

**My Memories of Leonid Khachiyan and a Personal Tribute for His Contributions
in Linear Programming**

Bahman Kalantari

Department of Computer Science
Rutgers University
New Brunswick, New Jersey

In the Fall of 1989 Louis Billera, a professor of mathematics and operations research at Cornell University, visiting Rutgers University that year, invited me to give an operations research colloquium talk at Cornell. He also told me that at the invitation of Mike Todd, Leonid Khachiyan would be arriving at Cornell to spend some time at the School of Operations Research and Industrial Engineering. My invitation to Cornell and the promise of meeting Khachiyan was really exciting. In 1979 Khachiyan had presented a new algorithm for the linear programming problem that in particular proved the problem was tractable. It was a groundbreaking result that even made it to the front page of the New York Times. Like many others that was the first time I read about Khachiyan's work which became known as the "ellipsoid method."

A few years prior to Khachiyan's work, when I was a graduate student in mathematics at the University of Minnesota and I knew that I did not want to get my Ph.D in classical mathematics, I had decided to explore and in doing so learned about the field of operations research and the subject of linear programming. For the most part I taught myself because the courses on linear programming offered at various departments were cookbook style and non-rigorous. Linear programming courses were offered at different departments such as computer science, industrial engineering and business, but not in the mathematics department. I liked the subject of linear programming by itself, and also because I had learned that, to my amazement, it overlapped with some combinatorial optimization problems such as the matching problem, and the network flow problem. These problems were unfamiliar to graduate students in mathematics. I remember mentioning them to a fellow graduate student, and she said, "Do you know what they say about operation researchers?" I shook my head no. So she responded, "mathematical deviates."

That attitude was shocking. I was not even speaking of operations research, rather the subject of linear programming and combinatorial optimization. I was looking at these problems from a mathematical point of view and found them quite interesting in their own right. Many years later, I still find my view of these problems unchanged in the sense that I am not concerned with their "practical applications." To me these problems represented a very interesting kind of mathematics because they dealt with interesting algorithms. I have always liked pure mathematics, but in retrospect I was more interested in algorithmic mathematics which at the time really didn't exist in the broad sense that it does today. Most likely, advancements in computer technology is one of the most important factors in making algorithmic mathematics a

more relevant subject today. So what did Khachiyan's discovery of a new algorithm do to linear programming and combinatorial optimization? Personally, I feel his algorithm turned these subjects into more rigorous, more respectable, and more popular disciplines that they used to be. I think he also proved it was okay to have deviated from the norm. Khachiyan could be labeled a mathematician, a computer scientist, an operations researcher, an algorithmic mathematician, or perhaps all of the above, but it is really immaterial. There is no definitive definition of any of these subjects anyway.

Before Khachiyan's work on linear programming in 1979, nobody knew of an algorithm that would solve every instance of the linear programming problem in a sound fashion - at least in the theoretical sense. The best known algorithm for linear programming was the "simplex method," which is mostly credited to George Dantzig who enormously popularized the subject of linear programming. Although in general the simplex method is a pretty good algorithm, one can cook up specific problems that would make the algorithm run a very long time. Just because a particular algorithm may not be a good algorithm for solving a particular problem, it does not mean that the particular problem is hard. It could be that the algorithm is not right. What Khachiyan did was to exhibit an algorithm for linear programming which was a good algorithm from the theoretical point of view. In theory, no particular instance of linear programming could break its back. And that was a monumental achievement! Khachiyan's pioneering work broke the barriers and opened the way for further discoveries in the entire field of optimization.

As a graduate student I was very excited about Khachiyan's algorithm because it was so different than the standard simplex-based algorithms. I certainly did not want to be called a mathematical deviate just because I did not want to specialize in classical mathematics. Somehow I felt that Khachiyan's work would revive the linear programming problem. One day I heard that Ben Rosen, a professor of computer science at University of Minnesota, well-known for his work in non-linear programming, was to give a talk about Khachiyan's algorithm. Rosen informally explained the main steps and spoke of a computational result on a very small example. While the example would be solved easily via the simplex method, using the ellipsoid method it took a lot of iterations. The fact that the ellipsoid method - despite its good theoretical behavior - was not a practical algorithm had been noticed before. In fact Dantzig himself had written a computational analysis pointing to the impracticality of the ellipsoid algorithm. He and others had concluded that the ellipsoid method was not a practical algorithm. Khachiyan, as I learned

from him many years later, never had any claims to the practicality of the ellipsoid algorithm. He was only interested in its theoretical implications.

At the time I heard Rosen's talk I did not know him personally. Later he would become my Ph.D thesis advisor in computer science. I wonder if Khachiyan's algorithm had anything to do with that. As I became more familiar with Khachiyan's algorithm I was getting increasingly interested in trying new ideas for solving the linear programming problem. I even implemented some of these trial algorithms. None amounted to anything and I was probably lucky when Rosen had me concentrate on some earthly research problems he had suggested. So I abandoned these futile attempts on linear programming. Being a naive graduate student it did not occur to me that there would be many who would be provoked and inspired by Kahchiyan's algorithm. Nor did it occur to me that there was nothing wrong with searching for yet newer algorithms for linear programming.

