Практическое задание по методам оптимизации

Гаврилов Олег Алексеевич

Рассматриваются 5 двумерных функций на квадрате $0 \le x, y \le 10$:

$$f_a(x, y) = (x - 3)^2 + y^2 + 1$$

$$f_b(x, y) = (x + 1)^2 + (2y - 8)^2 + 1$$

$$f_c(x, y) = x^2 + xy + y^2 - 16x - 17y + 94$$

$$f_d(x, y) = ((x - 7)^2 + (y - 8)^2) \cdot ((x - 8)^2 + (y - 7)^2) + 4$$

$$f_e(x, y) = (4xy - 19)^2 \cdot (\cos^2 \pi x + \cos^2 \pi y) - x - y + 24$$

Требуется найти минимальное значение каждой функции с некоторой точностью, а также число итераций (вычислений функции) n, которые потребовались для достижения заданной точности. Были выбраны три численных метода поиска минимума:

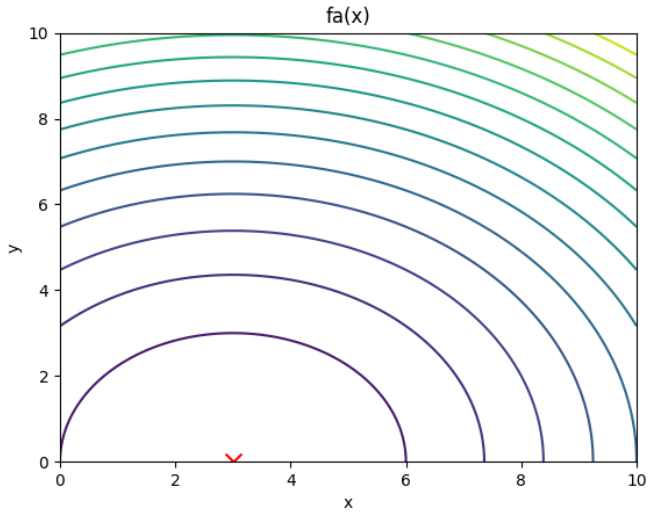
- двумерный полный перебор,
- метод переменных направлений с одномерным поиском методом бисекции,
- градиентный спуск (с дроблением шага и проекциями на область).

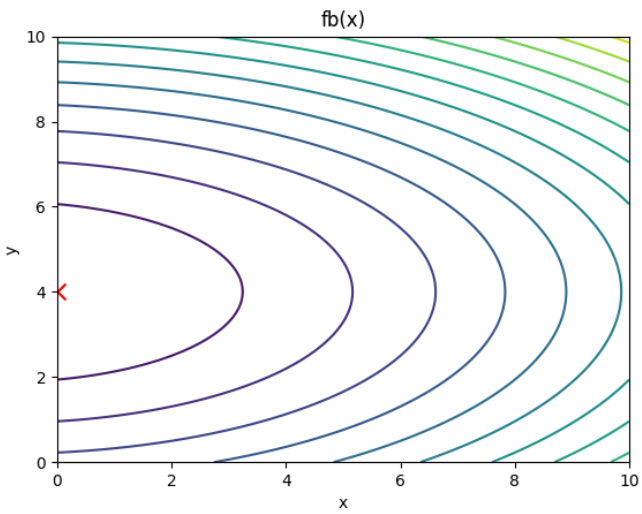
В последних двух методах в качестве начальной точки выбрано $(x_0, y_0) = (1, 2)$. Решение требуется вычислить с точностью $\varepsilon = 10^{-3}$.

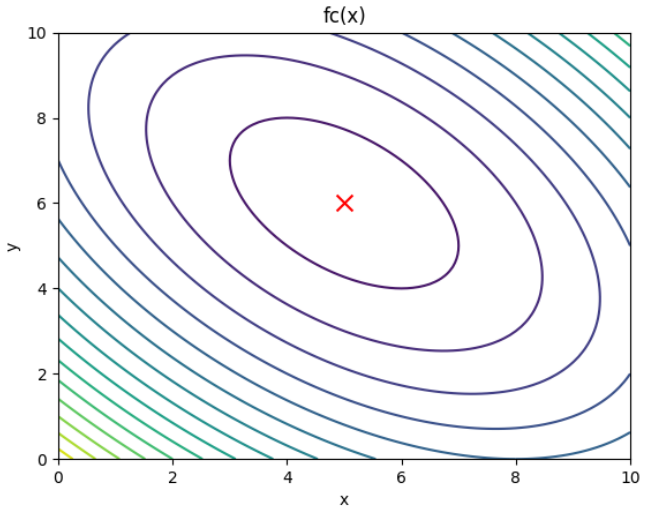
Для функций f_a, f_b, f_c, f_d точки минимума можно найти аналитически:

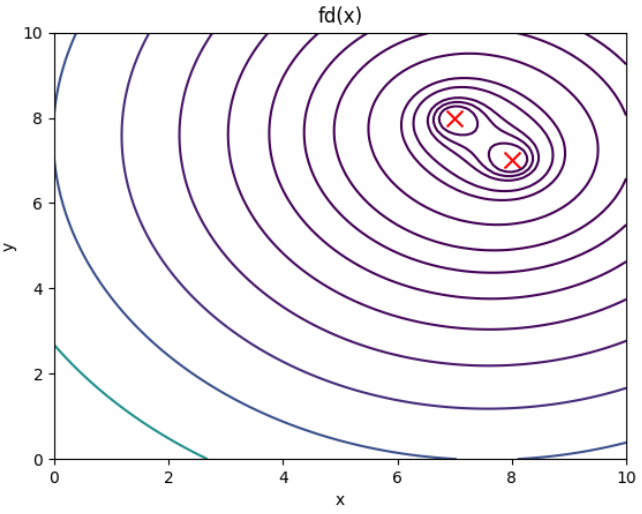
- $\arg\min f_a(x, y) = (3, 0)$, $\quad \min f_a = 1$
- $\arg\min f_b(x, y) = (0, 4)$, $\quad \min f_b = 2$
- $\arg\min f_c(x, y) = (5, 6)$, $\quad \min f_c = 3$
- $\arg\min f_d(x, y) = \{(7, 8),\ (8, 7)\}$, $\quad \min f_d = 4$

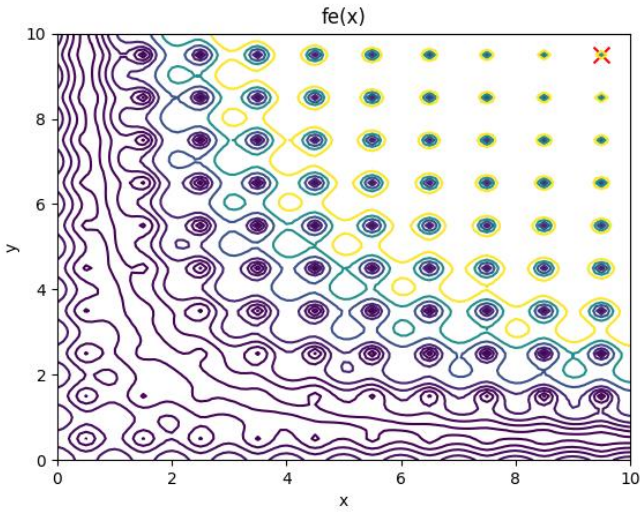
Вычисления реализованы на Python с использованием бибилотеки NumPy; визуализация – с помощью Matplotlib. Ниже представлены графики изолиний каждой из пяти функций.


fa(x)
x
y


fb(x)
x
y


fc(x)
x
y


fd(x)
x
y


fe(x)
x
y

Результаты вычислений

Метод	f	x^*	y^*	$f(x^*, y^*)$	n	**Примечание**
полный перебор 2D	f_a f_b f_c f_d f_e	3.0003 0.0000 5.0005 6.9997 9.4999	0.0000 4.0004 5.9996 7.9998 9.4999	1.0000 2.0000 3.0000 4.0000 5.0059	10^8 10^8 10^8 10^8 10^8	$\varepsilon = 0.001$ время вычислений на ноутбуке ~15 мин
переменных направлений с делением пополам	f_a f_b f_c f_d f_e	2.9999 0.0003 5.0000 7.0001 2.5015	0.0003 3.9999 6.0001 7.9999 2.5015	1.0000 2.0006 3.0000 4.0000 18.9986	112 112 448 224 2240	$\varepsilon = 0.001$
градиентный спуск	f_a f_b f_c f_d f_e	2.9997 0.0000 4.9998 7.4996 1.5093	0.0003 4.0000 5.9998 7.4996 3.5100	1.0000 2.0000 3.0000 4.2500 18.9895	39 24 28 141 1246	$\varepsilon = 0.001$, начальное значение шага $\alpha = 0.1$, если f увеличилось, то $\alpha := \alpha/2$

Выводы:

- Метод полного перебора наиболее точный в отношении функции $f_e(x)$. Поскольку она сильно осциллирует, другие два метода находят лишь некоторый локальный минимум.
- Градиентный спуск может остановиться не в точке локального экстремума, а в седловой точке, как в случае функции $f_d(x)$.
- Метод деления пополам считается методом нулевого порядка, поэтому число требуемых итераций получилось в разы больше, чем в методе градиентного спуска, который имеет первый порядок.