|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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

для данных индивидуального варианта;

1. организовать вывод на экран графика целевой функции, найденной точки минимума и последовательности отрезков содержащих точку искомого минимума (для последовательности отрезков следует предусмотреть возможность «отключения» вывода её на экран).

|  |  |
| --- | --- |
| **Целевая функция *f(x)*** | ***[a, b]*** |
|  | *[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. унимодальна на отрезке ;
2. достигает минимум во внутренней точке отрезка .

Выберем точки так, чтобы:

(\*)

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

Тогда в силу унимодальности функции .

Аппроксимируем целевую функцию параболой, переходящей через точки: .

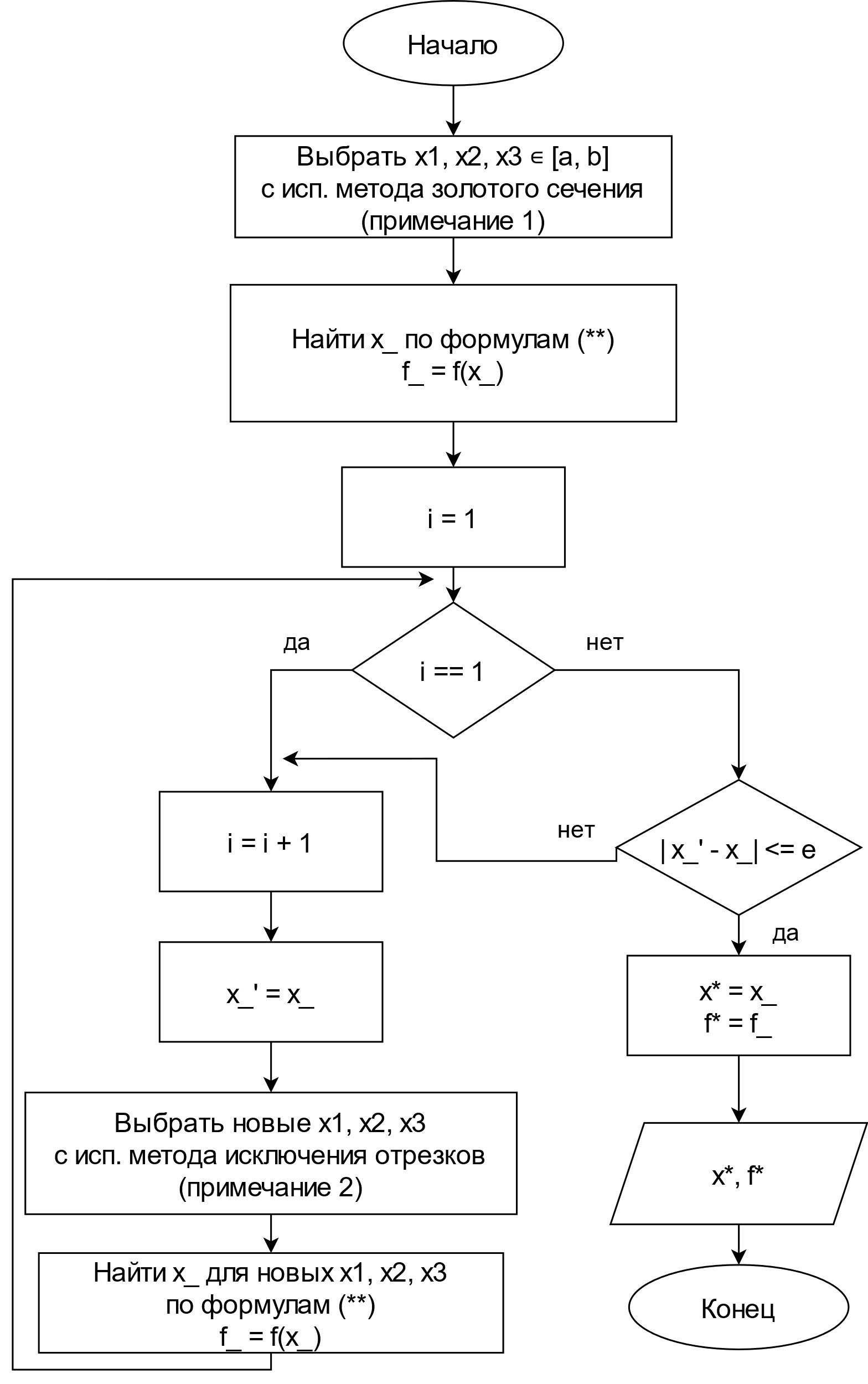
Тогда в силу выбора точек ветви этой параболы будут направлены вверх, а точка её минимума будет принадлежать отрезку . За очередное приближение точки принимается точка .

Пусть – уравнение искомой параболы, тогда можно доказать, что (\*\*):

Значение используется как очередное приближение значения . Далее выбираются новые точки и процедура повторяется до тех порка пока не выполнится неравенство .

Примечание 1. Для выбора точек 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 рассматриваются две пробные точки , для которых используется метод исключения отрезков. В новом отрезке в качестве выбирается та точка из , 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 |  |  |
| 1 | 0.01 | 6 | 0.7035258875 | -0.4652445869 |
| 2 | 0.0001 | 8 | 0.7054140240 | -0.4652516013 |
| 3 | 0.000001 | 12 | 0.7054664267 | -0.4652516064 |