When I graduated and became an assistant professor at Rutgers University in the Spring of 1984, I learned of a new polynomial-time algorithm for linear programming by Narendra Karmarkar. It was supposed to be good in theory and good in practice. This was in contrast to Khachiyan's algorithm that was good in theory, but slow in practice. Karmarkar's algorithm was quite amazing, another ingenious and beautiful polynomial-time linear programming algorithm! Karmarkar's algorithm generated a lot of new interest in linear programming and more generally mathematical programming and it resulted in the emergence of an area that has become known as "interior-point methods." I became interested in Karmarkar's algorithm as did many other researchers all over the world. I had some of my own findings. In particular I discovered a close relative of linear programming, called "matrix scaling," as well as some unusual dualities which many years later I called "scaling dualities."

I was fascinated by the matrix scaling problem and the scaling dualities and their connections with linear programming. Even today after many years I still find these connections fascinating and beautiful. An special case of the matrix scaling problem, called "nonnegative matrix scaling" has been studied by a group of researchers outside of the field of linear programming since the 1930's. But no one had noticed the connections to linear programming. I spoke about matrix scaling at a few mathematical programming conferences, but I did not catch anyone's attention. I could not even catch the attention of the non-negative matrix scaling people. It was

very frustrating. I remember when I talked about the relationship between matrix scaling and linear programming to a senior optimization person he said, “That’s not the way to do linear programming.” I was furious, but kept it to myself. I was thinking of Khachiyan’s algorithm and Karmarkar’s algorithm. Only after we have seen these algorithms and understand them do we believe their relevance to linear programming. But would they have started by thinking, “that’s not the way to do linear programming!”

That day in September 1989 as I was driving to Ithaca to give my colloquium talk at Cornell University, I was very excited that I would be meeting Khachiyan. I thought perhaps I could get him interested in the matrix scaling problem. Unfortunately when I arrived in Ithaca I learned that Khachiyan had not yet come. I gave my talk and left the next day. I do recall a private conversation about matrix scaling with James Renegar who had given the first complexity result for a path-following algorithm for linear programming. In order to formulate linear programming into the matrix scaling problem I needed the assumption that the coefficients be rational numbers. Renegar asked what if the coefficients are real numbers. It was a good question and I did not have an answer. But as I learned much later from Chvátal’s online lecture notes on linear programming, Renegar’s concern was not an issue after all, a simple but clever reduction would take care of it (see Appendix 1).

A few months after my trip to Ithaca there was a workshop at Cornell and I learned that Khachiyan had arrived. Mike Grigoridis, a senior optimization faculty at my department, and I attended the workshop and we planned to invite Khachiyan to visit the computer science department at Rutgers. I finally saw Khachiyan at the workshop. I had never seen him before, not even a picture. He was the center of attention and appeared cool and confident. He would easily get into a conversation about any field of optimization. Only after a few minutes of conversation with Khachiyan one would be assured that it was no accident he came up with the ellipsoid algorithm for linear programming. He was sharp as a sword. When I got my chance I introduced myself and had a chat. Then I invited him to visit Rutgers for a talk. He immediately accepted and I was delighted.

When Kahchiyan arrived at Newark Airport to give a talk at Rutgers, I was there to greet him. He was dressed in a trench coat and also was wearing a hat. He looked more like a detective. When we got to the parking ramp he asked if he could smoke a cigarette. I noticed he took

deep puffs. He was methodical in smoking. Actually I liked the fact that he smoked because at the time I was foolish enough to smoke myself. It seemed that he was more professional at it than I was then. Later I learned, to Khachiyan, putting one's feet up and smoking a cigarette was one of the most luxurious things a man could do. At first he appeared to be reserved and I was cautious. As I was driving back from the airport I told him that I could put him up at a comfortable hotel and it would be all paid for by the department. I also told him that I was living in an apartment with a spare bed and he would be welcomed to stay there. He immediately accepted the offer to stay with me. That was the beginning of a new friendship. I liked him very much from the outset. That evening we spoke about a lot of different things, including linear programming of course. I also spoke about the matrix scaling cousin to linear programming, but I didn't want to be pushy. The next day he gave a beautiful talk at Rutgers. I was hearing him lecture for the first time. It was clear that he had not thrown in the towel just because he already had his fame. At his lecture I also heard the first of his so-called two-liner proofs of difficult things that really didn't look like a proof. Perhaps it was clear to him but not to the others. I liked the way he used to pronounce the word "proof." He would say, "perproof."

Grigoriadis and I managed to get Khachiyan interested in Rutgers, and vice versa, and we ended up recruiting him for a visiting position that would later turn into a permanent one. Grigoriadis was heavily involved with the formal details in Khachiyan's recruiting process. He was also the main reason behind my attraction to Rutgers in 1984. During his first year at Rutgers Khachiyan decided to teach a one-semester seminar on the subject of self-concordance theory using the book of Nesterov and Nemirovsky who developed the theory. There were about a dozen or more faculty and students attending his first few lectures. But he managed to give such tough lectures that after the first few most people disappeared. For the most part there were only two students attending, Jeffrey Lagarias and myself. The three of us used to go to dinner after the lectures and would have a good time.

