

Эффективные численные методы решения задачи PageRank для дважды разреженных матриц

*Аникин Антон, Гасников Александр, Горнов Александр,
Камзолов Дмитрий, Максимов Юрий, Нестеров Юрий*

1. Введение

В данной работе мы сконцентрируемся на решении задачи PageRank и ее окрестностях. Хорошо известно (Брин–Пейдж, 1998 [1], [2]), что задача ранжирования web-страниц приводит к поиску вектора Фробениуса–Перрона стохастической матрицы $x^T P = x^T$. Размеры матрицы могут быть колоссальными (в современных реалиях это сто миллиардов на сто миллиардов). Выгрузить такую матрицу в оперативную память обычного компьютера не представляется возможным. Задачу PageRank можно переписать, как задачу выпуклой оптимизации (Назин–Поляк, 2011 [3]; Ю.Е. Нестеров, 2012 [4], [5]; Гасников–Дмитриев, 2015 [6]) в разных вариантах: минимизация квадратичной формы $\|Ax - b\|_2^2$ и минимизация бесконечной нормы $\|Ax - b\|_\infty$ на единичном симплексе. Здесь $A = P^T - I$, I – единичная матрица. К аналогичным задачам приводят задачи анализа данных (Ridge regression, LASSO; Hastie–Friedman–Tibshirani, 2009 [7]), задачи восстановления матрицы корреспонденций по замерам трафика на линках в компьютерных сетях [8] и многие другие задачи. Особенностью постановок этих задач, также как и в случае задачи PageRank, являются колоссальные размеры.

В данной статье планируется сосредоточиться на изучении роли разреженности матрицы A , на использовании рандомизированных подходов и на специфике множества, на котором происходит оптимизация. Симплекс является (в некотором смысле) наилучшим возможным множеством, которое может порождать (независимо от разреженности матрицы A) разреженность решения (см., например, п. 3.3 С. Бубек, 2014 [9], это тесно связано с l_1 -оптимизацией [10]).

Все о чем здесь написано хорошо известно на таком уровне грубости. Поясним, дополнительно погружаясь в детали, в чем заключаются отличия развиваемых в статье подходов от известных подходов. Прежде всего, мы вводим специальный класс *дважды разреженных матриц* (одновременно разреженных и по строкам и по столбцам).¹ Такие матрицы, например, возникают в методе конечных элементов, и в более общем контексте при изучении разностных схем.² Если считать, что матрица A имеет размеры n на n , а число элементов в каждой строке и столбце не больше чем $s \ll n$, то число ненулевых элементов в матрице может быть sn . Кажется, что это произведение точно должно

¹ Заметим, что за счет специального “раздутия” изначальной матрицы можно обобщить приведенные в статье оценки и подходы с класса дважды разреженных матриц, на матрицы, у которых есть небольшое число полных строк и/или столбцов. Такие задачи встречаются в приложениях заметно чаще.

² Применительно к ранжированию web-страниц, это свойство означает, что входящие и выходящие степени всех web-страниц равномерно ограничены.

возникать в оценках общей сложности (числа арифметических операций, типа умножения или сложения двух чисел типа double) решения задачи (с определенной фиксированной точностью). Оказывается, что для первой постановки (минимизация квадратичной формы на симплексе) сложность может быть сделана пропорциональна s^2 (разделы 2 и 3), а для второй (минимизация бесконечной нормы) и того меньше $s \ln n$ (раздел 4).

