# Path Finding II : An $\widetilde{O}(m\sqrt{n})$ Algorithm for the Minimum Cost Flow Problem

Yin Tat Lee
MIT
yintat@mit.edu

Aaron Sidford
MIT
sidford@mit.edu

**Abstract**

In this paper we present an $\widetilde{O}(m\sqrt{n}\log^{O(1)} U)$ time algorithm for solving the maximum flow problem on directed graphs with $m$ edges, $n$ vertices, and capacity ratio $U$. This improves upon the previous fastest running time of $O(m\min\{m^{1/2},n^{2/3}\}\log\left(n^2/m\right)\log(U))$ achieved over 15 years ago by Goldberg and Rao [10]. In the special case of solving dense directed unit capacity graphs our algorithm improves upon the previous fastest running times of of $O(m\min\{m^{1/2},n^{2/3}\})$ achieved by Even and Tarjan [7] and Karzanov [15] over 35 years ago and of $\widetilde{O}(m^{10/7})$ achieved recently by Mądry [25].

We achieve these results through the development and application of a new general interior point method that we believe is of independent interest. The number of iterations required by this algorithm is better than that achieved by analyzing the best self-concordant barrier of the feasible region. By applying this method to the linear programming formulations of maximum flow, minimum cost flow, and lossy generalized minimum cost flow analyzed by Daitch and Spielman [5] we achieve a running time of $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$ for these problems as well. Furthermore, our algorithm is parallelizable and using a recent nearly linear work polylogarithmic depth Laplacian system solver of Spielman and Peng [31] we achieve a $\widetilde{O}(\sqrt{n}\log^{O(1)}(U/\epsilon))$ depth and $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$ work algorithm for solving these problems.

## 1 Introduction

The maximum flow problem and its dual, the minimum $s$-$t$ cut problem, are two of the most well studied problems in combinatorial optimization [33]. These problems are key algorithmic primitives used extensively throughout both the theory and practice of computer science [1]. Numerous problems in algorithm design efficiently reduce to the maximum flow problem [2, 34] and techniques developed in the study of this problem have had far reaching implications [3, 2].

Study of the maximum flow problem dates back to 1954 when the problem was first posed by Harris [32]. After decades of work the current fastest running time for solving the maximum flow problem is due to a celebrated result of Goldberg and Rao in 1998 in which they produced a $O(m\min\{m^{1/2},n^{2/3}\}\log(n^2/m)\log(U))$ time algorithm for weighted directed graphs with $n$ vertices, $m$ edges and integer capacities of maximum capacity $U$ [10].[1] While there have been numerous improvements in the running time for solving special cases of this problem (see Section 1.1), the running time for solving the maximum flow problem in full generality has not been improved since 1998.

[1] Throughout this paper we restrict our attention to "weakly" polynomial time algorithms, that is algorithms which may depend polylogarithmically on $U$. The current fastest "strongly polynomial" running time is $O(nm)$ [30].

In this paper we provide an algorithm that solves the maximum flow problem with a running time of $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$,[2] yielding the first improvement to the running time for maximum flow in 15 years and the running time for solving dense unit capacity directed graphs in 35 years. Furthermore, our algorithm is easily parallelizable and using [31], we obtain a $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$ work $\widetilde{O}(\sqrt{n}\log^{O(1)}(U))$ depth algorithm. Using the same technique, we also solve the minimum cost flow problem in time $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$ time and produce $\epsilon$-approximate solutions to the lossy generalized minimum cost flow problem in $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$ time.

We achieve these running times through a novel extension of the work in Part I [22]. In particular, we show how to implement and analyze an algorithm that is essentially "dual" to our approach in [22] and we generalize this algorithm to work for a broader class of barrier functions. This extension is nontrivial as it ultimately yields a path following algorithm that achieves a convergence rate better than that of the best possible self-concordant barrier for feasible region. To the best of the authors' knowledge this is the first interior point method to break this long-standing barrier to the convergence rate of general interior point methods [27]. Furthermore, by applying our algorithm to the linear programming formulations of the maximum flow, minimum cost flow, and lossy generalized minimum cost flow problems analyzed in [5], and by using both the error analysis in [5] and nearly linear time algorithms for solving Laplacian systems [36, 18, 19, 17, 21, 23, 31], we achieve the desired running times.

While our approach is general and the analysis is technical, for the specific case of the maximum flow problem our linear programming algorithm has a slightly more straightforward interpretation. The algorithm simply alternates between re-weighting costs, solving electric flow problems to send more flow, and approximately computing the effective resistance of all edges in the graph to keep the effective resistance of all edges in the graph fairly small and uniform. Hence, by following the path of (almost) least (effective) resistance, we solve the maximum flow problem in $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$.

### 1.1 Previous Work

While the worst case asymptotic running time for solving the maximum flow problem has remained unchanged over the past 15 years, there have been significant breakthroughs on specific instances of the problem, generalizations of the problem, and the technical machinery used to solve the problem. Here we survey some of the key results that we leverage to achieve our running times.

Although the running time for solving general directed instances of maximum flow has remained relatively stagnant until recently [25], there have been significant improvements in the running time for computing maximum flows on undirected graphs over the past few decades. A beautiful line of work on faster algorithms for approximately solving the maximum flow problem on undirected graphs began with a result of Benzcur and Karger in which they showed how to reduce approximately computing minimum cuts in arbitrary undirected graphs to the same problem on sparse graphs, i.e. those with only a nearly linear number of vertices [3]. In later work, Karger also showed how reduce computing approximate maximum flow on dense undirected graphs to computing approximate maximum flows on sparse undirected graphs [13]. Pushing this idea further, in a series of results Karger and Levine showed how to compute the exact maximum flow in an unweighted undirected graph in time $\widetilde{O}(m + nF)$ where $F$ is the maximum flow value of the graph [12].

In 2004 a breakthrough result of Spielman and Teng [36] showed that a particular class of linear systems, Laplacians, can be solved in nearly linear time and Christiano, Kelner, Mądry, and Spielman [4] showed how to use these fast Laplacian system solvers to approximately solve the maximum flow problem on undirected graphs in time $\widetilde{O}(mn^{1/3}\epsilon^{-11/3})$. Later Lee, Rao and

[2] Here and in the remainder of the paper we use $\widetilde{O}(\cdot)$ to hide $\mathsf{polylog}(m)$ factors.

Srivastava [20] showed how to solve the problem in $\widetilde{O}(mn^{1/3}\epsilon^{-2/3})$ for undirected unweighted graphs. This exciting line of work culminated in recent breakthrough results of Sherman [34] and Kelner, Lee, Orecchia and Sidford [16] who showed how to solve the problem in time almost linear in the number of edges in the graph, $\widetilde{O}(m^{1+o(1)}\epsilon^{-2})$, using congestion-approximators, oblivious routings, efficient construction techniques developed by Mądry [24].

In the exact and directed setting, over the past few years significant progress has been made on solving the maximum flow problem and its generalizations using interior point methods, a powerful and general technique for convex optimization [14, 27]. In 2008, Daitch and Spielman [5] showed that, by careful application of interior point techniques, fast Laplacian system solvers [36], and a novel method for solving M-matrices, they could match (up to polylogarithmic factors) the running time of Goldberg Rao and achieve a running time of $\widetilde{O}(m^{3/2}\log^{O(1)}(U))$ not just for maximum flow but also for the minimum cost flow and lossy generalized minimum cost flow problems. Furthermore, very recently Mądry [25] achieved an astounding running time of $\widetilde{O}(m^{10/7})$ for solving the maximum flow problem on un-capacitated directed graphs by a novel application and modification of interior point methods. This shattered numerous barriers providing the first general improvement over the running time of $O(m\min\{m^{1/2},n^{2/3}\})$ for solving unit capacity graphs proven over 35 years ago by Even and Tarjan [7] and Karzanov [15] in 1975.

