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 «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3

Дисциплина Методы вычислений

Тема Метод парабол

Вариант №10

Студент Коноваленко В. Д.

Группа ИУ7-21М

Оценка (баллы) ______________

Преподаватель Власов П.А.

Москва.
2024 г.

Цель работы: изучение метода парабол для решения задачи одномерной минимизации.

Содержание работы

1. реализовать метод парабол в сочетании с методом золотого сечения в виде программы на ЭВМ;
2. провести решение задачи
$$\begin{cases} f(x) \to min \\ x \in [a, b] \end{cases}$$
для данных индивидуального варианта;
3. организовать вывод на экран графика целевой функции, найденной точки минимума $(x^*,\ f(x^*))$ и последовательности отрезков $[x_{1,i}, x_{3,i}]$, содержащих точку искомого минимума (для последовательности отрезков следует предусмотреть возможность «отключения» вывода её на экран).

Целевая функция *f(x)*	***[a, b]***
$\sin\left(\frac{x^4 + x^3 - 3x + 3 - 30^{\frac{1}{3}}}{2}\right) + th\left(\frac{4\sqrt{3}x^3 - 2x - 6\sqrt{2} + 1}{-2\sqrt{3}x^3 + x + 3\sqrt{2}}\right) + 1.2$	*[0, 1]*

Метод парабол является представителем группы методов, основанных на аппроксимации целевой функции некоторой другой функцией, точку минимума которой можно найти аналитически. Эта точка и принимается за очередное приближение искомого минимума целевой функции.

Пусть:

1) $f(x)$ унимодальна на отрезке $[a; b]$;

2) $f(x)$ достигает минимум во внутренней точке отрезка $[a; b]$.

Выберем точки $x_1, x_2, x_3 \in [a; b]$ так, чтобы:

$$(*) \quad \begin{cases} 1)\ x_1 < x_2 < x_3 \\ 2)\ f(x_1) \geq f(x_2) \leq f(x_3) \end{cases}$$

(причём по крайней мере одно неравенство из (2) должно быть строгим)

Тогда в силу унимодальности функции $f(x)$ $x^* \in [x1; x3]$.

Аппроксимируем целевую функцию $f(x)$ параболой, переходящей через точки: $(x_1; f_1), (x_2; f_2), (x_3; f_3)$, где $f_1 = f(x_1), f_2 = f(x_2), f_3 = f(x_3)$.

Тогда в силу выбора точек x_1, x_2, x_3 ветви этой параболы будут направлены вверх, а точка $\bar{x}$ её минимума будет принадлежать отрезку $[x_1; x_3]$. За очередное приближение точки x^* принимается точка $\bar{x}$.

Пусть $g(x) = a_0 + a_1(x - x_1) + a_2(x - x_1)(x - x_2)$ – уравнение искомой параболы, тогда можно доказать, что (**):

