58th Annual IEEE Symposium on Foundations of Computer Science

Much Faster Algorithms for Matrix Scaling

Zeyuan Allen-Zhu
Microsoft Research
zeyuan@csail.mit.edu

Yuanzhi Li
Princeton University
yuanzhil@cs.princeton.edu

Rafael Oliveira
University of Toronto
rafael@cs.toronto.edu

Avi Wigderson
Institute for Advanced Study
avi@ias.edu

***Abstract*—We develop several efficient algorithms for the classical *Matrix Scaling* problem, which is used in many diverse areas, from preconditioning linear systems to approximation of the permanent. On an input $n \times n$ matrix A, this problem asks to find diagonal (scaling) matrices X and Y (if they exist), so that XAY ε-approximates a doubly stochastic matrix, or more generally a matrix with prescribed row and column sums.**

We address the general scaling problem as well as some important special cases. In particular, if A has m nonzero entries, and if there exist X and Y with polynomially large entries such that XAY is doubly stochastic, then we can solve the problem in total complexity $\widetilde{O}(m + n^{4/3})$. This greatly improves on the best known previous results, which were either $\widetilde{O}(n^4)$ or $O(mn^{1/2}/\varepsilon)$.

Our algorithms are based on tailor-made first and second order techniques, combined with other recent advances in continuous optimization, which may be of independent interest for solving similar problems.

***Keywords*-matrix scaling, iterative algorithms, first-order method, second-order method, doubly stochastic**

I. Introduction

The *matrix scaling* problem is natural and simple to describe. Given a non-negative real matrix A, can one *scale* its rows and columns (namely *multiply* each by a non-negative constant) to yield prescribed row *sums* and column *sums*. Note that the number of constraints is the same as the number of degrees of freedom; however, what makes it interesting (beyond the many applications that we detail below) is that the constraints are additive while the scalings are multiplicative.

Taking real non-negative entries and computing the row and column sums actually capture a much more general problem: one can allow A to have complex entries and require the ℓ_p norms of rows and columns, after scaling, to equal prescribed values.[1]

For a non-negative $d \times n$ matrix A, we say A is an (r, c)-matrix if r and c are respectively the vectors of row and column sums of A. Given vectors r and c, the problem of *matrix (r, c)-scaling* is

> to find *positive diagonal* matrices X and Y for which the matrix XAY is an (r, c)-matrix.

When $d = n$ and $r = c = \mathbb{1} \in \mathbb{R}^n$ where $\mathbb{1}$ is the all-one vector, the matrix $(\mathbb{1}, \mathbb{1})$-scaling problem becomes the *doubly stochastic scaling problem.*

While the above exact scaling problem is of interest, its asymptotic version is even more so, both from the algorithmic viewpoint and from the structural one, as it captures natural combinatorial problems. We say that A is *asymptotically (r, c)-scalable* if the row and column sums can reach r and c asymptotically: that is, for every $\epsilon > 0$, there exist positive diagonal matrices X, Y such that, letting $B = XAY$, we have $\|B\mathbb{1} - r\| \le \varepsilon$ and $\|\mathbb{1}^\top B - c\| \le \varepsilon$.[2]

The combinatorial essence of asymptotic scaling follows from a well-known characterization (see Proposition II.2). A matrix A is asymptotically $(\mathbb{1}, \mathbb{1})$-scalable if and only if the permanent of A is positive, namely if the bipartite graph defined by the positive entries in A has a perfect matching. A matrix A is asymptotically (r, c)-scalable if and only if a natural flow on the same bipartite graph[3] has a solution. Duality (Hall's theorem and max-flow-min-cut theorem) gives simple certificates of non-scalability in terms of the patterns of 0's in the matrix A.

The main computational problem we study is: given a matrix A, vectors r, c and $\varepsilon > 0$, determine if A is ε-approximately (r, c) scalable, and if so, find the scaling matrices X, Y.[4]

Before diving into the history of matrix scaling, we explain one of its most basic applications, which also demonstrates its algorithmic importance.

The full (and better formatted) version of this paper is available at https://arxiv.org/abs/1704.02315.

Most of this work was done when Z. Allen-Zhu were a research member at the Institute for Advanced Study, and when Rafael Oliveira was a student at Princeton University.

Z. Allen-Zhu and A. Wigderson are partially supported by NSF grant CCF-1412958, and R. Oliveira is partially supported by NSF grant DMS-1451191, NSF grant CCF-1523816, and a Siebel scholarship.

[1] The simple reduction replaces any entry α in the matrix by $|\alpha|^p$.

[2] The choice of norm in these expressions is not too important, and can be taken to be ℓ_2.

[3] Connect the source to the row vertex i with capacity r_i, and the column vertex j to the sink with capacity c_j.

[4] We shall formally define ε-approximation in Section II. At a high level, it asks the matrix A, after scaling, to have row and column sums being ε-close to r and c respectively.

0272-5428/17 $31.00 © 2017 IEEE
DOI 10.1109/FOCS.2017.87

Preconditioning Linear Systems. When solving a linear system $Az = b$, it is often desirable —for numerical stability and efficiency purposes— to have matrix A be well-conditioned. When this is not the case, one tries to transform A into a "better conditioned" matrix A'. Matrix scaling provides a natural and efficient reduction to do so. For instance, one would *hope* that a scaled matrix A', in which e.g. all row and column p-norms are (say) 1, is better conditioned.[5] For this reason, we can use a matrix scaling algorithm to obtain diagonal matrices X, Y, and define $A' = XAY$. Now, the solution to $Az = b$ can be obtained by solving the (hopefully more numerically stable) linear system $A'z' = Xb$ and setting $z = Y^{-1}z'$. We stress here that A' and A have the same sparsity.

A. History and Prior Work

The matrix (r, c)-scaling problem is so natural and important that it was discovered independently by many different scientific communities, starting in 1937 with the work of Kruithof (6) in telephone forecasting, Deming and Stephan (7) in transportation science, Brown (8) in engineering, Stone (9) in economics, Wilkinson (10) in numerical analysis, and Friedlander (11) and Sinkhorn (12) in statistics. It has been applied in image reconstruction (13), operations research (14; 15), decision and control (16), theoretical computer science (17), and other scientific disciplines. For more references, we refer the reader to the survey (18), the paper (15) and references therein.

Perhaps the most famous algorithm for solving matrix (r, c)-scaling is the *RAS method* (12). [6]The RAS method alternatively applies row and column normalization, where in a row (resp. column) normalization we multiply each $A_{i,j}$ by $r_i \cdot \left(\sum_j A_{ij}\right)^{-1}$ (resp. by $c_j \cdot \left(\sum_i A_{ij}\right)^{-1}$).

In the original paper, Sinkhorn (12) only proved the convergence of the RAS method when A has only strictly positive entries and when $r = c = \mathbb{1}$. The best known complexity result for the RAS method is given by Kalantari *et al.* (19). In particular, they showed that if the entries of A are polynomially bounded, then the RAS method converges in $\widetilde{O}(n/\varepsilon^2)$ iterations[7] for $(\mathbb{1}, \mathbb{1})$-scaling, or in $\widetilde{O}(h^2/\varepsilon^2)$ iterations for (r, c)-scaling, where r and c are integral vectors and $h \stackrel{\text{def}}{=} \|r\|_1 = \|c\|_1$. Kalantari *et al.* also analyzed the RAS method when A is strictly positive in all n^2 entries, or is exactly (r, c)-scalable with polynomial scaling factors.[8] We summarize them in Table I and II.

Other algorithmic approaches were also developed for matrix scaling. The results (15; 20–22) proved asymptotic convergence without giving complexity bounds. Kalantari and Khachiyan (23) used the ellipsoid method, obtaining the first poly-logarithmic dependence on the approximation parameter ε, with total complexity $\widetilde{O}(n^4)$.[9] Balakrishnan (16) used interior point method and obtained a total complexity $\widetilde{O}(n^6)$. Rote and Zachariasen (24) reduced the (r, c)-scaling problem to running $\widetilde{O}(n^4)$ instances of mincost maximum flow. Linial *et al.* (17) proposed the first strongly polynomial algorithm with a total complexity $\widetilde{O}(n^7)$.

B. Our Improvements Over Known Results

We propose four algorithms to tackle the general matrix scaling problem and also some special cases. In all cases, we have outperformed all relevant previous results, and in some cases our complexities are close to linear in terms of the input size.

To state our complexity bounds let us discuss the input conventions we use. We denote by m the number of nonzero entries of A, and assume $m \geq n \geq d$. We assume all entries of A are rational numbers with polynomial sizes (i.e., at most $\mathsf{poly}(n)$ in numerators and denominators[10]), and both r and c are positive integral vectors with entries at most $\mathsf{poly}(n)$. Let $h \stackrel{\text{def}}{=} \|r\|_1 = \|c\|_1 \geq n$.

A complete listing of our results appear in Table I and Table II.

Our `Scaling0` can be viewed as an accelerated version of RAS. Its total complexity is

$$\boxed{\begin{array}{c}\widetilde{O}(mn/\varepsilon^{2/3}) \text{ for the } (\mathbb{1}, \mathbb{1})\text{-scaling, or} \\ \widetilde{O}(mn^{2/3}h^{1/3}/\varepsilon^{2/3}) \text{ for the general } (r, c)\text{-scaling.}\end{array}}$$

This improves the best result of RAS by a factor $\varepsilon^{-4/3}$ for $(\mathbb{1}, \mathbb{1})$-scaling, and a factor $h^{5/3}n^{-2/3}\varepsilon^{-4/3} \geq n\varepsilon^{-4/3}$ for (r, c)-scaling. We stress that even testing scalability requires $\varepsilon < 1/n$ (see (17)) so reducing the ε dependency from ε^{-2} to $\varepsilon^{-2/3}$ (and later to $\mathsf{polylog}(1/\varepsilon)$) is very meaningful.

