Практическое задание 1: Методы градиентного спуска и Ньютона.

Калугин Дмитрий

18 марта 2018 г.

1 Формулы для логистической регрессии

В задаче двухклассовой логистической регрессии задача минимизации ставится следующим образом:

$$f(x) = \frac{1}{m}\sum_{i=1}^{m}\ln\left(1+\exp(-b_i\langle a_i, x\rangle)\right) + \frac{\lambda}{2}\|x\|_2^2,$$

$$\min_{x\in\mathbb{R}^n} f(x),$$

где $b_i \in \mathbb{R}, a_i \in \mathbb{R}^n$. Введем обозначение $A = (a_1, \ldots, a_m)$. Найдем градиент функции $f(x)$:

$$Df(x)[v] = \frac{1}{m}\sum_{i=1}^{m}\frac{\exp(-b_i\langle a_i, x\rangle)(-b_i\langle a_i, v\rangle)}{1+\exp(-b_i\langle a_i, x\rangle)} + \lambda\langle x, v\rangle =$$

$$= \langle -\frac{1}{m}\sum_{i=1}^{m}\frac{\exp(-b_i\langle a_i, x\rangle)}{1+\exp(-b_i\langle a_i, x\rangle)}b_i a_i + \lambda x, v\rangle = \langle -\frac{1}{m}\sum_{i=1}^{m}\left(1-\frac{1}{1+\exp(-b_i\langle a_i, x\rangle)}\right)b_i a_i + \lambda x, v\rangle,$$

$$\nabla f(x) = -\frac{1}{m}\sum_{i=1}^{m}\frac{\exp(-b_i\langle a_i, x\rangle)}{1+\exp(-b_i\langle a_i, x\rangle)}b_i a_i + \lambda x = -\frac{1}{m}A^T b\frac{\exp(-bAx)}{1+\exp(-bAx)} + \lambda x$$

Аналогично найдем гессиан $f(x)$:

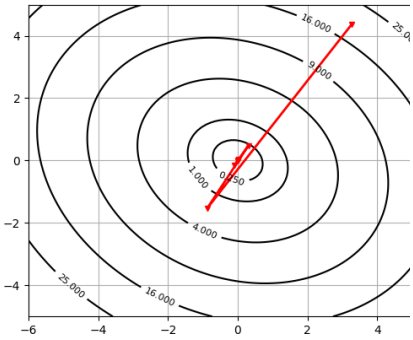
$$D^2 f(x)[v, v] = \langle \frac{1}{m}\sum_{i=1}^{m}\frac{\exp(-b_i\langle a_i, x\rangle)}{(1+\exp(-b_i\langle a_i, x\rangle))^2}(b_i a_i)^2 v + \lambda v, v\rangle,$$

$$\nabla^2 f(x) = \frac{1}{m}A^T\frac{1}{1+\exp(-b_i\langle a_i, x\rangle)}\left(1-\frac{1}{1+\exp(-b_i\langle a_i, x\rangle)}\right)A$$

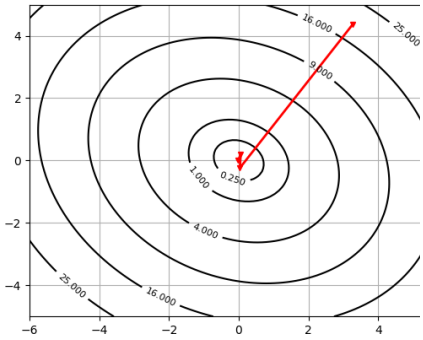
2 Эксперимент: Траектория градиентного спуска на квадратичной функции

В данном эксперименте сипользовались две разные матрицы A, одна с числом обусловленности 1.51, вторая — с числом обусловленности 11.356. Для Каждой матрицы запуски производились из трех различных точек. Результаты эксперимента приведены ниже.

