Методы оптимизации
Лабораторная работа №1

Гранкин Максим M3237 Назаров Георгий Панов Иван M3239

Университет ИТМО, КТ.
Четвертый семестр. Февраль 2021

1 Задача оптимизации. Вариант №7

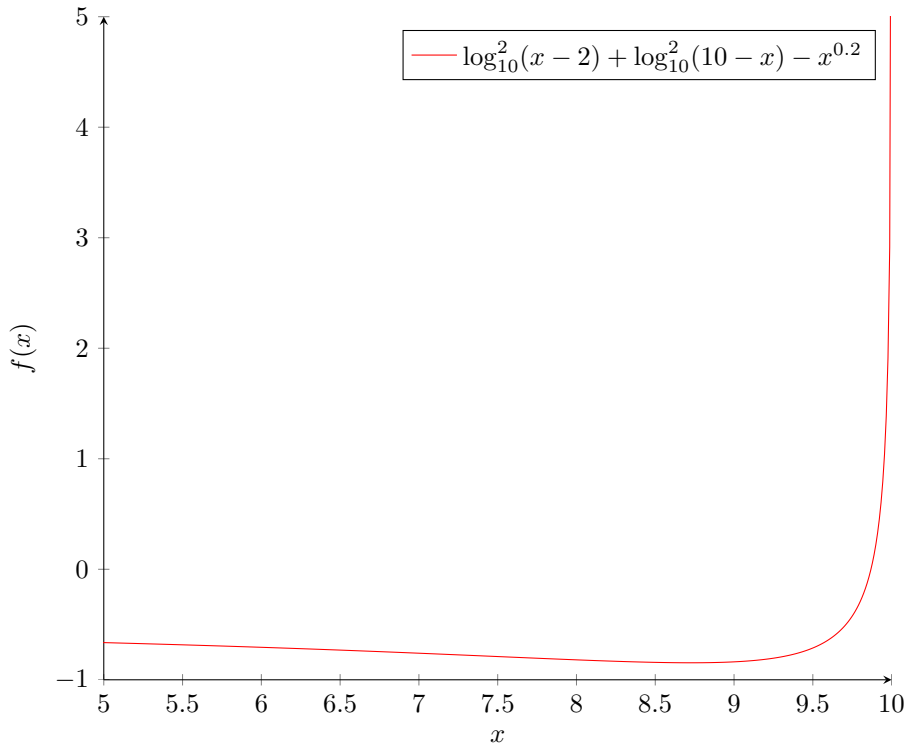
1.1 Постановка задачи

Изучить и реализовать алгоритмы одномерной минимизации функции:

- метод дихотомии
- метод золотого сечения
- метод Фибоначчи
- метод парабол
- комбинированный метод Брента

Протестировать реализованные алгоритмы на следующей функции:

$$f(x) = \log_{10}^2(x-2) + \log_{10}^2(10-x) - x^{0.2} \to min \text{ на } [6; 9.9]$$



1.2 Аналитическое решение

Найдём и классифицируем глобальные экстремумы следующей функции:

$$f(x) = \log_{10}(10-x) + \log(x-2) - x^{0.2}$$

Найдем критические точки функции $f(x)$:

Вычислим критические точки $-x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$

Область определения $f(x) = -x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$ это $\{x \in R : 2 < x < 10\}$:

$$f(x) = -x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10} \text{ если } 2 < x < 10$$

Чтобы найти точки, для начала посчитаем $f'(x)$:

$$\frac{d}{dx}\left(-x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}\right) = -\frac{0.2}{x^{0.8}} - \frac{2(\log(10-x))}{(10-x)\left(\log^2 10\right)} + \frac{2(\log(x-2))}{(x-2)\left(\log^2 10\right)} :$$

$$f'(x) = -\frac{0.2}{x^{0.8}} - \frac{2(\log(10-x))}{\left(\log^2 10\right)(10-x)} + \frac{2(\log(x-2))}{\left(\log^2 10\right)(x-2)}$$

Решаем $-\frac{0.2}{x^{0.8}} - \frac{2(\log(10-x))}{(10-x)\left(\log^2 10\right)} + \frac{2(\log(x-2))}{(x-2)\left(\log^2 10\right)} = 0$ получим $x = 8.72691$:

$$x = 8.72691$$

$f'(x)$ существует для всех x таких что $2 < x < 10$:

$-\frac{0.2}{x^{0.8}} - \frac{2(\log(10-x))}{\left(\log^2 10\right)(10-x)} + \frac{2(\log(x-2))}{\left(\log^2 10\right)(x-2)}$ существует для всех x таких что $2 < x < 10$

Единственная критическая точка $-x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$ это $x = 8.72691$:

$$x = 8.72691$$

Область определения $-x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$ это $\{x \in R : 2 < x < 10\}$:

Конечные точки $\{x \in R : 2 < x < 10\}$ это $x = 2$ и 10

Вычислим $-x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$ в точках $x = 2, 8.72691$ и 10 :

x	$f(x)$
2^+	∞
8.72691	-0.846037
10^-	∞

$f(x) = -x^{0.2} + \frac{\log^2(10-x)}{\log^2 10} + \frac{\log^2(x-2)}{\log^2 10}$ имеет один глобальный минимум

Наибольшее значение соответствует глобальному максимуму, а наименьшее значение соответствует глобальному максимуму :

x	$f(x)$	тип экстремума
2^+	∞	глобальный максимум
8.72691	-0.846037	глобальный минимум
10^-	∞	глобальный максимум

Удалим точки $x=$ 2^+и 10^-из таблицы

Они не могут быть глобальными экстремумами, так как значение $f(x)$ никогда не достигается :

x	$f(x)$	тип экстремума
8.72691	-0.846037	глобальный минимум

Ответ: $f(x)$ достигает глобальный минимум в точке $x = 8.72691$

2 Таблицы с результатами исследований

Тестирование производилось при $\varepsilon = 10^{-5}$.

2.1 Метод дихотомии

Тестирование производилось при $\delta = 10^{-6}$. Метод сошелся за 18 шагов. Найден минимум при $x = 8.7269039$ со значением $f(x) = -0.8460374$.