$$a_0 = f_1$$

$$a_1 = \frac{f_2 - f_1}{x_2 - x_1}$$

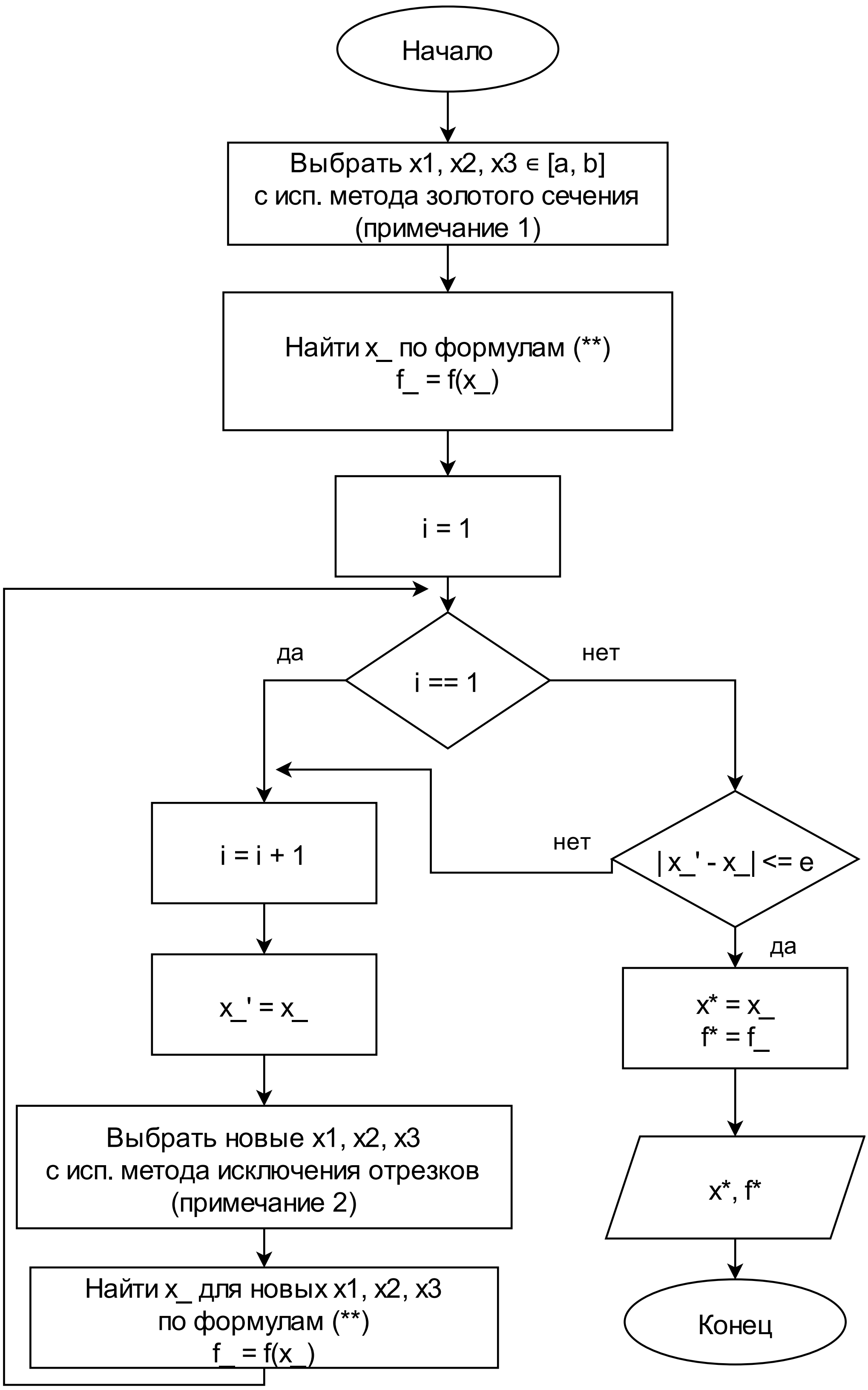
$$a_2 = \frac{1}{x_3 - x_2}\left[\frac{f_3 - f_1}{x_3 - x_1} - \frac{f_2 - f_1}{x_2 - x_1}\right]$$

$$\bar{x} = \frac{1}{2}\left[x_1 + x_2 - \frac{a_1}{a_2}\right]$$

Значение $\bar{x}$ используется как очередное приближение значения x^*. Далее выбираются новые точки x_1, x_2, x_3 и процедура повторяется до тех порка пока не выполнится неравенство $|\bar{x} - \bar{x}'| \le \varepsilon$.

Примечание 1. Для выбора точек x_1, x_2, x_3 на первой итерации выполняется метод золотого сечения до тех пор, пока для двух пробных точек этого метода и одной из граничных точек очередного отрезка не будут выполнены неравенства (*).

Примечание 2. При второй и последующих итерациях на отрезке $[x_1; x_3]$ рассматриваются две пробные точки x_2 и $\bar{x}$, для которых используется метод исключения отрезков. В новом отрезке $[x_1'; x_3']$ в качестве $x_2{}'$ выбирается та точка из x_2 и $\bar{x}$, которая оказалась внутри.