It was during the course of this seminar that I saw Khachiyan a lot and kept talking to him about the matrix scaling problem. I must admit in those early days it was not easy to get Khachiyan interested in the matrix scaling problem. He was teaching Nesterov and Nemirovsky's self-concordance theory which was supposed to be the state-of-the-art in convex programming. Why should the particular matrix scaling problem I was interested in - a special convex programming problem - be interesting anyway? I tried to convince him that the matrix scaling problem and

the scaling dualities were not a by-product of self-concordance theory and these dualities give rise to new algorithms. I tried hard to convince him that he should look into the matrix scaling problem. It took me a while, several attempts for sure, but finally I got through. Ironically I had managed to get Khachiyan interested in the doubly stochastic scaling of non-negative matrices months earlier. But the case of matrix scaling that was related to linear programming took much longer. When he finally did listen to me and realized the relevance of positive semidefinite matrix scaling to linear programming it was like, “Why didn’t you tell me sooner?” Well, I had been trying for a while!

Once Khachiyan grasped the relationship between linear programming and matrix scaling he absolutely became fascinated with it. It was like the cigarette smoking, at times I thought he had become more addicted to matrix scaling than I was. I would go to his apartment to pick him up to go school and right after I would enter he would start talking about matrix scaling. I recall once as he was talking about the matrix scaling problem he offered to make me a cup of coffee. He boiled a pot of water but forgot all about it till all the water had evaporated. He had to boil another pot and this time we decided to stand and watch the pot, but we were still in a heated conversation about matrix scaling. The water came to a boil and as he was speaking I saw him make a cup of instant coffee. It was for me I thought. But to my surprise he began to drink it himself! He had completely forgotten about me! I declared my presence, “What about me Leo?” Only then did he realize I was still waiting for my coffee, and we laughed. Another time he offered to boil a cup of soup but I convinced him that it would be safer if we went out to eat.

Khachiyan and I used to go out a lot for lunch or dinner. We had good times during these periods. I was also learning a lot from him. Khachiyan too always had general questions even about English. For some his English accent wasn’t always easy to understand. In restaurants he used to order Coke or diet Coke. But he would end up repeating the order several times before the waitress would understand. These exchanges between him and waitresses about Coke somehow reminded me of incidents happening to Peter Sellers as Inspector Clouseau in the Pink Panther movies. We used to laugh about these episodes. Once we came up with a brilliant solution to the problem: he would order Pepsi instead. Khachiyan had a good sense of humor and was witty.

Finally, all the time that Khachiyan and I spent talking about the matrix scaling problem paid-

off. We wrote our first joint article, “Diagonal Matrix Scaling and Linear Programming.” We both really liked the article and were excited about it. Neither one of us could have done it on his own alone. I know I could not have done it alone. For me it was a fantasy come true. Here I had written an article with Khachiyan, giving a very simple polynomial-time algorithm for the linear programming problem - the problem that was Khachiyan’s cup of tea and brought him fame in the first place. Not only that, our algorithm also solved the (positive semidefinite) diagonal matrix scaling problem in polynomial-time. It was killing two birds with one stone. As an algorithm for linear programming it was among those having the best known iteration complexity for linear programming. It was just fantastic.

Our algorithm used a path-following strategy, a strategy which was shown to produce an improved iteration complexity over Karmarkar’s algorithm, first by Renegar. We were not the only ones giving an algorithm of the same iteration complexity for linear programming as that of Renegar’s, but we felt that our algorithm was much simpler than all such algorithms and in fact simpler than any other polynomial-time algorithm for linear programming. Its amazing simplicity was due to its novelty in using matrix scaling. Even our path-following strategy was different than Renegar’s or any other path-following algorithm, including those in the book of Nesterov and Nemirovsky on self-concordance theory. While Renegar’s algorithm took over thirty article pages, our algorithm took less than five pages and accomplished more. Yet it is understandable to one familiar with undergraduate linear algebra. We even experimented with a MATLAB implementation of the algorithm. It worked marvelously. Both Khachiyan and I believed that if implemented properly it would even be a practical algorithm. Neither one of us had the interest or the expertise to do serious computational testing.

We submitted our manuscript to the SIAM Journal on Optimization. Ironically, it took more than fifteen months to get this crisp five-page article accepted and published in the journal. A referee, or may be the editor in charge, complained that the article was not written in the standard format. Our algorithm was so simple that Khachiyan didn’t see the need to state it in a formal fashion, i.e. using theorem-lemma format. And Khachiyan didn’t want to change anything because a referee did not like it. I was trying to be as accommodating to him as possible. For example when writing the article he did not agree with my suggestion to make an explicit mention of a separation theorem known as Gordan’s Theorem which I thought was a key ingredient (see Appendix 2). I believe Gordan’s Theorem, which has a very intuitive geometric

interpretation, is the simplest of the separation theorems, yet one of the most profound ones in linear and mathematical programming. Some linear programming books don't even mention the theorem. Indeed Karmarkar's famous algorithm can be shown to be an algorithmic attempt in proving Gordan's Theorem.