Шаг	$\|r - l\|$	ratio	l	r	$f(x_1)$	x_1	$f(x_2)$	x_2
1	3.9		6	9.9	-0.8167501	7.9499995	-0.8167502	7.9500005
2	1.9500005	0.5000001	7.9499995	9.9	-0.8419586	8.9249993	-0.8419586	8.9250003
3	0.9750008	0.5000003	7.9499995	8.9250003	-0.8403537	8.4374994	-0.8403538	8.4375004
4	0.4875009	0.5000005	8.4374994	8.9250003	-0.8458649	8.6812493	-0.8458649	8.6812503
5	0.2437509	0.500001	8.6812493	8.9250003	-0.8455011	8.8031243	-0.8455011	8.8031253
6	0.121876	0.5000021	8.6812493	8.8031253	-0.846017	8.7421868	-0.846017	8.7421878
7	0.0609385	0.5000041	8.6812493	8.7421878	-0.8460178	8.7117181	-0.8460178	8.7117191
8	0.0304697	0.5000082	8.7117181	8.7421878	-0.8460374	8.7269524	-0.8460374	8.7269534
9	0.0152354	0.5000164	8.7117181	8.7269534	-0.8460325	8.7193352	-0.8460325	8.7193362
10	0.0076182	0.5000328	8.7193352	8.7269534	-0.8460362	8.7231438	-0.8460362	8.7231448
11	0.0038096	0.5000656	8.7231438	8.7269534	-0.8460371	8.7250481	-0.8460371	8.7250491
12	0.0019053	0.5001312	8.7250481	8.7269534	-0.8460373	8.7260003	-0.8460373	8.7260013
13	0.0009531	0.5002624	8.7260003	8.7269534	-0.8460374	8.7264764	-0.8460374	8.7264774
14	0.0004771	0.5005246	8.7264764	8.7269534	-0.8460374	8.7267144	-0.8460374	8.7267154
15	0.000239	0.5010481	8.7267144	8.7269534	-0.8460374	8.7268334	-0.8460374	8.7268344
16	0.00012	0.5020917	8.7268334	8.7269534	-0.8460374	8.7268929	-0.8460374	8.7268939
17	0.0000605	0.504166	8.7268929	8.7269534	-0.8460374	8.7269227	-0.8460374	8.7269237
18	0.0000308	0.5082632	8.7268929	8.7269237	-0.8460374	8.7269078	-0.8460374	8.7269088

2.2 Метод золотого сечения

Метод сошелся за 26 шагов. Найден минимум при $x = 8.7269039$ со значением $f(x) = -0.8460374$.

Шаг	$\|r - l\|$	ratio	l	r	$f(x_1)$	x_1	$f(x_2)$	x_2
1	3.9		6	9.9	-0.7891519	7.4896674	-0.8393753	8.4103326
2	2.4103326	0.618034	7.4896674	9.9	-0.8393753	8.4103326	-0.8390312	8.9793349
3	1.4896674	0.618034	7.4896674	8.9793349	-0.8228178	8.0586698	-0.8393753	8.4103326
4	0.9206651	0.618034	8.0586698	8.9793349	-0.8393753	8.4103326	-0.8452592	8.6276721
5	0.5690023	0.618034	8.4103326	8.9793349	-0.8452592	8.6276721	-0.8459279	8.7619953
6	0.3516628	0.618034	8.6276721	8.9793349	-0.8459279	8.7619953	-0.8446975	8.8450117
7	0.2173396	0.618034	8.6276721	8.8450117	-0.8460151	8.7106884	-0.8459279	8.7619953
8	0.1343232	0.618034	8.6276721	8.7619953	-0.8458477	8.678979	-0.8460151	8.7106884
9	0.0830163	0.618034	8.678979	8.7619953	-0.8460151	8.7106884	-0.8460364	8.7302859
10	0.0513069	0.618034	8.7106884	8.7619953	-0.8460364	8.7302859	-0.8460164	8.7423978
11	0.0317094	0.618034	8.7106884	8.7423978	-0.8460359	8.7228003	-0.8460364	8.7302859
12	0.0195975	0.618034	8.7228003	8.7423978	-0.8460364	8.7302859	-0.8460318	8.7349123
13	0.0121119	0.618034	8.7228003	8.7349123	-0.8460374	8.7274267	-0.8460364	8.7302859
14	0.0074856	0.618034	8.7228003	8.7302859	-0.8460372	8.7256596	-0.8460374	8.7274267
15	0.0046263	0.618034	8.7256596	8.7302859	-0.8460374	8.7274267	-0.8460372	8.7285188
16	0.0028592	0.618034	8.7256596	8.7285188	-0.8460374	8.7267517	-0.8460374	8.7274267
17	0.0017671	0.618034	8.7256596	8.7274267	-0.8460374	8.7263346	-0.8460374	8.7267517
18	0.0010921	0.618034	8.7263346	8.7274267	-0.8460374	8.7267517	-0.8460374	8.7270095
19	0.000675	0.618034	8.7267517	8.7274267	-0.8460374	8.7270095	-0.8460374	8.7271689
20	0.0004172	0.618034	8.7267517	8.7271689	-0.8460374	8.7269111	-0.8460374	8.7270095
21	0.0002578	0.618034	8.7267517	8.7270095	-0.8460374	8.7268502	-0.8460374	8.7269111
22	0.0001593	0.618034	8.7268502	8.7270095	-0.8460374	8.7269111	-0.8460374	8.7269487
23	0.0000985	0.618034	8.7268502	8.7269487	-0.8460374	8.7268878	-0.8460374	8.7269111
24	0.0000609	0.618034	8.7268878	8.7269487	-0.8460374	8.7269111	-0.8460374	8.7269254
25	0.0000376	0.618034	8.7268878	8.7269254	-0.8460374	8.7269022	-0.8460374	8.7269111
26	0.0000232	0.618034	8.7268878	8.7269111	-0.8460374	8.7268967	-0.8460374	8.7269022

2.3 Метод Фибоначчи

Метод сошелся за 25 шагов. Найден минимум при $x = 8.7269113$ со значением $f(x) = -0.8460374$.

