|  |  |
| --- | --- |
| 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» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

Отчёт

по лабораторной работе № 3

по дисциплине «Теория Систем и Системный Анализ»

**Тема: «Исследование алгоритма имитации отжига»**

Вариант 16

Выполнил: Сердюкова М.Ю.,

студентка группы ИУ8-32

Проверил: Коннова Н. С.,

доцент каф. ИУ8

г. Москва,

2020 г.

# 1. Цель работы

Изучение метода имитации отжига для поиска экстремума на примере унимодальной и мультимодальной функций одного переменного.

# 2. Постановка задачи

Унимодальная функция:

Отрезок поиска:

1. На интервале [a,b] задана унимодальная функция одного переменного f(x) . Используя метод имитации отжига осуществить поиск минимума f(x).

2. При аналогичных исходных условиях осуществить поиск минимума f(x), модулированной сигналом sin(5x) , т.е. мультимодальной функции f(x)\* sin(5x).

# 3. Ход работы

Пусть задана функция f(x)=. График этой функции и функции, модулированной сигналом sin5x, т.е. мультимодальной функции f(x)=\*sin(5x), представлены на рис. 3.1 и 3.2 соответственно. Интервал поиска – [9, 14]. Реализовываем алгоритм имитации отжига следующим образом:

1. Изначально задаются начальная температура T\_max и конечная температура T\_min.

2. Случайно выбирается точка x1 на отрезке. Вычисляется значение функции в этой точке f(x1).

3. Пока Ti > Tmin:

1) Случайно выбирается точка xi на отрезке. Вычисляется значение функции в этой точке f(xi).

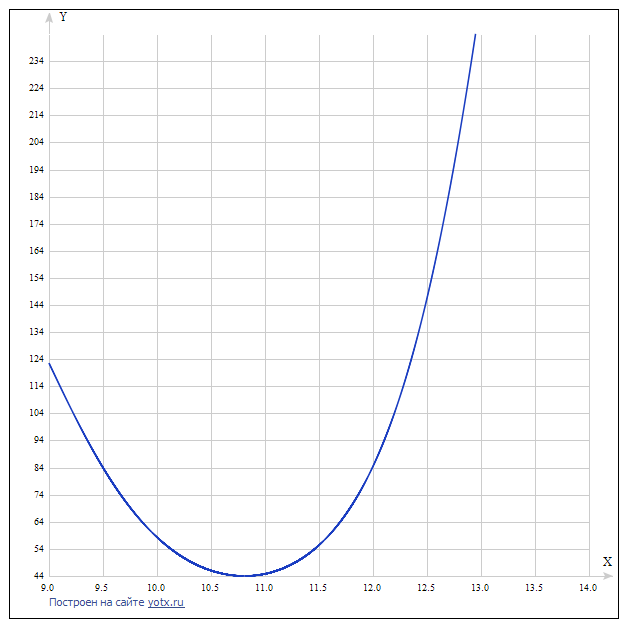
2) Определяется d\_f = f\_i - f\_i-1.

3) Если d\_f <= 0, то осуществляется переход в точку xi .

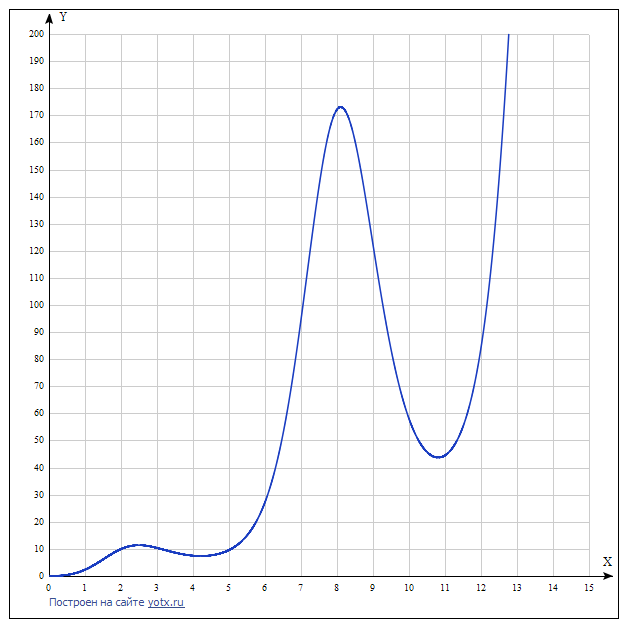
4) Если d\_f > 0, то переход осуществляется с вероятностью P(d\_f).