It is ironic that our article on linear programming and matrix scaling has received little attention in the mathematical programming literature. As I am sure was the case with Khachiyan, there is no doubt in my mind that matrix scaling is a legitimate relative of linear programming. And as I discovered later, a generalization of matrix scaling is a legitimate relative of convex programming problems. I don't know how Khachiyan felt about our article later on - it would really be immaterial today - but I know that at the time we wrote it he liked it very much. I may very well be biased here but I would like to think that our algorithm is one of the simplest and most elegant polynomial-time algorithms for linear programming. These are all due to the fact that it exploits the connections between linear programming and the new relative, matrix scaling. Khachiyan believed that when you want to look for a good algorithm for a particular problem, you must search outside of the field containing the problem. The ellipsoid algorithm certainly was one such algorithm for linear programming. So was Karmarkar's algorithm. What amazed Khachiyan was that the linear programming problem could be solved by such a simple algorithm coming from the matrix scaling problem. Our matrix scaling algorithm is certainly much simpler than the ellipsoid algorithm for linear programming. I am glad Khachiyan looked into the matrix scaling problem.

I remember when Khachiyan came to Rutgers he was surprised at the quantity of the articles that were written or being written on the subject of linear programming. Many of these articles followed Karmarkar's algorithm. Once he remarked - half seriously, half wittily - "In Russia there may be a dozen people working on linear programming. In US linear programming is a sport." Come to think of it, it was an interesting remark and one can draw parallels between sports and linear programming. The truth was that Khachiyan liked the sport of linear programming and he proved that he still liked to play the game when he got interested in the matrix scaling connection. But one thing is for sure, he was one of the most legendary players of the sport of linear programming. He was a player not interested in the quantity of the games he played, but in their quality. Perhaps Khachiyan was right about linear programming being a sport. Perhaps the reason people did not get interested in the matrix scaling problem is because

they have been busy playing their own sports. But perhaps some day matrix scaling will become a popular sport among other popular sports of mathematical programming.

In addition to our article on linear programming and matrix scaling, Khachiyan and I wrote several other articles, on the subject of nonnegative matrix scaling which has a life of its own and unrelated to linear programming. We did receive an NSF grant for research on the matrix scaling problem. But eventually we had our differences of style and the kind of problems we liked to look at. Khachiyan was strictly interested in algorithms. I liked algorithms, but I also liked generalities. Why does an algorithm work? Where is the limit of applicability of an algorithm? etc. Ironically when I suggested to Khachiyan that we should try to extend our matrix scaling/linear programming algorithm to more general convex optimization problems he said, "Our algorithm probably only works for linear programming." After all we did not agree on the choice of all research problems, besides we had our separate research interests. Finally, we split up. It was good for us to follow our own independent lines of research. It turned out that Khachiyan was wrong about generalization of our algorithm. Well, all legendary players falter some time. This generalization took me a few years of course. I managed to extend our linear programming/matrix scaling algorithm to the semidefinite programming problem and even to general self-concordant convex programming problems. Like its linear programming counterpart, the algorithm for semidefinite programming is very simple, yet novel. The most general algorithm, although makes use of the theory of self-concordance of Nesterov and Nemirovsky - which I learned from Khachiyan when attending his seminar - has many novel features of its own coming from matrix scaling concepts. Despite the relevance of matrix scaling to convex programming I have generally found optimization journals uninterested in the publication of my articles.

Despite the fact that I was unable to receive funding from NSF for my singular research on the subject of matrix scaling problem, and despite the lack of interest of optimization journals, I am very glad to have studied the problem and quite proud of the discoveries. I feel privileged to have had the opportunity to work on some aspects of the matrix scaling problem with the legendary Khachiyan. Getting him interested in matrix scaling and working with him made a world of difference and convinced me further of the worth of the problem. If nothing else I have a personal sense of fulfilment and triumph over some of the mysteries of the matrix scaling problem and its place in convex programming. However, I feel that even my individual research on the extension

of our matrix scaling algorithm to more general convex programming problems was indirectly influenced by Khachiyan (though he didn't believe it could be done). His untimely passing has prompted me to produce a polished and condensed version of these findings ("Matrix scaling dualities in convex programming" referenced in Appendix 2). These generalizations also made me appreciate even more the beauty of our joint linear programming/matrix scaling algorithm. Nothing can ever take away the pleasure and pride of having worked with Khachiyan. No one can erase the memory of the matrix scaling games we played.

Putting aside Khachiyan's brilliance and expertise, probably what made him so unique was having the courage to look into new problems he felt were significant. It did not matter to him how many were working on such problems, or who had worked on them. When he got interested in the matrix scaling problem nothing else mattered. Nothing looked unusual. Solving linear programming using the ellipsoid algorithm must have looked very unusual before he showed us how. That is why nobody bothered to look at linear programming through the ellipsoid method, despite the fact that the work horse of the ellipsoid algorithm was already known prior to Khachiyan's linear programming algorithm. He had the insight to use it in order to solve a problem for which nobody knew of a polynomial-time algorithm. Surely, Khachiyan shall always remain to be among the greatest and most legendary figures in the field of mathematical programming. I was lucky to have known him and to have had the opportunity to collaborate with him. And also to have been a friend.