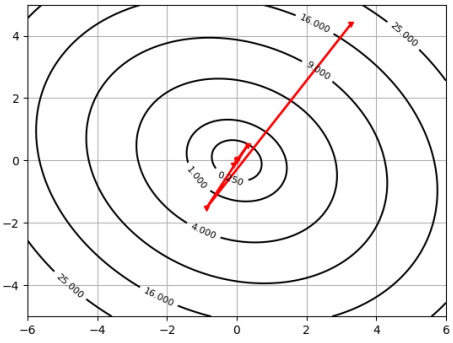
2.1 Матрица с маленьким числом обусловленности



(a) Constant step, 5 steps

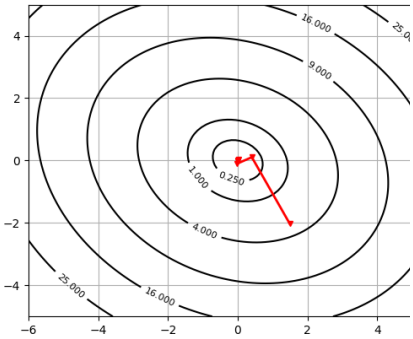


(b) Armijo rule, 8 steps

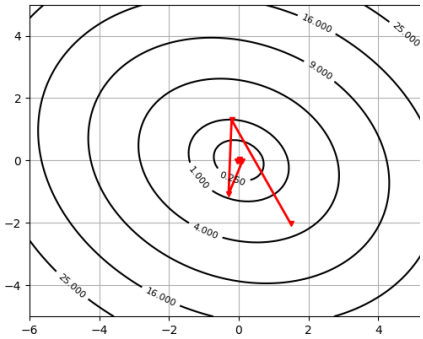


(c) Wolfe rule, 5 steps

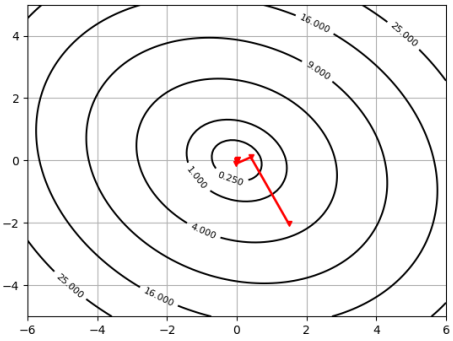
Рис. 1: Первая точка запуска



(a) Constant step, 5 steps

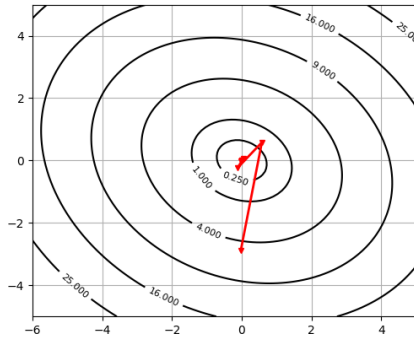


(b) Armijo rule, 9 steps

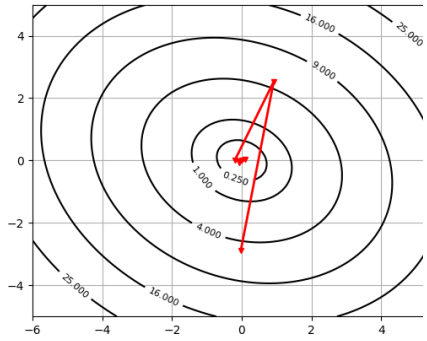


(c) Wolfe rule, 5 steps

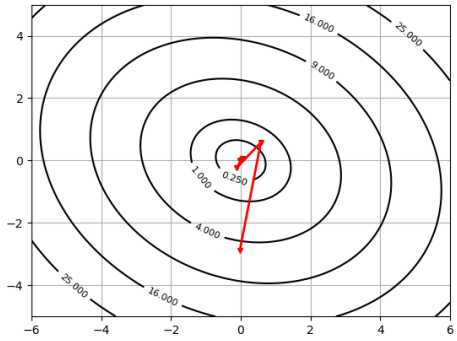
Рис. 2: Вторая точка запуска



(a) Constant step, 6 steps



(b) Armijo rule, 7 steps