	length	ratio	left	right	value	argument	secondValue	secondArgument
1	3.3		6.6	9.9	-0.8115627	7.8604878	-0.8454277	8.6395122
2	2.0395122	0.618034	7.8604878	9.9	-0.8454277	8.6395122	-0.8260357	9.1209757
3	1.2604878	0.618034	7.8604878	9.1209757	-0.8366735	8.3419513	-0.8454277	8.6395122
4	0.7790243	0.618034	8.3419513	9.1209757	-0.8454277	8.6395122	-0.845161	8.8234149
5	0.4814635	0.618034	8.3419513	8.8234149	-0.8430986	8.525854	-0.8454277	8.6395122
6	0.2975608	0.618034	8.525854	8.8234149	-0.8454277	8.6395122	-0.8460124	8.7097567
7	0.1839027	0.618034	8.6395122	8.8234149	-0.8460124	8.7097567	-0.8459766	8.7531703
8	0.1136581	0.618034	8.6395122	8.7531703	-0.8458771	8.6829257	-0.8460124	8.7097567
9	0.0702446	0.618034	8.6829257	8.7531703	-0.8460124	8.7097567	-0.8460374	8.7263392
10	0.0434135	0.618034	8.7097567	8.7531703	-0.8460374	8.7263392	-0.8460292	8.7365878
11	0.026831	0.6180341	8.7097567	8.7365878	-0.8460333	8.7200053	-0.8460374	8.7263392
12	0.0165825	0.6180338	8.7200053	8.7365878	-0.8460374	8.7263392	-0.8460364	8.7302538
13	0.0102486	0.6180344	8.7200053	8.7302538	-0.8460366	8.7239199	-0.8460374	8.7263392
14	0.0063339	0.6180328	8.7239199	8.7302538	-0.8460374	8.7263392	-0.8460373	8.7278345
15	0.0039146	0.6180371	8.7239199	8.7278345	-0.8460372	8.7254152	-0.8460374	8.7263392
16	0.0024193	0.6180258	8.7254152	8.7278345	-0.8460374	8.7263392	-0.8460374	8.7269105
17	0.0014953	0.6180555	8.7263392	8.7278345	-0.8460374	8.7269105	-0.8460374	8.7272633
18	0.000924	0.6179775	8.7263392	8.7272633	-0.8460374	8.7266921	-0.8460374	8.7269105
19	0.0005712	0.6181818	8.7266921	8.7272633	-0.8460374	8.7269105	-0.8460374	8.7270449
20	0.0003528	0.6176471	8.7266921	8.7270449	-0.8460374	8.7268265	-0.8460374	8.7269105
21	0.0002184	0.6190476	8.7268265	8.7270449	-0.8460374	8.7269105	-0.8460374	8.7269609
22	0.0001344	0.6153846	8.7268265	8.7269609	-0.8460374	8.7268769	-0.8460374	8.7269105
23	0.000084	0.625	8.7268769	8.7269609	-0.8460374	8.7269105	-0.8460374	8.7269273
24	0.0000504	0.6	8.7268769	8.7269273	-0.8460374	8.7268937	-0.8460374	8.7269105
25	0.0000336	0.6666667	8.7268937	8.7269273	-0.8460374	8.7269105	-0.8460374	8.7269105

2.4 Метод парабол

Метод сошелся за 19 шагов. Найден минимум при $x = 8.726906$ со значением $f(x) = -0.8460374$.