Appendix 1. (The Relevance of Matrix Scaling in Linear Programming)

Consider the polyhedron

$$P = \{x : Ax \leq b\},$$

where A is an $m \times n$ real matrix and $b \in \mathfrak{R}^m$. Assume A has no row of zeros. Let

$$C = \{(y, s) : A^T y = 0, b^T y + s = 0, (y, s) \geq (0, 0), (y, s) \neq (0, 0)\}.$$

It is well known that linear programming is equivalent to the feasibility problem, i.e. testing if P is nonempty. There is a strong relationship between the feasibility of P and the feasibility of C . The latter problem is referred as the “homogeneous feasibility problem.” If we let

$$B = \begin{pmatrix} A^T & 0 \\ b^T & 1 \end{pmatrix},$$

where 0 is the zero vector in \mathfrak{R}^n , then C can be written as

$$C = \{z \in \mathfrak{R}^{n+1} : Bz = 0, z \geq 0, z \neq 0\}.$$

Gordant’s Theorem says either C is feasible, or the strict linear system $B^T u < 0$, but not both. It follows that $Ax < b$ is feasible if and only if C is infeasible. It is interesting to note that Khachiyan’s algorithm was designed to test the feasibility of the strict system of linear inequalities, $Ax < b$. On the other hand the problem of testing if C is nonempty can be shown to be equivalent to Karmarkar’s canonical formulation of the linear programming problem. Thus Gordant’s Theorem gives a link between Khachiyan’s algorithm and Karmarkar’s algorithm, showing the algorithms are designed for problems which are dual to each other.

The following result, to be referred as “Chvátal’s Lemma,” relates the feasibility of P and C .

Lemma 1. (Chvátal’s Lemma [2]) If C is empty, then P is feasible. Assume (y, s) is in C . If $s > 0$. Then P is empty. If $s = 0$, for each $y_i > 0$ the corresponding constraint in P must be tight at every feasible point of P . This implies that we can eliminate at least one of the variables and one of the constraints in P , and repeat the above process.

In particular, Chvátal’s Lemma implies that, over the reals, linear programming - when formulated as the decision problem of testing if P is nonempty - is reducible to the homogeneous feasibility problem, i.e. testing if C is nonempty. It is well-known that any algorithm for the decision version of the feasibility problem can be used to yield a solution by applying the decision

problem several times (see [1], Problem 16.3). Motivated by Chvátal's Lemma we can prove a stronger result: when C is empty, not only P has a nonempty interior, but that it has interior points of particular form which can be computed directly via the matrix scaling algorithm [7]. Thus while Chvátal's Lemma connects the feasibility of P and C in a decision fashion, Theorem 1 relates them in a direct and constructive manner which in particular reveals the algorithmic significance of the diagonal matrix scaling algorithm.

Theorem 1. (Jin and Kalantari [3]) The following four conditions are equivalent:

- (1) C is empty.
- (2) There exists $w < 0$, and scalar $\delta > 0$ such that

$$AA^T w = b + \delta w^{-1}, \quad w^{-1} = (1/w_1, \dots, 1/w_n)^T.$$

- (3) There exists $w < 0$ such that $x = A^T w$ satisfies $Ax < b$.
- (4) $Ax < b$ is feasible.

The equivalence of (1) and (2) follows from a duality between the homogenous feasibility problem and diagonal scaling of the matrix $B^T B$: C is empty if and only if $ZB^T Bz = e$, for some $z > 0$, where $Z = \text{diag}(z)$, and e is the vector of ones. (2) implies (3), (3) implies (4), and by Gordan's Theorem (4) implies (1).

The algorithmic implication of Theorem 1 is as follows. In order to find an interior point of P , we test if C is empty using the diagonal matrix scaling algorithm in [7]. The algorithm either exclusively computes a point in C (implying $Ax < b$ is empty), or a vector $z > 0$ satisfying $\|ZB^T Bz - e\| < \epsilon$. It suffices to choose $\epsilon = 1$. The corresponding $z > 0$ must satisfy $B^T Bz > 0$. It is easy to see that such z also gives rise to w satisfying $AA^T w < b$, hence an interior point of P . Since linear programming is reducible to testing if P is nonempty, Theorem 1 reveals the intrinsic connection between linear programming and matrix scaling.

Remark. It is well-known that over the rationals the feasibility of $Ax \leq b$ can be replaced with the feasibility of the inflated system of strict inequality $Ax < b'$, where $b'_i = b_i + \epsilon$, for some $\epsilon > 0$ which depends on the *size* of the input. Thus from Theorem 1 it follows that over the rationals linear programming can be tested via a single homogeneous feasibility problem.