Первая задача может решаться обычным прямым градиентным методом с 1-нормой в прямом пространстве [11], [12] – нетривиальным тут является, в том числе, организация работы с памятью. В частности, требуется хранение градиента в массиве, обновление элементов которого и вычисления максимального/минимального элемента должно осуществляться за время, не зависящие от размера массива (то есть от n). Тут оказываются полезными фибоначчиевы и бродялевские кучи [13]. Основная идея – не рассчитывать на каждом шаге градиент функции заново $A^T A x_{k+1}$, а пересчитывать его, учитывая, что $x_{k+1} = x_k + e_k$, где вектор e_k состоит в основном из нулей: $A^T A x_{k+1} = A^T A x_k + A^T A e_k$. Детали будут изложены в разделе 2. Аналогичную оценку общей сложности планируется получить методом условного градиента Франк–Вульфа [14], большой интерес к которому появился в последние несколько лет в основном в связи с задачами Big Data (М. Ягги [15], Аршуи–Юдицкий–Немировский [16], Ю.Е. Нестеров [17]). Аккуратный анализ работы этого метода также при правильной организации работы с памятью позволяет (аналогично предыдущему подходу) находить такой x , что $\|Ax - b\|_2 \leq \varepsilon$ за число арифметических операций $O(n + s^2/\varepsilon^2)$, причем оценка оказывается вполне практической. Этому будет посвящен раздел 3.

Вторая задача (минимизации $\|Ax - b\|_\infty$ на единичном симплексе) с помощью техники Юдицкого–Немировского, 2012 (см. п. 6.5.2.3 [18]) может быть переписана как седловая задача на произведение двух симплексов. Если применять рандомизированный метод зеркального спуска (основная идея которого заключается в вычислении вместо градиента Ax стохастического градиента, рассчитываемого по следующему правилу: с вероятностью x_1 он равен первому столбцу матрицы A , с вероятностью x_2 второму и т.д.), то для того, чтобы обеспечить с вероятностью $\geq 1 - \sigma$ неравенство $\|Ax - b\|_\infty \leq \varepsilon$ достаточно выполнить $O(n \ln(n/\sigma)/\varepsilon^2)$ арифметических операций. К сожалению, этот подход не позволяет в полном объеме учесть разреженность матрицы A . В статье предлагается другой путь, восходящий к работе Григориадиса–Хачияна, 1995 [19] (см. также [20]). В основе этого подхода лежит рандомизация не при вычислении градиента, а при последующем проектировании (согласно KL -расстоянию, что отвечает экспоненциальному взвешиванию) градиента на симплекс. Предлагается вместо честной проекции на симплекс случайно выбирать одну из вершин симплекса (разреженный объект!) так, чтобы математическое ожидание проекции равнялось бы настоящей проекции [20]. В работе планируется показать, что это можно эффективно делать, используя специальный двоичные деревья пересчета компонент градиента [6]. В результате планируется получить следующую оценку для дважды разреженных матриц

$O(n + s \ln n \ln(n/\sigma)/\varepsilon^2)$. Напомним, что при этом число ненулевых элементов в матрице A может быть sn . Возникает много вопросов относительно того насколько все это практично. К сожалению, на данный момент наши теоретические оценки говорят о том, что константа в $O(\cdot)$ может иметь порядок 10^2 . Подробнее об этом будет написано в разделе 4.

В данной статье при специальных предположениях относительно матрицы PageRank P (дважды разреженная) мы предлагаем методы, которые работают по наилучшим известным сейчас оценкам для задачи PageRank в условиях отсутствия контроля спектральной щели матрицы PageRank [6]. А именно, полученная в статье оценка (см. разделы 2, 3) $O(n + s^2/\varepsilon^2)$ сложности поиска такого x , что $\|Ax - b\|_2 \leq \varepsilon$, является на данный момент наилучшей при $s \ll \sqrt{n}$ для данного класса задач (быстрее может работать только метод МСМС $O(\ln n \ln(n/\sigma)/(\alpha\varepsilon^2))$, требующий, чтобы спектральная щель α была достаточно большой [6]).