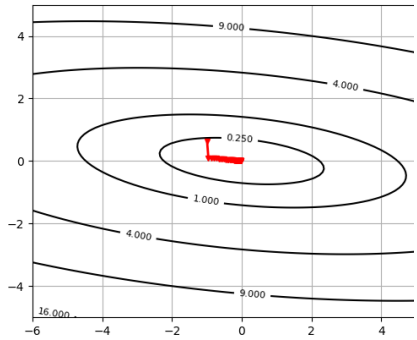


(c) Wolfe rule, 6 steps

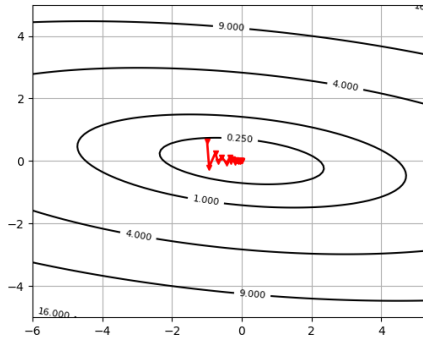
Рис. 3: Третья точка запуска

В этих экспериментах меньше всего шагов понадобилось алгоритмам с константным шагом и с шагом, выбраным по методу Вульфа, они ведут себя практически одинаково на всех точках. Методу Армихо требуется немного больше шагов, но в целом, на матрице с маленьким числом обусловленности все методы сходятся достаточно быстро(константы в методах Армихо и Вульфа были взяты значениями по умолчанию).

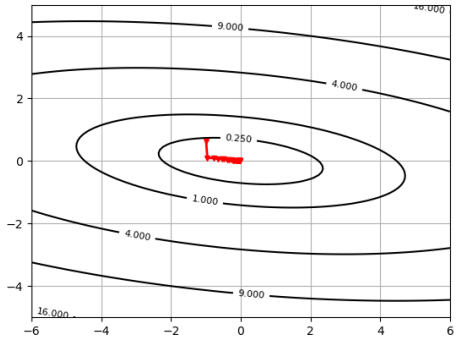
2.2 Матрица с большим числом обусловленности



