|  |  |
| --- | --- |
| 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 №2**  **Студент \_Брянская Е.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-21М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Власов П.А.** |  |

Москва.

2023 г.

**Цель работы:** изучение метода парабол для решения задачи одномерной оптимизации.

**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]* |

Общая идея метода заключается в том, что целевая функция аппроксимируется квадратичной функцией, точку минимума которой можно найти аналитически. При этом точка минимума аппроксимирующей функции принимается в качестве приближения точки минимума исходной целевой функции.

Выбираются пробные точки внутри рассматриваемого интервала [a, b], так что:

|  |  |
| --- | --- |
| 1. *,* где по крайней мере одно неравенство является строгим. | (\*) |

В силу унимодальности целевой функции можно утверждать, что точка минимума x\*, как и удовлетворяет условию x\* ∊ [].

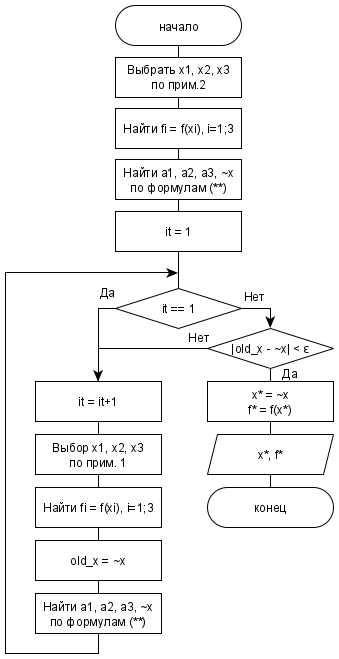
В методе парабол в качестве аппроксимирующей функции используется квадратичная. Она проходит через точки

Уравнение параболы:

|  |  |
| --- | --- |
|  | (\*\*) |

Прим.1: В качестве на следующей итерации используются точки, принадлежащие оставшемуся отрезку, и точка или , которая оказалась внутри.

Прим.2: Изначально точки выбираются при помощи метода золотого сечения. Делается несколько итераций, пока две пробные точки и одна из граничных не начнут удовлетворять



Текст программы представлен на Листинге 1

Листинг 1

|  |
| --- |
| function lab03()  clc();  debugFlg = 1;  delayS = 0.8;  a = 0;  b = 1;  eps = 1e-6;  fplot(@f, [a, b]);  hold on;  pause(3);  parabolic\_method(a, b, eps, debugFlg, delayS);  end  function parabolic\_method(a, b, eps, debugFlg, delayS)  tau = (sqrt(5)-1) / 2;  l = b - a;  x1 = b - tau\*l;  x2 = a + tau\*l;  f1 = f(x1);  f2 = f(x2);  fprintf('Golden ratio method (looking for initial points x1, x2, x3)\n');  i = 0;  if debugFlg  fprintf('Iteration %2d:\t [a, b] = [%.10f, %.10f], f(a) = %.10f, f(b) = %.10f\n', i, a, b, f(a), f(b));  line([a b], [f(a) f(b)], 'color', 'b');  hold on;  end  while l > 2\*eps  i = i + 1;  if debugFlg  line([a b], [f(a) f(b)], 'color', 'b');  hold on;  end  if f1 <= f2  b = x2;  l = b - a;  new\_x = b - tau\*l;  new\_f = f(new\_x);  if f1 <= new\_f  x3 = x2; f3 = f2;  x2 = x1; f2 = f1;  x1 = new\_x; f1 = new\_f;  break;  end  x2 = x1; f2 = f1;  x1 = new\_x; f1 = new\_f;  else  a = x1;  l = b - a;  new\_x = a + tau\*l;  new\_f = f(new\_x);  if f2 <= new\_f  x1 = a;  x3 = new\_x; f3 = new\_f;  break;  end  x1 = x2; f1 = f2;  x2 = new\_x; f2 = new\_f;  end  if debugFlg  fprintf('Iteration %2d:\t [a, b] = [%.10f, %.10f], f(a) = %.10f, f(b) = %.10f\n', i, a, b, f(a), f(b));  line([a b], [f(a) f(b)], 'color', 'r');  hold on;  pause(delayS);  end  end  if debugFlg  fprintf('Found points x1, x2, x3: %.10f, %.10f, %.10f\n', x1, x2, x3);  scatter(x1, f1, 'green', 'filled');  scatter(x2, f2, 'green', 'filled');  scatter(x3, f3, 'green', 'filled');  line([x1 x3], [f1 f3], 'color', 'b');  hold on;  pause(delayS\*2);  end  fprintf('Parabolic method\n');  a1 = (f2 - f1) / (x2 - x1);  a2 = ((f3 - f1)/(x3 - x1) - (f2 - f1)/(x2 - x1)) / (x3 - x2);  x\_ = 1 / 2 \* (x1 + x2 - a1/a2);  f\_ = f(x\_);  for i = 1:1000  old\_x\_ = x\_;  if x\_ > x2  x1 = x2; f1 = f2;  x2 = x\_; f2 = f\_;  else  x3 = x2; f3 = f2;  x2 = x\_; f2 = f\_;  end  if debugFlg  fprintf('Iteration %2d:\t [x1, x3] = [%.10f, %.10f], f(x1) = %.10f, f(x3) = %.10f\n', i, x1, x3, f1, f3);  fprintf('Current min point: x=%.10f, f(x)=%.10f\n', x\_, f\_);  line([x1 x3], [f1 f3], 'color', 'b');  hold on;  pause(delayS);  end  a1 = (f2 - f1) / (x2 - x1);  a2 = ((f3 - f1)/(x3 - x1) - (f2 - f1)/(x2 - x1)) / (x3 - x2);  x\_ = 1 / 2 \* (x1 + x2 - a1/a2);  f\_ = f(x\_);  if abs(old\_x\_ - x\_) <= eps  break  end  end  x\_res = x\_; f\_res = f\_;  if debugFlg  scatter(x\_res, f\_res, 'r', 'filled');  fprintf('Final result after %2d iterations: x=%.10f, f(x)=%.10f\n', i, x\_res, f\_res);  end  end  function y = f(x)  y = cos(power(x,5) - x + 3 + power(2, 1/3)) + atan((power(x,3) - 5 \* sqrt(2)\*x - 4) / (sqrt(6)\*x + sqrt(3))) + 1.8;  end |

**Результаты расчетов для задачи из индивидуального варианта.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | ε | N |  |  |
| 1 | 0.01 | 6 | 0.6626400573 | -0.2251325465 |
| 2 | 0.0001 | 9 | 0.6639224611 | -0.2251354835 |
| 3 | 0.000001 | 13 | 0.6639622119 | -0.2251354862 |