Министерство науки и высшего образования
Федеральное государтсвенное бюджетное образовательное учреждение высшего образования
Югорский государственный университет

Отчет о лабораторной работе №2
по дисциплине «Методы оптимизации»

Выполнил

Студент группы 1162б
________________ Панчишин И. Р.
«____» __________ 2019 г.

Принял

Доцент ИЦЭ
________________ Самарин В. А.
«____» __________ 2019 г.

Ханты-Мансийск, 2019

Цель

Изучить прямые методы минимизации.

Задачи

1. Реализовать следующие три метода минимизации: дихотомии, золотого сечения и Фибоначчи.

2. Изучить зависимость числа вычислений функции (скорости работы) от заданной точности.

Ход работы

Реализовал требуемые методы на языке программирования *Octave* (свободная реализация *Matlab*). Исходный код, представленный ниже, позволяет определить минимум функции (унимодальной), визуализировать изменение отрезка поиска на каждой итерации, а также построить зависимость скорости поиска от заданной точности.

```
addpath(../code);

set(0, defaultaxesfontsize, 14);
set(0, defaulttextfontsize, 14);


% исходные данные
f = @(X) X.^4 + exp(-X)
X = -1:0.01:1;
[a b] = deal(-0.5, 1)
e = 0.1

% минимум
[targetxm, targetym] = fminbnd(f, a, b)


% работа алгоритмов
Fm = {@dichotomy, @gold, @fib};
Name = {Дихотомии, Золотого сечения, Фибоначчи};

for i = 1:length(Fm)
    subplot(1, 3, i);

    plot(X, f(X), Color, b);
    xlabel(x);
    ylabel(y);
    hold on;

    plot(targetxm, targetym, bo, LineWidth, 3);


    [xm, ym, n, Approx] = Fm{i}(f, a, b, e)
    title([Name{i}, , n = , num2str(n)]);

    segment = plot([Approx(1, 1) Approx(1, 2)], [0 0], Color, r, LineWidth, 3);
    for j = 1:length(Approx) - 1
        st = Approx(j, :);
        en = Approx(j+1, :);
        if st(1) != en(1)
```

```
            shinkseg(st(1), en(1), segment, 1);
        end
        if st(2) != en(2)
            shinkseg(st(2), en(2), segment, 1);
        end
    end

    plot(xm, ym, ro, LineWidth, 3);
end

% зависимость от точности
figure;
hold on;

E = linspace(0.0001, 0.5, 20);
for i = 1:length(Fm)
    N = [];
    for e = E
        [xm ym n] = Fm{i}(f, a, b, e);
        N = [N n];
    end
    plot(E, N);
end

legend(Name);
xlabel(Погрешность);
ylabel(Вычислений);


pause
```

```
% метод дихотомии

% f - функция одной переменной
% a, b - точки, определяющие отрезок поиска минимума
% e - характеристика точности (чем меньше, тем точнее)

% Xm - точка минимума
% ym - минимум
% n - кол-во вычислений целевой функции (f)
% Approx - история приближения

function [xm, ym, n, Approx] = dichotomy(f, a, b, e)
    Approx = [a, b];
    d = e; %rand() * 2 * e;
    n = 0;

    while (b - a) / 2 > e
        [x1 x2] = deal((a + b - d) / 2, (a + b + d) / 2);

        n = n + 2;
        if (f(x2) > f(x1))
            b = x2;
        else
            a = x1;
        end

        Approx = [Approx; [a, b]];
    end

