Практическое задание 2

Рыбка Елизавета, 474

23 октября 2017 г.

1 Эксперимент: Зависимость числа итераций метода сопряженных градиентов от числа обусловленности и размерности пространства

Выясним какая есть зависимость между числом итераций метода и числом обусловленности k, размерностью пространства n. Проведем нижеуказанный эксперимент для метода сопряженных градиентов и градиентного спуска. Для каждого n, k указанных в таблице случайным образом 100 раз генерируем функцию и начальную точку, а затем усредняем количество итераций, которое понадобилось методу:

- Создаем список длины n, на первом месте в котором стоит k, на втором 1, далее идут случайные числа из интервала $[1, k)$. $A = diag(a)$. При таком построении число обусловленности будет равно k
- b генерируем как случайный вектор размерности n, с компонентами из интервала $[-2k, 2k]$ (случайно и равномерно).
- x_0 генерируем как случайный вектор размерности n, с компонентами из интервала $[-2k, 2k]$ (случайно и равномерно).

Результаты эксперимента представлены в таблицах (4 Приложения) и визуализированы на следующих графиках.

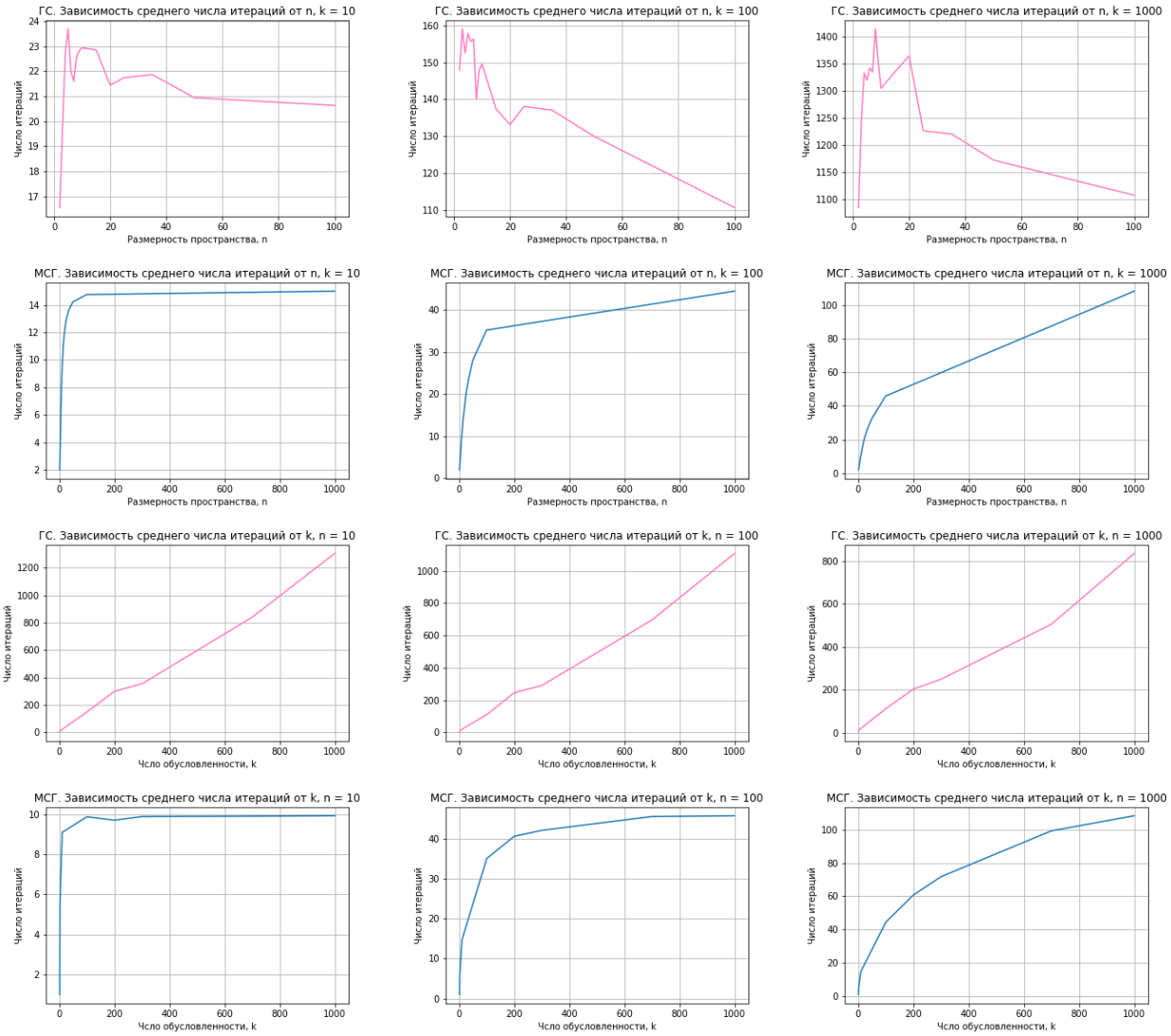

Рис. 1: Зависимость среднего числа итераций метода сопряженных градиентов от числа обусловленности и размерности пространства

Какие выводы можно сделать?

Мы видим что поведение МСГ и ГС разительно отличаются. При фиксированном k увеличение размерности ведет для них к противоположным результатам.

И если размерность не сильно влияет на повидение ГС, она принципиальна для МСГ. Это согласуется с результатами предыдущего задания и известными фактами об МСГ. Известно, что МСГ должен сходится за конечное число шагов $\leq n$, а нужная точность, как видно из графиков, может быть достигнута значительно раньше. И то когда настанет это раньше, исходя из 2го ряда графиков, зависит от числа обусловленности. Так же ,по идее, накопления погрешностей может нарушаться ортогональность базисных векторов и ухудшать сходимость.