(a) Constant step, 62 steps

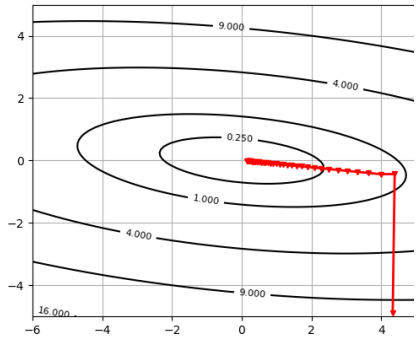


(b) Armijo rule, 32 steps

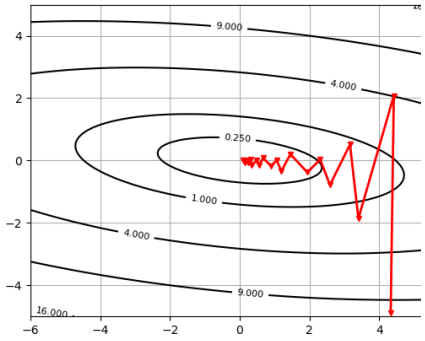


(c) Wolfe rule, 31 steps

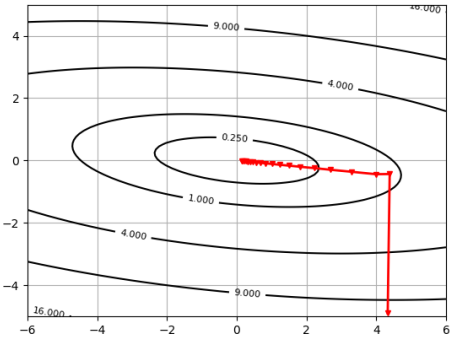
Рис. 4: Первая точка запуска



(a) Constant step, 61 steps

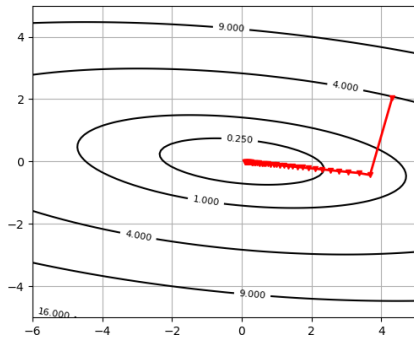


(b) Armijo rule, 33 steps

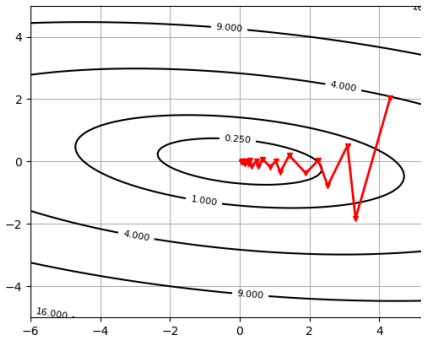


(c) Wolfe rule, 31 steps

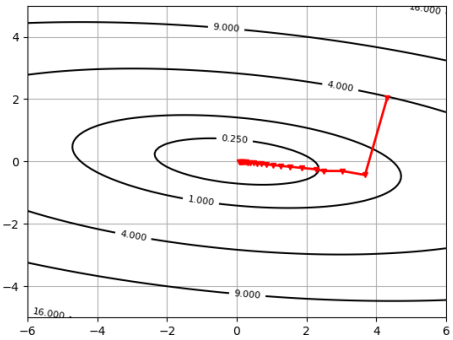
Рис. 5: Вторая точка запуска



(a) Constant step, 54 steps



(b) Armijo rule, 29 steps



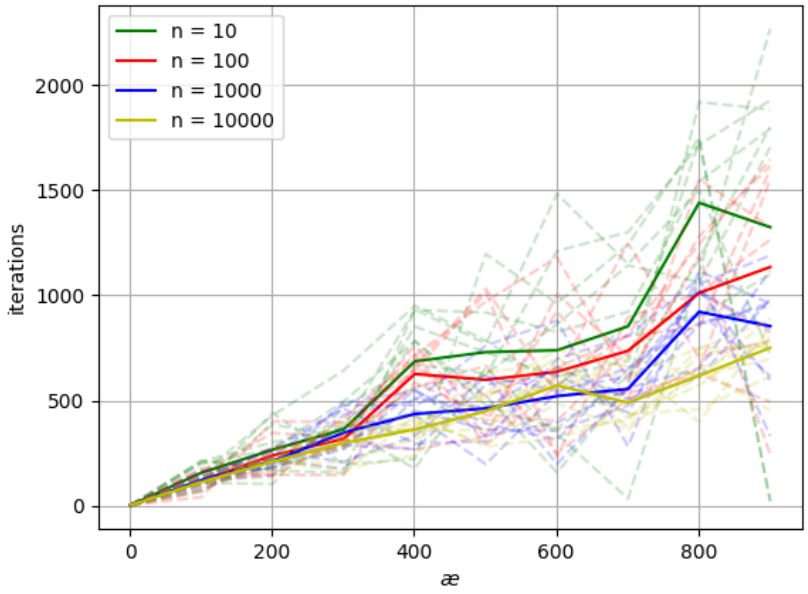
(c) Wolfe rule, 28 steps

Рис. 6: Третья точка запуска

В этом случае уже хорошо виден выигрыш методов Армихо и Вульфа перед константным методом. При этом метод Армихо, как видно, совершает сильные скачки, и "перепрыгивает"минимум по направлению, в та время как у метода Вульфа такого не наблюдается.

Тем не менее, общая проблема градиентных методов ярко видна — при сильной обусловленности матрицы метод начинает совершать много маленьких шагов вдоль большой оси.