Приведенные в статье подходы, применимы не только к квадратным матрицам A и не только к задаче PageRank. В частности, можно обобщать приведенные в статье подходы на композитные постановки [21]. Тем не менее, свойство двойной разреженности матрица A является существенным. Если это свойство не выполняется, и имеет место, скажем, только разреженность в среднем по столбцам, то приведенные в статье подходы могут быть доминируемы рандомизированными покомпонентными спусками. Если матрица A сильно вытянута по числу строк (такого рода постановки характерны для задач анализа данных), то покомпонентные методы применяются к двойственной задаче [22], [23], если по числу столбцов (такого рода постановки характерны для задач поиска равновесных конфигураций в больших транспортных/компьютерных сетях), то покомпонентные методы применяются к прямой задаче [24], [25]. Подчеркнем, что при условии двойной разреженности A с $s \ll \sqrt{n}$ нам не известны более эффективные (приведенных в данной статье) способы решения описанных задач. В частности, это относится и к упомянутым выше покомпонентным спускам.

В заключительном пятом разделе работы приводятся результаты численных экспериментов.

2. Прямой градиентный метод в 1-норме

Задача поиска вектора PageRank может быть сведена к следующей задаче выпуклой оптимизации [12]

$$f(x) = \frac{1}{2} \|Ax\|_2^2 + \frac{1}{2} \sum_{k=1}^n (-x_k)_+^2 \rightarrow \min_{\langle x, e \rangle = 1}$$

где $A = P^T - I$, I – единичная матрица, $e = (1, \dots, 1)^T$,

$$(y)_+ = \begin{cases} y, & y \geq 0 \\ 0, & y < 0 \end{cases}.$$

При этом мы считаем, что в каждом столбце и каждой строке матрицы P не более $s \ll \sqrt{n}$ элементов отлично от нуля (P – разрежена). Эту задачу предлагается решать обычным градиентным методом, но не в евклидовой норме, а в 1-норме (см., например, [11]):

$$x_{k+1} = x_k + \arg \min_{h: \langle h, e \rangle = 0} \left\{ f(x_k) + \langle \nabla f(x_k), h \rangle + \frac{L}{2} \|h\|_1^2 \right\},$$

где $L = \max_{i=1, \dots, n} \|A^{(i)}\|_2^2 + 1 \leq 3$. Для достижения точности ε^2 по функции потребуется сделать $O(LR^2/\varepsilon^2) = O(1/\varepsilon^2)$ итераций [11]. Не сложно проверить, что пересчет градиента на каждой итерации заключается в умножении $A^T A h$, что может быть сделано за $O(s^2)$. Связано это с тем, что вектор h всегда имеет только две компоненты отличные от нуля (такая разреженность получилась благодаря выбору 1-нормы), причем эти компоненты соответствуют $\arg \min_{i=1, \dots, n} \partial f(x_k)/\partial x^i$ и $\arg \max_{i=1, \dots, n} \partial f(x_k)/\partial x^i$, что пересчитывается (при использовании кучи для поддержания максимальной и минимальной компоненты градиента) за $O(s^2 \ln n)$ (логарифмический фактор можно убрать, если использовать, например, фибоначчиевы кучи [13]). Таким образом, с учетом препроцессинга и стоимости первой итерации $O(n)$ общая трудоемкость предложенного метода будет $O(n + s^2/\varepsilon^2)$, что заметно лучше многих известных методов [6].

Не большая “ложка дегтя” в том, что в решении могут быть маленькие отрицательные компоненты. Также численные эксперименты (см. раздел 5) на реальных web-матрицах и случайно сгенерированных разреженных матрицах показали, что на практике приведенная оценка с запасом не достигается (метод работает хуже, чем можно ожидать из приведенной оценки). Связано это, по-видимому, с тем, что при работе с такими матрицами мы не учитываем в полном объеме разреженность. Другое дело, когда нам заранее известно, скажем, что матрица A имеет отличные от нуля элементы только в $s/2$ окрестности диагонали. Другими словами, для того, чтобы сполна использовать разреженность требуется либо “структура разреженности”, либо препроцессинг (в нашем случае он заключается в представлении матрицы по строкам в виде списка смежности: в каждой строке отличный от нуля элемент хранит ссылку на следующий отличный от нуля элемент, аналогичное представление матрицы делается и по столбцам). Заметим, что препроцессинг помогает ускорять решения задач не только в связи с более полным учетом разреженности постановки, но и, например, в связи с более эффективной организацией рандомизации [6].