Из нижнего ряда графиков напрашивается интересный вывод о МСГ: зависимость от числа обусловленности k возрастает при увеличении размерности пространства n.

2 Эксперимент: Выбор размера истории в методе L-BFGS

Оценка требуемой памяти и сложности итерации метода

Итак, давайте, перед тем как перейти к практической части,сделаем теоретический анализ: оценим размер требуемой памяти и сложность итерации метода L-BFGS в зависимости от размера истории l и размерности пространства n.

Алгоритм 1 Рекурсивное умножение L-BFGS матрицы на вектор

function BFGS_MULTIPLY(v, $\mathcal{H}$, γ_0)
  if $\mathcal{H} = \emptyset$ **then**
    return $\gamma_0 v$
  end if
  $(s, y) \leftarrow$ последняя пара из $\mathcal{H}$.
  $\mathcal{H}' \leftarrow \mathcal{H}$ без последней пары.
  $v' \leftarrow v - \frac{\langle s,v \rangle}{\langle y,s \rangle} y$
  $z \leftarrow$ BFGS_MULTIPLY$(v', \mathcal{H}', \gamma_0)$
  return $z + \frac{\langle s,v \rangle - \langle y,z \rangle}{\langle y,s \rangle} s$.
end function

Имея в распоряжении указанную процедуру, направление d_k вычислить легко:

Алгоритм 2 Вычисление направления поиска d_k в методе L-BFGS

function LBFGS_DIRECTION
  $(s, y) \leftarrow$ последняя пара из $\mathcal{H}_k$
  $\gamma_0 \leftarrow \frac{\langle y,s \rangle}{\langle y,y \rangle}$
  return BFGS_MULTIPLY$(-\nabla f(x_k)$, $\mathcal{H}_k$, $\gamma_0)$
end function

Посмотрим на Алгоритм 2. Сначала вычисляется γ_0, для этого вычисляется два скалярных произведения; это $2n$ операций т.е. $O(n)$. Затем вызывается рекурсивный Алгоритм 1. Видим что каждый шаг в нем делает несколько скалярных произведений и сложений векторов, т.е. $O(n)$ действий. Всего шагов l, где l — длина истории. Таким образом, по времени весь LBFGS потрбует $O(nl)$.