5) Понижение температуры: T\_i+1 = T\_i\*0.95

В качестве начальной температуры возьмем T\_max =10000, конечной – T\_min =0.1.



**Рисунок 1** – график функции f(x)



**Рисунок 2** – график функции f(x)\*sin(5x)

**Таблица 1** – результаты программы для унимодальной функции

N T x f(x)

1 10000 7.02462 321.748

2 9500 7.769 96.9494

3 9025 7.57361 163.478

4 8573.75 9.07432 149.948

5 8145.06 9.98853 116.074

6 7737.81 9.57109 58.4698

7 7350.92 9.64469 79.1782

8 6983.37 7.57382 74.789

9 6634.2 8.28019 149.965

10 6302.49 8.49949 170.421

11 5987.37 7.02862 160.582

12 5688 9.76857 97.3456

13 5403.6 7.86309 68.1208

14 5133.42 8.7086 168.06

15 4876.75 7.59562 146.223

16 4632.91 7.21652 151.708

17 4401.27 8.453 116.322

18 4181.2 9.59788 163.193

19 3972.14 8.52091 77.5441

20 3773.54 9.26534 159.297

21 3584.86 9.33444 100.618

22 3405.62 9.33444 95.3575

23 3235.34 8.40269 95.3575

24 3073.57 9.8068 165.721

25 2919.89 9.49137 66.2445

26 2773.9 9.80409 84.2867

27 2635.2 9.01178 66.3747

28 2503.44 9.06059 121.319

29 2378.27 9.95281 117.221

30 2259.36 7.55244 59.8522

31 2146.39 9.723 148.209

32 2039.07 9.14336 70.4685

33 1937.11 8.12762 110.363

34 1840.26 7.21932 173.007

35 1748.25 7.80741 116.605

36 1660.83 9.59926 165.515

37 1577.79 8.32224 77.4611

38 1498.9 8.1497 169.054

39 1423.96 8.32942 172.873

40 1352.76 8.93467 168.794

41 1285.12 8.94408 127.817

42 1220.87 7.80626 127.024

43 1159.82 8.95513 165.458

44 1101.83 7.4403 126.094

45 1046.74 9.7362 138.306

46 994.403 8.86213 69.7757

47 944.682 7.87113 133.888

48 897.448 9.7743 168.386

49 852.576 8.43889 67.834

50 809.947 7.57253 163.934

51 769.45 9.54732 149.861

52 730.977 8.75155 80.6635

53 694.428 8.03898 142.875

54 659.707 9.36377 172.697

55 626.722 7.38611 93.1927

56 595.386 8.12295 133.184

57 565.616 7.23174 173.024

58 537.335 8.14763 117.864

59 510.469 9.87802 172.889

60 484.945 7.09903 62.9752

61 460.698 8.77779 104.399

62 437.663 8.89872 140.785

63 415.78 8.23389 130.836

64 394.991 7.11725 171.608

65 375.241 7.11725 106.241

66 356.479 7.7839 106.241

67 338.655 7.44835 164.294

68 321.723 7.44835 139.051

69 305.636 8.2556 139.051

70 290.355 8.06423 171.094

71 275.837 9.9345 172.925

72 262.045 9.83173 60.5878

73 248.943 9.02625 65.0671

74 236.496 9.02625 120.102

75 224.671 9.71724 120.102

76 213.437 9.71724 70.7738

77 202.765 9.84881 70.7738

78 192.627 8.45136 64.2807

79 182.996 7.62161 163.28

80 173.846 9.94438 153.714

81 165.154 9.34078 60.1883

82 156.896 8.64928 94.8863

83 149.051 7.79868 150.66

84 141.599 7.4366 165.072

85 134.519 8.44876 137.962

86 127.793 8.70107 163.418

87 121.403 7.81552 146.798

88 115.333 9.49051 165.916

89 109.566 7.28934 84.3438

90 104.088 9.75644 123.669

91 98.8836 8.81709 68.7338

92 93.9395 8.81468 137.601

93 89.2425 8.71948 137.797

94 84.7804 8.6322 145.384

95 80.5413 8.6322 151.891

96 76.5143 8.6322 151.891

97 72.6886 8.6322 151.891

98 69.0541 8.77879 151.891

99 65.6014 7.15757 140.705

100 62.3214 7.15757 110.331

101 59.2053 8.84784 110.331

102 56.245 7.73629 135.072

103 53.4328 7.97963 161.568

104 50.7611 8.91206 171.726

105 48.2231 8.91206 129.718

106 45.8119 9.79754 129.718

107 43.5213 9.62378 66.6911

108 41.3453 7.00319 76.0038

109 39.278 8.94428 94.8331

110 37.3141 9.48185 127.008