```

```
    xm = (a + b) / 2;
    ym = f(xm); ++n;
end
```

```
% метод Фибоначчи

% см. dichotomy.m для описания аргументов

function [xm, ym, n, Approx] = fib(f, a, b, e)
    Approx = [a, b];

    minfib = (b - a) / e;
    k = 1;
    while minfib > fibonacci(k)
        ++k;
    end

    right = @(a, b, k) a + fibonacci(k-1) / fibonacci(k) * len(a, b);
    left = @(a, b, k) a + fibonacci(k-2) / fibonacci(k) * len(a, b);

    [x1 x2] = deal(left(a, b, k), right(a, b, k));
    [y1 y2] = deal(f(x1), f(x2));
    n = 2;

    while k > 2
        ++n;
        if (y2 > y1)
            b = x2;
            [x2 y2] = deal(x1, y1);
            x1 = left(a, b, k);
            y1 = f(x1);
        else
            a = x1;
            [x1 y1] = deal(x2, y2);
            x2 = right(a, b, k);
            y2 = f(x2);
        end

        Approx = [Approx; [a, b]];

        --k;
    end

    xm = (a + b) / 2;
    ym = f(xm); ++n;
end

function res = fibonacci(n)
    if n < 0
        res = -1;
    elseif n == 0
        res = 0;
    elseif n == 1
        res = 1;
    else
        res = fibonacci(n - 1) + fibonacci(n - 2);
    end
end
```

```
% метод золотого сечения

% см. dichotomy.m для описания аргументов

function [xm, ym, n, Approx] = gold(f, a, b, e)
    Approx = [a, b];

    g = (sqrt(5) - 1) / 2;
    right = @(a, b) a + g * len(a, b);
    left = @(a, b) a + (1 - g) * len(a, b);

    [x1 x2] = deal(left(a, b), right(a, b));
    [y1 y2] = deal(f(x1), f(x2));
    n = 2;

    % точность для произвольной итерации
    %initLen = b - a;
    %1/2 * g^n * initLen

    while (b - a) / 2 > e
        ++n;
        if (y2 > y1)
            b = x2;
            [x2 y2] = deal(x1, y1);
            x1 = left(a, b);
            y1 = f(x1);
        else
            a = x1;
            [x1 y1] = deal(x2, y2);
            x2 = right(a, b);
            y2 = f(x2);
        end

        Approx = [Approx; [a, b]];
    end

    xm = (a + b) / 2;
    ym = f(xm); ++n;
end
```

```
% длина отрезка

function res = len(a, b)
    res = b - a;
end
```

```
% анимация изменения отрезка

function shinkseg(st, en, line, timeout)
    X = get(line, XData);

    for d = st : len(st, en) / 30 : en
        if (st > en)
            X(2) = d;
            set(line, XData, X);
        else
            X(1) = d;
            set(line, XData, X);
```

```
        end

        pause(0.01);
    end

    pause(timeout);
end
```

Найденный каждым методом минимум отмечен красной точкой на Рис. 1. Синей точкой отмечен минимум, найденный при помощи встроенной функции *fminbnd*. Минимумы не совпадают, так как заданная точность не достаточно велика. Отрезок, выделенный красным, является конечным отрезком поиска минимума.