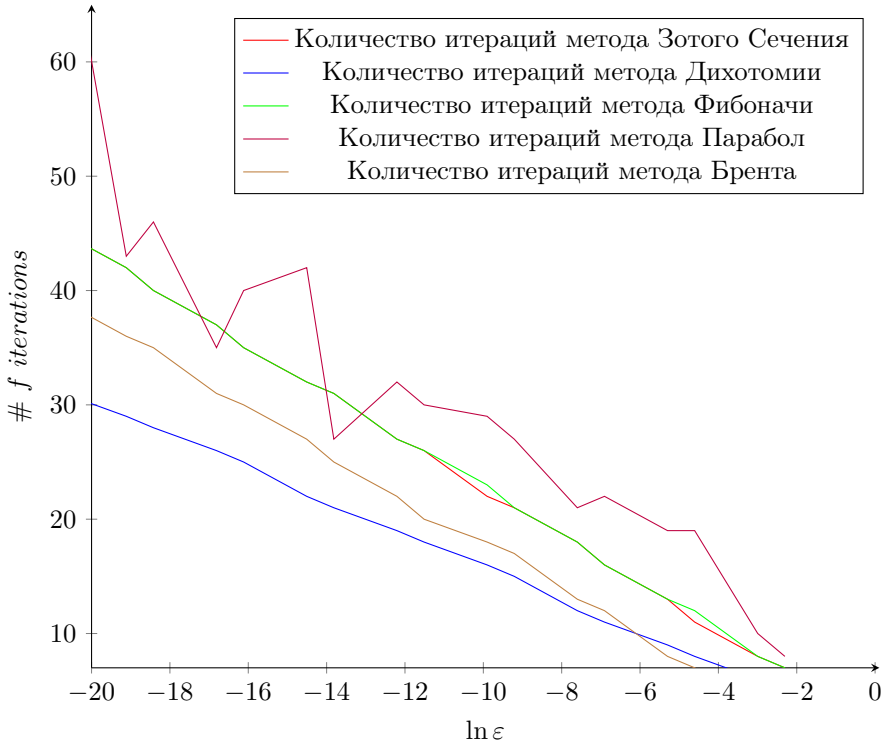
	$\|r-l\|$	ratio	l	r/x_3	$f(x_1)$	x_1	a0	a1	a2	u	$f(u)$	$f(x_2)$	$f(x_3)$	x_2
1	3.9		6	9.9	-0.7060166	6	-0.7060166	-0.056728	0.1445884	7.1253421	-0.7670129	-0.8114367	0.2240237	7.8583429
2	2.7746579	0.7114507	7.1253421	9.9	-0.7670129	7.1253421	-0.7670129	-0.0599189	0.1512877	7.332224	-0.7795237	-0.7680738	0.2240237	7.1430464
3	2.7569536	0.9936193	7.1430464	9.9	-0.7680738	7.1430464	-0.7680738	-0.0541414	0.3025344	7.9267929	-0.8154194	-0.8432509	0.2240237	8.5315798
4	1.9732071	0.7157201	7.9267929	9.9	-0.8154194	7.9267929	-0.8154194	-0.0439659	0.4360135	8.3093114	-0.8352759	-0.8446216	0.2240237	8.5909937
5	1.5906886	0.8061438	8.3093114	9.9	-0.8352759	8.3093114	-0.8352759	-0.0326824	0.5379072	8.4856476	-0.8419379	-0.8448164	0.2240237	8.6012254
6	1.4143524	0.8891447	8.4856476	9.9	-0.8419379	8.4856476	-0.8419379	-0.0061298	0.7322042	8.6781622	-0.8458413	-0.8442468	0.2240237	8.8623051
7	0.3766575	0.2663109	8.4856476	8.8623051	-0.8419379	8.4856476	-0.8419379	-0.0092524	0.0880306	8.7087929	-0.8460096	-0.8450947	-0.8442468	8.8268332
8	0.3411856	0.9058246	8.4856476	8.8268332	-0.8419379	8.4856476	-0.8419379	-0.0199903	0.0744366	8.7183903	-0.8460312	-0.8458746	-0.8450947	8.6825776
9	0.1442556	0.4228068	8.6825776	8.8268332	-0.8458746	8.6825776	-0.8458746	-0.0058385	0.0887355	8.7242421	-0.8460368	-0.845977	-0.8450947	8.7001104
10	0.1267229	0.8784607	8.7001104	8.8268332	-0.845977	8.7001104	-0.845977	-0.0035755	0.090943	8.7251944	-0.8460371	-0.8460158	-0.8450947	8.7109622
11	0.115871	0.9143657	8.7109622	8.8268332	-0.8460158	8.7109622	-0.8460158	-0.0004342	0.0939817	8.7266074	-0.8460374	-0.8460274	-0.8450947	8.737633
12	0.0266708	0.230177	8.7109622	8.737633	-0.8460158	8.7109622	-0.8460158	-0.0012559	0.0858161	8.7268272	-0.8460374	-0.8460373	-0.8460274	8.7280575
13	0.0170953	0.6409723	8.7109622	8.7280575	-0.8460158	8.7109622	-0.8460158	-0.0013805	0.0849713	8.7269	-0.8460374	-0.8460374	-0.8460373	8.7265907
14	0.0014668	0.0858006	8.7265907	8.7280575	-0.8460374	8.7265907	-0.8460374	-0.0000346	0.0861747	8.7269059	-0.8460374	-0.8460374	-0.8460373	8.7268198
15	0.0012377	0.8437945	8.7268198	8.7280575	-0.8460374	8.7268198	-0.8460374	-0.0000088	0.0861977	8.726906	-0.8460374	-0.8460374	-0.8460373	8.7268903
16	0.0011672	0.9430681	8.7268903	8.7280575	-0.8460374	8.7268903	-0.8460374	-5e-7	0.0862052	8.7269061	-0.8460374	-0.8460374	-0.8460373	8.7269164
17	0.0000262	0.0224086	8.7268903	8.7269164	-0.8460374	8.7268903	-0.8460374	-0.0000016	0.086117	8.7269061	-0.8460374	-0.8460374	-0.8460374	8.7269032
18	0.0000132	0.5059045	8.7269032	8.7269164	-0.8460374	8.7269032	-0.8460374	-5e-7	0.0860653	8.7269061	-0.8460374	-0.8460374	-0.8460374	8.7269033
19	0.0000131	0.993045	8.7269033	8.7269164	-0.8460374	8.7269033	-0.8460374	0	0.0861189	8.7269061	-0.8460374	-0.8460374	-0.8460374	8.7269088

2.5 Метод Брента

Метод сошелся за 19 шагов. Найден минимум при $x = 8.7268806$ со значением $f(x) = -0.8460374$.