In the $\mathsf{polylog}(1/\varepsilon)$ complexity regimes, our `Scaling1` and `Scaling3` have complexities

$$\boxed{\begin{array}{c}\widetilde{O}(mn + n^{7/3}) \text{ for } (\mathbb{1}, \mathbb{1})\text{-scaling, or} \\ \widetilde{O}\big(mn + \min\{n^{\frac{5}{2}}, n^{\frac{7}{3}} + mn^{\frac{1}{3}}h^{\frac{1}{2}}\}\big) \text{ for } (r, c)\text{-scaling.}\end{array}}$$

[5]This indeed is widely use in practice (see (1; 2)), and indeed tends to numerically stabilize systems (see (1; 3; 4)), although no theoretical bounds are known (see (5)).

[6]Also known as the Sinkhorn process, discovered by Sinkhorn in 1964 (12). The RAS method fits as an instance of the "alternate minimization" heuristic, of which this is one of the few known instances in which it converges quickly.

[7]Each iteration of the RAS method costs complexity $O(m)$, the number of non-zero entries in A. The necessary word-size is only polylogarithmic in n and $1/\varepsilon$.

[8]That is, A can be (r, c)-scaled with diagonal scaling matrices X, Y where each $X_{i,i}$ and $Y_{j,j}$ are in $\left[\frac{1}{\mathsf{poly}(n)}, \mathsf{poly}(n)\right]$.

[9]Throughout the paper, we use $\widetilde{O}$, $\widetilde{\Omega}$ and $\widetilde{\Theta}$ notions to hide polylogarithmic factors in n and $1/\varepsilon$.

[10]More generally, the complexities scale linearly with the bit-size of the matrix entries.

Subcase	Paper	Total Complexity
full positive matrix	RAS method (25, 1993)	$\widetilde{O}(n^{2+1/2}/\varepsilon)$ ♭
	`Scaling0`	$\widetilde{O}(n^{2+1/3}/\varepsilon^{2/3})$ ♭
	`Scaling1` or `Scaling3`	$\widetilde{O}(n^2)$
scaling factors poly bounded	RAS method (19, 2008)	$\widetilde{O}(mn^{1/2}/\varepsilon)$ ♭
	`Scaling0`	$\widetilde{O}(mn^{1/3}/\varepsilon^{2/3})$ ♭
	`Scaling1`	$\widetilde{O}(m+n^{3/2})$ ♭
	`Scaling2+Scaling3`	$\widetilde{O}(m+n^{4/3})$
general	RAS method (19, 2008)	$\widetilde{O}(mn/\varepsilon^2)$ ♭
	`Scaling0`	$\widetilde{O}(mn/\varepsilon^{2/3})$
	LSW method (17, 1998)	$\widetilde{O}(mn^5)$ ♭
	ellipsoid (23, 1996)	$\widetilde{O}(n^4)$ ♭
	interior point (16, 2004)	$\widetilde{O}(n^6)$ ♭
	max flow (24, 2007)	$\geq \widetilde{\Omega}(mn^4)$ ♭
	`Scaling1`	$\widetilde{O}(mn+n^{5/2})$ ♭
	`Scaling0+Scaling3`	$\widetilde{O}(mn+n^{7/3})$

Table I: $(\mathbb{1},\mathbb{1})$-scaling.

- We use $\widetilde{O}$ to hide log factors in n and $1/\varepsilon$.
- `Scaling0` is *simple, deterministic* just like RAS;
- `Scaling1`, `Scaling2`, and `Scaling3` use SDD linear system solvers and graph sparsification.
- ♭ indicates the complexity is outperformed.

Subcase	Paper	Total Complexity
full positive matrix	RAS method (19, 2008)	$\widetilde{O}(n^2h^{1/2}/\varepsilon)$ ♭
	`Scaling0`	$\widetilde{O}(n^2h^{1/3}/\varepsilon^{2/3})$ ♭
	`Scaling1`	$\widetilde{O}(n^2)$
scaling factors poly bounded	RAS method (19, 2008)	$\widetilde{O}(mh^{1/2}/\varepsilon)$ ♭
	`Scaling0`	$\widetilde{O}(mh^{1/3}/\varepsilon^{2/3})$ ♭
	`Scaling1`	$\widetilde{O}(m+n^{3/2})$
	`Scaling2+Scaling3`	$\widetilde{O}(m+h^{1/2}n^{5/6})$
general	RAS method (19, 2008)	$\widetilde{O}(mh^2/\varepsilon^2)$ ♭
	`Scaling0`	$\widetilde{O}(mn^{2/3}h^{1/3}/\varepsilon^{2/3})$
	LSW method (17, 1998)	$\widetilde{O}(mn^5)$ ♭
	ellipsoid (23, 1996)	$\widetilde{O}(n^4)$ ♭
	interior point (16, 2004)	$\widetilde{O}(n^6)$ ♭
	max flow (24, 2007)	$\geq \widetilde{\Omega}(mn^4)$ ♭
	`Scaling1`	$\widetilde{O}(mn+n^{5/2})$
	`Scaling0+Scaling3`	$\widetilde{O}(mn+n^{\frac{7}{3}}+mn^{\frac{1}{3}}h^{\frac{1}{2}})$

Table II: (r,c)-scaling.

- Following (19), we assume r and c are positive integral vectors and $h=\|r\|_1=\|c\|_1$. Obviously $h\geq n$.
- Since the complexity of maximum flow is at least $\Omega(m)$, we present a complexity lower bound to (24).

If A is (r,c)-scalable with polynomially large scaling factors,[11] our complexities reduce to

$$\widetilde{O}(m+n^{4/3}) \text{ for } (\mathbb{1},\mathbb{1})\text{-scaling, or}$$
$$\widetilde{O}(m+\min\{n^{3/2},h^{1/2}n^{5/6}\}) \text{ for } (r,c)\text{-scaling.}$$

Our Approaches. We have four algorithms `Scaling0`, `Scaling1`, `Scaling2`, and `Scaling3`, all based on tailor-made first and second-order techniques in continuous optimization. We also combine graph sparsification, SDD linear system solvers, and multiplicative weight updates into the optimization process. We now elaborate more on how this is done.

Matrix scaling can be written (in several ways) as the solution to convex optimization problems. We focus on a specific convex objective in this paper, which is the log of the capacity function (26):

$$f(x)\stackrel{\text{def}}{=}\sum_{i=1}^{d} r_i\log\left(\sum_{j\in[n]}A_{i,j}e^{x_j}\right)-c^\top x\ . \quad \text{(I.1)}$$

If A is asymptotically scalable, then the (approximate) minimizer of $f(x)$ corresponds to scaling matrices X,Y such that XAY is an ε-approximate (r,c)-matrix (see Proposition II.3). A similar objective was also studied by Kalantari *et al.* (19).

At a high level, `Scaling0` uses first-order optimization techniques to minimize $f(x)$, and all other methods `Scaling1`, `Scaling2`, and `Scaling3` use a *mixture* of first and second order techniques.

FIRST-ORDER FRAMEWORK. It was known that the RAS method can be viewed as a first-order method (19), but only with convergence rate $1/\varepsilon^2$. Since $f(x)$ is not Lipschitz smooth (i.e., $\nabla^2 f(x)$ does not have a bounded spectral norm), one cannot apply generic optimization methods. We propose first-order building blocks that are specific to the matrix scaling problem, and then use the linear coupling framework of (27) to combine gradient and mirror descent, in order to achieve the $1/\varepsilon^{2/3}$ convergence rate. We call this method `Scaling0` and it outperforms the RAS method in all relevant parameter regimes. Note that `Scaling0` is as simple to implement as the RAS method.

SECOND-ORDER FRAMEWORK. It turns out the Hessian $\nabla^2 f(x)$ is always symmetric diagonally dominant (SDD), so one can invert it efficiently using modern SDD linear system solvers and graph sparsification techniques. This gives hope for designing efficient second-order methods. Unfortunately, $f(x)$ is not self-concordant in the entire space (i.e., it does not satisfy the apologue of $|f'''(x)|\leq 2f''(x)^{3/2}$ in high dimensions), so we cannot apply standard second-order methods (e.g. Newton method). Instead, we show $f(x)$ satisfies a special property: the second-order Taylor approximation of $f(x+\delta)$ at point x is accurate for all vector δ with

[11] Namely, when A can be scaled to an (r,c) matrix with diagonal scaling matrices X,Y that satisfying each $X_{i,i}$ and $Y_{j,j}$ are within $\left[\frac{1}{\mathsf{poly}(n)},\mathsf{poly}(n)\right]$. This condition is satisfied at least when all entries of A are within $\left[\frac{1}{\mathsf{poly}(n)},\mathsf{poly}(n)\right]$.

$\|\delta\|_\infty \le 1/8$:

$$f(x) + \langle \nabla f(x), \delta\rangle + \tfrac{1}{6}\delta^\top \nabla^2 f(x)\delta \le f(x+\delta) \le f(x) + \langle \nabla f(x), \delta\rangle + \delta^\top \nabla^2 f(x)\delta \ .$$

This implies if we can repeatedly minimize

$$f(x) + \langle \nabla f(x), \delta\rangle + \tfrac{1}{6}\delta^\top \nabla^2 f(x)\delta \quad \text{over an } \ell_\infty \text{ constraint on } \delta, \qquad \text{(I.2)}$$

and update $x \leftarrow x + \frac{1}{6}\delta$, then we can have an $\log(1/\varepsilon)$ convergence rate as opposed to $1/\mathsf{poly}(\varepsilon)$.