Теперь поговорим о памяти. Нам необходимо хранить историю — $2l$ n-ых векторов. Итого: $O(nl)$. Более точно: $2ln + 4n$

Эксперимент
На графиках представлены результаты эксперимента. Отношение норм отличается от отношения квадратов норм в логарифмической шкале только коэффициентом 2 и следовательно не влияет на качественные зависимости. Для наглядности графики приведены только для первых 100 итераций.

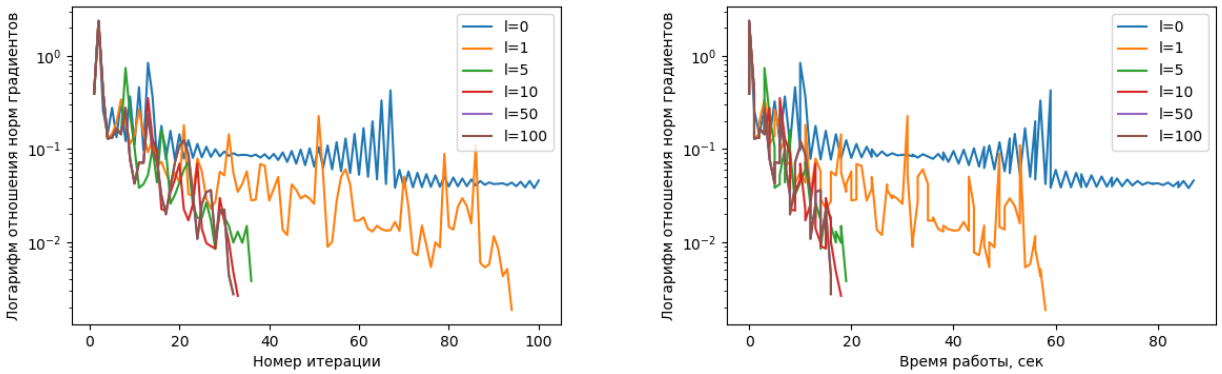

Рис. 2: Относительное уменьшение нормы градиента на данных gisette

Видим что переход от числа итераций к реальному времени не влияет на взаимное расположение графиков с ненулевой историей и ухудшает показатели для метода с $l = 0$, что лагично т.к. в этом случае это принципиально другой метод.
Посмотрим на график при $l = 0$ более подробно. В этом случае LBFGS совпадает с градиентым спуском. Метод не сошелся за 500 операций(на рисунках не представлено). На графиках видно принципиальное различие в поведении этого метода и настоящего LBFGS (насколько я понимаю основная причина — затаривание ГС в локальных минимумах или седловых точках).
Теперь обратим внимание на методы с ненулевой историей. Увеличение длины истории приводит к улучшению сходимости метода как по времени так и по числу операций. Но это происходит неравномерно. С некоторого момента происходит насыщение и дальнейшее увеличение длины истории оказывается нецелесообразным (вся суть как раз в том чтобы экономить память). Сходимость практически перестает улучшаться при $l = 10$ (поэтому размер истории в реализации LBFGS по дефолту 10)

3 Эксперимент: Сравнение методов на реальной задаче логистической регрессии

В таблице приведены размеры датасетов (в Mb). Т. е. на Рис. 3 датасеты отсортированы по размеру: более крупные — выше.