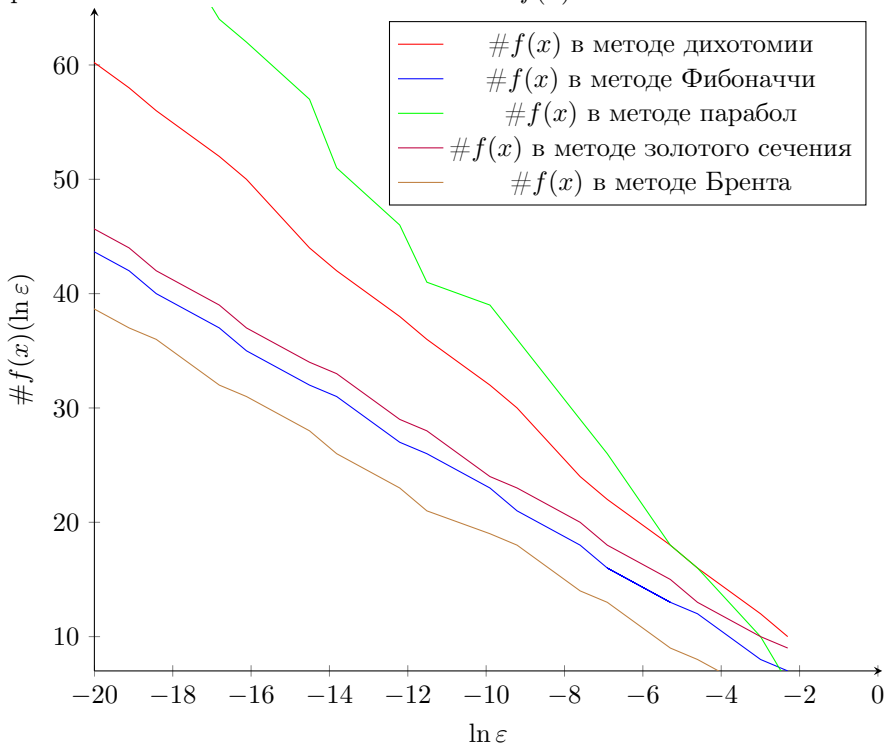
	$\|r-l\|$	ratio	l	r	1 min	2 min	p 1 min	1 min val	2 min val	p 2 min val	a0	a1	a2
1	3.9		6	9.9	7.4896674	7.4896674	7.4896674	-0.7891519	-0.7891519	-0.7891519			
2	2.4103326	0.618034	7.4896674	9.9	8.4103326	7.4896674	7.4896674	-0.8393753	-0.7891519	-0.7891519			
3	1.4896674	0.618034	7.4896674	8.9793349	8.4103326	8.9793349	7.4896674	-0.8393753	-0.8390312	-0.7891519	-0.8393753	0.0006046	0.0370256
4	0.5690874	0.3820231	8.4102475	8.9793349	8.4103326	8.4102475	8.9793349	-0.8393753	-0.8393721	-0.8390312			
5	0.5690023	0.9998505	8.4103326	8.9793349	8.6276721	8.4103326	8.4102475	-0.8452592	-0.8393753	-0.8393721			
6	0.3516628	0.618034	8.6276721	8.9793349	8.7619953	8.6276721	8.4103326	-0.8459279	-0.8452592	-0.8393753	-0.8459279	-0.0049787	0.0628263
7	0.1344118	0.382218	8.6276721	8.762084	8.7619953	8.762084	8.6276721	-0.8459279	-0.8459274	-0.8452592			
8	0.1343232	0.9993407	8.6276721	8.7619953	8.7106884	8.7619953	8.762084	-0.8460151	-0.8459279	-0.8459274			
9	0.0830163	0.618034	8.678979	8.7619953	8.7106884	8.7619953	8.762084	-0.8460151	-0.8459279	-0.8459274			
10	0.0513069	0.618034	8.7106884	8.7619953	8.7302859	8.7106884	8.7619953	-0.8460364	-0.8460151	-0.8459279	-0.8460364	-0.0010889	0.0878835
11	0.0196858	0.3836871	8.7106884	8.7303742	8.7302859	8.7303742	8.7106884	-0.8460364	-0.8460363	-0.8460151			
12	0.0075739	0.3847383	8.7228003	8.7303742	8.7302859	8.7303742	8.7228003	-0.8460364	-0.8460363	-0.8460359			
13	0.0074856	0.9883411	8.7228003	8.7302859	8.7274267	8.7302859	8.7303742	-0.8460374	-0.8460364	-0.8460363			
14	0.0046263	0.618034	8.7256596	8.7302859	8.7274267	8.7256596	8.7302859	-0.8460374	-0.8460372	-0.8460364			
15	0.0028592	0.618034	8.7256596	8.7285188	8.7274267	8.7256596	8.7285188	-0.8460374	-0.8460372	-0.8460372			
16	0.0017671	0.618034	8.7256596	8.7274267	8.7267517	8.7274267	8.7256596	-0.8460374	-0.8460374	-0.8460372			
17	0.0010921	0.618034	8.7263346	8.7274267	8.7267517	8.7274267	8.7263346	-0.8460374	-0.8460374	-0.8460374			
18	0.000675	0.618034	8.7267517	8.7274267	8.7270095	8.7267517	8.7274267	-0.8460374	-0.8460374	-0.8460374			
19	0.0004172	0.618034	8.7267517	8.7271689	8.7270095	8.7267517	8.7271689	-0.8460374	-0.8460374	-0.8460374			

3 Зависимость количества итераций от $\ln \varepsilon$



4 Зависимость количества вычислений функции от $\ln \varepsilon$

График зависимости количества вычислений $f(x)$ от $\ln \varepsilon$:



4.1 Метод дихотомии

Данные привидены при $\delta = 10^{-10}$.

ε	Вычислений $f(x)$	$x*$	$f(x*)$
0.100000000	10	8.74218749997969	-0.8460169934998543
0.050000000	12	8.71171874998047	-0.8460177831815177
0.010000000	16	8.719335937480274	-0.8460324786882153
0.005000000	18	8.723144531230176	-0.8460361662908555
0.001000000	22	8.726000976542604	-0.8460373102278015
0.000500000	24	8.726477050761343	-0.8460373648754617
0.000100000	30	8.726893615702739	-0.8460373807085817
0.000050000	32	8.726863861064068	-0.8460373805683742
0.000010000	36	8.72688617704307	-0.8460373806878243
0.000005000	38	8.726889896372905	-0.8460373806993943
0.000001000	42	8.72689268587028	-0.8460373807065082
0.000000500	44	8.726893150786509	-0.8460373807075637
0.000000100	50	8.72689355758821	-0.8460373807084567
0.000000050	52	8.726893586645474	-0.8460373807085194
0.000000010	56	8.726893579381159	-0.8460373807085035
0.000000005	58	8.726893583013316	-0.8460373807085114
0.000000001	62	8.726893585737434	-0.8460373807085175

4.2 Метод Фибоначчи

ε	Вычислений $f(x)$	$x*$	$f(x*)$
0.100000000	7	8.752941176470587	-0.845977627132542
0.050000000	8	8.694545454545455	-0.8459497376253309
0.010000000	12	8.731034482758622	-0.8460359075624683
0.005000000	13	8.730000000000002	-0.8460365541108549
0.001000000	16	8.727283281733742	-0.8460373684662246
0.000500000	18	8.72682926829268	-0.8460373802136981
0.000100000	21	8.726883483965524	-0.8460373806779596
0.000050000	23	8.726907030989668	-0.8460373807219252
0.000010000	26	8.726911277457358	-0.8460373807196901
0.000005000	27	8.726903383512019	-0.8460373807213654
0.000001000	31	8.726906625417282	-0.8460373807219762
0.000000500	32	8.726906266948653	-0.8460373807219976
0.000000100	35	8.726906048671534	-0.8460373807219997
0.000000050	37	8.726906120176544	-0.8460373807220001
0.000000010	40	8.726906118514858	-0.8460373807220001
0.000000005	42	8.726906138050609	-0.8460373807220001
0.000000001	45	8.726906117946992	-0.8460373807220001