Our `Scaling1` algorithm uses multiplicative weight update to solve (I.2), our `Scaling2` algorithm uses accelerated gradient descent to solve (I.2), and our final and most involved algorithm `Scaling3` uses more advanced multiplicative weight update in combination with first-order techniques to solve (I.2). We remark here that `Scaling3` needs a warm-start, that is, a point x where $f(x) - \inf_x\{f(x)\}$ is sufficiently small. We use `Scaling0` or `Scaling2` to find such a warm-start.

A Related Problem. In the matrix balancing problem, a symmetric matrix $B \in \mathbb{R}^{n\times n}$ is ℓ_p-balanced if the ℓ_p-norm of its i-th row equals that of its i-th column, for every $i \in [n]$. Given any $A \in \mathbb{R}^{n\times n}$, we wish to find a diagonal matrix D with positive diagonal entries, such that $B = DAD^{-1}$ is ℓ_p-balanced. Our techniques in this paper can also be extended to matrix balancing.

C. A Parallel Work

When preparing this paper, we found out that Cohen, Mądry, Tsipras, and Vladu also worked on the same problem. The two works are completely independent, so the two teams decided to put the results on arXiv on the same date, see

- https://arxiv.org/abs/1704.02315 (ours), and
- https://arxiv.org/abs/1704.02310 (theirs).

We summarize the main differences between the two results as follows:

- They worked on a different objective $g(x,y) = \sum_{ij} A_{i,j} e^{x_i - y_j} - r^\top x - c^\top y$ from our $f(x)$, see (I.1). However, $g(x,y)$ has the same properties as $f(x)$, so their algorithms also apply to $f(x)$, and our algorithms also apply to $g(x,y)$.
- Both results obtained the same *second-order framework* and reduced matrix scaling to solving (I.2). However, in this regime, their result solves (I.2) faster, because they have non-trivially adapted the SDD linear system techniques from (28). They obtained a nearly-linear time $\widetilde{O}(m)$ algorithm for $(\mathbb{1},\mathbb{1})$-scaling, when the scaling factors are polynomially bounded.
- Our *first-order framework* is not studied in Cohen et al. Since our first-order method `Scaling0` is as simple to implement as the RAS method, and also *deterministic*, it may be of practical and perhaps other interests. In particular, `Scaling0` applies to the problem of deterministic approximation of permanent (17).[12]

D. Roadmap

In Section II, we discuss preliminaries. In Section III, we show diameter bounds for the scaling parameters. In Section IV, we present our first-order framework and algorithm `Scaling0`. In Section V, we present our second-order framework. In Section VI, VII, and VIII respectively, we introduce our algorithms `Scaling1`, `Scaling2`, and `Scaling3`. Throughout this paper, we assume exact arithmetic operations for presenting the cleanest proofs; we discuss how to use logarithmic bit-length to implement our algorithms in the full version. Most of the proofs are in the appendix.

II. Notations and Preliminaries

Throughout the paper, we denote by $\|v\|_p$ the p-norm of vector v if $p \in [1,+\infty]$, and $\|v\|$ the Euclidean norm of v when it is clear from the context. We denote by $\|v\|_w \stackrel{\text{def}}{=} \left(\sum_{i=1}^n w_i v_i^2\right)^{1/2}$ the w-normalized Euclidean norm of vector v if w is a positive vector. We denote by $\|v\|_{\mathbf{A}} = (v^\top \mathbf{A} v)^{1/2}$ the matrix-Euclidean norm. We denote by $e^v = (e^{v_i})_i = (e^{v_1}, e^{v_2}, \dots)$, $\log(v) = (\log v_i)_i$, and $v^{-1} = (v_i^{-1})_i$ the component-wise exponentiation, logarithm, and inversion for vector v. Given vectors u, v, we denote by $u \le v$ the relationship that $u_i \le v_i$ for all coordinates i.

Given symmetric matrices $\mathbf{M}$ and $\mathbf{N}$, we write $\mathbf{M} \preceq \mathbf{N}$ if $\mathbf{N} - \mathbf{M}$ is positive semidefinite (PSD). We say a matrix $\mathbf{M}$ is Laplacian if (1) $\mathbf{M}$ is symmetric, (2) $\mathbf{M}_{i,j} \le 0$ for $i \ne j$, and (3) $\mathbf{M}_{i,i} = -\sum_{j\ne i} \mathbf{M}_{i,j}$. It satisfies $v^\top \mathbf{M} v = \sum_{i<j} |\mathbf{M}_{i,j}|(v_i - v_j)^2$ for every vector v. We say a matrix $\mathbf{M}$ is symmetric diagonally dominant (or SDD for short) if (1) $\mathbf{M}$ is symmetric and (2) $\mathbf{M}_{i,i} \ge \sum_{j\ne i} |\mathbf{M}_{i,j}|$. Obviously, a Laplacian matrix is SDD; and an SDD matrix is PSD.

Throughout this paper, $\mathbf{A} \in \mathbb{R}^{d\times n}_{\ge 0}$ is non-negative and of dimensions $d \times n$. We denote by m the number of non-zero entries of $\mathbf{A}$. Without loss of generality, we assume $d \le n \le m$ and the maximum entry of each row of $\mathbf{A}$ is exactly 1. We denote by $\mathbf{A}_i$ the i-th row vector of $\mathbf{A}$. We assume all the positive entries of $\mathbf{A}$ are in the range $\left[\frac{1}{\mathsf{poly}(n)}, 1\right]$ and represented by rational numbers with numerators and denominators at most $\mathsf{poly}(n)$. We also assume $r \in \mathbb{R}^d_{>0}$ and $c \in \mathbb{R}^n_{>0}$ are positive integral

[12] In contrast, due to the randomness in efficient SDD linear system solvers, the second-order algorithms of both this paper and Cohen et al. are randomized.

vectors and each $r_i, c_j \in \{1, 2, \ldots, \mathsf{poly}(n)\}$.[13] We let $h \stackrel{\text{def}}{=} \|r\|_1 = \|c\|_1$.

Definition II.1. *Given $r \in \mathbb{R}^d_{>0}, c \in \mathbb{R}^n_{>0}$ and $\mathbf{A} \in \mathbb{R}^{d\times n}_{\geq 0}$, and denote by $r' \in \mathbb{R}^d_{\geq 0}$ (resp. $c' \in \mathbb{R}^d_{\geq 0}$) the vector of row sums (resp. column sums) of $\mathbf{A}$. We say*

- *$\mathbf{A}$ is an (r,c)-matrix if $r = r'$ and $c = c'$.*
- *$\mathbf{A}$ is an ε-approximate (r,c)-matrix if $r' = r$ and $\|c' - c\|^2_{c^{-1}} = \sum_{j\in[n]} c_j^{-1}(c'_j - c_j)^2 \leq \varepsilon^2$.*[14]
- *$\mathbf{A}$ is (r,c)-scalable if there exists positive diagonal matrices $\mathbf{X}, \mathbf{Y}$ so $\mathbf{XAY}$ is an (r,c)-matrix.*
- *$\mathbf{A}$ is asymptotically (r,c)-scalable if for every $\varepsilon > 0$, there exist positive diagonal matrices $\mathbf{X}, \mathbf{Y}$ so that $\mathbf{XAY}$ is an ε-approximate (r,c)-matrix.*

It is known that the existence of (r,c)-scaling can be characterized by the following proposition.

Proposition II.2 ((15))**.** *A non-negative matrix $\mathbf{A} \in \mathbb{R}^{d\times n}_{\geq 0}$ is exactly (r,c)-scalable if and only if $\|r\|_1 = \|c\|_1$ and for every zero minor $R \times C \subseteq [d] \times [n]$ of $\mathbf{A}$,*

1) *$\sum_{i\in[d]\setminus R} r_i \geq \sum_{j\in C} c_j$ or equivalently $\sum_{i\in R} r_i \leq \sum_{j\in[n]\setminus C} c_j$.*
2) *Equality in 1 holds iff the minor $([d]\setminus R)\times([d]\setminus C)$ is all zero as well.*

A nonnegative matrix $\mathbf{A}$ is asymptotically (r,c)-scalable if condition 1 holds.

Proposition II.3. *Our objective $f(x)$ in (I.1) is convex and*

- *$\nabla_j f(x) = \sum_{i=1}^d \frac{r_i \mathbf{A}_{i,j}}{\langle \mathbf{A}_i, e^x\rangle} e^{x_j} - c_j$.*
- *If $\|\nabla f(x)\|^2_{c^{-1}} \leq \varepsilon$, then $\left(\frac{r_i \mathbf{A}_{i,j}\cdot e^{x_j}}{\langle \mathbf{A}_i, e^x\rangle}\right)_{i,j}$ is an ε-approximate (r,c)-matrix.*
- *If $\mathbf{A}$ is exactly (r,c)-scalable, then there exists x^* so that $f(x^*) = \min_x\{f(x)\}$ and $\nabla f(x^*) = 0$.*
- *If $\mathbf{A}$ is asymptotically (r,c)-scalable, then $\inf_x\{f(x)\} > -\infty$.*
- *$\mathbf{A}$ is not asymptotically (r,c)-scalable if and only if $\inf_x\{f(x)\} = -\infty$.*

III. New Bounds on Scaling Parameters

We first recall a few bounds for (r,c)-scalable matrices that are essentially from prior work.

