Национальный исследовательский университет ИТМО
Факультет информационных технологий и программирования
Прикладная математика и информатика

Методы оптимизации
Отчет по лабораторной работе №3

Работу выполнили:
Попов В. О.
Пульникова А. А.
Рассадников Г. С.
Преподаватель:
Ким С.Е.

Санкт-Петербург
2023

1. Реализовать методы **Gauss-Newton** и **Powell Dog Leg** для решения нелинейной регрессии. Сравнить эффективность с методами, реализованными в предыдущих работах.

Gauss-Newton

Алгоритм используется для решения задач нелинейным методом наименьших квадратов. Алгоритм не требует вычисления вторых производных, что является преимуществом алгоритма.

Алгоритм итеративно находит значения переменных, которые минимизируют сумму квадратов, если нам даны m функций от n переменных $\sum_{i=1}^{m} \alpha_i^2$, где α_i функции от n переменных, которые имеют вид:

$$\alpha_i(\beta) = y_i - f(x_i, \beta)$$

Алгоритм начинает с начального приближения β_0, далее алгоритм будет действовать так:

$$\beta_{s+1} = \beta_s - (J_f^T \times J_f)^{-1} \times J_f^T \times \alpha(\beta_s)$$

Где $J = \frac{\partial f_i}{\partial x_j}$, в формуле рассмотрим функции и переменные, как вектора. $(J_f^T \times J_f)^{-1} \times J_f^T$ является псевдообратной матрицей к J_f.

Мы требуем, чтобы m ≥ n, поскольку иначе матрица $J_f^T \times J_f$ не имеет обратной и поставленную задачу нельзя решить.

Powell Dog Leg

Алгоритм представляет собой итеративный алгоритм оптимизации для решения нелинейных задач наименьших квадратов. Использует алгоритм Gauss-Newton с градиентным спуском, но использует доверительную область, то есть на каждом шаге если алгоритм Gauss-Newton находится в доверительной области, он используется для обновления, иначе алгоритм ищет минимум вдоль крутого спуска, то есть точки Коши. Если точка Коши находится за пределами доверительного интервала, то она усекается до границы, а если она находится внутри области, то обновление происходит так: пересечение области и линии, соединяющей точку Коши и шаг Gauss-Newton.

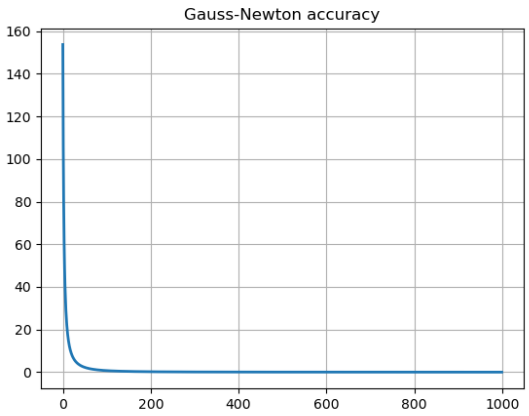
Точка Коши – точка, лежащая на градиенте, который минимизирует квадратичную модель, при условии, что шаг находится в доверительной области.

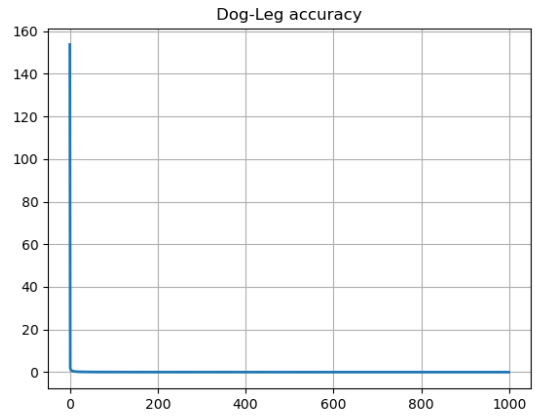
Минимизируем функцию $F(x) = \|f(x)\|^2 = \sum_{i=1}^{m}(f_i(x))^2$, где $f_i : R^n \to R$. Алгоритм ищет оптимальную точку с помощью последовательности $x_k = x_{k-1} + \delta_k$, последовательность сходится к нужной точке.

Учитывая доверительную область δ_k выбирается так:

- $\delta_{gn} = -(J^T \times J)^{-1} \times J^T \times f(x)$, где $J = \frac{\partial f_i}{\partial x_j}$, при условии, что шаг Gauss-Newton находится в пределах доверительной области, т. е. $\|\delta_{gn}\| \leq \Delta$
- $\frac{\Delta}{\|\delta_{sd}\|} \times \delta_{sd}$, где $\delta_{sd} = -J^T \times f(x)$, при условии, что шаг Gauss-Newton и шаг наискорейшего спуска находятся вне доверительного спуска, т. е. $t \times \|\delta_{sd}\| \geq \Delta$ и $\|\delta_{gn}\| \geq \Delta$
- $t \times \delta_{sd} + s(\delta_{gn} + t \times \delta_{sd})$ с s таким, чтобы $\|\delta\| = \Delta$, где $t = \frac{\|\delta_{sd}\|^2}{\|J\delta_{sd}\|^2}$, при условии, что $\|\delta_{gn}\| \geq \Delta$ и $t \times \|\delta_{sd}\| \leq \Delta$

Приведем графики сходимости алгоритмов.