4.3 Метод парабол

ε	Вычислений $f(x)$	$x*$	$f(x*)$
0.100000000	6	8.752941176470587	-0.845977627132542
0.050000000	10	8.694545454545455	-0.8459497376253309
0.010000000	16	8.731034482758622	-0.8460359075624683
0.005000000	18	8.730000000000002	-0.8460365541108549
0.001000000	26	8.727283281733742	-0.8460373684662246
0.000500000	29	8.72682926829268	-0.8460373802136981
0.000100000	36	8.726883483965524	-0.8460373806779596
0.000050000	39	8.726907030989668	-0.8460373807219252
0.000010000	41	8.726911277457358	-0.8460373807196901
0.000005000	46	8.726903383512019	-0.8460373807213654
0.000001000	51	8.726906625417282	-0.8460373807219762
0.000000500	57	8.726906266948653	-0.8460373807219976
0.000000100	62	8.726906048671534	-0.8460373807219997
0.000000050	64	8.726906120176544	-0.8460373807220001
0.000000010	73	8.726906118514858	-0.8460373807220001
0.000000005	75	8.726906138050609	-0.8460373807220001
0.000000001	80	8.726906117946992	-0.8460373807220001

4.4 Метод золотого сечения

ε	Вычислений $f(x)$	$x*$	$f(x*)$
0.100000000	9	8.694833721937705	-0.8459512704097805
0.050000000	10	8.720487176055734	-0.846033852677419
0.010000000	13	8.732599094114786	-0.8460345754264209
0.005000000	15	8.72654313508526	-0.8460373693803985
0.001000000	18	8.72654313508526	-0.8460373693803985
0.000500000	20	8.727089200570301	-0.846037377834328
0.000100000	23	8.726929860809168	-0.8460373806733724
0.000050000	24	8.726899429622671	-0.8460373807181703
0.000010000	28	8.726903869472945	-0.8460373807215724
0.000005000	29	8.72690661345132	-0.8460373807219773
0.000001000	33	8.726906213110272	-0.8460373807219989
0.000000500	34	8.726905965685896	-0.8460373807219986
0.000000100	37	8.72690608250384	-0.8460373807220001
0.000000050	39	8.726906140912812	-0.8460373807219999
0.000000010	42	8.726906140912812	-0.8460373807219999
0.000000005	44	8.726906139669506	-0.8460373807219999
0.000000001	47	8.726906138719702	-0.8460373807219999

4.5 Метод Брента

ε	Вычислений $f(x)$	$x*$	$f(x*)$
0.100000000	3	8.694833721937705	-0.8459512704097805
0.050000000	5	8.482075408034591	-0.8418273966634982
0.010000000	8	8.739143698621444	-0.8460243419626269
0.005000000	9	8.694833721937705	-0.8459512704097805
0.001000000	13	8.731926739667713	-0.8460352002314014
0.000500000	14	8.728856305600065	-0.846037052621843
0.000100000	18	8.726543135085262	-0.8460373693803985
0.000050000	19	8.726868531777464	-0.8460373806004717
0.000010000	21	8.7268806221151	-0.846037380666108
0.000005000	23	8.726902689392734	-0.8460373807209992
0.000001000	26	8.726899429622673	-0.8460373807181703
0.000000500	28	8.726906613451323	-0.8460373807219772
0.000000100	31	8.726906613451321	-0.8460373807219772
0.000000050	32	8.726906213110272	-0.8460373807219989
0.000000010	36	8.726906074468038	-0.846037380722
0.000000005	37	8.726906046405109	-0.846037380722
0.000000001	40	8.726906063448617	-0.8460373807220001

5 Выводы

1. Наилучшие результаты в плане точности метод Фибоначчи.
Относительная погрешность:

- Метод дихотомии — 0.000114 %
- Метод золотого сечения — 0.000069 %
- Метод Фибоначчи — 0.000015 %
- Метод парабол — 0.000047 %
- Метод Брента — 0.000337 %

2. Наилучшую сходимость показывают методы дихотомии и Брента (график в разделе 3).
3. Наименьшее количество вычислений минимизируемой функции достигается методами Брента и золотого сечения. Метод парабол показывает худшие результаты по сравнению с этими методами, потому что на почти всем промежутке минимизируемая фукнция ведет себя почти как горизонтальная линия.
4. Все перечисленные методы могут найти (но не гарантируют этого) минимум для многомодальных функций, если в процессе выполнения попадут в нужный унимодальный промежуток.