Название	Размер
real-sim	33.6
news20	25.5
rcv1	13
w8a	3.3

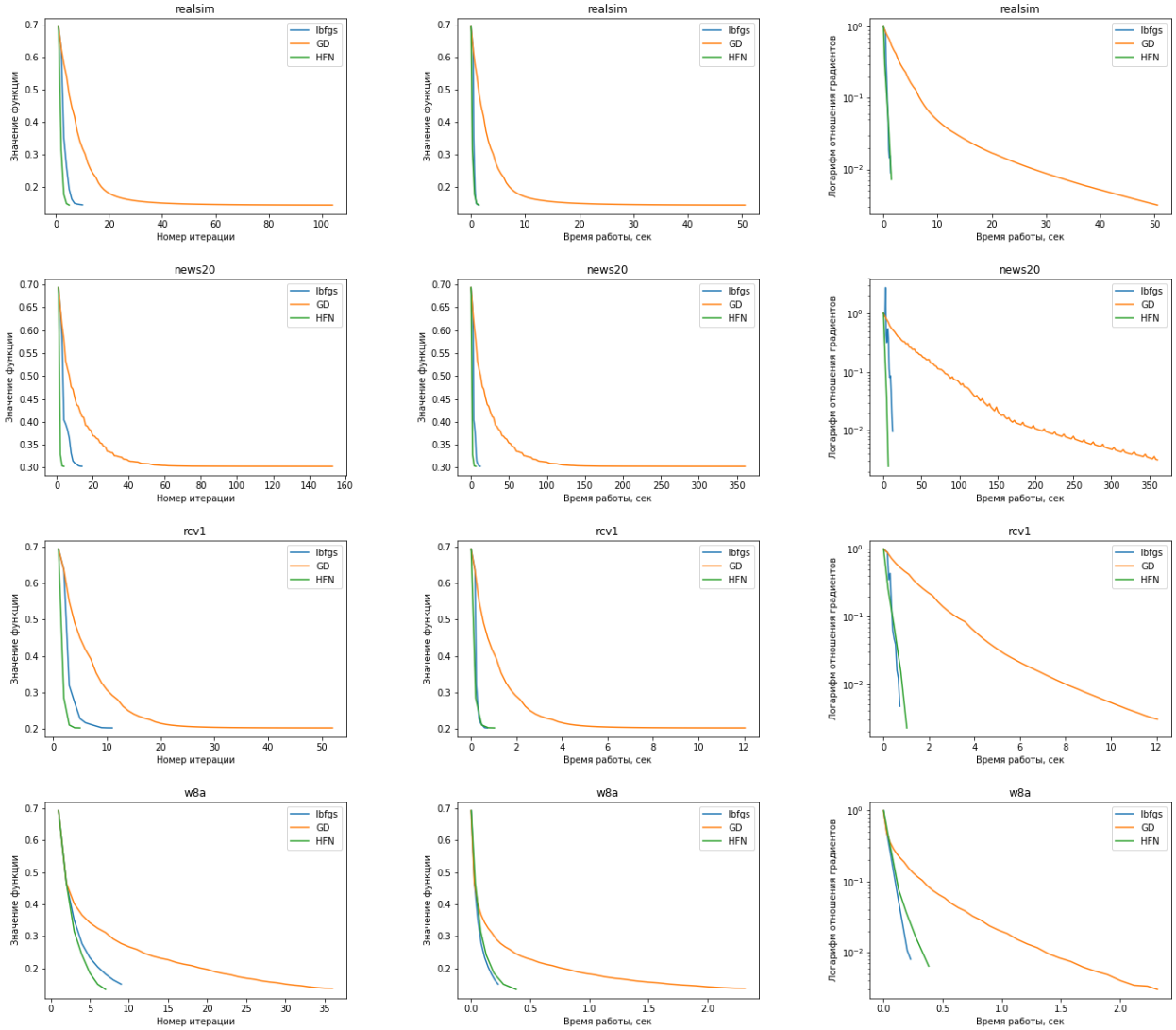

Рис. 3: Поведение усеченного Ньютона(HFN), L-BFGS, градиентного спуска(GD) на различных датасетах

Правые графики в логарифмическом масштабе $\Rightarrow$ по ним судим о скорости сходимости. Она как видим, линейная для HFN и L-BFGS.