Lemma III.1 (objective bound)**.** *For every x satisfying $\|x\|_\infty \leq N$, we have $f(0) - f(x) \leq 2hN$.*

Proof of Lemma III.1: Denoting $x' = x + \|x\|_\infty \mathbb{1}$, we know that $f(x') = f(x)$ and $\|x'\|_\infty \leq 2\|x\|_\infty$. On the other hand, since for every $i \in [d]$, we have $\langle \mathbf{A}_i, e^{x'}\rangle \geq \langle \mathbf{A}_i, \mathbb{1}\rangle \geq 1$, it satisfies $f(0) - f(x') \leq c^\top x' \leq 2hN$. ∎

Lemma III.2 (diameter bound)**.** *If $\mathbf{A}$ is exactly (r,c) scalable, and all non-zero entries of $\mathbf{A}$ are within $[\nu, 1]$ for some $\nu > 0$. Then, the following holds:*

1) *If $\mathbf{A}$ is full (i.e., $\forall i,j, \mathbf{A}_{i,j} > 0$) then there exists a minimizer x^* of $f(x)$ s.t. $\|x^*\|_\infty \leq \ln \frac{hn}{\nu}$.*
2) *If $\mathbf{A}$ is not full, then there exists a minimizer x^* of $f(x)$ such that $\|x^*\|_\infty \leq (h + 1/2)\ln \frac{h}{\nu}$.*

In this paper, we improve (the second item of) Lemma III.2 in two aspects. First, we allow $\mathbf{A}$ to be *asymptotically* (r,c)-scalable. Second, we improve the diameter bound from $\widetilde{O}(h)$ to $\widetilde{O}(n)$ for arbitrary (r,c). (Recall that r and c are integral so $h \geq n$.)

Lemma III.3 (diameter bounds for the asymptotic case)**.** *If $\mathbf{A}$ is asymptotically (r,c)-scalable, and all non-zero entries of $\mathbf{A}$ are within $[\nu, 1]$ for some $\nu > 0$, then, for every $\varepsilon > 0$, there exists $x^*_\varepsilon \in \mathbb{R}^n$ such that*

$$\|x^*_\varepsilon\|_\infty = O\left(n \ln \tfrac{nh}{\nu\varepsilon}\right) \quad , \quad \|\nabla f(x^*_\varepsilon)\|_\infty \leq \varepsilon \ ,$$
$$\text{and} \quad f(x^*_\varepsilon) - \inf_x \left\{f(x)\right\} \leq \varepsilon \ .$$

One can verify that Lemma III.3 is tight (up to constant factors) for instance when $\mathbf{A}$ is a square upper-triangular matrix and the diagonal of $\mathbf{A}$ equals $r = c$.

IV. A New First-Order Framework

In this section, we minimize $f(x)$ using a specially designed first-order optimization method, and finds an ε-approximate (r,c)-scaling with a total complexity that scales with $\varepsilon^{-2/3}$.

High-Level Intuition. We first illustrate why the convergence rate $\varepsilon^{-2/3}$ is reasonable from an optimization standpoint. Recall that if we are given a convex function $g(x)$ that is $O(1)$-Lipschitz smooth —meaning that its Hessian $\nabla^2 g(x)$ has a bounded spectral norm— then, using accelerated gradient descent (29; 30), one can find a point x_1 satisfying $g(x_1) - g(x^*) \leq O\left(\frac{\|x^*\|_2^2}{T^2}\right)$ in T iterations, where x^* is a minimizer of $g(x)$. At the same time, also recall that each step of gradient descent $x' = x - \nabla g(x)$ decreases the objective by at least $g(x) - g(x') \geq \frac{1}{2}\|\nabla g(x)\|_2^2$, so we can apply another T steps of gradient descent on top of x_1, and obtain a point x_2 satisfying $\|\nabla g(x_2)\|_2^2 \leq O\left(\frac{\|x^*\|_2^2}{T^3}\right)$. In other words, we reach x_2 with $\|\nabla g(x_2)\|_2^2 \leq \varepsilon^2$ in $T \propto \varepsilon^{-2/3}$ iterations.

Unfortunately, the function $f(x)$ we are dealing in this paper is not Lipschitz smooth, so we cannot apply the above approach. This is also why previous results using first-order techniques only achieve $1/\varepsilon^2$ rate in

[13] This assumption was also made for instance by Kalantari *et al.* (19).

[14] In certain literature people have also used $\|c' - c\|_2^2 \leq \varepsilon^2$ as the definition of ε-approximation (19). However, their performance loses a factor of $\|c\|_\infty$ so we used this $\|\cdot\|_{c^{-1}}$ notation to simplify our and their statements.

general and $1/\varepsilon$ rate in some special cases (see Table I and II).[15]

Instead, we use the linear-coupling framework of (27) to recover this $\varepsilon^{-2/3}$ convergence rate without using smoothness. To apply linear coupling, we need to design

- a problem-specific gradient descent step, which is a direction δ to move so that the objective decrease $f(x) - f(x+\delta)$ is sufficiently large;
- a problem-specific mirror descent step, which is an online update rule which ensures $\langle \nabla f(x), x - u\rangle$ is small for "any" vector u; and
- a linear combination of the analysis of the two for a faster convergence.

Furthermore, due to technical difficulties, we need to ensure the updates are always inside some infinite-norm box. This adds some extra difficulty in the proofs.

Roadmap. We introduce our gradient and mirror descent steps in Section IV-A and IV-B respectively, and present our linear coupling method `LC` in Algorithm 1, and analyze it in Section IV-C. In Section IV-D we build our algorithm `Scaling0` using `LC` as a subroutine, and present the final theorems. We introduce the following notion for convenience:[16]

Definition IV.1 (gradient split)**.** *At any $x \in \mathbb{R}^n$, define small and large gradients $\nabla^{\mathsf{s}}, \nabla^{\mathsf{l}} \in \mathbb{R}^n$ by*

$$\forall j \in [n]\colon \quad \nabla^{\mathsf{s}}_j \stackrel{\text{def}}{=} \min\left\{c_j, \nabla_j f(x)\right\} \in [-c_j, c_j] \quad \textit{and}$$
$$\nabla^{\mathsf{l}}_j \stackrel{\text{def}}{=} \nabla_j f(x) - \nabla^{\mathsf{s}}_j \geq 0 \ .$$

Also, define small and large coordinates $\Lambda^{\mathsf{s}}, \Lambda^{\mathsf{l}} \subseteq [n]$ by

$$\Lambda^{\mathsf{s}} \stackrel{\text{def}}{=} \left\{j \in [n]\colon \nabla_j \in [-c_j, c_j]\right\} \quad \textit{and}$$
$$\Lambda^{\mathsf{l}} \stackrel{\text{def}}{=} [n] \setminus \Lambda^{\mathsf{s}} = \left\{j \in [n]\colon \nabla_j > c_j\right\} \ .$$

(The above definition has used the trivial fact that $\nabla_j f(x) \in [-c_j, +\infty)$ for any j.)

A. A Specific Gradient Descent

We now introduce a problem-specific gradient descent. Recall that when analyzing a smooth function $g(x)$, one can show a quadratic lower bound

$$g(x) - g(x+\delta) \geq Q(x,\delta) \stackrel{\text{def}}{=} -\langle \nabla g(x), \delta\rangle - \tfrac{1}{2}\|\delta\|_2^2 \ ,$$

and thus choosing $\delta = \arg\max_\delta \left\{Q(\delta)\right\} = -\nabla g(x)$ gives a decrease $g(x) - g(x+\delta) \geq \frac{1}{2}\|\nabla g(x)\|_2^2$.

For our function $f(x)$, we show a similar quadratic lower bound:

[15]For instance, the RAS method can be viewed as performing a gradient descent step $x' = x - \nabla f(x)$ (19).

[16]Recall that each coordinate $\nabla_j f(x)$ is in the interval $[-c_j, \infty)$. This gradient splitting technique was earlier introduced to solve positive linear programming and semidefinite programming (31–33).

Lemma IV.2. *Given $x \in \mathbb{R}^n$, denote by $\nabla = \nabla f(x)$ and $\Lambda^{\mathsf{s}}, \Lambda^{\mathsf{l}} \subseteq [n]$ the set of small and large coordinates (see Def. IV.1). Then, for every $\delta \in \mathbb{R}^n$ where $\|\delta\|_\infty \leq 1/2$, we have*

- *if $\delta \geq 0$, then $f(x) - f(x+\delta) \geq Q^+(x,\delta) \stackrel{\text{def}}{=} \sum_{j\in\Lambda^{\mathsf{s}}}\left(-\nabla_j\cdot\delta_j - \frac{4}{3}c_j\cdot\delta_j^2\right) + \sum_{j\in\Lambda^{\mathsf{l}}}\left(-\frac{7}{3}\nabla_j\cdot\delta_j\right)$.*
- *if $\delta \leq 0$, then $f(x) - f(x+\delta) \geq Q^-(x,\delta) \stackrel{\text{def}}{=} \sum_{j\in\Lambda^{\mathsf{s}}}\left(-\nabla_j\cdot\delta_j - \frac{4}{3}c_j\cdot\delta_j^2\right) + \sum_{j\in\Lambda^{\mathsf{l}}}\left(-\frac{1}{2}\nabla_j\cdot\delta_j\right)$.*

(Recall that $\delta \geq 0$ or $\delta \leq 0$ means entry-wise non-negativity or non-positivity.)