111 35.4484 9.08182 84.9208

112 33.676 8.65309 115.448

113 31.9922 8.57092 150.382

114 30.3926 8.18547 156.111

115 28.8729 9.87778 172.48

116 27.4293 9.87778 62.9858

117 26.0578 9.82662 62.9858

118 24.7549 9.86849 65.3056

119 23.5172 9.86849 63.3959

120 22.3413 9.86849 63.3959

121 21.2243 9.80005 63.3959

122 20.1631 9.80005 66.5695

123 19.1549 9.80005 66.5695

124 18.1972 9.80005 66.5695

125 17.2873 9.80005 66.5695

126 16.4229 9.80005 66.5695

127 15.6018 9.80005 66.5695

128 14.8217 9.80005 66.5695

129 14.0806 9.80005 66.5695

130 13.3766 9.80005 66.5695

131 12.7078 9.80005 66.5695

132 12.0724 9.80005 66.5695

133 11.4687 9.80005 66.5695

134 10.8953 9.80005 66.5695

135 10.3505 9.80005 66.5695

136 9.83302 9.80005 66.5695

137 9.34136 9.80005 66.5695

138 8.8743 9.80005 66.5695

139 8.43058 9.80005 66.5695

140 8.00905 9.80005 66.5695

141 7.6086 9.80005 66.5695

142 7.22817 9.80005 66.5695

143 6.86676 9.80005 66.5695

144 6.52342 9.80005 66.5695

145 6.19725 9.80005 66.5695

146 5.88739 9.80005 66.5695

147 5.59302 9.80005 66.5695

148 5.31337 9.80005 66.5695

149 5.0477 9.80005 66.5695

150 4.79532 9.80005 66.5695

151 4.55555 9.80005 66.5695

152 4.32777 9.80005 66.5695

153 4.11138 9.80005 66.5695

154 3.90581 9.80005 66.5695

155 3.71052 9.78122 66.5695

156 3.525 9.78122 67.4904

157 3.34875 9.78122 67.4904

158 3.18131 9.78122 67.4904

159 3.02224 9.61652 67.4904

160 2.87113 9.61652 76.4313

161 2.72758 9.61652 76.4313

162 2.5912 9.61652 76.4313

163 2.46164 9.61652 76.4313

164 2.33856 9.61652 76.4313

165 2.22163 9.61652 76.4313

166 2.11055 9.61652 76.4313

167 2.00502 9.61652 76.4313

168 1.90477 9.61652 76.4313

169 1.80953 9.61652 76.4313

170 1.71905 9.61652 76.4313

171 1.6331 9.95 76.4313

172 1.55145 9.95 59.9639

173 1.47387 9.95 59.9639

174 1.40018 9.95 59.9639

175 1.33017 9.95 59.9639

176 1.26366 9.95 59.9639

177 1.20048 9.95 59.9639

178 1.14045 9.95 59.9639

179 1.08343 9.95 59.9639

180 1.02926 9.95 59.9639

181 0.977798 9.95 59.9639

182 0.928908 9.95 59.9639

183 0.882462 9.95 59.9639

184 0.838339 9.95 59.9639

185 0.796422 9.95 59.9639

186 0.756601 9.95 59.9639

187 0.718771 9.95 59.9639

188 0.682833 9.95 59.9639

189 0.648691 9.95 59.9639

190 0.616256 9.95 59.9639

191 0.585444 9.95 59.9639

192 0.556171 9.95 59.9639

193 0.528363 9.95 59.9639

194 0.501945 9.95 59.9639

195 0.476847 9.95 59.9639

196 0.453005 9.95 59.9639

197 0.430355 9.95 59.9639

198 0.408837 9.95 59.9639

199 0.388395 9.95 59.9639

200 0.368975 9.95 59.9639

201 0.350527 9.95 59.9639

202 0.333 9.95 59.9639

203 0.31635 9.95 59.9639

204 0.300533 9.95 59.9639

205 0.285506 9.95 59.9639

206 0.271231 9.95 59.9639

207 0.257669 9.95 59.9639

208 0.244786 9.95 59.9639

209 0.232547 9.95 59.9639

210 0.220919 9.95 59.9639

211 0.209873 9.95 59.9639

212 0.19938 9.95 59.9639

213 0.189411 9.95 59.9639

214 0.17994 9.95 59.9639

215 0.170943 9.95 59.9639

216 0.162396 9.95 59.9639

217 0.154276 9.95 59.9639

218 0.146562 9.95 59.9639

219 0.139234 9.95 59.9639

220 0.132272 9.95 59.9639

221 0.125659 9.95 59.9639

222 0.119376 9.95 59.9639

223 0.113407 9.95 59.9639

224 0.107737 9.95 59.9639

225 0.10235 9.95 59.9639

Result Xmin = 9.95281 Fmin = 59.8522

**Таблица 2** – результаты программы для мультимодальной функции

N T x f(x)