While our algorithm for solving the maximum flow problem is new, we make extensive use of these breakthroughs on the maximum flow problem. We use sampling techniques first discovered in the context of graph sparsification [35], but not to sparsify a graph but rather to re-weight the graph so that we make progress at a rate commensurate with the number of vertices and not the number of edges. We use fast Laplacian system solvers as in [4, 20], but we use them to make the cost of interior point iterations cheap as in [5, 25]. We then use reductions and error analysis in Daitch and Spielman [5] as well as their solvers for M-matrices to apply our framework to flow problems. Furthermore, as in Mądry we use weights to change the central path (albeit for a slightly different purpose). We believe this further emphasizes the power of these tools as general purpose techniques for algorithm design.

| Year | Author | Running Time |
|---|---|---|
| 1972 | Edmonds and Karp [6] | $\tilde{O}(m^2\log(U))$ |
| 1984 | Tardos [37] | $O(m^4)$ |
| 1984 | Orlin [28] | $\tilde{O}(m^3)$ |
| 1986 | Galil and Tardos [9] | $\tilde{O}\left(mn^2\right)$ |
| 1987 | Goldberg and Tarjan [11] | $\tilde{O}(mn\log(U))$ |
| 1988 | Orlin [29] | $\tilde{O}(m^2)$ |
| 2008 | Daitch and Spielman [5] | $\tilde{O}(m^{3/2}\log^2(U))$ |
| 2013 | This paper | $\tilde{O}(m\sqrt{n}\log^{O(1)}(U))$ |

Figure 1.1: Here we summarize the running times of algorithms for the minimum cost flow problem. $U$ denotes the maximum absolute value of capacities and costs. For simplicity, we only list exact algorithms which yielded polynomial improvements.

## 1.2 Our Approach

Our approach to the maximum flow problem is motivated by our work in Part I [22]. In Part I we provided a new method for solving a general linear program written in the *dual* of standard form

$$\min_{\vec{y}\in\mathbb{R}^n\,:\,\mathbf{A}\vec{y}\geq\vec{c}}\vec{b}^T\vec{y} \tag{1.1}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\vec{b} \in \mathbb{R}^n$, and $\vec{c} \in \mathbb{R}^m$. We showed how to solve (1.1) in $\widetilde{O}(\sqrt{\mathrm{rank}(\mathbf{A})}\log(U/\epsilon))$ iterations while only solving $\widetilde{O}(1)$ linear systems in each iteration.[3] Whereas previous comparable linear program solvers required $\sqrt{\max\{m,n\}}$ iterations when $\mathbf{A}$ was full rank, ours only required $\sqrt{\min\{m,n\}}$ in a fairly general regime.

Unfortunately, this result was insufficient to produce faster algorithms for the maximum flow problem and its generalizations. Given an arbitrary minimum cost maximum flow instance there is a natural linear program that one can use to express the problem:

$$\min_{\substack{\vec{x} \in \mathbb{R}^m \;:\; \mathbf{A}^T \vec{x} = \vec{b} \\ \forall i \in [m] \;:\; l_i \leq x_i \leq u_i}} \vec{c}^T \vec{x} \tag{1.2}$$

where the variables $x_i$ denote the flow on an edge, the $l_i$ and $u_i$ denote lower and upper bounds on how much flow we can put on the edge, and $\mathbf{A}$ is the incidence matrix associated with the graph [5]. In this formulation, $\mathrm{rank}(\mathbf{A})$ is less than the number of vertices in the graph and using fast Laplacian system solvers [36, 18, 19, 17, 21, 23, 31] we can solve linear systems involving $\mathbf{A}$ in time nearly linear in the number of edges in the graph. Thus, if we could perform similar error analysis as in Daitch and Spielman [5] and solve (1.2) in time comparable to that we achieve for solving (1.1) this would immediately yield a $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$ algorithm for the maximum flow problem. Unfortunately, it is not clear how to apply our previous results in this more general setting and naive attempts to write (1.2) in the form of (1.1) without increasing $\mathrm{rank}(\mathbf{A})$ fail.

Even more troubling, achieving a faster than $\widetilde{O}(\sqrt{m}L)$ iterations interior point method for solving general linear programs in this form would break a long-standing barrier for the convergence rate of interior point methods. In a seminal result of Nesterov and Nemirovski [27], they provided a unifying theory for interior point methods and showed that given the ability to construct a *$v$-self concordant barrier* for a convex set, one can minimize linear functions over that convex set with a convergence rate of $O(\sqrt{v})$. Furthermore, they showed how to construct such barriers for a variety of convex sets and thereby achieve fast running times.

To the best of the authors knowledge, there is no general purpose interior point method that achieves a convergence rate faster than the self concordance of the best barrier of the feasible region. Furthermore, using lower bounds results of Nesterov and Nemirovski, it is not hard to see that any general barrier for (1.2) must have self-concordance $\Omega(m)$. To be more precise, note the following result of Nesterov and Nemirovski.

**Theorem 1** ([27, Proposition 2.3.6])**.** *Let $\Omega$ be a convex polytope in $\mathbb{R}^m$. Suppose there is a vertices of the polytope belongs exactly to $k$ linearly independent $(m-1)$-dimensional facets. Then, the self-concordance of any barrier on $\Omega$ is at least $k$.*

Consequently, even if our maximum flow instance just consisted of $O(m)$ edges in parallel Theorem 1 implies that a barrier for the polytope must have self-concordance at least $\Omega(m)$. Note that this does not rule out a different reduction of the problem to minimizing a linear function over a convex body for which there is a $O(n)$ self-concordant barrier. However, it does reflect the difficulty of using standard analysis of interior point methods.

## 1.3 Our Contributions

In this paper we provide an $\tilde{O}(\sqrt{\mathrm{rank}(\mathbf{A})}\log(U/\epsilon))$ iteration algorithm for solving linear programs of the form (1.2). This is the first general interior point method we aware of that converges at a

[3] Throughout this paper we use $U$ to denote the width of a linear program defined in Theorem 18

faster rate than the self-concordance of the best barrier of the feasible region. Each iteration of our algorithm involves solving of $\widetilde{O}(1)$ linear systems of the form $\mathbf{A}^T\mathbf{D}\mathbf{A}\vec{x}=\vec{d}$. By applying this method to the linear program formulation of lossy generalized minimum cost flow analyzed in Daitch and Spielman [5], we achieve a running time of $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$ for solving this problem.

We achieve this running time by a novel extension of the ideas in [22] to work with the primal linear program formulation (1.2) directly. Using an idea from [8], we create a 1-self concordant barrier for each of the $l_i \leq x \leq u_i$ constraints and run a primal path following algorithm with the sum of these barriers. While this would naively yield a $O(\sqrt{m}\log(U/\epsilon))$ iteration method, we show how to use weights in a similar manner as in [22] to improve the convergence rate to $\tilde{O}(\sqrt{\mathrm{rank}(\mathbf{A})}\log(U/\epsilon))$.

While there are similarities between this analysis and the analysis in Part I, we cannot use that result directly. Changing from weighted path following in the dual linear program formulation to this primal formulation changes the behavior of the algorithm and in essentially shifts degeneracies in maintaining weights to degeneracies to maintaining feasibility. This simplifies some parts of the analysis and makes others make some parts of the analysis simpler and some more complicated.

On the positive side, the optimization problem we need to solve to computes the weights becomes better conditioned. Furthermore, inverting the behavior of the weights obviates the need for $r$-steps that were key to our analysis in our previous work.