Обратим внимание на то, что число элементов в матрице P отличных от нуля, даже при наложенном условии разреженности (по строкам и столбцам), все равно может быть достаточно большим sn . Удивляет то, что в оценке общей трудоемкости этот размер фактически никак не присутствует. Это в перспективе (при правильной организации работы с памятью) позволяет решать задачи колоссальных размеров. Более того, даже в случае небольшого числа не разреженных ограничений вида $\langle a_i, x \rangle = b_i, i = 1, \dots, m = O(1)$, можно “раздуть” пространство (не более чем в два раза), в котором происходит оптимизация (во многих методах, которые учитывают разреженность, это число входит подобно рассмотренному примеру, так что такое раздутие не приведет к серьезным затратам), и переписать эту систему в виде $Ax = b$, где матрица будет иметь размеры $O(n) \times O(n)$, но число отличных от нуля элементов в каждой строке и столбце будет $O(1)$. Таким образом, допускается небольшое число “плотных” ограничений.

3. Метод условного градиента Франк–Вульфа

Перепишем задачу поиска вектора PageRank следующим образом

$$f(x) = \frac{1}{2} \|Ax\|_2^2 \rightarrow \min_{x \in S_n(1)},$$

где $S_n(1)$ – единичный симплекс в \mathbb{R}^n . Для решения этой задачи будем использовать метод условного градиента Франк–Вульфа [14] – [17]. Напомним, в чем состоит метод.

Выберем одну из вершин симплекса и возьмем точку старта x_0 в этой вершине.

Далее по индукции, шаг которой имеет следующий вид. Решаем задачу

$$\langle \nabla f(x_k), y \rangle \rightarrow \min_{y \in S_n(1)}.$$

Обозначим решение этой задачи через

$$y_k = (0, \dots, 0, 1, 0, \dots, 0),$$

где 1 стоит на позиции

$$i_k = \arg \min_{i=1, \dots, n} \partial f(x_k) / \partial x^i.$$

Положим

$$x_{k+1} = (1 - \gamma_k) x_k + \gamma_k y_k, \gamma_k = \frac{2}{k+1}.$$

Имеет место следующая оценка [14] – [17]

$$f(x_N) = f(x_N) - f_* \leq \frac{2L_p R_p^2}{N+1},$$

где

$$R_p^2 \leq \max_{x, y \in S_n(1)} \|y - x\|_p^2, \quad L_p \leq \max_{\|h\|_p \leq 1} \langle h, A^T A h \rangle = \max_{\|h\|_p \leq 1} \|A h\|_2^2, \quad 1 \leq p \leq \infty.$$

С учетом того, что оптимизация происходит на симплексе, мы выберем $p = 1$. Не сложно показать, что этот выбор оптимален. В результате получим, что $R_1^2 = 4$, $L_1 = \max_{i=1, \dots, n} \|A^{(i)}\|_2^2 + 1 \leq 3$. Таким образом, чтобы $f(x_N) \leq \varepsilon^2/2$ достаточно сделать $N = 48\varepsilon^{-2}$ итераций. В действительности, число 48 можно почти на порядок уменьшить немного более тонкими рассуждениями (детали мы вынуждены здесь опустить).

Сделав препроцессинг стоимостью $O(n)$, можно так организовать процедуру пересчета $\nabla f(x_k)$ и вычисления $\arg \min_{i=1, \dots, n} \partial f(x_k)/\partial x^i$, что каждая итерация будет иметь сложность $O(s^2)$. Таким образом, итоговая оценка сложности описанного метода будет $O(n + s^2/\varepsilon^2)$.