The above quadratic lower bounds distinct from the classical one $Q(x,\delta)$ in two aspects. First, for large coordinates $j \in \Lambda^{\mathsf{l}}$, we only have a linear lower bound. Second, Q^+ and Q^- have different forms for $\delta \geq 0$ and $\delta \leq 0$. Here is an explanation for such two distinctions.

Consider even a simple univariate function $h(x) = e^x - 1$. First, we do not have $h(0) - h(\delta) \geq -h'(0)\delta - C\delta^2$ for any constant C, so we cannot have a quadratic lower bound. Second, one can try to show inequalities like

$$h(0) - h(\delta) \geq -C_1 h'(0)\delta \text{ for } \delta \geq 0 \quad \text{and}$$
$$h(0) - h(\delta) \geq -C_2 h'(0)\delta \text{ for } \delta \leq 0.$$

However, it must satisfy $C_1 > 1$ and $C_2 < 1$, so the two constants must be distinct for $\delta \geq 0$ and $\delta \leq 0$. We choose to let $C_1 = \frac{7}{3}$ and $C_2 = \frac{1}{2}$.

Lemma IV.2 suggests us to perform gradient descent as (one of) the minimizer of Q^+ and Q^-:

Definition IV.3 (gradient descent)**.** *Given x satisfying $\|x\|_\infty \leq N$, define the projected gradient descent step $x' \leftarrow \texttt{Grad}^N(x)$ where $\texttt{Grad}^N(x) \stackrel{\text{def}}{=} \arg\min_{y\in\{y_1,y_2\}}\left\{f(y)\right\}$ where*

$$y_1 = x + \arg\max_{\delta\in\Omega^+_{N,x}}\left\{Q^+(x,\delta)\right\} \quad \textit{and}$$
$$\Omega^+_{N,x} \stackrel{\text{def}}{=} \left\{\delta \geq 0 \mid \|x+\delta\|_\infty \leq N,\ \|\delta\|_\infty \leq 1/2\right\}$$
$$y_2 = x + \arg\max_{\delta\in\Omega^-_{N,x}}\left\{Q^-(x,\delta)\right\} \quad \textit{and}$$
$$\Omega^-_{N,x} \stackrel{\text{def}}{=} \left\{\delta \leq 0 \mid \|x+\delta\|_\infty \leq N,\ \|\delta\|_\infty \leq 1/2\right\}$$

Obviously, $\texttt{Grad}^N(x)$ can be computed in complexity $O(n+m)$.

Note that in the definition above, we have specified a parameter N which ensures that the output $x' = \texttt{Grad}^N(x)$ is also in the box $\|x'\|_\infty \leq N$. One can also let $N = +\infty$ and this means that we put no constraint on $\|x'\|_\infty$. The next two are direct corollaries of Lemma IV.2:

Corollary IV.4. *If $x' = \texttt{Grad}^N(x)$, then we have*

$$f(x) - f(x') \geq \frac{1}{2}\Big(\max_{\delta \in \Omega^+_{N,x}} \{Q^+(x,\delta)\} + \max_{\delta \in \Omega^-_{N,x}} \{Q^-(x,\delta)\} \Big) \geq 0 \ .$$

Corollary IV.5. *If $x' = \texttt{Grad}^\infty(x)$ and $\nabla f(x) = \nabla^{\mathsf{s}} + \nabla^{\mathsf{l}}$ (see Def. IV.1), we have*

$$\|x' - x\|_\infty \leq 1/2$$

and

$$f(x) - f(x') \geq \frac{3}{32}\|\nabla^{\mathsf{s}}\|^2_{c^{-1}} + \frac{1}{4}\|\nabla^{\mathsf{l}}\|_1 \geq \Omega\big(\|\nabla^{\mathsf{s}}\|^2_{c^{-1}} + \|\nabla^{\mathsf{l}}\|_1\big) \ .$$

Remark IV.6. Corollary IV.5 replaces the classical gradient descent statement on smooth functions $g(x)$ that says $g(x) - g(x') \geq \frac{1}{2}\|\nabla g(x)\|_2^2$. Corollary IV.4 is the constrained version of Corollary IV.5.

B. A Specific Mirror Descent

The mirror descent step we take is a constrained minimization with respect to the $\|\cdot\|_c^2$ norm:

Definition IV.7 (mirror descent)**.** *Given z satisfying $\|z\|_\infty \leq N$, a feedback vector $v \in \mathbb{R}^n$, define the projected mirror descent step $z' \leftarrow \texttt{Mirr}^N(z, v)$ as*

$$\texttt{Mirr}^N(z,v) \stackrel{\text{def}}{=} \arg\min_{\|z'\|_\infty \leq N} \big\{ \langle v, z' \rangle + \tfrac{1}{2}\|z' - z\|_c^2 \big\} \ .$$

Obviously, $\texttt{Mirr}^N(z, v)$ can be computed in complexity $O(n)$.

The following lemma is classical for mirror descent:

Lemma IV.8. *If $z' = \texttt{Mirr}^N(z, v)$, then for every u satisfying $\|u\|_\infty \leq N$, we have*

$$\langle v, z - u \rangle \leq \langle v, z - z' \rangle - \tfrac{1}{2}\|z - z'\|_c^2 + \tfrac{1}{2}\|z - u\|_c^2 - \tfrac{1}{2}\|z' - u\|_c^2 \ .$$

C. Linear Coupling

We now introduce our linear-coupling algorithm LC (see Algorithm 1). Starting from two initial vectors y_0 and $z_0 = 0$, in each iteration $k = 0, 1, \ldots, T-1$, our LC chooses a linear combination $x_{k+1} = \tau_k z_k + (1 - \tau_k) y_k$ for some parameter $\tau_k \in (0, 1)$, and performs two updates: $y_{k+1} = \texttt{Grad}^{15N}(x_{k+1})$ and $z_{k+1} = \texttt{Mirr}^N(z_k, \alpha_k \nabla^{\mathsf{s}})$. Here, $\alpha_k > 0$ is the learning rate for mirror descent. The choices of τ_k and α_k are in Algorithm 1. From the description:

Fact IV.9. *We always have $\|z_k\|_\infty \leq N$, $\|x_k\|_\infty \leq 15N$, and $\|y_k\|_\infty \leq 15N$.*

Proof of Fact IV.9: y_0 and $z_0 = 0$ both satisfy norm bounds. y_k comes from gradient descent with range $15N$ so $\|y_k\|_\infty \leq 15N$; z_K comes from mirror descent with range N so $\|z_k\|_\infty \leq N$; finally, x_k is a convex combination of y_{k-1} and z_{k-1} so satisfies $\|x_k\|_\infty \leq 15N$. ■

We show the following lemma which describes the one-iteration behavior of LC:

Lemma IV.10. *If $\tau_k \alpha_k \leq 3/64$, $\tau_k \in \big(0, \frac{1}{32N}\big]$, and u is any vector satisfying $\|u\|_\infty \leq N$, then*

$$0 \leq \tfrac{1-\tau_k}{\tau_k}\big(f(y_k) - f(u)\big) - \tfrac{1}{\tau_k}\big(f(y_{k+1}) - f(u)\big) + \frac{1}{2\alpha_k}\|z_k - u\|_c^2 - \frac{1}{2\alpha_k}\|z_{k+1} - u\|_c^2 \ .$$

Lemma IV.10 is the main technical contribution of this section, and relies on careful applications of Lemma IV.2 and Lemma IV.8, together with tailor-made analysis for our $f(x)$. The next theorem is a corollary of Lemma IV.10 by appropriate choices τ_k and α_k, and telescoping $k = 0, 1, \ldots, T-1$.

Theorem IV.11 (LC)**.** *If y_0 satisfies $\|y_0\|_\infty \leq 15N$ and $T \geq 1$, then the output $y_T = \texttt{LC}(\mathbf{A}, N, T, y_0)$ (see Algorithm 1) satisfies that for every $u \in \mathbb{R}^n$ and $\|u\|_\infty \leq N$:*

$$\|y_T\|_\infty \leq 15N$$

and

$$f(y_T) - f(u) \leq O\Big(\frac{N^2\big(f(y_0) - f(u) + h\big)}{(N+T)^2}\Big) \ .$$

D. Complexity Statements

The $\frac{N^2}{(N+T)^2}\big(f(y_0) - f(u)\big)$ term in Theorem IV.11 can hurt the performance of LC.[17] For this reason, as a warm start, one needs to repeatedly apply LC for $\log N$ times, each with $T = \Theta(N)$. We summarize this final algorithm as Scaling0 in Algorithm 2 and present the final theorem:

Theorem IV.12 (Scaling0)**.** *If $N \geq 1$, then $(z_1, z) = \texttt{Scaling0}(\mathbf{A}, N, T)$ satisfies*

- *If $T \geq N$, then for every u satisfying $\|u\|_\infty \leq N$, we have*

  $$\|z_1\|_\infty \leq 15N \quad \text{and} \quad f(z_1) - f(u) \leq O\big(\tfrac{N^2 h}{T^2}\big) \ .$$

- *If $T \geq (N^2 h)^{1/3}$ and there exists u so that $\|u\|_\infty \leq N$ and $f(u) - \inf_x\{f(x)\} \leq 1$, then*

  $$\|\nabla f(z)\|^2_{c^{-1}} \leq O\big(\tfrac{N^2 h}{T^3}\big) \ .$$

The total complexity of Scaling0 *is $O(m(N \log N + T))$.*

(Due to technical reasons, we do not have bound on $\|z\|_\infty$.)

Recall that to obtain an ε-approximate (r, c)-scaling, it suffices to find z with $\|\nabla f(z)\|^2_{c^{-1}} \leq$

[17] For instance, the general upper bound on $f(0) - f(x^*)$ is only $\widetilde{O}(Nh)$ (see Lemma III.1).