On the negative side, we have to regularize the weight computation so that weight changes do not undo newton steps on the feasible point and we have to do further work to show that Newton steps on the current feasible point are stable. In the dual formulation it was easy to assert that small Newton steps on the current point do not change the point multiplicatively. However, for this primal analysis this is no longer the case and we need to explicitly bound the size of the Newton step in both the $\ell_\infty$ norm and a weighted $\ell_2$ norm. Hence, we measure the centrality of our points by the size of the Newton step in a mixed norm of the form $\|\cdot\| = \|\cdot\|_\infty + C_{\mathrm{norm}}\|\cdot\|_{\mathbf{W}}$ to keep track of these two quantities simultaneously.

Measuring of Newton step size both with respect to the mixed norm helps to explain how our method outperforms the self-concordance of the best barrier for the space. Self-concordance is based on $\ell_2$ analysis and the lower bounds for self-concordance are precisely the failure of the sphere to approximate a box. While ideally we would just perform optimization over the $\ell_\infty$ box directly, $\ell_\infty$ is ripe with degeneracies that makes this analysis difficult. Nevertheless, unconstrained minimization over a box is quite simple and by working with this mixed norm and choosing weights to improve the conditioning, we are taking advantage of the simplicity of minimizing $\ell_\infty$ over most of the domain and only paying for the $n$-self-concordance of a barrier for the smaller subspace induce by the requirement that $\mathbf{A}^T\vec{x} = \vec{b}$. We hope that this analysis may find further applications.

### 1.4 Paper Organization

The rest of our paper is structured as follows. In Section 2 and Section 3 we cover preliminaries. In Section 4 we introduce our path finding framework and in Section 5 we present the key lemmas used to analyze progress along paths and in Section 6 we introduce the weight function we use to find paths. In Section 7 provide our linear programming algorithm and in Section 8 we discuss the requirements of the linear system solvers we use in the algorithm. In Section 9 we use these results to achieve our desired running times for the maximum flow problem and its generalizations.

Some of the analysis in this paper is similar to our previous work in Part I [22] and when the analysis is nearly the same we often omit details. We encourage the reader to look at Part I [22] for more detailed analysis and longer expositions of the machinery we use in this paper. Note that throughout this paper we make no attempt to reduce polylogarithmic factors in our running times.

## 2 Notation

Here we introduce various notation that we will use throughout the paper. This section should be used primarily for reference as we reintroduce notation as needed later in the paper. (For a summary of linear programming specific notation we use, see Appendix A.)

**Variables:** We use the vector symbol, e.g. $\vec{x}$, to denote a vector and we omit the symbol when we denote the vectors entries, e.g. $\vec{x} = (x_1, x_2, \ldots)$. We use bold, e.g. $\mathbf{A}$, to denote a matrix. For integers $z \in \mathbb{Z}$ we use $[z] \subseteq \mathbb{Z}$ to denote the set of integers from 1 to $z$. We let $\vec{\mathbb{1}}_i$ denote the vector that has value 1 in coordinate $i$ and is 0 elsewhere.

**Vector Operations:** We frequently apply scalar operations to vectors with the interpretation that these operations should be applied coordinate-wise. For example, for vectors $\vec{x}, \vec{y} \in \mathbb{R}^n$ we let $\vec{x}/\vec{y} \in \mathbb{R}^n$ with $[\vec{x}/\vec{y}]_i \stackrel{\mathrm{def}}{=} (x_i/y_i)$ and $\log(\vec{x}) \in \mathbb{R}^n$ with $[\log(\vec{x})]_i = \log(x_i)$ for all $i \in [n]$ .

**Matrix Operations:** We call a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ positive semidefinite (PSD) if $\vec{x}^T \mathbf{A} \vec{x} \geq 0$ for all $\vec{x} \in \mathbb{R}^n$ and we call $\mathbf{A}$ positive definite (PD) if $\vec{x}^T \mathbf{A} \vec{x} > 0$ for all $\vec{x} \in \mathbb{R}^n$. For a positive definite matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ we denote let $\| \cdot \|_{\mathbf{A}} : \mathbb{R}^n \to \mathbb{R}$ denote the norm such that for all $\vec{x} \in \mathbb{R}^n$ we have $\left\|\vec{x}\right\|_{\mathbf{A}} \stackrel{\mathrm{def}}{=} \sqrt{\vec{x}^T \mathbf{A} \vec{x}}$. For symmetric matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ we write $\mathbf{A} \preceq \mathbf{B}$ to indicate that $\mathbf{B} - \mathbf{A}$ is PSD (i.e. $\vec{x}^T \mathbf{A} \vec{x} \leq \vec{x}^T \mathbf{B} \vec{x}$ for all $\vec{x} \in \mathbb{R}^n$) and we write $\mathbf{A} \prec \mathbf{B}$ to indicate that $\mathbf{B} - \mathbf{A}$ is PD (i.e. that $\vec{x}^T \mathbf{A} \vec{x} < \vec{x}^T \mathbf{B} \vec{x}$ for all $\vec{x} \in \mathbb{R}^n$). We define $\succ$ and $\succeq$ analogously. For $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}$, we let $\mathbf{A} \circ \mathbf{B}$ denote the Schur product, i.e. $[\mathbf{A} \circ \mathbf{B}]_{ij} \stackrel{\mathrm{def}}{=} \mathbf{A}_{ij} \cdot \mathbf{B}_{ij}$ for all $i \in [n]$ and $j \in [m]$, and we let $\mathbf{A}^{(2)} \stackrel{\mathrm{def}}{=} \mathbf{A} \circ \mathbf{A}$. We use $\mathrm{nnz}(\mathbf{A})$ to denote the number of nonzero entries in $\mathbf{A}$. For any norm $\| \cdot \|$ and matrix $\mathbf{M}$, the operator norm of $\mathbf{M}$ is defined by $\left\|\mathbf{M}\right\| = \sup_{\|\vec{x}\|=1} \left\|\mathbf{M}\vec{x}\right\|$.

**Diagonal Matrices:** For $\mathbf{A} \in \mathbb{R}^{n \times n}$ we let $\mathrm{diag}(\mathbf{A}) \in \mathbb{R}^n$ denote the vector such that $\mathrm{diag}(\mathbf{A})_i = \mathbf{A}_{ii}$ for all $i \in [n]$. For $\vec{x} \in \mathbb{R}^n$ we let $\mathbf{diag}(\vec{x}) \in \mathbb{R}^{n \times n}$ be the diagonal matrix such that $\mathrm{diag}(\mathbf{diag}(\vec{x})) = \vec{x}$. For $\mathbf{A} \in \mathbb{R}^{n \times n}$ we let $\mathbf{diag}(\mathbf{A})$ be the diagonal matrix such that $\mathrm{diag}(\mathbf{diag}(\mathbf{A})) = \mathrm{diag}(\mathbf{A})$. For $\vec{x} \in \mathbb{R}^n$ when the meaning is clear from context we let $\mathbf{X} \in \mathbb{R}^{n \times n}$ denote $\mathbf{X} \stackrel{\mathrm{def}}{=} \mathbf{diag}(\vec{x})$.

**Multiplicative Approximations:** Frequently in this paper we need to convey that two vectors $\vec{x}$ and $\vec{y}$ are close multiplicatively. We often write $\|\mathbf{X}^{-1}(\vec{y} - \vec{x})\|_\infty \leq \epsilon$ to convey the equivalent facts that $y_i \in [(1-\epsilon)x_i, (1+\epsilon)x_i]$ for all $i$ or $(1-\epsilon)\mathbf{X} \preceq \mathbf{Y} \preceq (1+\epsilon)\mathbf{X}$. At times we find it more convenient to write $\|\log \vec{x} - \log \vec{y}\|_\infty \leq \epsilon$ which is approximately equivalent for small $\epsilon$. In Lemma 37, we bound the quality of this approximation.