Appendix 2. (The Linear Programming-Matrix Scaling Algorithm Revisited) Here I will derive the polynomial-time linear programming/matrix scaling algorithm described in [7], but from a fresh point of view. For the extension of the algorithm to semidefinite programming see [5], and for its extension to much more general convex programming problems see [6]. Consider the problem of computing a nonnegative nontrivial zero of a given positive semidefinite symmetric $n \times n$ matrix, Q . This is equivalent to testing if $\mu = \min\{\phi(x) = \frac{1}{2}x^T Qx : x \geq 0, \|x\| = 1\}$ is zero. Also consider the problem of computing a diagonal matrix D with positive diagonal entries such that $DQDe = e$, where e is the vector of ones. We will consider ϵ -approximate version of these problems. Define ϵ -HP as the problem of computing $d > 0, \|d\| = 1$, such that $\phi(d) \leq \epsilon$, or proving its unsolvability. Define ϵ -ASP as the problem of computing a positive definite diagonal matrix D such that $\|DQDe - e\| \leq \epsilon$, or proving its unsolvability. For a given $d > 0$, let $D = \text{diag}(d)$.

Lemma 1. (Corollary of Gordan's Theorem) If $\mu = 0$, then for all $d > 0$, $\|DQDe - e\| \geq 1$.

Proof. Suppose there exists $d > 0$ such that $\|DQDe - e\| < 1$. Let $y = DQDe$. Then $\sum_i^n |y_i - 1|^2 < 1$. Thus for all i , $|y_i - 1| < 1$. This implies $y_i > 0$. Thus $y = DQDe > 0$. This implies $QDe = Qd > 0$. From Gordan's Theorem the system $Qx = 0, x \geq 0, x \neq 0$ has no solution. Thus, $\mu > 0$, a contradiction. QED

Lemma 2. Let $u \in \Re^n$ be arbitrary. Let γ be a number in $(0, 1]$. Given $t \in (0, 1]$, suppose there exists $d > 0$ such that

$$\|tDQDe + tDu - e\| \leq \frac{1}{2}\gamma \quad (1)$$

Let $\hat{d} = \sqrt{td}$. If

$$\|\hat{D}Q\hat{D}e - e\| \geq \gamma, \quad (2)$$

then

$$\phi\left(\frac{d}{\|d\|}\right) \leq C(\gamma)t, \quad \text{where } C(\gamma) = \frac{\|u\|^2}{\gamma^2}[2n + \gamma(\sqrt{n} + 1)]. \quad (3)$$

Before proving Lemma 2 we explain how the above two lemmas together offer the main strategy for solving the two approximate problems. To solve ϵ -HP when $\mu = 0$, from (3) we can select t such that $C(\gamma)t \leq \epsilon$. Next it suffices to compute d such that (1) is satisfied for that t . Then from Lemma 1 the corresponding \hat{d} must satisfy the inequality (2) and hence $d/\|d\|$ is an ϵ -approximate solution. The actual computation of such d is accomplished via a path-following

Newton iteration scheme to be described in the next two lemmas. To solve ϵ -ASP when $\mu > 0$, since $\phi(d/\|d\|) \geq \mu$, we choose the value of t so that $C(\gamma)t \geq \mu/2$, then from Lemma 2 if we compute a d such that (1) is satisfied, the corresponding \hat{d} will violate inequality (2). Such \hat{d} can be used to compute an ϵ -ASP solution within a few Newton iterations.

Proof. Let $v = tDQDe + tDu - e$. Since $\hat{D} = \sqrt{t}D$, clearly $v = (\hat{D}Q\hat{D}e - e) + \sqrt{t}\hat{D}u$. From Cauchy-Schwarz inequality, and the assumed upper bound on v , we get

$$e^T v = 2\phi(\hat{d}) + \sqrt{t}u^T \hat{d} - n \leq \|e\| \|v\| \leq \frac{\sqrt{n}}{2}\gamma.$$

Since $-\sqrt{t}u^T \hat{d} \leq \sqrt{t}\|u\|\|\hat{d}\|$, we have

$$2\phi(\hat{d}) \leq n + \frac{\sqrt{n}}{2}\gamma + \sqrt{t}\|\hat{d}\| \|u\|. \quad (4)$$

Since

$$\|\sqrt{t}\hat{D}u\| = \|\hat{D}Q\hat{D}e - e - v\| \geq \|\hat{D}Q\hat{D}e - e\| - \|v\| \geq \gamma - \frac{1}{2}\gamma = \frac{1}{2}\gamma,$$

and since $\|\hat{d}\| \|u\| \geq \|\hat{D}\| \|u\| \geq \|\hat{D}u\|$, we get $1/\|\hat{d}\| \leq 2\sqrt{t}\|u\|/\gamma$. Dividing (4) by $\|\hat{d}\|^2$, and using the bound on $1/\|\hat{d}\|$, we get the desired result. QED

For a given $t \in (0, 1]$, let y be the solution to

$$(tQ + D^{-2})y = -(tQd + tu - d^{-1}), \quad d^{-1} = (1/d_1, \dots, 1/d_n).$$

Let $d' = d + y$ and $\lambda = y^T(tQ + D^{-2})y$. Set $z = D^{-1}y$. The point d' is the Newton iterate and can be shown to be the minimizer of the quadratic approximation to the function $t\phi(x) + tu^T x - \sum_{i=1}^n \ln x_i$ at d . It is easy to see that z and λ satisfy

$$(tDQD + I)z = -(tDQDe + tDu - e), \quad \lambda = z^T(tDQD + I)z. \quad (5)$$

Lemma 3. Given $d > 0$, and $t \in (0, 1]$, let d' be the Newton iterate defined above, and $D' = \text{diag}(d')$. Let $Z = \text{diag}(z)$. We have

$$\|tD'QD'e + tD'u - e\| = \|Zz\| \leq \|z\|^2 \leq \lambda^2 \leq \|tDQDe + tDu - e\|^2.$$