Стоит отметить, что функционал, выбранный в этом примере, обеспечивает намного лучшую оценку $\|Ax\|_2 \leq \varepsilon$ по сравнению с функционалом из раздела 4, который обеспечивает лишь $\|Ax\|_\infty \leq \varepsilon$. Наилучшая (в разреженном случае без, условий на спектральную щель матрицы P [6]) из известных нам на данный момент оценок $O(n + s \ln n \ln(n/\sigma)/\varepsilon^2)$ (см. раздел 4) для $\|Ax\|_\infty$ может быть улучшена приведенной в этом разделе оценкой, поскольку $\|Ax\|_2$ может быть (и так часто бывает) в $\sim \sqrt{n}$ раз больше $\|Ax\|_\infty$, а $s \ll \sqrt{n}$.

4. Седловое представление задачи PageRank и рандомизированный зеркальный спуск в форме Григориадиаса–Хачияна

Перепишем задачу поиска вектора PageRank следующим образом

$$f(x) = \|Ax\|_\infty \rightarrow \min_{x \in S_n(1)}.$$

Следуя [18] эту задачу можно переписать седловым образом

$$\min_{x \in S_n(1)} \max_{\|y\|_1 \leq 1} \langle Ax, y \rangle.$$

В свою очередь последнюю задачу можно переписать как

$$\min_{x \in S_n(1)} \max_{\omega \in S_{2n}(1)} \langle Ax, J\omega \rangle,$$

где $J = \text{diag}\{I_n, -I_n\}$.

В итоге задачу можно переписать, с сохранением свойства разреженности, как

$$\min_{x \in S_n(1)} \max_{\omega \in S_{2n}(1)} \langle \omega, \tilde{A}x \rangle.$$

Следуя [20], опишем эффективный и интерпретируемый способ решения этой задачи. Пусть есть два игрока А и Б. Задана матрица (антагонистической) игры $\tilde{A} = \|\tilde{a}_{ij}\|$, где $|\tilde{a}_{ij}| \leq 1$, \tilde{a}_{ij} – выигрыш игрока А (проигрыш игрока Б) в случае когда игрок А выбрал стратегию i , а игрок Б стратегию j . отождествим себя с игроком Б. И предположим, что игра повторяется $N \gg 1$ раз (это число может быть заранее неизвестно [20], однако для простоты изложения будем считать это число известным). Введем функцию потерь (игрока Б) на шаге k $f_k(x) = \langle \omega^k, \tilde{A}x \rangle$, $x \in S_n(1)$, где $\omega^k \in S_{2n}(1)$ – вектор (вообще говоря, зависящий от всей истории игры до текущего момента включительно, в частности, как-то зависящий и от текущей стратегии (не хода) игрока Б, заданной распределением вероятностей (результат текущего разыгрывания (ход Б) игроку А не известен)) со всеми компонентами равными 0, кроме одной компоненты, соответствующей ходу А на шаге k , равной 1. Хотя функция $f_k(x)$ определена на единичном симплексе, по “правилам игры” вектор x^k имеет ровно одну единичную компоненту, соответствующую ходу Б на шаге k , остальные компоненты равны нулю. Обозначим цену игры

$$C = \max_{\omega \in S_{2n}(1)} \min_{x \in S_n(1)} \langle \omega, \tilde{A}x \rangle = \min_{x \in S_n(1)} \max_{\omega \in S_{2n}(1)} \langle \omega, \tilde{A}x \rangle. \text{ (теорема фон Неймана о минимаксе)}$$

Пару векторов (ω, x) , доставляющих решение этой минимаксной задачи (т.е. седловую точку), назовем равновесием Нэша. По определению (это неравенство восходит к Ханнани [26])

$$\min_{x \in S_n(1)} \frac{1}{N} \sum_{k=1}^N f_k(x) \leq C.$$

Тогда, если мы (игрок Б) будем придерживаться следующей рандомизированной стратегии (см., например, [6], [19], [20]), выбирая $\{x^k\}$:

1. $p^1 = (n^{-1}, \dots, n^{-1})$;
2. Независимо разыгрываем случайную величину $j(k)$ такую, что $P(j(k) = j) = p_j^k$;
3. Полагаем $x_{j(k)}^k = 1$, $x_j^k = 0$, $j \neq j(k)$;

4. Пересчитываем

$$p_j^{k+1} \sim p_j^k \exp\left(-\sqrt{\frac{2 \ln n}{N}} \tilde{a}_{i(k)j}\right), \quad j=1, \dots, n,$$

где $i(k)$ – номер стратегии, которую выбрал игрок А на шаге k ; ³

то с вероятностью $\geq 1 - \sigma$

$$\frac{1}{N} \sum_{k=1}^N f_k(x^k) - \min_{x \in S_n(1)} \frac{1}{N} \sum_{k=1}^N f_k(x) \leq \sqrt{\frac{2}{N}} \left(\sqrt{\ln n} + 2\sqrt{2 \ln(\sigma^{-1})} \right),$$

т.е. с вероятностью $\geq 1 - \sigma$ наши (игрока Б) потери ограничены

$$\frac{1}{N} \sum_{k=1}^N f_k(x^k) \leq C + \sqrt{\frac{2}{N}} \left(\sqrt{\ln n} + 2\sqrt{2 \ln(\sigma^{-1})} \right).$$

Самый плохой для нас случай (с точки зрения такой оценки) – это когда игрок А тоже действует (выбирая $\{\omega^k\}$) согласно аналогичной стратегии (только игрок А решает задачу максимизации).⁴ Если и А и Б будут придерживаться таких стратегий, то они сойдутся к равновесию Нэша (седловой точке), причем чрезвычайно быстро [20]:

$$N = 8 \frac{\ln(2n) + 4 \ln(\sigma^{-1})}{\varepsilon^2} - \text{число итераций};$$

$$O\left(n + \frac{s \ln n (\ln n + \ln(\sigma^{-1}))}{\varepsilon^2}\right) - \text{общее число арифметических операций},$$

³ Заметим, что эта стратегия имеет естественную интерпретацию. Мы (игрок Б) описываем на текущем шаге игрока А вектором, компонентны которого – сколько раз игрок А использовал до настоящего момента соответствующую стратегию. Согласно этому вектору частот мы рассчитываем вектор своих ожидаемых потерь (при условии, что игрок А будет действовать согласно этому частотному вектору). Далее, вместо того, чтобы выбирать наилучшую (для данного вектора частот А) стратегию, дающую наименьшие потери, мы используем распределение Гиббса с параметром $\sqrt{2 \ln n / N}$ (экспоненциально взвешиваем с этим параметром вектор ожидаемых потерь с учетом знака). С наибольшей вероятностью будет выбрана наилучшая стратегия, но с ненулевыми вероятностями могут быть выбраны и другие стратегии.

⁴ Игрок А пересчитывает

$$p_i^{k+1} \sim p_i^k \exp\left(\sqrt{\frac{2 \ln(2n)}{N}} \tilde{a}_{ij(k)}\right), \quad i=1, \dots, 2n,$$

где $j(k)$ – номер стратегии, которую выбрал игрок В на шаге k .

где число $s \ll \sqrt{n}$ – ограничивает сверху, число ненулевых элементов в строках и столбцах матрицы P .

Нетривиальным местом здесь является оценка сложности одной итерации $O(s \ln n)$. Получается эта оценка исходя из оценки наиболее тяжелых действий шаг 2 и 4. Сложность тут в том, что чтобы сгенерировать распределение дискретной случайной величины, принимающей n различных значений (в общем случае) требуется $O(n)$ арифметических операций – для первого генерирования (приготовления памяти). Последующие генерирования могут быть сделаны за эффективнее – за $O(\ln n)$. Однако в нашем случае есть специфика, заключающаяся в том, что при переходе на следующий шаг s вероятностей в распределении могли как-то поменяться. Если не нормировать распределение вероятностей, то можно считать, что остальные вероятности остались неизменными. Оказывается, что такая специфика позволяет вместо оценки $O(n)$ получить оценку $O(s \ln n)$.