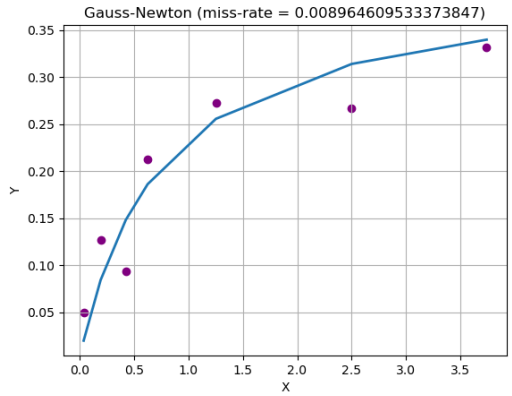


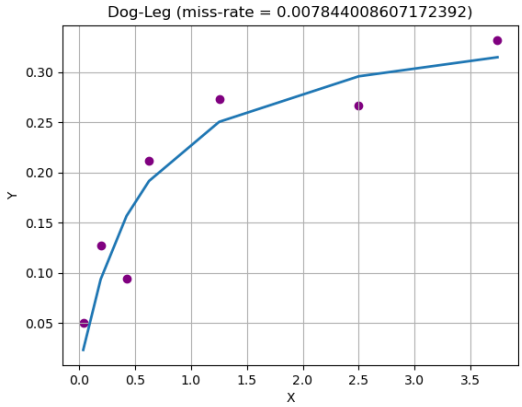
Сравним получившиеся функции с нахождением регрессии через градиентный спуск. На функции $kx + b$ получим следующие результаты (сравнимаем количество итераций при примерно одинаковых значений ошибок)

Алгоритм	Итерации	Miss-rate
GD	1385	0,0473
GN	1721	0,0488
DL	666	0,0489

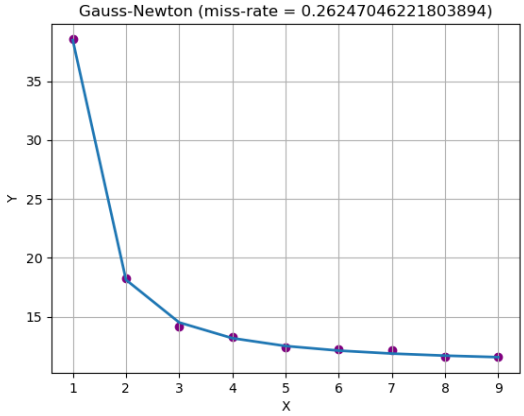
Однако, важным моментом является факт, что реализованные алгоритмы (в отличие от GD) работают не только для нахождения линейной регрессии.
Приведем примеры функций, на которых исследуем наши алгоритмы.

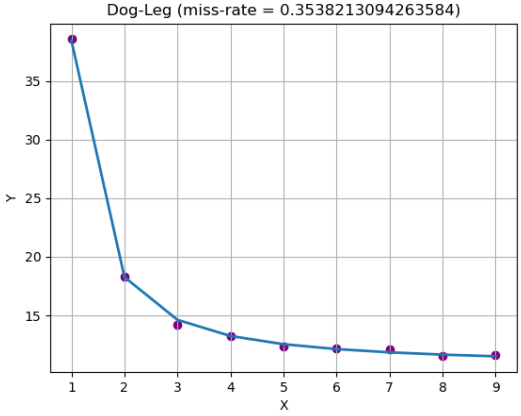
1) $\frac{\beta_0 x}{\beta_1 + x}$



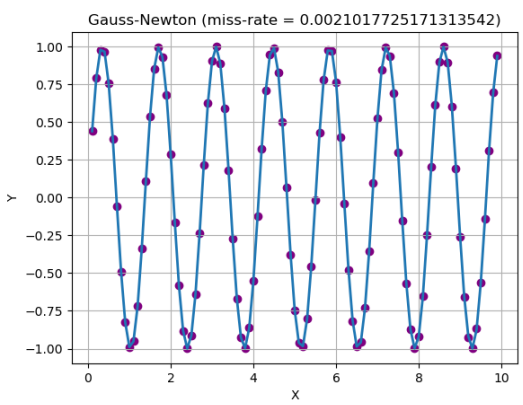


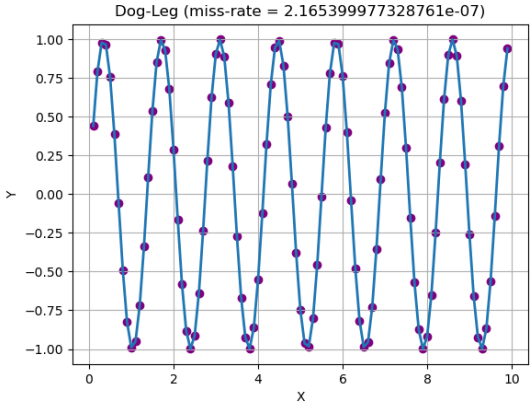
2) $\frac{(\beta_1 x^2 + \beta_0) \times (\beta_0 x + \beta_2)}{\beta_2 x^3}$