Algorithm 1 $\texttt{LC}(\mathbf{A}, N, T, y_0)$

Input: $\mathbf{A} \in \mathbb{R}^{d\times n}$, a non-negative matrix; $N \geq 1$, a diameter bound; $T \geq 1$, number of iterations; $y_0 \in \mathbb{R}^n$ a starting vector satisfying $\|y_0\|_\infty \leq 15N$;

$z_0 \leftarrow 0$ and $\tau_0 \leftarrow \frac{1}{32N}$;
for $k = 0$ **to** $T-1$ **do**
  $\tau_k \leftarrow$ the unique positive root of the quadratic equation $\frac{\tau_k^2}{\tau_{k-1}^2} + \tau_k - 1 = 0$;
  $x_{k+1} \leftarrow \tau_k z_k + (1-\tau_k) y_k$; ◇ $\tau_k \in (0,1)$
  $y_{k+1} \leftarrow \texttt{Grad}^{15N}(x_{k+1})$; ◇ *see Def. IV.3*
  Define $\nabla^{\mathsf{s}} \in \mathbb{R}^n$ where $\nabla^{\mathsf{s}}_j \leftarrow \min\{\nabla_j f(x_{k+1}), 1\}$;
  $z_{k+1} \leftarrow \texttt{Mirr}^N\big(z_k,\, \alpha_k \nabla^{\mathsf{s}}\big)$ where $\alpha_k = \frac{3}{64\tau_k}$; ◇ *see Def. IV.7*
end for
return y_T. ◇ y_T *satisfies* $\|y_T\|_\infty \leq 15N$

Algorithm 2 $\texttt{Scaling0}(\mathbf{A}, N, T)$

Input: $\mathbf{A} \in \mathbb{R}^{d\times n}$, a non-negative matrix; $N \geq 1$, a diameter bound; $T \geq 1$, number of iterations;

$z_0 \leftarrow 0$;
for $k = 0$ **to** $\log N$ **do**
  $z_0 \leftarrow \texttt{LC}(\mathbf{A}, N, \Theta(N), z_0)$;
$z_1 \leftarrow \texttt{LC}(\mathbf{A}, N, T, z_0)$;
for $k = 1$ **to** T **do**
  $z_{k+1} \leftarrow \texttt{Grad}^{\infty}(z_k)$;
$z \leftarrow \arg\min_{z\in\{z_1,\ldots,z_T\}} \{\|\nabla f(z)\|^2_{c^{-1}}\}$.
return (z_1, z). ◇ z_1 *satisfies* $\|z_1\|_\infty \leq 15N$

ε^2 (see Proposition II.3). Therefore, we can combine Theorem IV.12 with bounds on the scaling parameters: namely, $N \leq \widetilde{O}(n)$ for the general (r,c)-scaling (see Lemma III.3), or $N \leq \widetilde{O}(1)$ if the scaling parameters are polynomially bounded (see Footnote 11). This gives us the claimed results of `Scaling0` in Table I and Table II.

V. A New Second-Order Framework

In this section, we propose a second-order framework in order to minimize $f(x)$. Our methods `Scaling1`, `Scaling2` and `Scaling3` in subsequent sections are all be based on this framework.

We show that near any point x, the function value $f(x+\delta)$ is well approximated by the second-order Taylor expansion of $f(x)$, as long as $\|\delta\|_\infty \leq 1/8$:

Lemma V.1 (second-order approximation)**.** *For every $x, \delta \in \mathbb{R}^n$ with $\|\delta\|_\infty \leq 1/8$, we have*

$$f(x) + \langle \nabla f(x), \delta\rangle + \frac{1}{6}\delta^\top \nabla^2 f(x)\delta \leq f(x+\delta) \leq f(x) + \langle \nabla f(x), \delta\rangle + \delta^\top \nabla^2 f(x)\delta \ .$$

Note that if $f(x)$ were an arbitrary convex function, such a quadratic approximation would only work for a very small region of δ. It is the special property of the matrix scaling problem that allows us to prove Lemma V.1 for all $\|\delta\|_\infty \leq 1/8$. We include the details in the full version.

Also, one may carefully verify that $\nabla^2 f(x)$ is a Laplacian matrix that may contain up to n^2 non-zero entries even if the original matrix $\mathbf{A}$ is sparse. Using classical graph sparsification techniques (see full version), with total complexity $\widetilde{O}(m)$, one can find another Laplacian matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ satisfying $\mathbf{H} \preceq \nabla^2 f(x) \preceq 1.1\mathbf{H}$, where $\mathbf{H}$ only has $\widetilde{O}(n)$ non-zero entries.

High-Level Intuition. Using Lemma V.1, it becomes natural to study the minimization question $\langle \nabla f(x), \delta\rangle + \frac{1}{6}\delta^\top \mathbf{H}\delta$ over all $\|\delta\|_\infty \leq 1/8$. If δ^* is such a minimizer, then one can show $f(x) - f(x+\delta^*) \geq \Omega\big(\frac{1}{\|x-x^*\|_\infty}\big)(f(x) - f(x^*))$ where x^* is the minimizer of $f(x)$. This sounds like we only needed $O(N\log(1/\varepsilon))$ iterations in total if $\|x^*\|_\infty \leq N$.

Unfortunately, this approach fails because $\|x - x^*\|_\infty$ may increase by $1/8$ per iteration, so the convergence rate may drop to $1/\varepsilon$ as opposed to $\log(1/\varepsilon)$. We fix this issue by restricting our attention only to the region $\{x \in \mathbb{R}^n \mid \|x\|_\infty \leq N\}$. If this region contains x^*, and if we can minimize $\langle \nabla f(x), \delta\rangle + \frac{1}{6}\delta^\top \mathbf{H}\delta$ over the intersection of $\|x+\delta\|_\infty \leq N$ and $\|\delta\|_\infty \leq 1/8$, then we can always have $f(x) - f(x+\delta^*) \geq \Omega\big(\frac{1}{N}\big)(f(x) - f(x^*))$ and thus converge in $O(N\log(1/\varepsilon))$ iterations.

For the reason above, we wish to repeatedly solve the following minimization problem

$$\boxed{\min_{\delta \in \mathsf{box}^N(x)} \Big\{\langle \nabla f(x), \delta\rangle + \frac{1}{6}\delta^\top \mathbf{H}\delta\Big\}} \qquad \text{(V.1)}$$

Definition V.2. *Given any point $x \in \mathbb{R}^n$ satisfying $\|x\|_\infty \leq N$ for some $N > 1$, we define*

$$\mathsf{box}^N(x) \stackrel{\text{def}}{=} \left\{\delta \in \mathbb{R}^n \;\middle|\; \|\delta - \alpha\|_\infty \leq \frac{1}{32}\right\}$$

where

$$\alpha_i \stackrel{\text{def}}{=} \begin{cases} (\frac{1}{32} - N - x_i) \in (0, \frac{1}{32}], & \text{if } x_i - \frac{1}{32} < -N; \\ (N - x_i - \frac{1}{32}) \in [-\frac{1}{32}, 0), & \text{if } x_i + \frac{1}{32} > N; \\ 0, & \text{otherwise.} \end{cases}$$

Fact V.3. *For all $\delta \in \mathsf{box}^N(x)$, we have $\|x+\delta\|_\infty \le N$ and $\|\delta\|_\infty \le \frac{1}{16}$. We also have $0 \in \mathsf{box}^N(x)$.*

Our next Lemma V.4 says that if we can solve (V.1) up to a small additive error, then we can decrease the objective distance to $f(u)$ by a factor of $1 - \frac{1}{900N}$ up to the same small additive error.

Lemma V.4. *Given x with $\|x\|_\infty \le N$ and $\mathbf{H}$ with $\mathbf{H} \preceq \nabla^2 f(x) \preceq 1.1\mathbf{H}$, the following holds:*

(a) For any $u \in \mathbb{R}^n$ with $\|u\|_\infty \le N$,

$$-\min_{\delta \in \mathsf{box}^N(x)} \left\{ \langle \nabla f(x), \delta \rangle + \tfrac{1}{6}\delta^\top \mathbf{H} \delta \right\} \ge \frac{1}{64N}\big(f(x) - f(u)\big) \ .$$

(b) If we are given $\widehat{\delta}$ satisfying $\|\widehat{\delta}\|_\infty \le 1/8$ and for $\varepsilon \ge 0$:

$$\langle \nabla f(x), \widehat{\delta} \rangle + \tfrac{1}{6}\widehat{\delta}^\top \mathbf{H} \widehat{\delta} \le \min_{\delta \in \mathsf{box}^N(x)} \left\{ \langle \nabla f(x), \delta \rangle + \frac{1}{6}\delta^\top \mathbf{H} \delta \right\} + \varepsilon \ ,$$

then it satisfies that for every $u \in \mathbb{R}^n$ with $\|u\|_\infty \le N$, $f(x) - f\big(x + \frac{\widehat{\delta}}{6.6}\big) \ge \frac{1}{900N}\big(f(x) - f(u)\big) - \varepsilon$.

VI. Second-Order Method 1: via Multiplicative Weight Updates

In this section, we propose `Scaling1` which uses *multiplicative weight update (MWU)* and an ℓ_2 constrained SDD linear system solver to tackle problem (V.1).

High-Level Intuitions. Denote by $h(\delta) \stackrel{\text{def}}{=} \langle \nabla f(x), \delta \rangle + \frac{1}{6}\delta^\top \mathbf{H} \delta$ for notation simplicity.