Замечание. Опишем точнее эту процедуру. У нас есть сбалансированное двоичное дерево высоты $O(\log_2 n)$ с n листом (дабы не вдаваться в технические детали, считаем что число n – есть степень двойки, понятно, что это не так, но можно взять, скажем, наименьшее натуральное m такое, что $2^m > n$ и рассматривать дерево с 2^m листом) и с $O(n)$ общим числом вершин. Каждая вершина дерева (отличная от листа) имеет неотрицательный вес равный сумме весов двух ее потомков. Первоначальная процедура приготовления дерева, отвечающая одинаковым весам листьев, потребует $O(n)$ операций. Но сделать это придется всего один раз. Процедура генерирования дискретной случайной величины с распределением, с точностью до нормирующего множителя, соответствующим весам листьев может быть осуществлена с помощью случайного блуждания из корня дерева к одному из листьев. Отметим, что поскольку дерево двоичное, то прохождение каждой его вершины при случайном блуждании, из которой идут два ребра в вершины с весами $a > 0$ и $b > 0$, осуществляется путем подбрасывания монетки (“приготовленной” так, что вероятность пойти в вершину с весом a – есть $a/(a+b)$). Понятно, что для осуществления этой процедуры нет необходимости в дополнительном условии нормировки: $a+b=1$. Если вес какого-то листа алгоритму необходимо поменять по ходу работы, то придется должным образом поменять дополнительно веса тех и только тех вершин, которые лежат на пути из корня к этому листу. Это необходимо делать, чтобы поддерживать свойство: каждая вершина дерева (отличная от листа) имеет вес равный сумме весов двух ее потомков.

Итак выше в этом разделе была описана процедура генерирования последовательности x^k со сложностью $O(n + s \ln n \ln(n/\sigma)/\varepsilon^2)$, на основе которой можно построить такой частотный вектор

$$\bar{x}^N = \frac{1}{N} \sum_{k=1}^N x^k,$$

компоненты – частоты, с которыми мы использовали соответствующие стратегии, что $\|A\bar{x}^N\|_{\infty} \leq \varepsilon$ с вероятностью $\geq 1 - \sigma$.

5. Численные эксперименты

To Do.

Работа выполнена при поддержке гранта РФФИ 15-31-20571-мол_а_вед. Исследование части авторов также частично было поддержано компанией Яндекс.

Литература

- [1]. *Brin S., Page L.* The anatomy of a large-scale hypertextual web search engine // Comput. Network ISDN Syst. 1998. V. 30(1–7). P. 107–117.
- [2]. *Langville A.N., Meyer C.D.* Google’s PageRank and beyond: The science of search engine rankings. Princeton University Press, 2006.
- [3]. *Назин А.В., Поляк Б.Т.* Рандомизированный алгоритм нахождения собственного вектора стохастической матрицы с применением к задаче PageRank // Автоматика и телемеханика. 2011. № 2. С. 131–141.
- [4]. *Nesterov Y.E.* Subgradient methods for huge-scale optimization problems // CORE Discussion Paper; 2012/2, 2012.
- [5]. *Nesterov Y.E.* Efficiency of coordinate descent methods on large scale optimization problem // SIAM Journal on Optimization. 2012. V. 22. № 2. P. 341–362.
- [6]. *Гасников А.В., Дмитриев Д.Ю.* Об эффективных рандомизированных алгоритмах поиска вектора PageRank // ЖВМиМФ. 2014. Т. 54. № 3. С. 355–371. [arXiv:1410.3120](https://arxiv.org/abs/1410.3120)
- [7]. *Hastie T., Tibshirani R., Friedman R.* The Elements of statistical learning: Data mining, Inference and Prediction. Springer, 2009.