**Matrices:** We use $\mathbb{R}^m_{>0}$ to denote the vectors in $\mathbb{R}^m$ where each coordinate is positive and for a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and vector $\vec{x} \in \mathbb{R}^m_{>0}$ we define the following matrices and vectors

- Projection matrix $\mathbf{P}_{\mathbf{A}}(\vec{x}) \in \mathbb{R}^{m \times m}$: $\mathbf{P}_{\mathbf{A}}(\vec{x}) \stackrel{\mathrm{def}}{=} \mathbf{X}^{1/2}\mathbf{A}(\mathbf{A}^T\mathbf{X}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{X}^{1/2}$.
- Leverage scores $\vec{\sigma}_{\mathbf{A}}(\vec{x}) \in \mathbb{R}^m$: $\vec{\sigma}_{\mathbf{A}}(\vec{x}) \stackrel{\mathrm{def}}{=} \mathrm{diag}(\mathbf{P}_{\mathbf{A}}(\vec{x}))$.
- Leverage matrix $\mathbf{\Sigma}_{\mathbf{A}}(\vec{x}) \in \mathbb{R}^{m \times m}$: $\mathbf{\Sigma}_{\mathbf{A}}(\vec{x}) \stackrel{\mathrm{def}}{=} \mathbf{diag}(\mathbf{P}_{\mathbf{A}}(\vec{x}))$.
- Projection Laplacian $\mathbf{\Lambda}_{\mathbf{A}}(\vec{x}) \in \mathbb{R}^{m \times m}$: $\mathbf{\Lambda}_{\mathbf{A}}(\vec{x}) \stackrel{\mathrm{def}}{=} \mathbf{\Sigma}_{\mathbf{A}}(\vec{x}) - \mathbf{P}_{\mathbf{A}}(\vec{x})^{(2)}$.

The definitions of projection matrix and leverage scores are standard when the rows of $\mathbf{A}$ are reweighed by the values in vector $\vec{x}$.

**Convex Sets:** We call a set $U \subseteq \mathbb{R}^k$ *convex* if for all $\vec{x}, \vec{y} \in \mathbb{R}^k$ and all $t \in [0,1]$ it holds that $t \cdot \vec{x} + (1-t) \cdot \vec{y} \in U$. We call $U$ *symmetric* if $\vec{x} \in \mathbb{R}^k \Leftrightarrow -\vec{x} \in \mathbb{R}^k$. For any $\alpha > 0$ and convex set

$U \subseteq \mathbb{R}^k$ we let $\alpha U \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{R}^k | \alpha^{-1}\vec{x} \in U\}$. For any $p \in [1, \infty]$ and $r > 0$ we refer to the symmetric convex set $\{\vec{x} \in \mathbb{R}^k | \|\vec{x}\|_p \leq r\}$ as *the $\ell_p$ ball of radius $r$*.

**Calculus:** For $f : \mathbb{R}^n \to \mathbb{R}$ differentiable at $x \in \mathbb{R}^n$, we let $\nabla f(\vec{x}) \in \mathbb{R}^n$ denote the gradient of $f$ at $\vec{x}$, i.e. $[\nabla f(\vec{x})]_i = \frac{\partial}{\partial x_i} f(\vec{x})$ for all $i \in [n]$. For $f \in \mathbb{R}^n \to \mathbb{R}$ twice differentiable at $x \in \mathbb{R}^n$, we let $\nabla^2 f(\vec{x})$ denote the hessian of $f$ at $x$, i.e. $[\nabla f(\vec{x})]_{ij} = \frac{\partial^2}{\partial x_i \partial x_j} f(\vec{x})$ for all $i, j \in [n]$. Often we will consider functions of two vectors, $g : \mathbb{R}^{n_1 \times n_2} \to \mathbb{R}$, and wish to compute the gradient and Hessian of $g$ restricted to one of the two vectors. For $\vec{x} \in \mathbb{R}^{n_1}$ and $\vec{y} \in \mathbb{R}^{n_2}$ we let $\nabla_{\vec{x}} \vec{g}(\vec{a}, \vec{b}) \in \mathbb{R}^{n_1}$ denote the gradient of $\vec{g}$ for fixed $\vec{y}$ at point $\{\vec{a}, \vec{b}\} \in \mathbb{R}^{n_1 \times n_2}$. We define $\nabla_{\vec{y}}$, $\nabla^2_{\vec{x}\vec{x}}$, and $\nabla^2_{\vec{y}\vec{y}}$ similarly. For $h : \mathbb{R}^n \to \mathbb{R}^m$ differentiable at $\vec{x} \in \mathbb{R}^n$ we let $\mathbf{J}(\vec{h}(\vec{x})) \in \mathbb{R}^{m \times n}$ denote the Jacobian of $\vec{h}$ at $\vec{x}$ where for all $i \in [m]$ and $j \in [n]$ we let $[\mathbf{J}(\vec{h}(\vec{x}))]_{ij} \stackrel{\text{def}}{=} \frac{\partial}{\partial x_j} h(\vec{x})_i$. For functions of multiple vectors we use subscripts, e.g. $\mathbf{J}_{\vec{x}}$, to denote the Jacobian of the function restricted to the $\vec{x}$ variable.

# 3 Preliminaries

## 3.1 The Problem

The central goal of this paper is to efficiently solve the following linear program

$$\begin{array}{rl} \min & \vec{c}^T \vec{x} \\ \vec{x} \in \mathbb{R}^m \ : \ \mathbf{A}^T \vec{x} = \vec{b} & \\ \forall i \in [m] \ : \ l_i \leq x_i \leq u_i & \end{array} \tag{3.1}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\vec{b} \in \mathbb{R}^n$, $\vec{c} \in \mathbb{R}^m$, $l_i \in \mathbb{R} \cup \{-\infty\}$, and $u_i \in \mathbb{R} \cup \{+\infty\}$.[4] We assume that that for all $i \in [m]$ the domain of variable $x_i$, $\text{dom}(x_i) \stackrel{\text{def}}{=} \{x : l_i \leq x \leq u_i\}$, is non-degenerate. In particular we assume that $\text{dom}(x_i)$ is not the empty set, a singleton, or the entire real line, i.e. $l_i < u_i$ and either $l_i \neq -\infty$ or $u_i \neq +\infty$. Furthermore we make the standard assumptions that $\mathbf{A}$ has full column rank, and therefore $m \geq n$, and we assume that the interior of the polytope, $\Omega^0 \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{R}^m \ : \ \mathbf{A}^T \vec{x} = \vec{b}, l_i < x_i < u_i\}$, is non-empty.[5]

The linear program (3.1) is a generalization of standard form, the case where for all $i \in [m]$ we have $l_i = 0$ and $u_i = +\infty$. While it is well known that all linear programs can be written in standard form, the transformations to rewrite (3.1) in standard form may increase the rank of $\mathbf{A}$ and therefore we solve (3.1) directly.

## 3.2 Coordinate Barrier Functions

Rather than working directly with the different domain of the $x_i$ we take a slightly more general approach and for the remainder of the paper assume that for all $i \in [m]$ we have a *barrier function*, $\phi_i : \text{dom}(x_i) \to \mathbb{R}$, such that

$$\lim_{x \to l_i} \phi_i(x) = \lim_{x \to u_i} \phi_i(x) = +\infty.$$

More precisely, we assume that each $\phi_i$ is a *1-self-concordant barrier function.*

[4]Typically (3.1) is written as $\mathbf{A}\vec{x} = \vec{b}$ rather than $\mathbf{A}^T \vec{x} = \vec{b}$. We chose this formulation to be consistent with the dual formulation in [22] and to be consistent with the standard use of $n$ to denote the number of vertices and $m$ to denote the number of edges in a graph in the linear program formulation of flow problems.

[5]For techniques to relax these assumptions see Appendix E of Part I [22].