Given any weight vector $w \in \Delta$ where $\Delta \stackrel{\text{def}}{=} \{w \in [1/2, n]^n \mid \sum_i w_i = n\}$, instead of minimizing $h(\delta)$ over all $\delta \in \mathsf{box}^N(x) = \left\{\delta \in \mathbb{R}^n \,\middle|\, \|\delta - \alpha\|_\infty \le \frac{1}{32}\right\}$, we can minimize $h(\delta)$ over a larger set $\Omega_w = \left\{\delta \in \mathbb{R}^n \,\middle|\, \|\delta - \alpha\|_w^2 \le \frac{n}{1024}\right\}$.[18] We would like do so because ℓ_2 constrained minimization is computationally cheap: minimizing $h(\delta)$ over Ω_w can be done using a variant of SDD linear system solvers in total complexity $\widetilde{O}(n)$ (see full version).

Next, we wish to apply the multiplicative weight update framework. Starting from some $w_0 \in \Delta$, in each round $k = 0, 1, \ldots, T-1$, we minimize $h(\delta)$ over set Ω_{w_k} and let $\delta_k \in \Omega_{w_k}$ be an approximate minimizer. Then, we update w_{k+1} from w_k by penalizing the coordinates i in δ_k where $|\delta_{k,i} - \alpha_i|$ is large. A variant of the MWU theory implies that, as long as $T = \widetilde{\Omega}(\sqrt{n})$, the average $\overline{\delta} = \frac{1}{T}\sum_{k=0}^{T-1} \delta_k$ satisfies $\|\overline{\delta} - \alpha\|_\infty \le O(1)$. At the same time, since objective δ_k minimizes (V.1) over a larger set $\Omega_w \supseteq \mathsf{box}^N(x)$, we also have $h(\overline{\delta}) \le \frac{1}{T}\sum_{k=0}^{T-1} h(\delta_k) \le \min_{\delta \in \mathsf{box}^N(x)} h(\delta)$. This gives an approximate solution to (V.1), and the total complexity is $\widetilde{O}(nT) = \widetilde{O}(n^{3/2})$ if $\mathbf{H}$ is given.

We summarize the above process as `MWUbasic` (see Algorithm 3), and show the following lemma:

Lemma VI.1 (`MWUbasic`)**.** *If $\mathbf{H} \in \mathbb{R}^{n \times n}$ is Laplacian, $K \ge 1$, $T \ge \Omega((n^{1/2}K + K^2)\log n)$, $\|\alpha\|_\infty \le 1/32$, and $\varepsilon > 0$, then the output $\overline{\delta} = \mathtt{MWUbasic}(\mathbf{A}, \mathbf{H}, \alpha, T, K, \varepsilon)$ satisfies*

$$\|\overline{\delta} - \alpha\|_\infty \le \frac{1}{32} + \frac{1}{8K}$$

and

$$\langle \nabla, \overline{\delta} \rangle + \frac{1}{6}\overline{\delta}^\top \mathbf{H} \overline{\delta} \le \min_{\|\delta - \alpha\|_\infty \le 1/32} \left\{ \langle \nabla, \delta \rangle + \frac{1}{6}\delta^\top \mathbf{H} \delta \right\} + \varepsilon \ .$$

With Lemma VI.1, we can repeatedly apply `MWUbasic` to minimize (V.1) for $\widetilde{O}(N \log(1/\varepsilon))$ times. We summarize the algorithm as `Scaling1` (in Algorithm 4) and have the following final theorem:

Theorem VI.2 (`Scaling1`)**.** *If $N \ge 1$ and $\varepsilon \in (0, 1)$, the output $y = \mathtt{Scaling1}(\mathbf{A}, N, \varepsilon)$ satisfies $\|y\|_\infty \le 2N$ and*

$$f(y) - f(u) \le \varepsilon \quad \text{for every } u \text{ with } \|u\|_\infty \le N.$$

Furthermore, if there exists u satisfying $f(u) - \inf_x\{f(x)\} \le \varepsilon$ and $\|u\|_\infty \le N$, then we also have $\|\nabla f(y)\|_{c^{-1}}^2 \le \varepsilon$. The total complexity is $\widetilde{O}(N(m + n^{3/2}))$.

We can combine Theorem VI.2 with bounds on scaling parameters: namely, $N \le \widetilde{O}(n)$ for the general (r, c)-scaling (see Lemma III.3), or $N \le \widetilde{O}(1)$ if the scaling parameters are polynomially bounded (see Footnote 11). This gives us the claimed results of `Scaling1` in Table I and Table II.

VII. Second-Order Method 2: via Accelerated Gradient Descent

In this section, we propose `Scaling2` (see Algorithm 5) which directly solves the constrained minimization problem (V.1) using a constrained version of accelerated gradient descent (29; 30). We shall not directly use `Scaling2` to solve the matrix scaling problem; instead, we shall later use `Scaling2` as a warm-start for `Scaling3`.

We have the following main lemma to estimate the per-iteration performance of `Scaling2`:

[18] It is easy to verify that $\mathsf{box}^N(x) \subseteq \Omega_w$ and conversely $\|\delta - \alpha\|_\infty \le O(\sqrt{n})$ for every $\delta \in \Omega_w$.

Algorithm 3 $\texttt{MWUbasic}(\nabla, \mathbf{H}, \alpha, T, K, \varepsilon)$

Input: $\nabla \in \mathbb{R}^n$; $\mathbf{H} \in \mathbb{R}^{n\times n}$ a Laplacian matrix; $\alpha \in \mathbb{R}^n$ satisfying $\|\alpha\|_\infty \le 1/32$; $T \ge 1$ number of rounds; $K \ge 1$ a parameter; $\varepsilon > 0$ an accuracy parameter.

1: $\Delta \leftarrow \{w \in [1/2, n]^n : \sum_i w_i = n\}$ and $w_0 \leftarrow (1, 1, \ldots, 1) \in \Delta$;
2: **for** $k = 0$ **to** $T - 1$ **do**
3: Use SDD solver (see full version) to find a vector $\delta_k \in \mathbb{R}^n$ satisfying $\|\delta_k - \alpha\|^2_{w_k} \le \frac{n}{1024}$ and $\langle \nabla f(x), \delta_k\rangle + \frac{1}{6}\delta_k^\top \mathbf{H}\delta_k \le \min_{\|\delta-\alpha\|^2_{w_k} \le n/1024} \left\{\langle \nabla f(x), \delta\rangle + \frac{1}{6}\delta^\top \mathbf{H}\delta\right\} + \varepsilon$
4: Define loss vector $\ell_k \in \mathbb{R}^n$ by $\ell_{k,i} \leftarrow -|\delta_{k,i} - \alpha_i|$.
5: $w_{k+1} \leftarrow \arg\min_{z\in\Delta} \left\{\eta\langle \ell_k, z\rangle + \sum_{i\in[n]} \left(z_i \log \frac{z_i}{w_{k,i}} + w_{k,i} - z_i\right)\right\}$
◇ *a multiplicative weight update with parameter $\eta = 1/(\sqrt{n} + K)$, see full version*
6: **end for**
7: **return** $\overline{\delta} \leftarrow \frac{1}{T}\sum_{k=0}^{T-1} \delta_k$;

Algorithm 4 $\texttt{Scaling1}(\mathbf{A}, N, \varepsilon)$

Input: $\mathbf{A} \in \mathbb{R}^{d\times n}$ non-negative matrix; $N \ge 1$ diameter bound; $\varepsilon \in (0, 1)$ accuracy parameter.

1: $x_0 \leftarrow 0$, $K \leftarrow \Theta(\log(1/\varepsilon))$, and $T \leftarrow \widetilde{\Theta}(\sqrt{n})$;
2: **for** $t = 0$ **to** NK **do**
3: Define $\mathsf{box}^N(x_t)$ and $\alpha \in [-1/32, 1/32]^n$ using Def. V.2;
4: $\mathbf{H} \leftarrow$ a matrix with $\widetilde{O}(n)$ nonzeros satisfying $\mathbf{H} \preceq \nabla^2 f(x_t) \preceq 1.1\mathbf{H}$;
5: $\overline{\delta} \leftarrow \texttt{MWUbasic}(\nabla f(x_t), \mathbf{H}, \alpha, T, K, \frac{\varepsilon}{900N})$;
6: $x_{t+1} \leftarrow x + \frac{\overline{\delta}}{6.6}$ and $N \leftarrow N + \frac{1}{50K}$; ◇ *so x_{t+1} satisfies $\|x_{t+1}\|_\infty \le N$ for this new N*
7: **end for**
8: **return** $y \leftarrow$ the last x_t.

Algorithm 5 $\texttt{Scaling2}(\mathbf{A}, N, T)$

Input: $\mathbf{A} \in \mathbb{R}^{d\times n}$, a non-negative matrix; $N \ge 1$, a diameter bound; $T \ge 1$, number of iterations;

1: $x_0 \leftarrow 0$;
2: **for** $t = 0$ **to** $\widetilde{O}(N)$ **do**
3: $\mathbf{H} \leftarrow$ a matrix with $\widetilde{O}(n)$ nonzeros satisfying $\mathbf{H} \preceq \nabla^2 f(x_t) \preceq 1.1\mathbf{H}$;
◇ *see full version for details.*
4: $\delta \leftarrow$ approximate minimizer for $\min_{\delta\in\mathsf{box}^N(x_t)} \left\{\langle \nabla f(x_t), \delta\rangle + \frac{1}{6}\delta^\top \mathbf{H}\delta\right\}$.
◇ *compute δ by applying T steps of constrained accelerated gradient descent. See Lemma VII.1*
5: $x_{t+1} \leftarrow x_t + \delta$;
6: **end for**
7: **return** $y \leftarrow$ the last x_t. ◇ *y satisfies $\|y\|_\infty \le N$*

Lemma VII.1. *In each iteration t of* Scaling2*, if $\|x_t\|_\infty \le N$, then we can compute x_{t+1} in complexity $\widetilde{O}(m + Tn)$, and it satisfies $\|x_{t+1}\|_\infty \le N$ and*

either (1): $f(x_{t+1}) - f(u) \le O\big(\frac{Nh}{T^2}\big)$ *or*

(2): $f(x_t) - f(x_{t+1}) \ge \Omega\big(\frac{1}{N}\big)(f(x_t) - f(u))$.