Proof. If H is an $n \times n$ symmetric positive definite matrix all of whose eigenvalues are bounded below by one, then it is easy to show for any $w \in \mathfrak{R}^n$, $\|w\|^2 \leq w^T H w \leq \|Hw\|^2$. To prove

the last two inequalities in the lemma we take $H = tDQDe + I$, $w = z$, and use (5). Since $\|Zz\| \leq \|z\|^2$, to complete the proof of lemma it suffices to prove the identity

$$tD'QD' + tD'u - e = -Zz. \quad (6)$$

To prove (6), from the first equation in (5) and regrouping terms we get

$$tDQD(e + z) = e - z - tDu. \quad (7)$$

Since $D(e + z) = d + y = d'$, from the above we get

$$tDQd' = e - z - tDu. \quad (8)$$

Note that $D'D^{-1} = D^{-1}D' = D^{-1}(D + DZ) = I + Z$. Multiplying the left-hand side of (8) by $D'D^{-1}$ we get $tD'QD'e$. Multiplying the right-hand side of (8) by $I + Z$, we get

$$(I + Z)(e - z - tDu) = (I + Z)(e - z) - D'D^{-1}tDu = e - z + z - Zz - tD'u.$$

This proves (6). QED

The next lemma shows how to go from a pair (d, t) satisfying (1) in Lemma 2, to a new pair (d', t') again satisfying (1), but where $t' = r_*t$ with $r_* < 1$, a constant independent of t .

Lemma 4. Fix $\gamma_0 \in (0, 1)$. Given $t \in (0, 1]$, suppose $d > 0$ satisfies $\|tDQDe + tDu - e\| \leq \gamma_0$. Then, $d' = d + y > 0$, and if $t' = tr_*$, where $r_* = (\frac{\sqrt{n} - \gamma_0}{\sqrt{n} - \gamma_0^2})$, then $\|t'D'QD'e + tD'u - e\| \leq \gamma_0$.

Proof. Since $\|tDQDe + tDu - e\| \leq \gamma_0 < 1$, from Lemma 3 we get $\|z\| < 1$. Thus $|z_i| < 1$. Thus $e + z > 0$. Hence

$$d' = d + y = De + Dz = D(e + z) > 0.$$

Also, from this lemma it follows that $\|tD'QD'e + tD'u - e\| \leq \gamma_0^2$. Let $a = D'QD'e + D'u$. For any $\tau \in (0, 1]$, we have

$$\|\tau a - e\| = \|\frac{\tau}{t}ta - \frac{\tau}{t}e + \frac{\tau}{t}e - e\| \leq \frac{\tau}{t}\|ta - e\| + (1 - \frac{\tau}{t})\|e\| \leq \sqrt{n} - \frac{\tau}{t}(\sqrt{n} - \gamma_0^2).$$

Setting the right-hand-side of the above equal to γ_0 , and solving for $\tau \equiv t'$ gives the desired result.

The Algorithm.

Initialization. Set $u = e - Qe$, $t = 1$, $d = e$, r_* as in Lemma 4, and γ_0 a fixed number in $(0, 1)$.

Input $t_* \in (0, 1)$. Input $\epsilon \in (0, 1)$, if ϵ -ASP is the problem of interest.

Phase I. While $t > t_*$, replace (d, t) with (d', t') , where $d' = d + Dz$, z the solution to $(tDQD + I)z = -(tDQDe + tDu - e)$, and $t' = r_*t$.

Phase II. Let $\hat{D} = \sqrt{t}D$. While $\|\hat{D}Q\hat{D}e - e\| > \epsilon$, replace \hat{d} with $\hat{d}' = \hat{d} + \hat{D}\hat{z}$, where \hat{z} is the solution to the linear system $(\hat{D}Q\hat{D} + I)\hat{z} = -(\hat{D}Q\hat{D}e - e)$.

Theorem 1. If $\mu = 0$, the algorithm solves ϵ -HP in $O(\sqrt{n} \ln \frac{n\|u\|}{\epsilon})$ iterations of Phase I. If $\mu > 0$, the algorithm solves ϵ -ASP in $O(\sqrt{n} \ln \frac{n\|u\|}{\mu} + \ln \ln \frac{1}{\epsilon})$ combined iterations of the two phase.

Proof. For a given t_* , let the k -th iterate of Phase I of the algorithm be denoted by (d^k, t_k) . Since $(d^0, t_0) = (e, 1)$ satisfies (1) of Lemma 1, from Lemma 4 so must d^k , for all k . We have $t_k = r_*^k \leq \exp(k(r_* - 1))$. Thus, if the number of iterations of Phase I is k_1 , then $t_* \leq \exp(k_1(r_* - 1))$. This implies $k_1 = O(\sqrt{n} \ln \frac{1}{t_*})$.