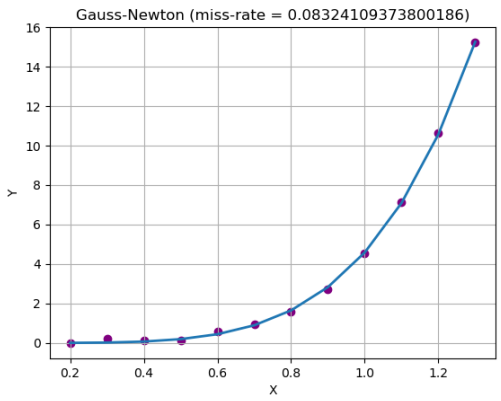


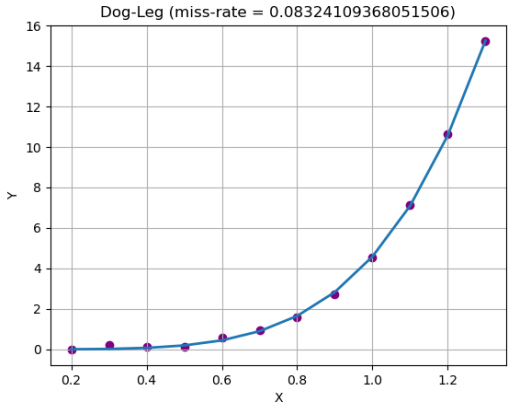
3) $sin(\beta_0 x)$



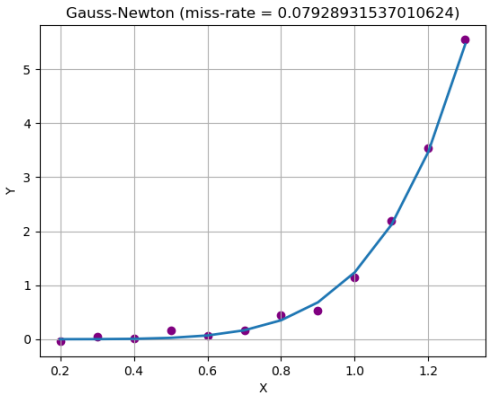


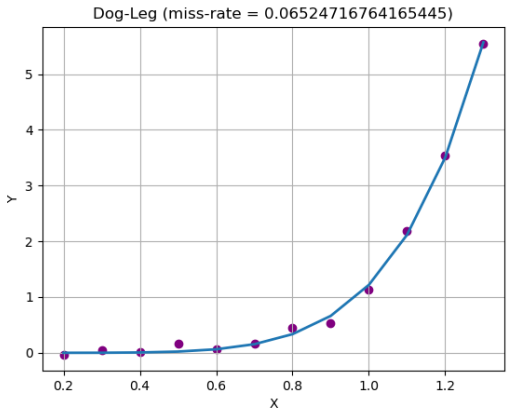
4) $\beta_0 x^{\beta_0}$



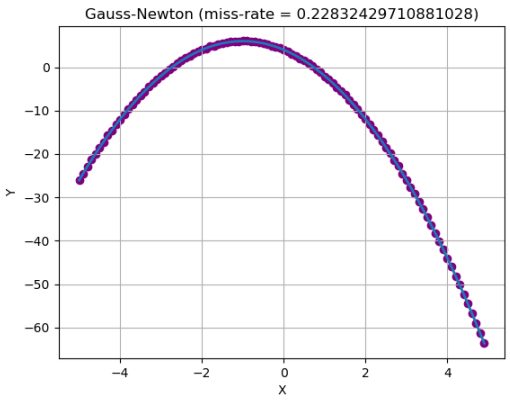


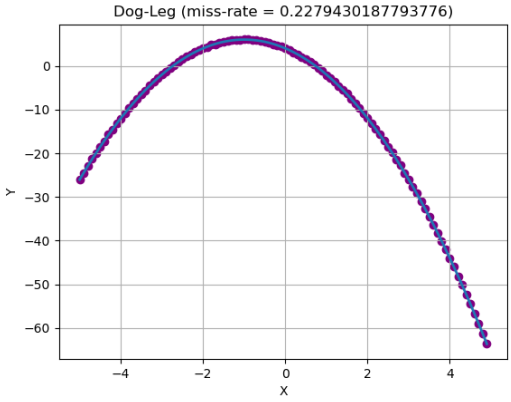
5) $\beta_0 x^{\beta_1}$



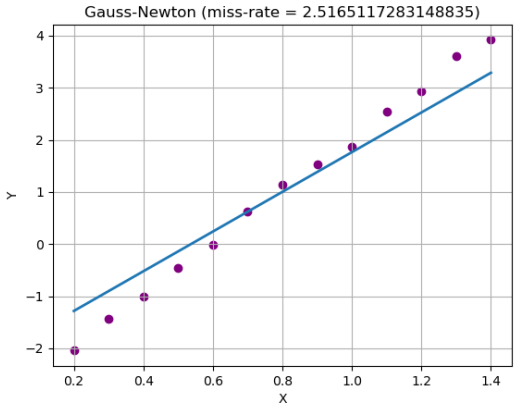


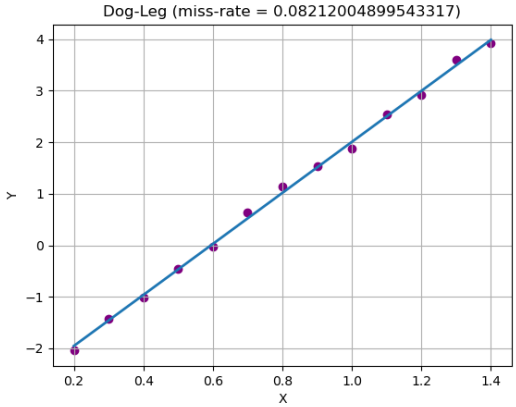
6) $\beta_0 x^2 + \beta_1 x + \beta_2$





7) $\beta_0 x + \beta_1$





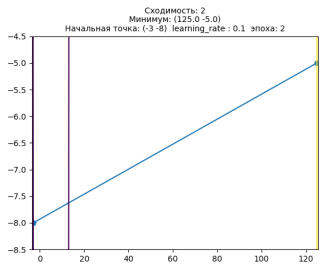
Заметим, что в последней функции Gauss-Newton не сходится.