1 10000 9.2019 -3.23353

2 9500 8.32484 94.8095

3 9025 9.49579 -119.246

4 8573.75 9.81029 -29.2018

5 8145.06 9.98939 -61.9173

6 7737.81 8.9193 -18.3007

7 7350.92 8.82783 74.4069

8 6983.37 8.80388 21.3605

9 6634.2 7.96228 5.14668

10 6302.49 9.09958 146.815

11 5987.37 8.26559 113.799

12 5688 9.99622 -79.986

13 5403.6 9.91788 -16.3244

14 5133.42 9.69838 -38.3392

15 4876.75 8.57015 -70.317

16 4632.91 7.18786 -141.34

17 4401.27 9.4554 -111.39

18 4181.2 7.52807 -13.2227

19 3972.14 8.66101 -8.58505

20 3773.54 8.85738 -93.872

21 3584.86 7.25905 40.2729

22 3405.62 9.36746 -118.948

23 3235.34 7.17207 26.2675

24 3073.57 9.87555 -107.815

25 2919.89 9.08837 -48.9315

26 2773.9 7.53527 114.193

27 2635.2 7.74507 -3.33729

28 2503.44 7.70844 138.65

29 2378.27 8.65805 119.341

30 2259.36 8.90596 -95.7314

31 2146.39 7.44082 67.7907

32 2039.07 7.4897 -65.7255

33 1937.11 9.77901 -35.4123

34 1840.26 9.51075 -66.2473

35 1748.25 9.51075 -34.5962

36 1660.83 9.38105 -34.5962

37 1577.79 8.63855 19.9429

38 1498.9 9.87614 -107.524

39 1423.96 9.53352 -48.7935

40 1352.76 7.19139 -42.1836

41 1285.12 7.3686 -112.103

42 1220.87 9.16781 -99.3182

43 1159.82 8.9854 103.97

44 1101.83 9.36213 100.108

45 1046.74 8.75464 28.7555

46 994.403 7.39659 -29.6065

47 944.682 8.9576 -88.0944

48 897.448 8.12806 90.8048

49 852.576 7.8549 34.4352

50 809.947 8.40594 167.715

51 769.45 7.68866 -153.645

52 730.977 7.30188 107.373

53 694.428 9.21562 -115.96

54 659.707 8.31978 90.4499

55 626.722 7.18847 -116.301

56 595.386 8.81127 -111.517

57 565.616 7.74104 10.2187

58 537.335 7.09844 136.727

59 510.469 7.09844 -83.9313

60 484.945 7.02459 -83.9313

61 460.698 9.89965 -51.9406

62 437.663 7.54399 -43.065

63 415.78 9.70628 3.07641

64 394.991 9.49141 -70.4115

65 375.241 7.04259 -27.5621

66 356.479 7.60031 -60.1803

67 338.655 7.56684 45.2973

68 321.723 7.56684 20.1182

69 305.636 9.44992 20.1182

70 290.355 9.54759 -10.9194

71 275.837 7.12806 -46.4683

72 262.045 9.10883 -94.8058

73 248.943 7.8595 113.201

74 236.496 7.93527 167.847

75 224.671 9.63654 156.703

76 213.437 9.63654 -65.6085

77 202.765 8.63629 -65.6085

78 192.627 8.63629 -108.841

79 182.996 8.63629 -108.841

80 173.846 7.23401 -108.841

81 165.154 7.23401 -117.991

82 156.896 7.22595 -117.991

83 149.051 7.22595 -117.277

84 141.599 7.00459 -117.277

85 134.519 8.25816 -42.6223

86 127.793 8.25816 -74.4051

87 121.403 7.37192 -74.4051

88 115.333 7.37192 -98.1109

89 109.566 7.37192 -98.1109

90 104.088 7.37192 -98.1109

91 98.8836 9.91102 -98.1109

92 93.9395 9.91102 -40.1427

93 89.2425 9.91102 -40.1427

94 84.7804 9.91102 -40.1427

95 80.5413 7.41185 -40.1427

96 76.5143 9.78867 -80.9898

97 72.6886 7.03674 -65.0588

98 69.0541 8.45819 -57.5216

99 65.6014 8.45819 -161.731