3 Эксперимент: Зависимость числа итераций градиентного спуска от числа обусловленности и размерности пространства



kappa).png

Рис. 7: Зависимость числа итераций градиентного спуска от числа обусловленности при разных значениях размерности пространства

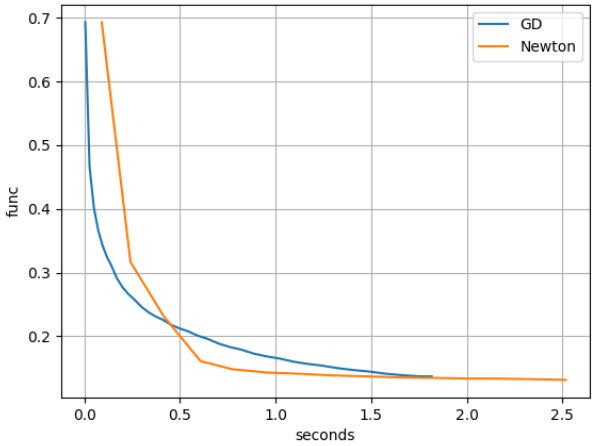
Из графика хорошо видно, что количество итераций практиески не меняется в зависимости от размерности задачи, в то время как от числа обусловленности оно растет линейно.

Из этого можно сделать вывод, что для хорошей работы градиентого спуска очень выгодно использовать предобуславливание матрицы.

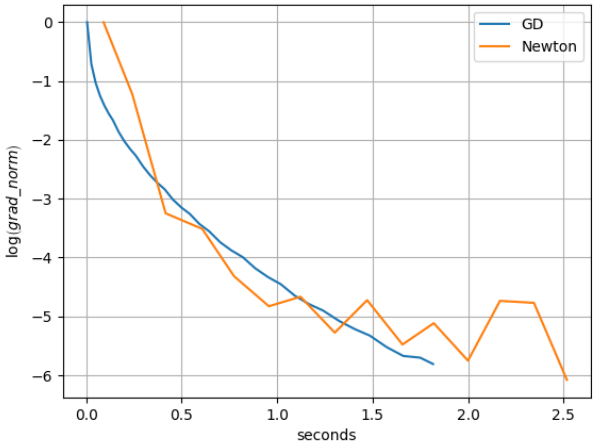
4 Эксперимент: Сравнение методов градиентного спуска и Ньютона на реальной задаче логистической регрессии

Опишем учетные стоимости операций этих методов. Градиентный спуск требует $O(n)$ памяти и $O(qn)$ времени на одну итерацию, где q — сложность вычисления f или одной из компонент градиента или гессиана. Метод Ньютона требует $O(n^2)$ памяти и $O(qn^2 + n^3)$ времени на одну итерацию(оценка времени складывается из времен $O(qn^2)$, которое необходимо на подсчет гессиана, и $O(n^3)$, которое необходимо на решение системы $\nabla^2 f(x_k)d_k = -\nabla f(x_k)$). При этом градиентный метод обладает линейеной скоростью сходимости, а метод Ньютона — квадратичной.

Эксперименты были проведены на трех различных датасетах: *w8a*, *gisette* и *real-sim*. При этом на последнем датасете не использовался метод Ньютона, так как на ноутбуке с 6 ГБ ОЗУ подсчет соответствующего гессиана и решение соответствующей линейной системы вычислительно слишком сложено.