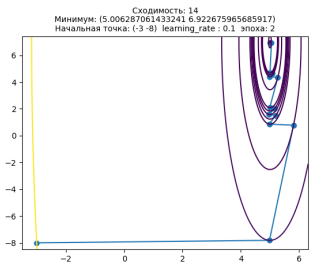
2. Реализовать **метод BFGS** и исследовать его сходимость при минимизации различных функций. Сравнить с другими реализованными алгоритмами.
Итерационный метод численной оптимизации, где гессиан функции вычисляется приближенно.
Мы хотим оптимизировать задачу нахождения минимума функции.

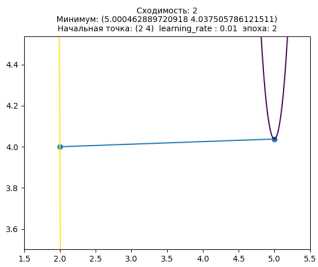
- Выбираем начальную точку x_0
- Выбираем точность ε> 0
- Определим начальное приближение $H_0 = B_0^{-1}$, где B_0^{-1} обратный гессиан функции, а гессиан функции — это матрица $a_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$
- Находим точку, в направлении которой будем работать $p_k = -H_k \times \nabla f_k$
- $x_{k+1} = x_k + \alpha_k \times p_k$, где α_k коэффициент линейного поиска, который находиться с использованием условия Вольфе
- Шаг алгоритма $s_k = x_{k+1} - x_k$, изменение градиента $y_k = \nabla f_{k+1} - \nabla f_k$
- Обновляем гессиан $H_{k+1} = (I - \rho_k \times s_k \times y_k^T) \times H_k \times (I - \rho_k \times y_k \times s_k^T) + \rho_k \times s_k \times s_k^T$ где $\rho_k = \frac{1}{y_k^T \times s_k}$ ($y_k^T \times s_k$ внешнее произведение векторов, по сути матричное произведение), I единичная матрица
- Продолжаем алгоритм пока $|\nabla f_k| > \varepsilon$

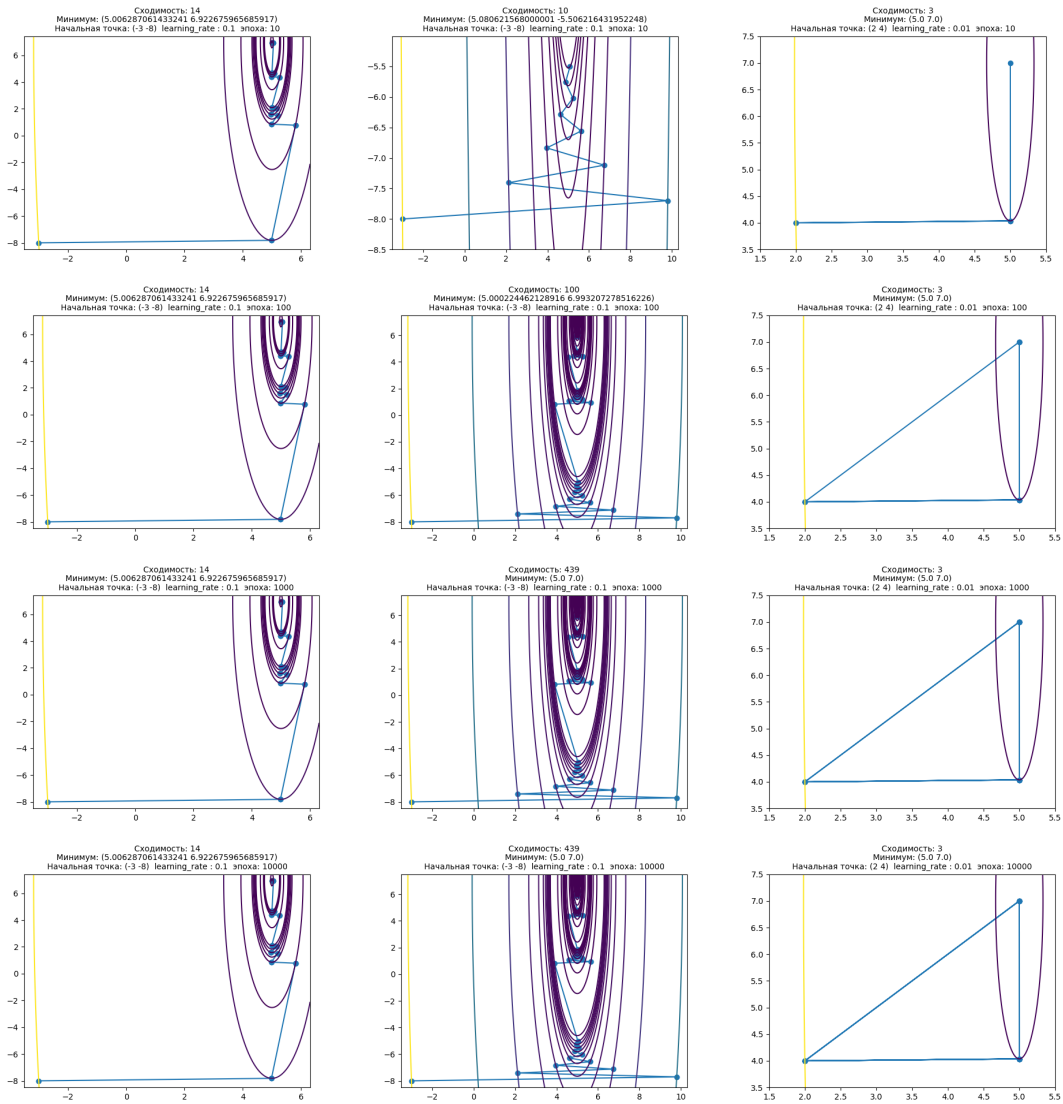
Приведем исследование сходимости BFGS и сравнение с другими алгоритмами из предыдущих лабораторных работ при минимизации различных функций.