**Definition 2** (1-Self-Concordant Barrier Function [26])**.** A thrice differentiable real valued barrier function $\phi$ on a convex subset of $\mathbb{R}$ is called a *1-self-concordant barrier function* if

$$\left|\phi'''(x)\right| \leq 2(\phi''(x))^{3/2} \text{ for all } x \in \mathrm{dom}(\phi) \tag{3.2}$$

and

$$\left|\phi'(x)\right| \leq \sqrt{\phi''(x)} \text{ for all } x \in \mathrm{dom}(\phi). \tag{3.3}$$

The first condition (3.2) bounds how quickly the second order approximation to the function can change and the second condition (3.3) bounds how much force the barrier can exert.

The existence of a self-concordant barrier for the domain is a standard assumption for interior point methods [27]. However, for completeness, here we show how for each possible setting of the $l_i$ and $u_i$ there is an explicit 1-self-concordant barrier function we can use:

- *Case (1): $l_i$ finite and $u_i = \infty$*: Here we use a *log barrier* defined as $\phi_i(x) \stackrel{\mathrm{def}}{=} -\log(x - l_i)$. For this barrier we have

$$\phi_i'(x) = -\frac{1}{x - l_i} \quad , \quad \phi_i''(x) = \frac{1}{(x-l_i)^2} \quad , \text{and} \quad \phi_i'''(x) = -\frac{2}{(x-l_i)^3}$$

  and therefore clearly $|\phi_i'''(x)| = 2(\phi_i''(x))^{3/2}$ , $|\phi_i'(x)| = \sqrt{\phi_i''(x)}$, and $\lim_{x \to l_i} \phi_i(x) = \infty$.

- *Case (2): $l_i = -\infty$ and $u_i$ finite*: Here we use a *log barrier* defined as $\phi_i(x) \stackrel{\mathrm{def}}{=} -\log(u_i - x)$. For this barrier we have

$$\phi_i'(x) = \frac{1}{u_i - x} \quad , \quad \phi_i''(x) = \frac{1}{(u_i - x)^2} \quad , \text{and} \quad \phi_i'''(x) = -\frac{2}{(u_i - x)^3}$$

  and therefore clearly $|\phi_i'''(x)| = 2(\phi_i''(x))^{3/2}$, $|\phi_i'(x)| = \sqrt{\phi_i''(x)}$, and $\lim_{x \to u_i} \phi_i(x) = \infty$.

- *Case (3): $l_i$ finite and $u_i$ finite*: Here we use a *trigonometric barrier*[6] defined as $\phi_i(x) \stackrel{\mathrm{def}}{=} -\log\cos(a_i x + b_i)$ for $a_i = \frac{\pi}{u_i - l_i}$ and $b_i = -\frac{\pi}{2}\frac{u_i + l_i}{u_i - l_i}$. Note for this choice as $x \to u_i$ we have $a_i x + b_i \to \frac{\pi}{2}$ and as $x \to l_i$ we have $a_i x + b_i \to \frac{-\pi}{2}$ and in both cases $\phi_i(x) \to \infty$. Furthermore,

$$\phi_i'(x) = a_i \tan(a_i x + b_i) \quad , \quad \phi_i''(x) = \frac{a_i^2}{\cos^2(a_i x + b_i)} \quad , \textit{and} \quad \phi_i''' = \frac{2a_i^3 \sin(a_i x + b_i)}{\cos^3(a_i x + b_i)}.$$

  Therefore, we have

$$\left|\phi_i'''(x)\right| = \left|\frac{2a_i^3 \sin(a_i x + b_i)}{\cos^3(a_i x + b_i)}\right| \leq \frac{2a_i^3}{|\cos(a_i x + b_i)|^3} = 2(\phi''(x))^{3/2}$$

  and $|\phi_i'(x)| \leq \frac{a_i}{|\cos(a_i x + b_i)|} = \sqrt{\phi_i''(x)}$.

For the remainder of this paper we will simply assume that we have a 1 self-concordant barrier $\phi_i$ for each of the $\mathrm{dom}(\phi_i)$ and not use any more structure about the barriers.

While there is much theory regarding properties of self-concordant barrier functions we will primarily use two common properties about self-concordant barriers functions. The first property, Lemma 3, shows that the Hessian of the barrier cannot change to quickly, and the second property, Lemma 4 we use to reason about how the force exerted by the barrier changes over the domain.

[6]The authors are unaware of this barrier being used previously. In [8] they considered a similar setting of 0, 1, or 2 sided constraints in (3.1) however for the finite $l_i$ and $u_i$ case they considered either the the barrier $-\log(u_i - x_i) - \log(x_i - l_i)$, for which the proof of condition (3.3) in Definition 2 is more subtle or the barrier $-\log(\min\{u_i - x, x - l_i\}) + \min\{u_i - x_i, x_i - l_i\}/((u_i - l_i)/2$ which is not thrice differentiable. The "trigonometric barrier" we use arises as the (unique) solution of the ODE $\phi''' = 2\left(\phi''\right)^{3/2}$ such that the function value goes to infinity up at $u_i$ and $l_i$.

**Lemma 3** ([26, Theorem 4.1.6])**.** *Suppose $\phi$ is a 1-self-concordant barrier function. For all $s \in \mathrm{dom}(\phi)$ if $r \stackrel{\mathrm{def}}{=} \sqrt{\phi''(s)}\,|s-t| < 1$ then $t \in \mathrm{dom}(\phi)$ and*

$$(1-r)\sqrt{\phi''(s)} \leq \sqrt{\phi''(t)} \leq \frac{\sqrt{\phi''(s)}}{1-r}.$$

**Lemma 4** ([26, Theorem 4.2.4])**.** *Suppose $\phi$ is a 1-self-concordant barrier function. For all $x, y \in \mathrm{dom}(\phi)$ , we have*

$$\phi'(x) \cdot (y - x) \leq 1.$$

# 4 Weighted Path Finding

In this paper we show how (3.1) can be solved using *weighted path finding.*[7] Our algorithm is essentially "dual" to the algorithm in Part I [22] and our analysis holds in a more general setting. In this section we formally introduce this weighted central path (Section 4.1) and define key properties of the path (Section 4.2) and the weights (Section 4.3) that we will use to produce an efficient path finding scheme.

## 4.1 The Weighted Central Path

Our linear programming algorithm maintains a feasible point $\vec{x} \in \Omega^0$, weights $\vec{w} \in \mathbb{R}^m_{>0}$, and minimizes the following *penalized objective function*

$$\min_{\mathbf{A}^T\vec{x}=\vec{b}} f_t(\vec{x},\vec{w}) \quad \text{where} \quad f_t(\vec{x},\vec{w}) \stackrel{\mathrm{def}}{=} t \cdot \vec{c}^T\vec{x} + \sum_{i\in[m]} w_i \phi_i(\vec{x}_i) \tag{4.1}$$

for increasing $t$ and small $\vec{w}$. For every fixed set of *weights*, $\vec{w} \in \mathbb{R}^m_{>0}$ the set of points $\vec{x}_{\vec{w}}(t) = \arg\min_{\vec{x}\in\Omega^0} f_t(\vec{x},\vec{w})$ for $t \in [0,\infty)$ form a path through the interior of the polytope that we call the *weighted central path.* We call $\vec{x}_{\vec{w}}(0)$ a *weighted center* of the polytope and note that $\lim_{t\to\infty} \vec{x}_{\vec{w}}(0)$ is a solution to the linear program.

While all weighted central paths converge to a solution of the linear program, different paths may have different algebraic properties either increasing or decreasing the difficult of a path following scheme (see Part 1 [22]). Consequently, our algorithm alternates between advancing down a central path (i.e. increasing $t$), moving closer to the weighted central path (i.e. updating $\vec{x}$), and picking a better path (i.e. updating the weights $\vec{w}$).

Ultimately, our weighted path finding algorithm follows a simple iterative scheme. We assume we have a feasible point $\{\vec{x},\vec{w}\} \in \{\Omega^0 \times \mathbb{R}^m_{>0}\}$ and a weight function $\vec{g}(\vec{x}) : \Omega^0 \to \mathbb{R}^m_{>0}$, such that for any point $\vec{x} \in \mathbb{R}^m_{>0}$ the function $\vec{g}(\vec{x})$ returns a good set of weights that suggest a possibly better weighted path. Our algorithm then repeats the following.