6 Многомодальные функции

Для тестирования возьмем функцию

$$h = 2.2, f = -10.0, g = -0.8, a = -10, b = 6.1, c = 10, d = 2.3$$

$$hx^8 + ax^7 + bx^6 + cx^5 + dx^4 + ex^3 + fx^2 + gx$$

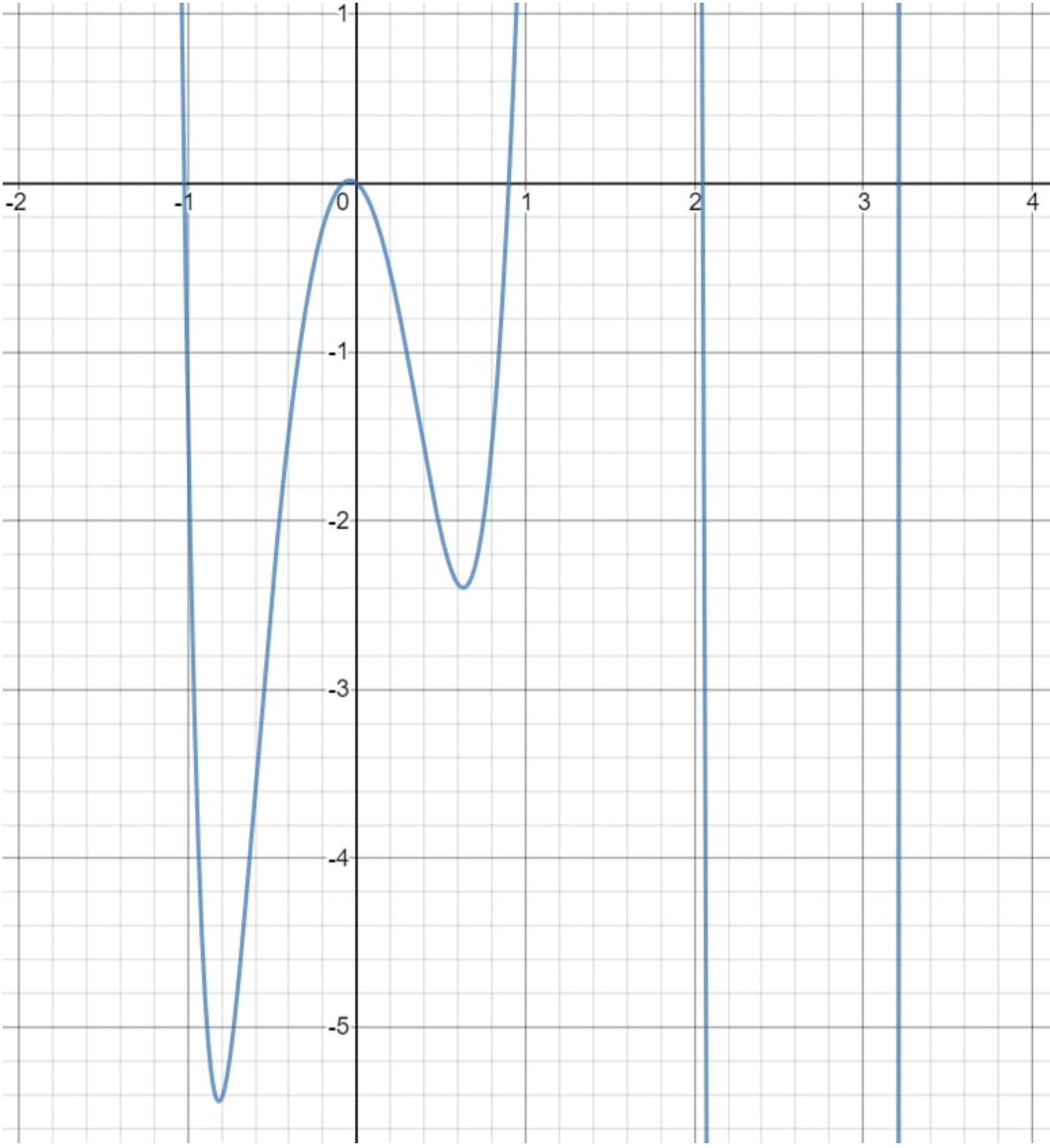
Ниже приведем результаты работы наших методов.
Границы: $(-1.0, 2.0)$

Метод оптимизации	x	$f(x)$
Метод золотого сечения	0.6310257923957638	-2.398552721226936
Метод дихотомии	0.6310256710368249	-2.39855272122661
Метод Фибоначчи	0.6310257927935599	-2.3985527212269364
Метод парабол	0.6310261520102378	-2.3985527212238997
Метод Брента	0.631025793319578	-2.3985527212269364

Границы: $(-1.0, 0.925)$

Метод оптимизации	x	$f(x)$
Метод золотого сечения	-0.8158158144523461	-5.438575458377366
Метод дихотомии	0.6308119767114729	-2.3964728000603026
Метод Фибоначчи	-0.8158158143246539	-5.4385754583773664
Метод парабол	0.6308122161575884	-2.3964728000596947
Метод Брента	-0.8158158154976147	-5.4385754583773664

Внимательный читатель может заметить что результаты наших методов не являются верными. Наши методы расчитывают на унимодальность функций, однако так как данная функция не унимодальна, то нельзя расчитывать корректность выполнения алгоритмов. Это связано с тем что каждый метод считает, что за одной из границ локальный минимум не достигается. В данной функции локальный минимум достигается на отсекаемой границе.


1
-2
-1
0
1
2
3
4
-1
-2
-3
-4
-5

7 Програмный код

7.1 Метод Дихотомии

```
public void calculate() {
    parameters.clear();
    double b = right, a = left;

    while ((b - a) / 2.0 > eps) {
        double x1 = (b + a - delta) / 2.0;
        double x2 = (b + a + delta) / 2.0;

        double f1 = function.apply(x1);
        double f2 = function.apply(x2);

        parameters.add(new DichotomyParameters(a, b, f1, x1, f2, x2));

        if (f1 <= f2) {
            b = x2;
        } else {
            a = x1;
        }
    }

    minArgument = (a + b) / 2.0;
    minValue = function.apply(minArgument);
}
```

7.2 Метод Фибоначчи

```
private void calculateFibonacciNumbers() {
    fibonacciNumbers.add(1.0);
    fibonacciNumbers.add(1.0);
    fibonacciNumbers.add(2.0);

    stepsCount = 0;
    double boundingValue = (right - left) / eps;

    while (boundingValue >= fibonacciNumbers.get(stepsCount + 2)) {
        Double f1 = fibonacciNumbers.get(fibonacciNumbers.size() - 2);
        Double f2 = fibonacciNumbers.get(fibonacciNumbers.size() - 1);
        fibonacciNumbers.add(f1 + f2);
        stepsCount++;
    }
}

public void calculate() {
    calculateFibonacciNumbers();
    validate();
    final double startDelta = right - left;
    double x1 = left + fibonacciNumbers.get(stepsCount - 1) / fibonacciNumbers.get(stepsCount +
       1) * startDelta;
    double x2 = left + right - x1;
    double f1 = function.apply(x1);
    double f2 = function.apply(x2);
    for (int i = 1; i < stepsCount; i++) {
        if (f1 > f2) {
            left = x1;
            x1 = x2;
            f1 = f2;
            x2 = left + fibonacciNumbers.get(stepsCount - i - 1)/
                fibonacciNumbers.get(stepsCount - i) * (right - left);
            f2 = function.apply(x2);
        } else {
            right = x2;
            x2 = x1;
            f2 = f1;
            x1 = left + fibonacciNumbers.get(stepsCount - i - 2)/
                fibonacciNumbers.get(stepsCount - i) * (right - left);
```

```
            f1 = function.apply(x1);
        }
    }
    minArgument = (x1 + x2) / 2.0;
    minValue = function.apply(minArgument);
}
```