1) $80(x-5)^2 + (y-7)^2$
Исследуем влияние числа эпох на сходимость алгоритмов, таких как GD, дихотомия, BFGS.



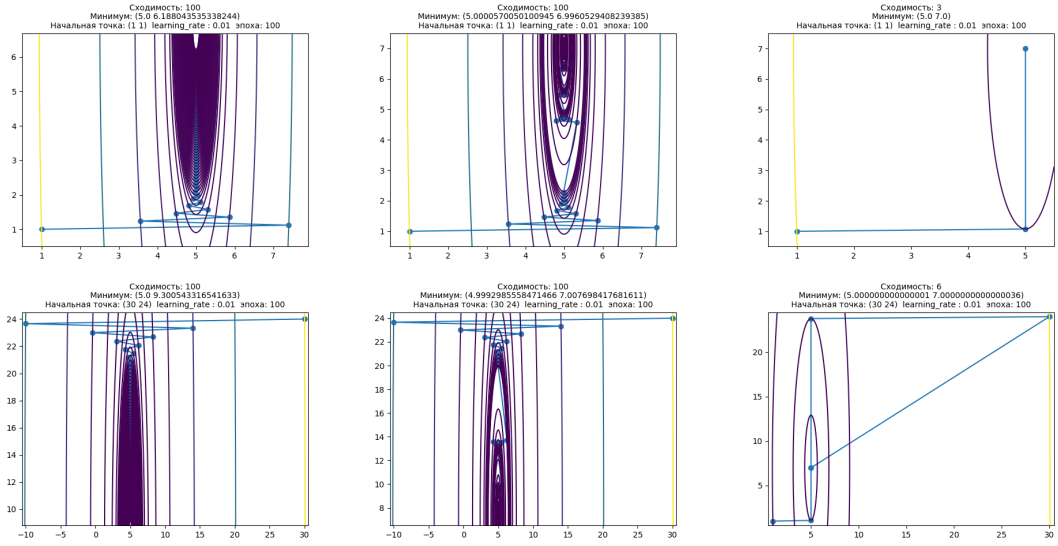


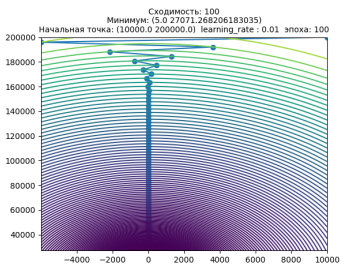


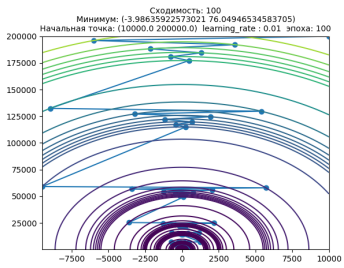


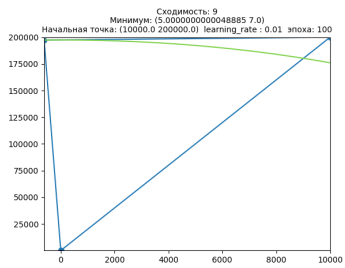
В отличии от алгоритмов, реализованных ранее BFGS, не требует большого числа итераций и находит точку минимума всего за три шага, что подтверждает эффективность его работы на простых данных в двумерных случаях.

Исследуем влияние выбора начальной точки на сходимость алгоритмов, таких как GD, Wolfe, BFGS.





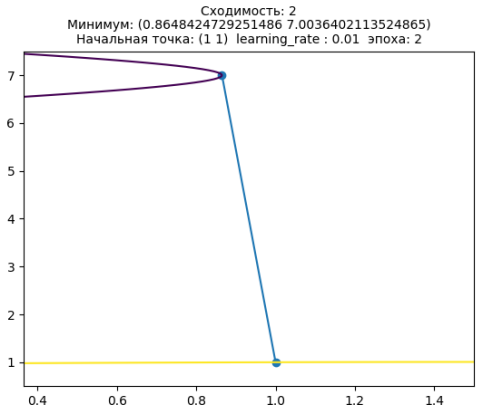


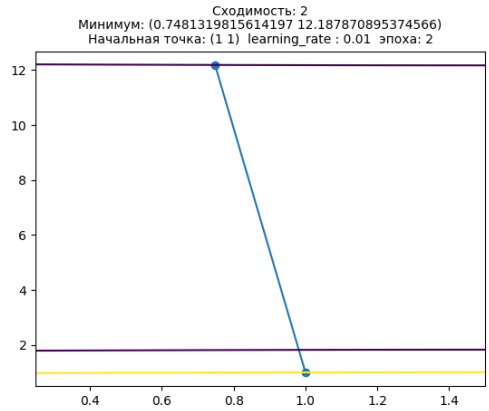


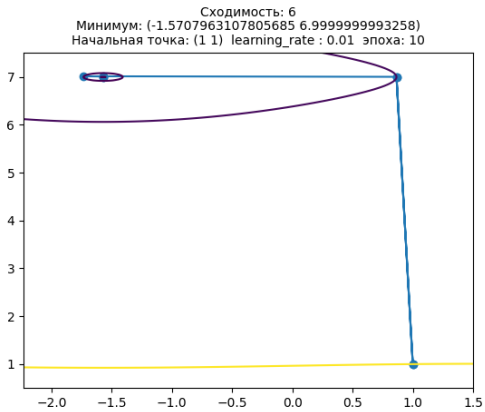
Выбор начальной точки почти не мешает BFGS, при отдалении начальной точки от точки минимума, очевидно, увеличивается число итераций, но благодаря большой эффективности алгоритма, удается найти абсолютно точную точку минимума за очень маленькое число итераций.