100 62.3214 7.11248 -161.731

101 59.2053 7.27522 -89.2698

102 56.245 7.27522 -118.519

103 53.4328 8.33919 -118.519

104 50.7611 8.33919 -127.116

105 48.2231 8.33919 -127.116

106 45.8119 8.33919 -127.116

107 43.5213 8.33919 -127.116

108 41.3453 8.52978 -127.116

109 39.278 8.52978 -154.299

110 37.3141 8.52978 -154.299

111 35.4484 7.2643 -154.299

112 33.676 7.2643 -118.9

113 31.9922 7.2643 -118.9

114 30.3926 7.30193 -118.9

115 28.8729 7.30193 -115.953

116 27.4293 8.6168 -115.953

117 26.0578 8.6168 -119.673

118 24.7549 8.6168 -119.673

119 23.5172 8.6168 -119.673

120 22.3413 8.6168 -119.673

121 21.2243 8.6168 -119.673

122 20.1631 8.6168 -119.673

123 19.1549 8.6168 -119.673

124 18.1972 8.6168 -119.673

125 17.2873 8.6168 -119.673

126 16.4229 8.6168 -119.673

127 15.6018 8.6168 -119.673

128 14.8217 8.6168 -119.673

129 14.0806 8.6168 -119.673

130 13.3766 8.6168 -119.673

131 12.7078 8.6168 -119.673

132 12.0724 8.6168 -119.673

133 11.4687 8.6168 -119.673

134 10.8953 8.6168 -119.673

135 10.3505 8.6168 -119.673

136 9.83302 8.6168 -119.673

137 9.34136 8.44683 -119.673

138 8.8743 8.44683 -160.956

139 8.43058 8.44683 -160.956

140 8.00905 8.44683 -160.956

141 7.6086 8.44683 -160.956

142 7.22817 8.44683 -160.956

143 6.86676 8.44683 -160.956

144 6.52342 8.44683 -160.956

145 6.19725 8.44683 -160.956

146 5.88739 8.44683 -160.956

147 5.59302 8.44683 -160.956

148 5.31337 8.44683 -160.956

149 5.0477 8.44683 -160.956

150 4.79532 8.44683 -160.956

151 4.55555 8.44683 -160.956

152 4.32777 8.44683 -160.956

153 4.11138 8.44683 -160.956

154 3.90581 8.44683 -160.956

155 3.71052 8.435 -160.956

156 3.525 8.435 -159.565

157 3.34875 8.435 -159.565

158 3.18131 8.435 -159.565

159 3.02224 8.435 -159.565

160 2.87113 8.435 -159.565

161 2.72758 8.435 -159.565

162 2.5912 8.435 -159.565

163 2.46164 8.435 -159.565

164 2.33856 8.435 -159.565

165 2.22163 8.435 -159.565

166 2.11055 8.435 -159.565

167 2.00502 8.435 -159.565

168 1.90477 8.435 -159.565

169 1.80953 8.435 -159.565

170 1.71905 8.435 -159.565

171 1.6331 8.435 -159.565

172 1.55145 8.435 -159.565

173 1.47387 8.435 -159.565

174 1.40018 8.435 -159.565

175 1.33017 8.435 -159.565

176 1.26366 8.435 -159.565

177 1.20048 8.435 -159.565

178 1.14045 8.435 -159.565

179 1.08343 8.435 -159.565

180 1.02926 8.435 -159.565

181 0.977798 8.435 -159.565

182 0.928908 8.435 -159.565

183 0.882462 8.435 -159.565

184 0.838339 8.435 -159.565

185 0.796422 8.46621 -159.565

186 0.756601 8.46621 -161.951

187 0.718771 8.46621 -161.951

188 0.682833 8.46621 -161.951

189 0.648691 8.46621 -161.951

190 0.616256 8.46621 -161.951

191 0.585444 8.46621 -161.951

192 0.556171 8.46621 -161.951

193 0.528363 8.46621 -161.951

194 0.501945 8.46621 -161.951

195 0.476847 8.46621 -161.951

196 0.453005 8.46621 -161.951