If $\mu = 0$, from Lemma 1 and Lemma 2 it follows that $\phi(d^k/\|d^k\|) \leq C(\gamma_0)t_k$. Thus, to solve ϵ -HP it suffices to choose t_* satisfying $C(\gamma_0)t_* = \epsilon$. From this and the definition of $C(\gamma_0)$, we have $\frac{1}{t_*} = O(\frac{n\|u\|^2}{\epsilon})$. Hence the claimed complexity for solving ϵ -HP.

If $\mu > 0$, since $\phi(d^k/\|d^k\|) \geq \mu$, from Lemma 2 it suffices to implement the algorithm with t_* satisfying $C(\frac{1}{2}\epsilon)t_* = \frac{1}{2}\mu$. Then Phase I of the algorithm will terminate with (d, t) such that $d > 0$, $t \leq t_*$, and if $\hat{D} = \sqrt{t}D$, then $\|\hat{D}Q\hat{D}e - e\| \leq \gamma_0$. In this case $\frac{1}{t_*} = O(\frac{n\|u\|^2}{\mu})$. Hence, $k_1 = O(\sqrt{n} \ln \frac{n\|u\|}{\mu})$. Denote the iterates of Phase II by \hat{d}_j , $j = 0, 1, \dots$, and $\hat{d}_0 = \hat{d}$. From Lemma 3, setting $t = 1$, $u = 0$, we have

$$\|\hat{D}_jQ\hat{D}_je - e\| \leq \|\hat{D}_{j-1}Q\hat{D}_{j-1}e - e\|^2 \leq \dots \leq \|\hat{D}_0Q\hat{D}_0e - e\|^{2^j} \leq \gamma_0^{2^j}.$$

To make the upper bound less than ϵ it suffices to iterate Phase II only $O(\ln \ln \frac{1}{\epsilon})$ times. QED

Remark 1. To solve ϵ -HP we simply run the algorithm in the expected number of iteration. If $\mu = 0$, then we will produce the desired approximate solution. Otherwise, μ must be positive. As usual when the entries of Q are rational number, or more generally algebraic numbers, we can choose ϵ so that an approximate solution can be rounded to an exact desired zero of ϕ . If $\mu > 0$, then to solve ϵ -ASP we need to have a lower bound on μ . Such lower bound can easily

be obtained for rational or algebraic inputs (see [4]).

Remark 2. Note that while the algorithm is motivated by the matrix scaling equation $DQDe = e$, its correctness and complexity are not dependent on it. In fact the algorithm can be used to prove the existence of d satisfying the matrix equation.

Theorem 2. $\mu > 0$ if and only if there exists $d > 0$ such that $DQDe = e$.

Proof. From Theorem 1, since $\mu > 0$, there exists a point $d > 0$ such that $\|DQDe - e\| \leq \gamma_0 < 1$. Set $d^0 = d$, and for each $k \geq 0$ define $d^{k+1} = d^k + D^k z^k$ where z^k is the solution to $(D^k Q D^k + I)z^k = -(D^k Q D^k e - e)$. Since in Lemma 3 we can set $u = 0$ and $t = 1$ it follows that $\|D^k Q D^k e - e\|$ converges to zero. To prove the theorem we only need to show that d^k 's form a bounded sequence. This follows since we have

$$\|d^{k+1}\| \leq \|d^k\| + \|D^k\| \|z^k\| \leq \|d^k\|(1 + \|z^k\|).$$

From Lemma 3, $\|z^k\| \leq \|z^{k-1}\|^2$. Thus, $\|d^k\| \leq \|d^0\| \sum_{i=1}^k \|z^0\|^i < \frac{\gamma_0}{1-\gamma_0} \|d_0\| < \infty$. QED

References

- [1] V. Chvátal, *Linear Programming*, W.H. Freeman and Company, New York, 1983.
- [2] V. Chvátal, *Notes on the Khachiyan-Kalantari algorithm*, Rutgers University, New Brunswick, NJ (2002). <http://www.cs.rutgers.edu/~chvatal/521/khaka1.pdf>
- [3] Y. Jin and B. Kalantari, *Chvátal's Lemma and matrix scaling*. Technical Report DCS-TR-581, Department of Computer Science, Rutgers University, New Bruswick, NJ, 2004.
- [4] B. Kalantari and M.R. Eramy-K, *On linear programming and matrix scaling over the algebraic numbers*, *Linear Algebra and its Applications*, 262, 1997, pp.283-306.
- [5] B. Kalantari, *Semidefinite programming and matrix scaling over the semidefinite cone*, *Linear Algebra and its Applications*, 375, 2003, pp. 221-243.
- [6] B. Kalantari, *Matrix scaling dualities and convex programming*. Technical Report, Department of Computer Science, Rutgers University, New Bruswick, NJ, 2005.
<http://www.cs.rutgers.edu/~kalantar/convex2005.pdf>
- [7] L. Khachiyan and B. Kalantari, *Diagonal matrix scaling and linear programming*, *SIAM J. Optim.*, 4 (1992), pp. 668-672.