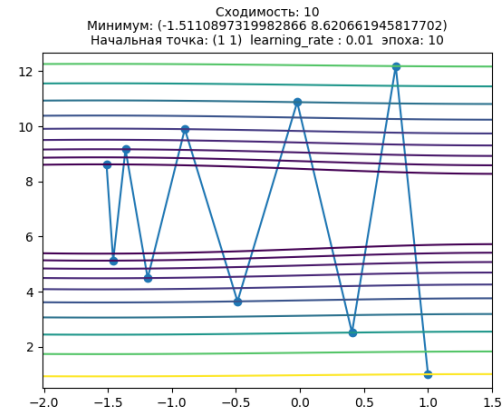
2) $\frac{1}{2}sin(x) + (y-7)^2 + 1$

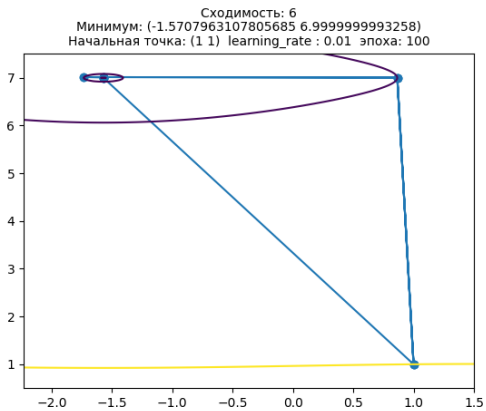
Исследуем влияние числа эпох на сходимость алгоритмов, таких как Wolfe, BFGS.

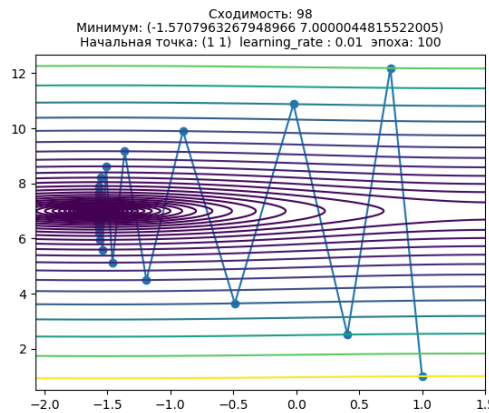






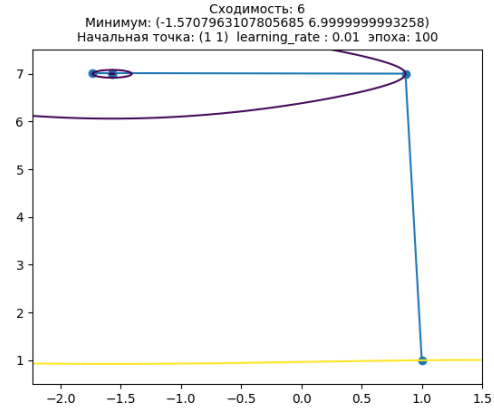


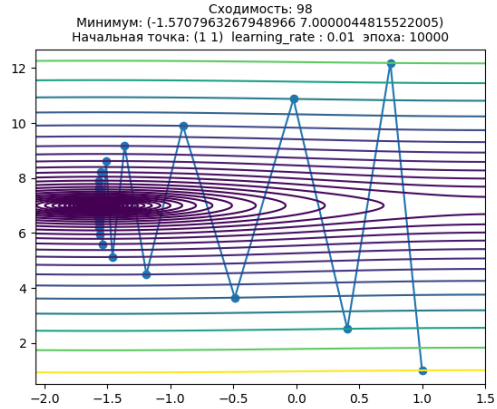


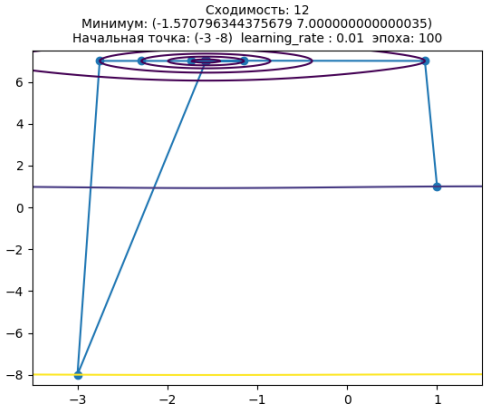


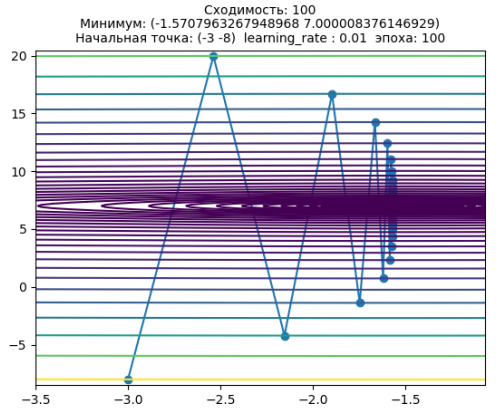
В отличии от алгоритма Wolfe BFGS находит точку минимума за маленькое число итераций, хотя конечный результат сравним по точности с результатом Wolfe.