1. If $\vec{x}$ close to $\arg\min_{\vec{y}\in\Omega} f_t(\vec{y},\vec{w})$, then increase $t$.

2. Otherwise, use projected Newton step to update $\vec{x}$ and move $\vec{w}$ closer to $\vec{g}(\vec{x})$.

3. Repeat.

In the remainder of this section we present how we measure both the quality of a current feasible point $\{\vec{x},\vec{w}\} \in \{\Omega^0 \times \mathbb{R}^m_{>0}\}$ and the quality of the weight function. In Section 4.2 we derive and present both how we measure how close $\{\vec{x},\vec{w}\}$ is to the weighted central path and the step we take to improve this *centrality.* Then in Section 4.3 we present how we measure the quality of a weight function, i.e. how good the weighted paths it finds are.

[7]See Part I [22] for more motivation regarding weighted paths.

### 4.2 Measuring Centrality.

Here we explain how we measure the distance from $\vec{x}$ to the minimum of $f_t(\vec{x},\vec{w})$ for fixed $\vec{w}$. This distance is a measure of how close $\vec{x}$ is to the weighted central path and we refer to it as the *centrality* of $\vec{x}$, denote $\delta_t(\vec{x},\vec{w})$. Whereas in Part I [22] we simply measured centrality by the size of the Newton step in the Hessian norm, here we use a slightly more complicated definition in order to reason about multiplicative changes in the Hessian (See Section 1.3).

To motivate our centrality measure we first compute a projected Newton step for $\vec{x}$. For all $\vec{x}\in\Omega^0$, we define $\vec{\phi}(\vec{x})\in\mathbb{R}^m$ by $\vec{\phi}(\vec{x})_i=\phi_i(\vec{x}_i)$ for $i\in[m]$. We define $\vec{\phi}'(\vec{x})$, $\vec{\phi}''(\vec{x})$, and $\vec{\phi}'''(\vec{x})$ similarly and let $\mathbf{\Phi},\mathbf{\Phi}',\mathbf{\Phi}'',\mathbf{\Phi}'''$ denote the diagonal matrices corresponding to these matrices. Using this, we have[8]

$$\nabla_x f_t(\vec{x},\vec{w})=t\cdot\vec{c}+\vec{w}\vec{\phi}'(\vec{x})\quad\text{ and }\quad\nabla_{xx}f_t(\vec{x},\vec{w})=\mathbf{W}\mathbf{\Phi}''(\vec{x})\,.$$

Therefore, a Newton step for $\vec{x}$ is given by

$$\begin{aligned}\vec{h}_t(\vec{x},\vec{w})&=-\left(\mathbf{W}\mathbf{\Phi}''(\vec{x})\right)^{-1/2}\mathbf{P}_{\mathbf{A}^T(\mathbf{W}\mathbf{\Phi}''(\vec{x}))^{-1/2}}\left(\mathbf{W}\mathbf{\Phi}''(\vec{x})\right)^{-1/2}\nabla_x f_t(\vec{x},\vec{w})\\&=-\mathbf{\Phi}''(\vec{x})^{-1/2}\mathbf{P}_{\vec{x},\vec{w}}\mathbf{W}^{-1}\mathbf{\Phi}''(\vec{x})^{-1/2}\nabla_x f_t(\vec{x},\vec{w})\end{aligned}\tag{4.2}$$

where $\mathbf{P}_{\mathbf{A}^T(\mathbf{W}\mathbf{\Phi}''(\vec{x}))^{-1/2}}$ is the orthogonal projection onto the kernel of $\mathbf{A}^T(\mathbf{W}\mathbf{\Phi}''(\vec{x}))^{-1/2}$ and $\mathbf{P}_{\vec{x},\vec{w}}$ is the orthogonal projection onto the kernel of $\mathbf{A}^T(\mathbf{\Phi}''(\vec{x}))^{-1/2}$ with respect to the norm $\|\cdot\|_{\mathbf{W}}$, i.e.

$$\mathbf{P}_{\vec{x},\vec{w}}\stackrel{\text{def}}{=}\mathbf{I}-\mathbf{W}^{-1}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\quad\text{ for }\quad\mathbf{A}_x\stackrel{\text{def}}{=}\mathbf{\Phi}''(\vec{x})^{-1/2}\mathbf{A}\,.\tag{4.3}$$

As with standard convergence analysis of Newton's method, we wish to keep the Newton step size in the Hessian norm, i.e. $\|\vec{h}_t(\vec{x},\vec{w})\|_{\mathbf{W}\mathbf{\Phi}''(\vec{x})}=\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\|_{\mathbf{W}}$, small and the multiplicative change in the Hessian, $\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\|_\infty$, small (See Lemma 4). While in the unweighted case we can bound the multiplicative change by the change in the hessian norm (since $\|\cdot\|_\infty\leq\|\cdot\|_2$), here we would like to use small weights and this comparison would be insufficient.

To track both these quantities simultaneously, we define the *mixed norm* for all $\vec{y}\in\mathbb{R}^m$ by

$$\|\vec{y}\|_{\vec{w}+\infty}\stackrel{\text{def}}{=}\|\vec{y}\|_\infty+C_{\text{norm}}\|\vec{y}\|_{\mathbf{W}}\tag{4.4}$$

for some $C_{\text{norm}}>0$ that we define later. Note that $\|\cdot\|_{\vec{w}+\infty}$ is indeed a norm for $\vec{w}\in\mathbb{R}^m_{>0}$ as in this case both $\|\cdot\|_\infty$and $\|\cdot\|_{\mathbf{W}}$ are norms. However, rather than measuring centrality by the quantity $\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\|_{\vec{w}+\infty}=\left\|\mathbf{P}_{\vec{x},\vec{w}}\left(\frac{\nabla_x f_t(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''}}\right)\right\|_{\vec{w}+\infty}$, we instead find it more convenient to use the following idealized form

$$\delta_t(\vec{x},\vec{w})\stackrel{\text{def}}{=}\min_{\vec{\eta}\in\mathbb{R}^n}\left\|\frac{\nabla_x f_t(\vec{x},\vec{w})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}.$$

We justify this definition by showing these two quantities differ by at most a multiplicative factor of $\|\mathbf{P}_{\vec{x},\vec{w}}\|_{\vec{w}+\infty}$ as follows

$$\delta_t(\vec{x},\vec{w})\leq\left\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\right\|_{\vec{w}+\infty}\leq\|\mathbf{P}_{\vec{x},\vec{w}}\|_{\vec{w}+\infty}\cdot\delta_t(\vec{x},\vec{w}).\tag{4.5}$$

This a direct consequence of the more general Lemma 36 that we prove in the appendix.

We summarize this section with the following definition.

[8] Recall that $\vec{w}\vec{\phi}'(\vec{x})$ denotes the entry-wise multiplication of the vectors $\vec{w}$ and $\vec{\phi}'(\vec{x})$.