Хуже всего всегда работает градиентный спуск. Гораздо. На больших датасетах на порядок.Что в целом логично, т.к. два другие метода являются квазиньютоновскими. Дальше обсуждать его неинтересно.

По уменьшению значения функции HFN выигрывает у L-BFGS по количеству итераций, но проигрывает по реальному времени. Это видно на маленьких объемах, на больших методы сходятся практически одинаково. Однако правильнее рассуждать о сходимости метода смотря на относительное изменение градиента (самый правый столбец графиков). Здесь мы видим, что HFN и L-BFGS ведут себя очень похоже. На графике real-sim они вообще не различимы (L-BFGS сходятся немного быстрее). На news20 HFN работает более равномерно и сходится быстрее. На меньших объемах данных выигрывает L-BFGS.

В итоге выбрать однозначно между HFN и L-BFGS не удалось. Однако градиентый спуск трогать не стоит:)

4 Приложения:

Таблица 1: МСГ. Среднее число итераций

$n\backslash k$	1	2	10	100	200	300	700	1000
2	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
3	1.0	2.99	3.0	3.0	3.0	3.0	3.0	3.0
4	1.0	3.88	4.0	4.0	4.0	4.0	4.0	4.0
5	1.0	4.64	4.96	4.99	4.99	5.0	4.99	5.0
6	1.0	5.06	5.95	5.99	5.99	6.0	5.99	6.0
7	1.0	5.33	6.83	7.0	6.99	6.99	7.0	6.99
8	1.0	5.35	7.64	7.93	7.95	7.98	7.98	7.99
9	1.0	5.54	8.33	8.84	8.86	8.94	8.95	8.99
10	1.0	5.55	9.09	9.87	9.7	9.88	9.9	9.92
15	1.0	5.84	11.1	13.94	13.99	14.13	14.04	14.37
20	1.0	5.97	12.13	16.73	17.66	17.44	17.93	18.39
25	1.0	5.99	12.85	19.72	20.69	20.71	21.19	21.74
35	1.0	5.99	13.62	23.57	25.37	25.39	26.99	26.67
50	1.0	6.0	14.21	27.95	30.81	30.7	32.72	32.43
100	1.0	6.0	14.74	35.16	40.74	42.23	45.7	45.89
1000	1.0	6.0	14.99	44.39	60.74	71.71	99.22	108.26

Таблица 2: ГС. Среднее число итераций

$n\backslash k$	1	2	10	100	200	300	700	1000
2	2.0	3.21	16.55	147.87	278.81	246.96	898.72	1084.46
3	2.0	6.28	19.68	159.09	275.73	363.77	912.83	1242.4
4	2.0	7.27	22.75	152.52	291.46	358.93	992.85	1332.51
5	2.0	7.98	23.69	157.82	310.54	349.09	846.36	1319.05
6	2.0	8.25	21.99	155.69	279.93	378.39	894.95	1342.06
7	2.0	8.43	21.6	156.33	280.45	372.64	871.31	1334.57
8	2.0	8.55	22.56	140.01	314.71	351.91	859.88	1414.08
9	2.0	8.96	22.82	147.8	269.99	352.4	838.61	1354.52
10	2.0	8.99	22.93	149.57	300.28	354.72	839.74	1304.54
15	2.0	9.58	22.85	137.43	271.67	328.31	824.48	1334.96
20	2.0	9.19	21.44	133.12	293.21	316.58	773.77	1364.31
25	2.0	9.33	21.74	138.07	269.9	350.81	796.99	1226.03
35	2.0	9.98	21.86	137.05	264.26	303.86	764.67	1220.35
50	2.0	9.69	20.94	129.93	253.5	314.23	712.17	1172.12
100	2.0	9.92	20.63	110.64	245.87	290.62	695.66	1107.17
1000	2.0	10.22	19.92	111.64	202.93	248.96	505.24	835.26