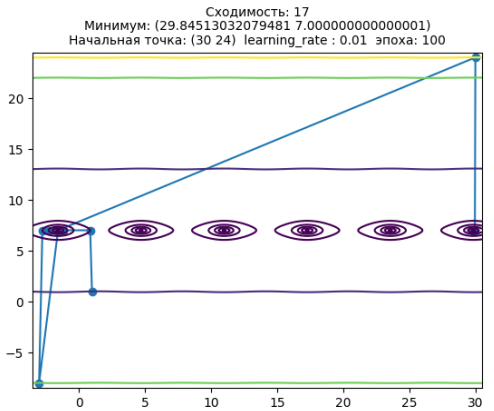
Исследуем влияние выбора начальной точки на сходимость алгоритмов, таких как Wolfe, BFGS.

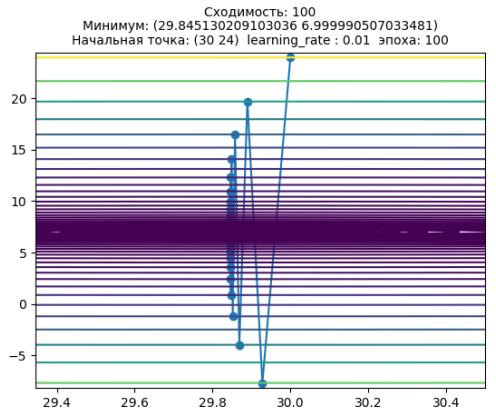












Выбор начальной точки влияет на найденную точку минимума, так как функция имеет бесконечно много точек минимума. При разных точках старта алгоритмы находят ближайшую точку минимума, которая имеет вид $(-\frac{\pi k}{2}, 7)$. В отличии от Wolfe BFGS совершает меньшее число итераций, находя точку с наилучшей точностью.

3) Исследуем работу алгоритма на многомерной функции (n = 100) и его отличие в работе с GD.
Функция: $x_1^2 + 2x_2^2 + 3x_3^2 + \ldots + 100x_{100}^2$
Выберем начальную точку: (1, 1 … 1)

Шаг градиента: 0,01

	BFGS		GD		
№	Число итераций	Расхождение ответа с реальной точкой минимума	Число итераций	Расхождение ответа с реальной точкой минимума	Расхождение ответа алгоритмов
1	2	5,05	2	5,77	3,91
2	12	0,98	10	2,30	2,35
3	97	1e-7	100	1,01	1,01
4	182	1e-7	1000	1	1
5	267	1e-8	10000	1	1

При n = 100 оба алгоритма справляются с задачей поиска минимума довольно эффективно, но алгоритм BFGS занимает в n раз больше времени на каждую итерацию. Сходимость BFGS получается довольно быстрой и точной по сравнению с GD.

4) Посмотрим на работу алгоритмов при n = 10000.
Функция: $x_1^2 + 2x_2^2 + 3x_3^2 + \ldots + 10000x_{10000}^2$.
Выберем начальную точку: (1, 1 … 1)

	BFGS		GD	
№	Число итераций	Расхождение ответа с реальной точкой минимума	Число итераций	Расхождение ответа с реальной точкой минимума
1	-	-	2	3e5
2	-	-	10	1e4
3	-	-	100	100

Алгоритм BFGS требует хранение матрицы размера n*n, для вычисления гессиана, поэтому при достаточно больших n алгоритм нельзя использовать для исследования сходимости при минимизации различных функций.

3. Дополнительное задание

Реализовать и исследовать алгоритм **L-BFGS**

BFGS с ограниченной памятью, алгоритм поддерживает историю последних m обновлений x и ∇f, где обычно m < 10. Это позволяет не хранить матрицу гессиана на каждой итерации цикла, что дает преимущество в используемой памяти при большом значении n в многомерных случаях.

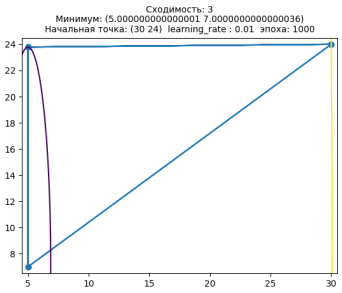
Алгоритм

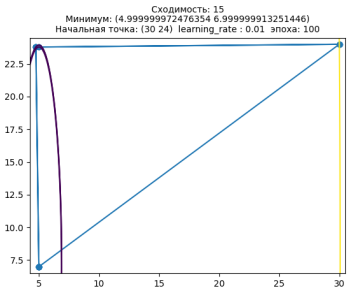
$q = \nabla f_k$
$for\ i = k-1, k-2, \ldots, k-m$
$\alpha_i = \rho_i s_i^T q$
$q = q - \alpha_i y_i$
$\gamma_k = \frac{s_{k-1} y_{k-1}}{y_{k-1}^T y_{k-1}}$
$H_k^0 = \gamma_k I$
$z = H_k^0 q$
$for\ i = k-m, k-m+1, \ldots, k-1$
$\beta_i = \rho_i y_i^T z$
$z = z + s_i(\alpha_i - \beta_i)$
$p = z$

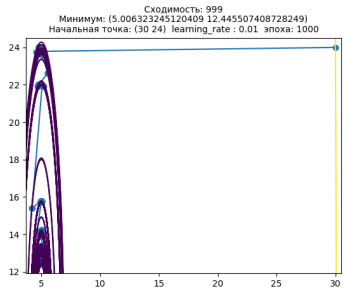
Проведем исследование сходимости L-BFGS метода и сравним его работу с BFGS.