197 0.430355 8.46621 -161.951

198 0.408837 8.46621 -161.951

199 0.388395 8.46621 -161.951

200 0.368975 8.46621 -161.951

201 0.350527 8.46621 -161.951

202 0.333 8.46621 -161.951

203 0.31635 8.46621 -161.951

204 0.300533 8.46621 -161.951

205 0.285506 8.46621 -161.951

206 0.271231 8.46621 -161.951

207 0.257669 8.46621 -161.951

208 0.244786 8.46621 -161.951

209 0.232547 8.46621 -161.951

210 0.220919 8.46621 -161.951

211 0.209873 8.46621 -161.951

212 0.19938 8.46621 -161.951

213 0.189411 8.46621 -161.951

214 0.17994 8.46621 -161.951

215 0.170943 8.46621 -161.951

216 0.162396 8.46621 -161.951

217 0.154276 8.46621 -161.951

218 0.146562 8.46621 -161.951

219 0.139234 8.46621 -161.951

220 0.132272 8.46621 -161.951

221 0.125659 8.46621 -161.951

222 0.119376 8.46621 -161.951

223 0.113407 8.46621 -161.951

224 0.107737 8.46621 -161.951

225 0.10235 8.46621 -161.951

Result Xmin = 8.46621 Fmin = -161.951

# 4. Выводы

Как видно из полученных результатов, применимость метода имитации отжига не зависит от того, является ли функция унимодальной или мультимодальной. Принципиальным в алгоритме SA является то, что, в отличие от большинства других стохастических алгоритмов поисковой оптимизации, он допускает шаги, приводящие к увеличению значений фитнес-функции.

# Приложение 1. Код программы main.cpp.

#include <iostream>

#include <math.h>

#include <random>

double f(const double &x) {

double f = pow(x, 2) \* exp(sin(x)) ;

return f;

}

double fm(const double &x) {

double f = (pow(x, 2) \* exp(sin(x))) \* (sin(5 \* x));

return f;

}

double randPoint(int a, int b) {

std::random\_device rd;

std::mt19937 gen(rd());

std::uniform\_real\_distribution<> dist(a, b);

return dist(gen);

}

double P(double Ti, double f, double fi) {

double P = pow(2.7, (-(fi - f)) / Ti);

return P;

}

void table(std::string type) {

const double Tmin = 0.1, Tmax = 10000;

double x, xi, Ti = Tmax, ff, ffi;

int i = 0;

x = randPoint(9, 14);

double mini = x, min = {type == "унимодальная" ? f(x) : fm(x)};

std::cout << " N T x f(x)\n";

while (Ti > Tmin) {

i++;

xi = randPoint(7, 10);

if (type == "унимодальная") {

ff = f(x);

ffi = f(xi);

} else {

ff = fm(x);

ffi = fm(xi);

}

if (ffi <= ff) {

x = xi;

} else {

if (randPoint(0, 1) < P(Ti, ff, ffi))

x = xi;

}

if (ff < min) {

min = {type == "унимодальная" ? f(x) : fm(x)};

mini = x;

}

std::cout.width(7);

std::cout << i;

std::cout.width(12);

std::cout << Ti;

std::cout.width(12);

std::cout << x;

std::cout.width(12);

std::cout << ff << '\n';

Ti \*= 0.95;

}

std::cout << "Result Xmin = " << mini << " Fmin = " << min << '\n';

}

int main() {

std::cout.setf(std::ios::left);

table("унимодальная");

table("модальная");

return 0;

}

**Ссылка на git-репозиторий:** https://github.com/SerdukovaM/TS\_lab3