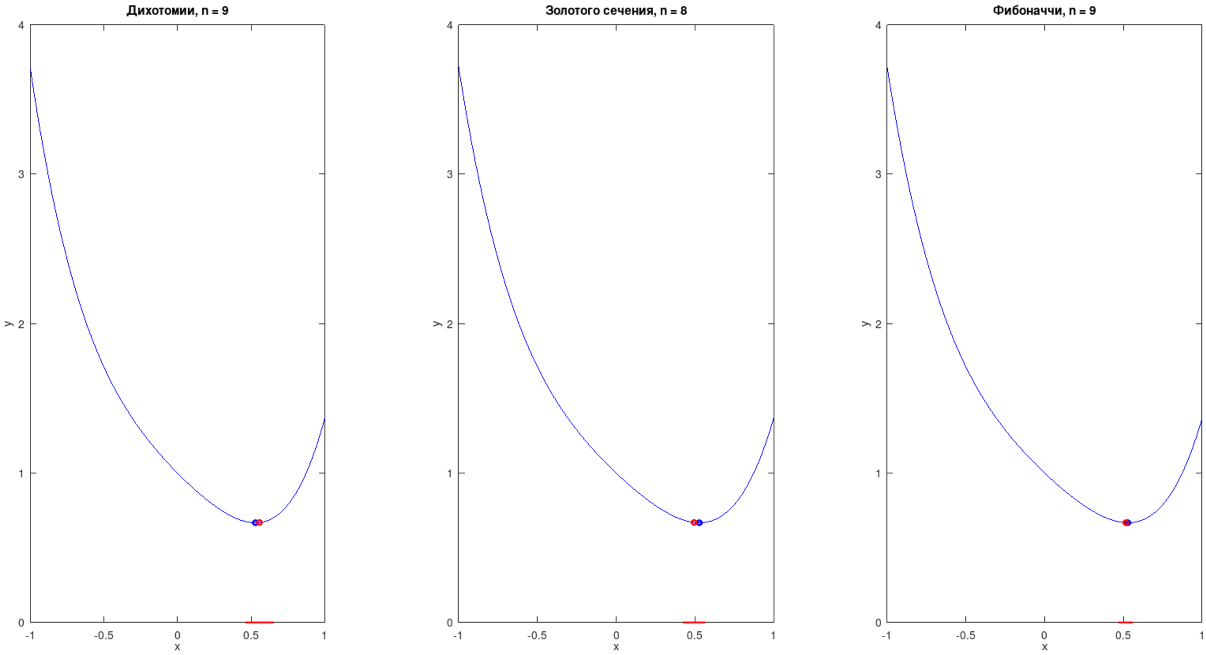

Рис. 1: Минимум функции

Зависимость числа вычислений от точности представлена на Рис. 2. Наблюдается экспоненциальный рост количества вычислений с увеличением точности.

Вывод

Реализовал прямые методы минимизации: дихотомии, золотого сечения и Фибоначчи. Сравнил их работу. Метод золотого сечения показал себя лучше остальных рассмотренных методов, однако его преимущество становится ощутимым только при большой точности, если точность небольшая, то подойдет любой метод.

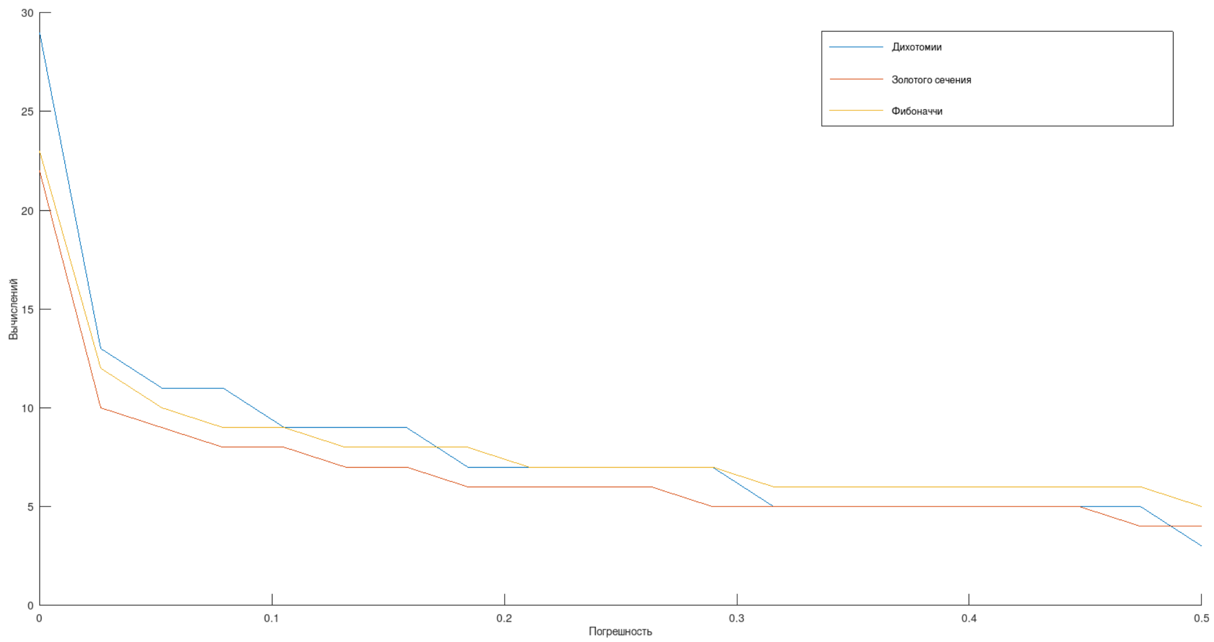

Рис. 2: Зависимость скорости от точности