1) $80(x-5)^2+(y-7)^2$

Исследуем сходимость BFGS, LBFGS с параметром m = 5, LBFGS с параметром m = 1. Параметр m позволяет экономить используемую память, но нарушает корректность аналитического решения и требует большего числа итераций для сходимости. При выборе m слишком маленьким (m=1) корректность алгоритма нарушается т. к. на каждой итерации цикла нет истории данных, что дает плохую сходимость, поэтому алгоритм продолжает поиск точки минимума.

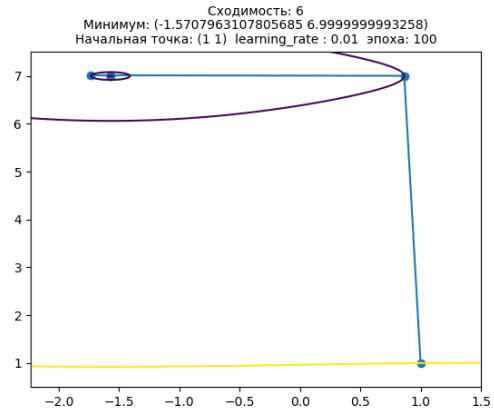


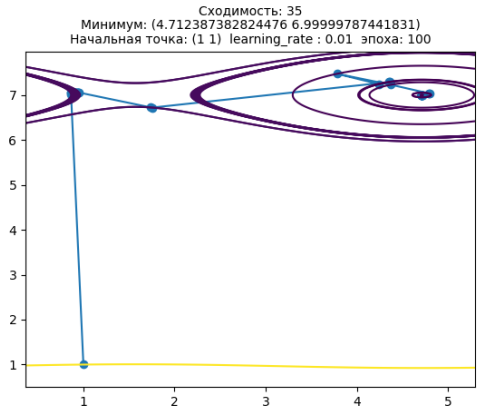


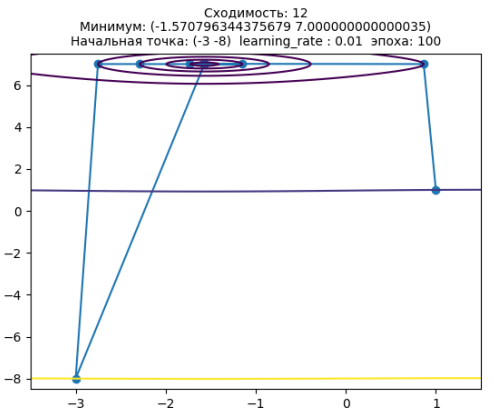


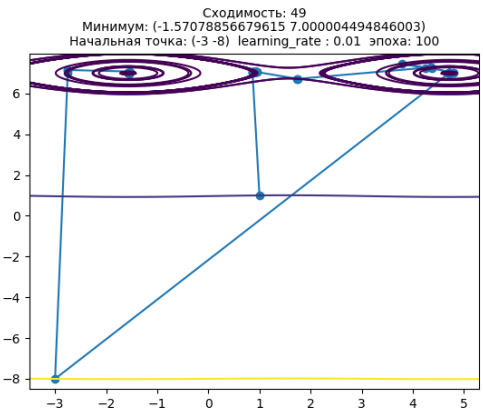
2) $\frac{1}{2}sin(x)+(y-7)^2+1$

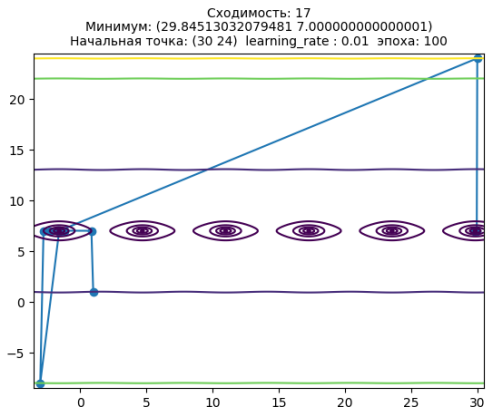
Исследуем выбор точки старта на примере сложной двумерной функции и сходимость BFGS и LBFGS (m = 5) алгоритмов.

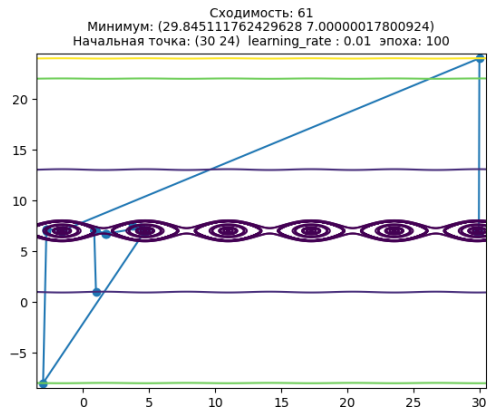












Т. к. LBFGS опирается на историю, то для сходимости требуется большое число итераций, однако алгоритм довольно быстро и точно сходится.

3) Посмотрим на работу алгоритмов при n = 10000. m = 7.
Функция: $x_1^2 + 2x_2^2 + 3x_3^2 + \ldots + 10000x_{10000}^2$.
Выберем начальную точку: (1, 1, … 1)

	LBFGS		GD	
№	Число итераций	Расхождение ответа с реальной точкой минимума	Число итераций	Расхождение ответа с реальной точкой минимума
1	2	365151	2	1e7
2	10	1146	10	1e4
3	100	103	100	1487

LBFGS позволяет исследовать сходимость в многомерных случаях при больших n. Притом результат получается точнее, чем при обычном градиентном спуске. Однако не удается найти идеальную точку минимума даже на простой функции, т. к. для хорошей сходимости требуется либо большое число итераций, либо хранение большой истории данных за последние m итераций. Увеличение данных параметров приводит к увеличению вычислительной нагрузке работы программы.