**Definition 5** (Centrality Measure)**.** For $\{\vec{x},\vec{w}\}\in\{\Omega^{0}\times\mathbb{R}_{>0}^{m}\}$ and $t\geq0$, we let $\vec{h}_{t}(\vec{x},\vec{w})$ denote the *projected newton step* for $\vec{x}$ on the penalized objective $f_{t}$ given by

$$\vec{h}_{t}(\vec{x},\vec{w})\stackrel{\mathrm{def}}{=}-\frac{1}{\sqrt{\vec{\phi}''(\vec{x})}}\mathbf{P}_{\vec{x},\vec{w}}\left(\frac{\nabla_{x}f_{t}(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)$$

where $\mathbf{P}_{\vec{x},\vec{w}}$ is the orthogonal projection onto the kernel of $\mathbf{A}^{T}(\mathbf{\Phi}'')^{-1/2}$ with respect to the norm $\|\cdot\|_{\mathbf{W}}$ (see 4.4). We measure the *centrality* of $\{\vec{x},\vec{w}\}$ by

$$\delta_{t}(\vec{x},\vec{w})\stackrel{\mathrm{def}}{=}\min_{\vec{\eta}\in\mathbb{R}^{n}}\left\Vert \frac{\nabla_{x}f_{t}(\vec{x},\vec{w})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert _{\vec{w}+\infty} \tag{4.6}$$

where for all $\vec{y}\in\mathbb{R}^{m}$ we let $\|\vec{y}\|_{\vec{w}+\infty}\stackrel{\mathrm{def}}{=}\|\vec{y}\|_{\infty}+C_{\mathrm{norm}}\|\vec{y}\|_{\mathbf{W}}$ for some $C_{\mathrm{norm}}>0$ we define later.

### 4.3 The Weight Function

With the Newton step and centrality conditions defined, the specification of our algorithm becomes more clear. Our algorithm is as follows

1. If $\delta_{t}(\vec{x},\vec{w})$ is small, then increase $t$.
2. Set $\vec{x}^{(\mathrm{new})}\leftarrow\vec{x}+\vec{h}_{t}(\vec{x},\vec{w})$ and move $\vec{w}^{(\mathrm{new})}$ towards $\vec{g}(\vec{x}^{(\mathrm{new})})$.
3. Repeat.

To prove this algorithm converges, we need to show what happens to $\delta_{t}\left(\vec{x},\vec{w}\right)$ when we change $t$, $\vec{x}$, $\vec{w}$. At the heart of this paper is understanding what conditions we need to impose on the weight function $\vec{g}(\vec{x}):\Omega^{0}\rightarrow\mathbb{R}_{>0}^{m}$ so that we can bound this change in $\delta_{t}(\vec{x},\vec{w})$ and hence achieve fast converge rates. In Lemma 7 we show that the effect of changing $t$ on $\delta_{t}$ is bounded by $C_{\mathrm{norm}}$ and $\left\Vert \vec{g}(\vec{x})\right\Vert _{1}$, in Lemma 8 we show that the effect that a Newton Step on $\vec{x}$ has on $\delta_{t}$ is bounded by $\left\Vert \mathbf{P}_{\vec{x},\vec{g}(\vec{x})}\right\Vert _{\vec{g}(\vec{x})+\infty}$, and in Lemma 9 and 10 we show the change of $\vec{w}$ as $\vec{g}(\vec{x})$ changes is bounded by $\left\Vert \mathbf{G}(\vec{x})^{-1}\mathbf{G}'(\vec{x})(\mathbf{\Phi}''(\vec{x}))^{-1/2}\right\Vert _{\vec{g}(\vec{x})+\infty}$.

Hence for the remainder of the paper we assume we have a weight function $\vec{g}(\vec{x}):\Omega^{0}\rightarrow\mathbb{R}_{>0}^{m}$ and make the following assumptions regarding our weight function. In Section 6 we prove that such weight function exists.

**Definition 6** (Weight Function)**.** A *weight function* is a differentiable function from $\vec{g}:\Omega^{0}\rightarrow\mathbb{R}_{>0}^{m}$ such that for constants $c_{1}(\vec{g})$, $c_{\gamma}(\vec{g})$, and $c_{\delta}(\vec{g})$, we have the following for all $\vec{x}\in\Omega^{0}$:

- *Size* : The size $c_{1}(\vec{g})=\left\Vert \vec{g}(\vec{x})\right\Vert _{1}$.
- *Slack Sensitivity*: The slack sensitivity $c_{\gamma}(\vec{g})$ satisfies $1\leq c_{\gamma}(\vec{g})\leq\frac{5}{4}$ and $\left\Vert \mathbf{P}_{\vec{x},\vec{w}}\right\Vert _{\vec{w}+\infty}\leq c_{\gamma}(\vec{g})$ for any $\vec{w}$ such that $\frac{4}{5}\vec{g}\left(\vec{x}\right)\leq\vec{w}\leq\frac{5}{4}\vec{g}\left(\vec{x}\right)$.
- *Step Consistency* : The step consistency $c_{\delta}(\vec{g})$ satisfies $c_{\delta}(\vec{g})\cdot c_{\gamma}(\vec{g})<1$ and
$$\left\Vert \mathbf{G}(\vec{x})^{-1}\mathbf{G}'(\vec{x})(\mathbf{\Phi}''(\vec{x}))^{-1/2}\right\Vert _{\vec{g}(\vec{x})+\infty}\leq c_{\delta}\leq1.$$
- *Uniformity* : The weight function satisfies $\left\Vert \vec{g}(\vec{x})\right\Vert _{\infty}\leq2$.

# 5 Progressing Along Weighted Paths

In this section, we provide the main lemmas we need for an $\tilde{O}(\sqrt{\text{rank}(\mathbf{A})}\log(U/\epsilon))$ iterations weighted path following algorithm for (3.1) assuming a weight function satisfying Definition 4.3. In Section 5.1, 5.2, and 5.3 we show how centrality, $\delta_t(\vec{x},\vec{w})$, is affected by changing $t$, $\vec{x}\in\Omega^0$, and $\vec{w}\in\mathbb{R}^m_{>0}$ respectively. In Section 5.4 we then show how to use these Lemmas to improve centrality using approximate computations of the weight function, $\vec{g}:\Omega^0\to\mathbb{R}^m_{>0}$.

## 5.1 Changing $t$

Here we bound how much centrality increases as we increase $t$. We show that this rate of increase is governed by $C_{\text{norm}}$ and $\left\Vert \vec{w}\right\Vert_1$.

**Lemma 7.** *For all $\{\vec{x},\vec{w}\}\in\{\Omega^0\times\mathbb{R}^m_{>0}\}$, $t>0$ and $\alpha\geq 0$, we have*

$$\delta_{(1+\alpha)t}(\vec{x},\vec{w}) \quad\leq\quad (1+\alpha)\delta_t(\vec{x},\vec{w})+\alpha\left(1+C_{norm}\sqrt{\left\Vert \vec{w}\right\Vert_1}\right).$$

*Proof.* Let $\vec{\eta}_t\in\mathbb{R}^n$ be such that

$$\delta_t(\vec{x},\vec{w})=\left\Vert \frac{\nabla_x f_t(\vec{x},\vec{w})+\mathbf{A}\vec{\eta}_t}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}=\left\Vert \frac{t\cdot\vec{c}+\vec{w}\vec{\phi}'(\vec{x})+\mathbf{A}\vec{\eta}_t}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}.$$

Applying this to the definition of $\delta_{(1+\alpha)t}$ and using that $\left\Vert\cdot\right\Vert_{\vec{w}+\infty}$ is a norm then yields

$$\begin{aligned}
\delta_{(1+\alpha)t}(\vec{x},\vec{w}) &= \min_{\vec{\eta}\in\mathbb{R}^n}\left\Vert \frac{(1+\alpha)t\cdot\vec{c}+\vec{w}\vec{\phi}'(\vec{x})+\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}\\
&\leq \left\Vert \frac{(1+\alpha)t\cdot\vec{c}+\vec{w}\vec{\phi}'(\vec{x})+\mathbf{A}(1+\alpha)\vec{\eta}_t}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}\\
&\leq (1+\alpha)\left\Vert \frac{t\cdot\vec{c}+\vec{w}\vec{\phi}'(\vec{x})+\mathbf{A}\vec{\eta}_t}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}+\alpha\left\Vert \frac{\vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\vec{w}+\infty}\\
&= (1+\alpha)\delta_t(\vec{x},\vec{w})+\alpha\left(\left\Vert \frac{\vec{\phi}'(\vec{x})}{\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_\infty+C_{\text{norm}}\left\Vert \frac{\vec{\phi}'(\vec{x})}{\sqrt{\vec{\phi}''(\vec{x})}}\right\Vert_{\mathbf{W}}\right)
\end{aligned}$$

Using that $|\phi_i'(\vec{x})|\leq\sqrt{\phi_i''(\vec{x})}$ for all $i\in[m]$ and $\vec{x}\in\mathbb{R}^m$ by Definition 2 yields the result. $\square$

## 5.2 Changing $\vec{x}$

Here we analyze the effect of a Newton step on $\vec{x}$ on centrality. We show for sufficiently central $\{\vec{x},\vec{w}\}\in\{\Omega^0\times\mathbb{R}^m_{>0}\}$ and $\vec{w}$ sufficiently close to $\vec{g}(\vec{x})$ Newton steps converge quadratically.