Текст программы представлен на Листинге 1

Листинг 1

```
function lab3()
    clc();
```

```
    warning('off', 'all');

    debug = 1;
    delaySeconds = 1.0;

    a = 0;
    b = 1;
    e = 1e-2;

    fplot(@f, [a, b]);
    hold on;


    [x, y, N] = parabolicMethod(a, b, e, debug, delaySeconds);

    fprintf('RESULT: e = %f | N = %d | x* = %.10f | f(x*) = %.10f', e, N, x, y)

    scatter(x, y, 'g', 'filled');

    hold off;
end

function y = f(x)
    y = sin((power(x, 4) + power(x, 3) - 3 * x + 3 - power(30, 1/3)) / 2) + tanh((4
* sqrt(3) * power(x, 3) - 2 * x - 6 * sqrt(2) + 1) / (-2 * sqrt(3) * power(x, 3) +
x + 3 * sqrt(2))) + 1.2;
end

function [x, y, N] = parabolicMethod(a, b, e, debug, delaySeconds)
    [x1, f1, x2, f2, x3, f3, N] = findStartPointsByGoldenRation(a, b, e, debug,
delaySeconds);

    fprintf("Found start points: x1 = %.10f (%.10f) | x2 = %.10f (%.10f) | x3 =
%.10f (%.10f)\n", x1, f1, x2, f2, x3, f3);

    fprintf("Finding minimum via parabolic method...\n");

    i = 0;

    while 1
        i = i + 1;

        a0 = f1;
        a1 = (f2 - f1)/(x2 - x1);
        a2 = ((f3 - f1)/(x3 - x1) - (f2 - f1)/(x2 - x1))/(x3 - x2);

        q = parabola_function(a0, a1, a2, x1, x2);

        if i ~= 1
            old_x_tilt = x_tilt;
        end

        x_tilt = (x1 + x2 - a1/a2)/2;
        f_tilt = f(x_tilt);

        if debug

            fprintf("%i. x1 = %.10f | x2 = %.10f | x3 = %.10f | x_tilt = %.10f\n",
i, x1, x2, x3, x_tilt);

            fplot(q, [a, b], 'LineStyle', ':', 'Color', 'r');
```

```
            scatter(x1, f1, 'r');
            scatter(x2, f2, 'r');
            scatter(x3, f3, 'r');
            scatter(x_tilt, f_tilt, 'r', 'filled');
            pause(delaySeconds);
            fplot(q, [a, b], 'LineStyle', ':', 'Color', 'b');
            scatter(x1, f1, 'b');
            scatter(x2, f2, 'b');
            scatter(x3, f3, 'b');
            scatter(x_tilt, f_tilt, 'b', 'filled');
        end

        if x2 < x_tilt
            if f2 <= f_tilt
                x3 = x_tilt;
                f3 = f_tilt;
            else % f2 > f_tilt
                x1 = x2;
                f1 = f2;
                x2 = x_tilt;
                f2 = f_tilt;
            end
        else % x2 > x_tilt
            if f_tilt <= x2
                x3 = x2;
                f3 = f2;
                x2 = x_tilt;
                f2 = f_tilt;
            else % f_tilt > x2
                x1 = x_tilt;
                f1 = f_tilt;
            end
        end

        if i ~= 1
            if abs(x_tilt - old_x_tilt) <= e
                break;
            end
        end
    end

    x = x_tilt;
    y = f_tilt;
    N = N + i;
end

function [result_x1, result_f1, result_x2, result_f2, result_x3, result_f3, N] =
findStartPointsByGoldenRation(a, b, e, debug, delaySeconds)
    t = (sqrt(5) - 1) / 2;
    l = b - a;

    x1 = b - t * l;
    f1 = f(x1);
    x2 = a + t * l;
    f2 = f(x2);

    fprintf("Finding start points via golden ratio method...\n")

    i = 1;

    if debug
```

```
        fprintf('%i: a%i = %.10f | b%i = %.10f\n', i, i, a, i, b);
        line([a, b], [f(a), f(b)], 'Color', 'red', 'LineStyle', '--');
        pause(delaySeconds);
        line([a, b], [f(a), f(b)], 'Color', 'blue', 'LineStyle', '--');
    end

    while 1
        if l > 2 * e
            i = i + 1;


           if f1 <= f2
                b = x2;
                l = b - a;

                new_x = b - t * l;
                new_f = f(new_x);

                if debug
                    fprintf('%i: a%i = %.10f | b%i = %.10f\n', i, i, a, i, b);
                    line([a, b], [f(a), f(b)], 'Color', 'red', 'LineStyle', '--');
                    pause(delaySeconds);
                    line([a, b], [f(a), f(b)], 'Color', 'blue', 'LineStyle', '--');
                end

                if new_f > f1
                    break;
                end

                x2 = x1;
                f2 = f1;

                x1 = new_x;
                f1 = new_f;
            else % f1 > f2
                a = x1;
                l = b - a;

                new_x = a + t * l;
                new_f = f(new_x);

                if debug
                    fprintf('%i: a%i = %.10f | b%i = %.10f\n', i, i, a, i, b);
                    line([a, b], [f(a), f(b)], 'Color', 'red', 'LineStyle', '--');
                    pause(delaySeconds);
                    line([a, b], [f(a), f(b)], 'Color', 'blue', 'LineStyle', '--');
                end

                if new_f > f2
                    break;
                end

                x1 = x2;
                f1 = f2;

                x2 = new_x;
                f2 = new_f;
           end

        else
            break
```

```
        end
    end

    result_x1 = x1;
    result_f1 = f1;

    result_x2 = x2;
    result_f2 = f2;

    result_x3 = new_x;
    result_f3 = new_f;

    N = i + 1;
end

function q = parabola_function(a0, a1, a2, x1, x2)
    q = @(x) ((a0 + a1*(x - x1) + a2 * (x - x1)*(x - x2)));
end
```

Результаты расчетов для задачи из индивидуального варианта.

№ п/п	ε	N	x^*	$f(x^*)$
1	0.01	6	0.7035258875	-0.4652445869
2	0.0001	8	0.7054140240	-0.4652516013
3	0.000001	12	0.7054664267	-0.4652516064