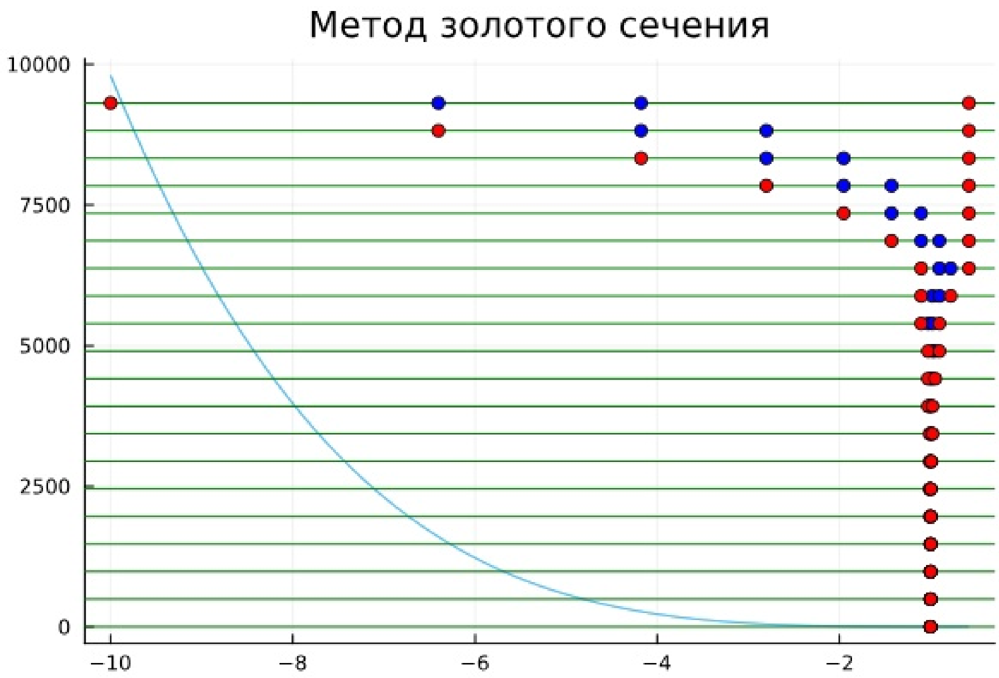

Рис. 2 — Результат работы алгоритма методом золотого сечения

3 Выводы

Исходя из результатов исследования можно сделать вывод о том, что метод Фибоначи является в данном случае наиболее подходящим, судя по скорости сходимости алгоритма, в то время как метод золотого сечения показал себя лучше метода схлодящимися отрезками, также известного как метод Дихотомии.

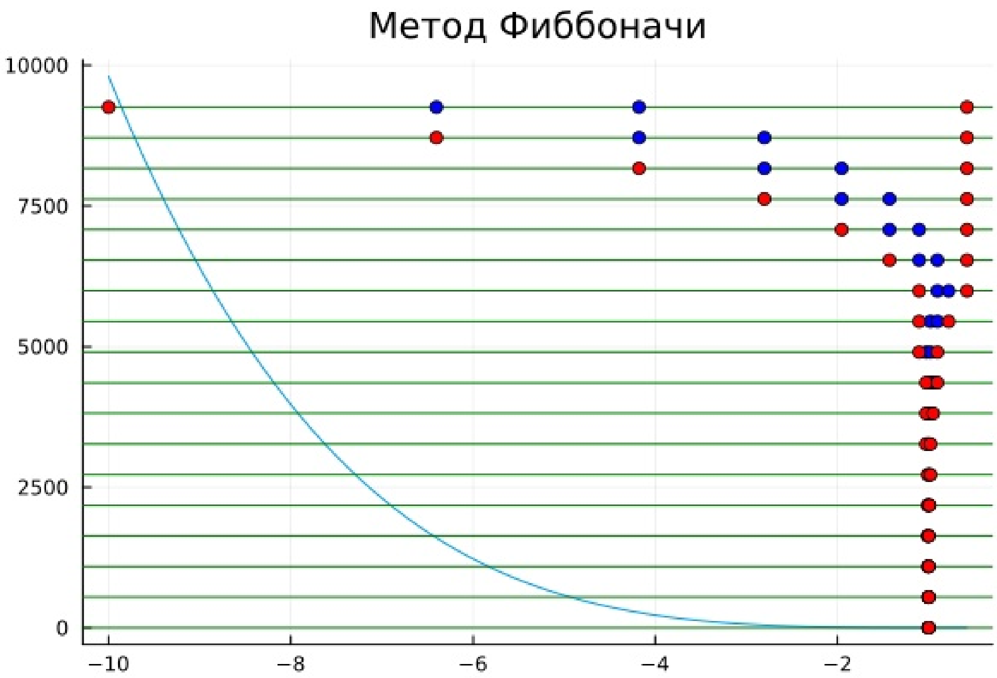

Рис. 3 — Результат работы алгоритма методом Фиббоначи