7.3 Метод парабол

```
private Double findX2(Double x1, Double x3) {
    return Math.random() * (x3 - x1) + x1;
}

public static class Parabola {
    private final Double a0;
    private final Double a1;
    private final Double a2;
    private final Double xMin;
    private final Double fxMin;

    Parabola(Function<Double, Double> function, Double x1, Double x2, Double x3, Double f1,
        Double f2, Double f3) {
        a0 = f1;
        a1 = (f2 - f1) / (x2 - x1);
        a2 = ((f3 - f1) / (x3 - x1) - (f2 - f1) / (x2 - x1)) / (x3 - x2);

        xMin = (x1 + x2 - a1 / a2) / 2.0;
        fxMin = function.apply(xMin);
    }

    public Double getA0() {
        return a0;
    }

    public Double getA1() {
        return a1;
    }

    public Double getA2() {
        return a2;
    }

    public Double getxMin() {
        return xMin;
    }

    public Double getFxMin() {
        return fxMin;
    }
}

public void calculate() {
    Double x1 = left, x3 = right;

    Double f1 = function.apply(x1);
    Double f3 = function.apply(x3);
    while (x3 - x1 > eps) {
        Double x2;
        Double f2;
        Double a0;
        Double a1;
        Double a2;

        Double xMin;
        Double fxMin;

        do {
            do {
```

```
                x2 = findX2(x1, x3);
                f2 = function.apply(x2);
            } while (f1 < f2 || f2 > f3);
            Parabola parabola = new Parabola(function, x1, x2, x3, f1, f2, f3);
            a0 = parabola.getA0();
            a1 = parabola.getA1();
            a2 = parabola.getA2();
            xMin = parabola.getxMin();
            fxMin = parabola.getFxMin();
        } while (Math.abs(xMin - x2) < eps / 4);

        left = x1;
        right = x3;
        Parameters step = new ParabolaParameters(left, right,
                f1, x1,
                a0, a1, a2,
                xMin, fxMin,
                f2, f3,
                x2, x3);

        parameters.add(step);

        if (fxMin < f2) {
            if (xMin < x2) {
                x3 = x2;
                f3 = f2;
            } else {
                x1 = x2;
                f1 = f2;
            }
        } else {
            if (xMin < x2) {
                x1 = xMin;
                f1 = fxMin;
            } else {
                x3 = xMin;
                f3 = fxMin;
            }
        }
    }

    minArgument = (x3 + x1) / 2.0;
    minValue = function.apply(minArgument);

}
```

7.4 Метод золотого сечения

```
private static final double sqrt5 = Math.sqrt(5.0);
private static final double tau = (sqrt5 - 1.0) / 2.0;
private static final double tauInv = (3.0 - sqrt5) / 2.0;

public void calculate() {
    parameters.clear();
    double length = intervalLength();
    double x1 = left + tauInv * length;
    double x2 = left + tau * length;
    double f1 = function.apply(x1);
    double f2 = function.apply(x2);

    double currentEps = length / 2;
    while (currentEps > eps) {
        parameters.add(new GoldenRatioParameters(
                left,
                right,
                f1, x1, f2, x2
        ));
        currentEps *= tau;
        if (f1 < f2) {
            f2 = f1;
            right = x2;
            x2 = x1;
            x1 = right - tau * intervalLength();
            f1 = function.apply(x1);
        } else {
            f1 = f2;
            left = x1;
            x1 = x2;
            x2 = left + tau * intervalLength();
            f2 = function.apply(x2);
        }
    }
    minArgument = (right + left) / 2.0;
    minValue = function.apply(minArgument);
}
```

7.5 Метод Брента

```
private boolean equals(final double a, final double b) {
    return Math.abs(a - b) < eps;
}

private boolean pointsNotEquals(final double a, final double b, final double c) {
    return !equals(a, b) && !equals(a, c) && !equals(b, c);
}
public void calculate() {
    double a = left;
    double c = right;
    double x = a + K * (c - a);
    double w = x;
    double v = x;
    double fx = function.apply(x);
    double fw = function.apply(x);
    double fv = function.apply(x);
    double d = c - a;
    double e = c - a;

    while (true) {
        double g = e;
        e = d;
        double tol = eps * Math.abs(x)  + eps / 10.0;
        if (Math.abs(x - (a + c) / 2.0) + (c - a) / 2.0 < 2 * tol) {
            break;
        }
        boolean parabolaMethodPassed = false;
        double u = a;

        if (pointsNotEquals(x, w, v) && pointsNotEquals(fx, fw, fv)) {
            Parabola parabola = new Parabola(function, x, w, v, fx, fw, fv);
            u = parabola.getxMin();
            if (a < u && u < c && Math.abs(u - x) < g / 2.0) {
                u = x - Math.signum(x - (a + c) / 2.0) * tol;
                parabolaMethodPassed = true;
            }
        }

        if (!parabolaMethodPassed) {
            if (x < (a + c) / 2.0) {
                u = x + K * (c - x);
                e = c - x;
            } else {
                u = x - K * (x - a);
                e = x - a;
            }
        }

        if (Math.abs(u - x) < tol) {
            u = x + Math.signum(u - x) * tol;
        }
        d = Math.abs(u - x);
        double fu = function.apply(u);
        if (fu < fx) {
            if (u > x) {
                a = x;
            } else {
                c = x;
            }
            v = w;
            w = x;
            x = u;
            fv = fw;
            fw = fx;
            fx = fu;
        } else {
            if (u > x) {
```

```
                c = u;
            } else {
                a = u;
            }
            if (fu < fw || equals(w, x)) {
                v = w;
                w = u;
                fv = fw;
                fw = fu;
            } else if (fu < fv || equals(v, x) || equals(v, w)) {
                v = u;
                fv = fu;
            }
        }
    }

    minArgument = (a + c) / 2.0;
    minValue = function.apply(minArgument);
}
```

7.6 Диаграмма классов