- [8]. *Zhang Y., Roughan M., Duffield N., Greenberg A.* Fast Accurate Computation of Large-Scale IP Traffic Matrices from Link Loads // In ACM Sigmetrics. 2003. San Diego, CA.
- [9]. *Bubeck S.* Theory of convex optimization for machine learning // e-print, 2014. [arXiv:1405.4980](https://arxiv.org/abs/1405.4980)
- [10]. *Candes E.J., Wakin M.B., Boyd S.P.* Enhancing Sparsity by Reweighted ℓ_1 Minimization // J. Fourier Anal. Appl. 2008. V. 14 P. 877–905.
- [11]. *Allen-Zhu Z., Orecchia L.* Linear coupling: An ultimate unification of gradient and mirror descent // e-print, 2014. [arXiv:1407.1537](https://arxiv.org/abs/1407.1537)
- [12]. *Гасников А.В., Дзуреченский П.Е., Нестеров Ю.Е.* Стохастические градиентные методы с неточным оракулом // Автоматика и телемеханика. 2016. (принята к печати) [arxiv:1411.4218](https://arxiv.org/abs/1411.4218)
- [13]. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: Построение и анализ. М.: МЦНМО, 2002.
- [14]. *Frank M., Wolfe P.* An algorithm for quadratic programming // Naval research logistics quarterly. 1956. V. 3. № 1-2. P. 95–110.
- [15]. *Jaggi M.* Revisiting Frank–Wolfe: Projection-free sparse convex optimization // Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. <https://sites.google.com/site/frankwolfegreedytutorial/>
- [16]. *Harchaoui Z., Juditsky A., Nemirovski A.* Conditional gradient algorithms for norm-regularized smooth convex optimization // Math. Program. Ser. B. 2015. http://www2.isye.gatech.edu/~nemirovs/ccg_revised_apr02.pdf
- [17]. *Nesterov Yu.* Complexity bounds for primal-dual methods minimizing the model of objective function // CORE Discussion Papers. 2015/03. 2015.
- [18]. *Juditsky A., Nemirovski A.* First order methods for nonsmooth convex large-scale optimization, II. In: Optimization for Machine Learning. Eds. S. Sra, S. Nowozin, S. Wright. MIT Press, 2012. <http://www2.isye.gatech.edu/~nemirovs/MLOptChapterII.pdf>
- [19]. *Grigoriadis M., Khachiyan L.* A sublinear-time randomized approximation algorithm for matrix games // Oper. Res. Lett. 1995. V. 18. № 2. P. 53–58.
- [20]. *Гасников А.В., Нестеров Ю.Е., Спокойный В.Г.* Об эффективности одного метода рандомизации зеркального спуска в задачах онлайн оптимизации // ЖВМ и МФ. Т. 55. № 4. 2015. С. 55–71. [arXiv:1410.3118](https://arxiv.org/abs/1410.3118)
- [21]. *Nesterov Yu.* Gradient methods for minimizing composite functions // Math. Prog. 2013. V. 140. № 1. P. 125–161.

- [22]. *Shalev-Shwartz S., Zhang T.* Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization // In Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014. P. 64–72.
[arXiv:1309.2375](#)
- [23]. *Zheng Q., Richtárik P., Zhang T.* Randomized dual coordinate ascent with arbitrary sampling // e-print, 2015. [arXiv:1411.5873](#)
- [24]. *Fercoq O., Richtarik P.* Accelerated, Parallel and Proximal Coordinate Descent // e-print, 2013. [arXiv:1312.5799](#)
- [25]. *Qu Z., Richtarik P.* Coordinate Descent with Arbitrary Sampling I: Algorithms and Complexity // e-print, 2014. [arXiv:1412.8060](#)
- [26]. *Lugosi G., Cesa-Bianchi N.* Prediction, learning and games. New York: Cambridge University Press, 2006.