Here, u is any vector satisfying $\|u\|_\infty \le N$.

The following theorem is a direct corollary of Lemma VII.1.

Theorem VII.2 (Scaling2)**.** *If $T \ge 1$, the output $y = \texttt{Scaling2}(\mathbf{A}, N, T)$ satisfies $\|y\|_\infty \le N$ and $f(y) - f(u) \le O\Big(\frac{Nh}{T^2}\Big)$ for every u with $\|u\|_\infty \le N$. The total complexity $\widetilde{O}\big(mN + NnT\big)$.*

Proof of Theorem VII.2 from Lemma VII.1: Whenever Line 3 is reached, either we have $f(x_{t+1}) - f(u) \le O\big(\frac{Nh}{T^2}\big)$ so we are done, or we have $f(x_{t+1}) - f(u) \le \big(1 - \Omega\big(\frac{1}{N}\big)\big)(f(x_t) - f(u))$. The latter cannot happen more than $\widetilde{O}(N)$ times. ■

VIII. Second-Order Method 3: via More Advanced MWU

Due to space limitation, we include `Scaling3` only in our arXiv version of the paper.

References

[1] T. Pock and A. Chambolle, "Diagonal preconditioning for first order primal-dual algorithms in convex optimization," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1762–1769.

[2] E. Chu, B. OíDonoghue, N. Parikh, and S. Boyd, "A primal-dual operator splitting method for conic optimization," *Stanford Internal Report*, 2013.

[3] O. E. Livne and G. H. Golub, "Scaling by binormalization," *Numerical Algorithms*, vol. 35, no. 1, pp. 97–120, 2004.

[4] A. M. Bradley, "Algorithms for the equilibration of matrices and their application to limited-memory quasi-newton methods," Ph.D. dissertation, Stanford University, 2010.

[5] R. Takapoui and H. Javadi, "Preconditioning via diagonal scaling," *arXiv preprint arXiv:1610.03871*, 2016.

[6] J. Kruithof, "Telefoonverkeersrekening," *De Ingenieur*, vol. 52, pp. E15–E25, 1937.

[7] W. E. Deming and F. F. Stephan, "On a least squares adjustment of a sampled frequency table when the expected marginal totals are known," *The Annals of Mathematical Statistics*, vol. 11, no. 4, pp. 427–444, 1940.

[8] D. T. Brown, "A note on approximations to discrete probability distributions," *Information and Control*, vol. 2, no. 4, pp. 386–392, 1959.

[9] R. Stone, *Multiple classifications in social accounting*. University of Cambridge, Department of Applied Economics, 1964.

[10] J. H. Wilkinson, "Rounding errors in algebraic processes." in *IFIP Congress*, 1959, pp. 44–53.

[11] D. Friedlander, "A technique for estimating a contingency table, given the marginal totals and some supplementary data," *Journal of the Royal Statistical Society. Series A (General)*, pp. 412–420, 1961.

[12] R. Sinkhorn, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *The annals of mathematical statistics*, vol. 35, no. 2, pp. 876–879, 1964.

[13] G. T. Herman and A. Lent, "Iterative reconstruction algorithms," *Computers in biology and medicine*, vol. 6, no. 4, pp. 273–294, 1976.

[14] T. Raghavan, "On pairs of multidimensional matrices," *Linear Algebra and its Applications*, vol. 62, pp. 263–268, 1984.

[15] U. G. Rothblum and H. Schneider, "Scalings of matrices which have prespecified row sums and column sums via optimization," *Linear Algebra and its Applications*, vol. 114, pp. 737–764, 1989.

[16] H. Balakrishnan, Inseok Hwang, and C. Tomlin, "Polynomial approximation algorithms for belief matrix maintenance in identity management," in *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, vol. 5. IEEE, 2004, pp. 4874–4879 Vol.5. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1429569http://ieeexplore.ieee.org/document/1429569/

[17] N. Linial, A. Samorodnitsky, and A. Wigderson, "A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*. New York, New York, USA: ACM Press, 1998, pp. 644–652. [Online]. Available: http://portal.acm.org/citation.cfm?doid=276698.276880

[18] M. Idel, "A review of matrix scaling and sinkhorn's normal form for matrices and positive maps," *arXiv preprint arXiv:1609.06349*, 2016.

[19] B. Kalantari, I. Lari, F. Ricca, and B. Simeone, "On the complexity of general matrix scaling and entropy minimization via the RAS algorithm," *Mathematical Programming*, vol. 112, no. 2, pp. 371–401, 2008.

[20] W. Gorman, "Estimating trends in leontief matrices: a note on mr. bacharach's paper," *Nuffield College, Oxford, duplicated*, 1963.

[21] A. W. Marshall and I. Olkin, "Scaling of matrices to achieve specified row and column sums," *Numerische Mathematik*, vol. 12, no. 1, pp. 83–90, 1968.

[22] S. M. Macgill, "Theoretical properties of biproportional matrix adjustments," *Environment and Planning A*, vol. 9, no. 6, pp. 687–701, 1977.

[23] B. Kalantari and L. Khachiyan, "On the complexity of nonnegative-matrix scaling," *Linear Algebra and its Applications*, vol. 240, pp. 87–103, jun 1996. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/002437959400188X

[24] G. Rote and M. Zachariasen, "Matrix Scaling by Network Flow," in *SODA '07*, 2007, pp. 848–854. [Online]. Available: http://page.mi.fu-berlin.de/rote/Papers/pdf/Matrix+scaling+by+network+flows.pdf

[25] B. Kalantari and L. Khachiyan, "On the rate of convergence of deterministic and randomized ras matrix scaling algorithms," *Operations research letters*, vol. 14, no. 5, pp. 237–244, 1993.

[26] L. Gurvits and A. Samorodnitsky, "A deterministic algorithm for approximating the mixed discriminant and mixed volume, and a combinatorial corollary," *Discrete & Computational Geometry*, vol. 27, no. 4, pp. 531–550, 2002.

[27] Z. Allen-Zhu and L. Orecchia, "Linear Coupling: An Ultimate Unification of Gradient and Mirror Descent," in *Proceedings of the 8th Innovations in Theoretical Computer Science*, ser. ITCS '17, 2017, full version available at http://arxiv.org/abs/1407.1537.

[28] R. Kyng, Y. T. Lee, R. Peng, S. Sachdeva, and D. A. Spielman, "Sparsified Cholesky and multigrid solvers for connection laplacians," in *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing - STOC 2016*. New York, New York, USA: ACM Press, 2016, pp. 842–850. [Online]. Available: http://arxiv.org/abs/1512.01892http://dl.acm.org/citation.cfm?doid=2897518.2897640

[29] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," in *Doklady AN SSSR (translated as Soviet Mathematics Doklady)*, vol. 269, 1983, pp. 543–547.

[30] ——, *Introductory Lectures on Convex Programming Volume: A Basic course*. Kluwer Academic Publishers, 2004, vol. I.

[31] Z. Allen-Zhu and L. Orecchia, "Using optimization to break the epsilon barrier: A faster and simpler width-independent algorithm for solving positive linear programs in parallel," in *Proceedings of the 26th ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '15, 2015.

[32] ——, "Nearly-Linear Time Positive LP Solver with Faster Convergence Rate," in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, ser. STOC '15, 2015.

[33] Z. Allen-Zhu, Y. T. Lee, and L. Orecchia, "Using optimization to obtain a width-independent, parallel, simpler, and faster positive SDP solver," in *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '16, 2016.

[34] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th International Conference on Machine Learning*, ser. ICML 2003, 2003, pp. 928–936.

[35] A. Rakhlin, "Lecture notes on online learning," *Draft*, 2009, available at http://www-stat.wharton.upenn.edu/~rakhlin/courses/stat991/papers/lecture_notes.pdf.

[36] R. Peng and D. A. Spielman, "An efficient parallel solver for SDD linear systems," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing - STOC '14*. New York, New York, USA: ACM Press, 2014, pp. 333–342. [Online]. Available: http://arxiv.org/abs/1311.3286http://dl.acm.org/citation.cfm?doid=2591796.2591832

[37] D. A. Spielman and S.-H. Teng, "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing - STOC '04*. New York, New York, USA: ACM Press, 2004, p. 81.

[38] J. A. Kelner, L. Orecchia, A. Sidford, and Z. A. Zhu, "A simple, combinatorial algorithm for solving SDD systems in nearly-Linear time," in *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, ser. STOC '13, 2013.

[39] R. Kyng and S. Sachdeva, "Approximate Gaussian Elimination for Laplacians - Fast, Sparse, and Simple," in *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, oct 2016, pp. 573–582. [Online]. Available: https://arxiv.org/pdf/1605.02353.pdfhttp://ieeexplore.ieee.org/document/7782972/

[40] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization*. Society for Industrial and Applied Mathematics, Jan. 2013.