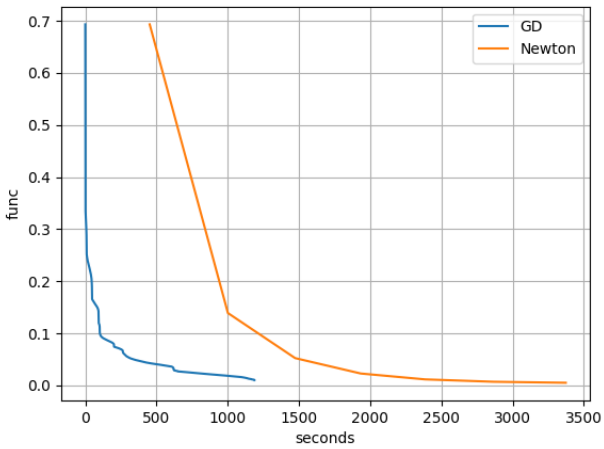


(a) Значение функции от времени

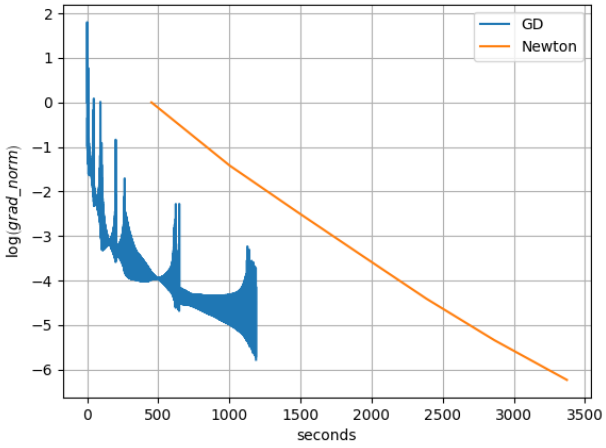


(b) Значение относительной нормы градиента от времени в логарифмической шкале

Рис. 8: *w8a* dataset

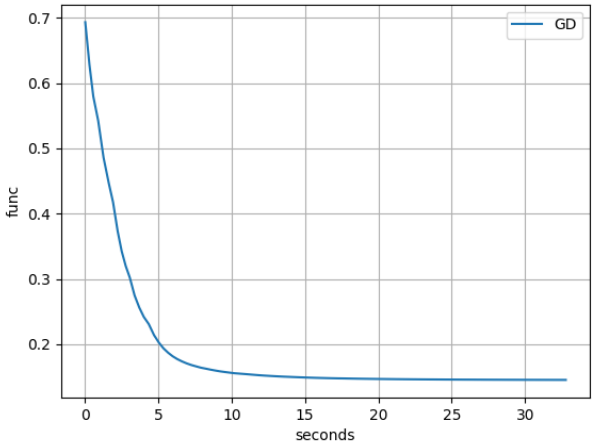


(a) Значение функции от времени

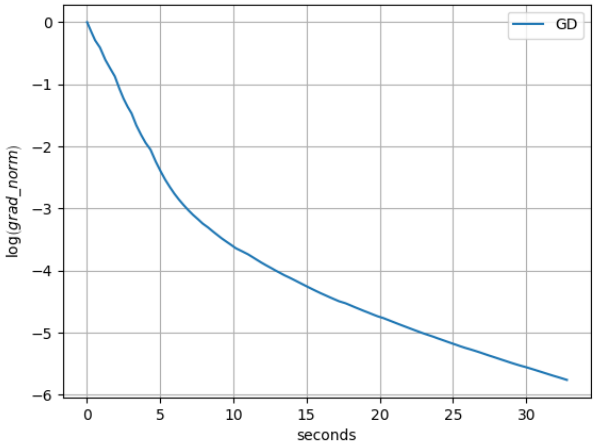


(b) Значение относительной нормы градиента от времени в логарифмической шкале

Рис. 9: *gisette* dataset



(a) Значение функции от времени



(b) Значение относительной нормы градиента от времени в логарифмической шкале

Рис. 10: *real-sim* dataset

Поведение функций на графиках понятно — значение функции всегда монотонно убывает, в то время как значение нормы градиента может убывать немонотонно, ведь наши методы гарантируют минимизацию функции, а про поведение нормы граадиента ничего сказать не могут.

Из графиков можно сделать следующие выводы. Не смотря на квадратичную сходимость метода Ньютона, использовать на практике его целесообразно только при малеьнкой размерности пространства признаков(датасет *w8a*), в этом случае это действительно дает выигрыш по времени. Также, методу Ньютона требуется совсем немного итераций. Так, на втором датасете ему понадобилось меньше 10 итераций, против 2000 итераций градиентного спуска. Тем не менее, за счет того, что каждая отдельная операция в градентном спуске делается гораздо быстрее, он оказывается эффективнее, и, более того, может применяться и на задачах больших размерностей.