**Lemma 8.** *Let $\{\vec{x}_0,\vec{w}\}\in\{\Omega^0\times\mathbb{R}^m_{>0}\}$ such that $\delta_t(\vec{x}_0,\vec{w})\leq\frac{1}{10}$ and $\frac{4}{5}\vec{g}(\vec{x})\leq\vec{w}\leq\frac{5}{4}\vec{g}(\vec{x})$ and consider a Newton step $\vec{x}_1=\vec{x}_0+\vec{h}_t(\vec{x},\vec{w})$. Then, $\delta_t(\vec{x}_1,\vec{w})\leq 4\left(\delta_t(\vec{x}_0,\vec{w})\right)^2$.*

*Proof.* Let $\vec{\phi}_0 \stackrel{\text{def}}{=} \vec{\phi}(\vec{x}_0)$ and let $\vec{\phi}_1 \stackrel{\text{def}}{=} \vec{\phi}(\vec{x}_1)$. By the definition of $\vec{h}_t(\vec{x}_0,\vec{w})$ and the formula of $\mathbf{P}_{\vec{x}_0,\vec{w}}$ we know that there is some $\vec{\eta}_0 \in \mathbb{R}^n$ such that

$$-\sqrt{\vec{\phi}_0''}\vec{h}_t(\vec{x}_0,\vec{w}) = \frac{t\cdot\vec{c}+\vec{w}\vec{\phi}_0' - \mathbf{A}\vec{\eta}_0}{\vec{w}\sqrt{\vec{\phi}_0''}}.$$

Therefore, $\mathbf{A}\vec{\eta}_0 = \vec{c}+\vec{w}\phi_0' + \vec{w}\phi_0'' h_t(\vec{x}_0,\vec{w})$. Recalling the definition of $\delta_t$ this implies that

$$\begin{aligned}
\delta_t(\vec{x}_1,\vec{w}) &= \min_{\vec{\eta}\in\mathbb{R}^n}\left\|\frac{t\cdot\vec{c}+\vec{w}\vec{\phi}_1'-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}_1''}}\right\|_{\vec{w}+\infty} \le \left\|\frac{t\cdot\vec{c}+\vec{w}\vec{\phi}_1'-\mathbf{A}\vec{\eta}_0}{\vec{w}\sqrt{\vec{\phi}_1''}}\right\|_{\vec{w}+\infty}\\
&\le \left\|\frac{\vec{w}(\vec{\phi}_1'-\vec{\phi}_0')-\vec{w}\vec{\phi}_0''\vec{h}_t(\vec{x}_0,\vec{w})}{\vec{w}\sqrt{\vec{\phi}_1''}}\right\|_{\vec{w}+\infty} = \left\|\frac{(\vec{\phi}_1'-\vec{\phi}_0')-\vec{\phi}_0''\vec{h}_t(\vec{x}_0,\vec{w})}{\sqrt{\vec{\phi}_1''}}\right\|_{\vec{w}+\infty}
\end{aligned}$$

By the mean value theorem, we have $\vec{\phi}_1'-\vec{\phi}_0' = \vec{\phi}''(\vec{\theta})\vec{h}_t(\vec{x}_0,\vec{w})$ for some $\vec{\theta}$ between $\vec{x}_0$ and $\vec{x}_1$ coordinate-wise. Hence,

$$\begin{aligned}
\delta_t(\vec{x}_1,\vec{w}) &\le \left\|\frac{\vec{\phi}''(\vec{\theta})\vec{h}_t(\vec{x}_0,\vec{w})-\vec{\phi}_0''\vec{h}_t(\vec{x}_0,\vec{w})}{\sqrt{\vec{\phi}_1''}}\right\|_{\vec{w}+\infty} = \left\|\frac{\left(\vec{\phi}''(\vec{\theta})-\vec{\phi}_0''\right)}{\sqrt{\vec{\phi}_1''}\sqrt{\vec{\phi}_0''}}\left(\sqrt{\phi_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right)\right\|_{\vec{w}+\infty}\\
&\le \left\|\frac{\vec{\phi}''(\vec{\theta})-\vec{\phi}_0''}{\sqrt{\vec{\phi}_1''}\sqrt{\vec{\phi}_0''}}\right\|_{\infty}\cdot\left\|\sqrt{\phi_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right\|_{\vec{w}+\infty}.
\end{aligned}$$

To bound the first term, we use Lemma 3 as follows

$$\begin{aligned}
\left\|\frac{\left(\vec{\phi}''(\vec{\theta})-\vec{\phi}_0''\right)}{\sqrt{\vec{\phi}_1''}\sqrt{\vec{\phi}_0''}}\right\|_{\infty} &\le \left\|\frac{\vec{\phi}''(\vec{\theta})}{\vec{\phi}_0''}-\vec{\mathbb{1}}\right\|_{\infty}\cdot\left\|\frac{\sqrt{\vec{\phi}_0''}}{\sqrt{\vec{\phi}_1''}}\right\|_{\infty}\\
&\le \left|\left(1-\left\|\sqrt{\vec{\phi}_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right\|_{\infty}\right)^{-2}-1\right|\cdot\left(1-\left\|\sqrt{\vec{\phi}_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right\|_{\infty}\right)^{-1}.
\end{aligned}$$

Using (4.5), i.e. Lemma 36, the bound $c_\gamma \le 2$, and the assumption on $\delta_t(\vec{x}_0,\vec{w})$, we have

$$\left\|\sqrt{\vec{\phi}_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right\|_{\infty} \le \left\|\sqrt{\vec{\phi}_0''}\vec{h}_t(\vec{x}_0,\vec{w})\right\|_{\vec{w}+\infty} \le c_\gamma\cdot\delta_t(\vec{x}_0,\vec{w}) \le \frac{1}{5}.$$

Using $\left((1-t)^{-2}-1\right)\cdot(1-t)^{-1} \le 4t$ for $t\le 1/5$, we have

$$\left\|\frac{\left(\vec{\phi}''(\vec{\theta})-\vec{\phi}_0''\right)}{\sqrt{\vec{\phi}_1''}\sqrt{\vec{\phi}_0''}}\right\|_{\infty} \le 4\left\|\sqrt{\vec{\phi}_0''}h_t(\vec{x}_0,\vec{w})\right\|_{\infty}.$$

Combining the above formulas yields that $\delta_t(\vec{x}_1,\vec{w}) \le 4\left(\delta_t(\vec{x}_0,\vec{w})\right)^2$ as desired. $\square$

### 5.3 Changing $\vec{w}$

In the previous subsection we used the assumption that the weights, $\vec{w}$, were multiplicatively close to the output of the weight function, $\vec{g}(\vec{x})$, for the current point $\vec{x}\in\Omega^0$. In order to maintain this invariant when we change $\vec{x}$ we will need to change $\vec{w}$ to move it closer to $\vec{g}(\vec{x})$. Here we bound how much $\vec{g}(\vec{x})$ can move as we move $\vec{x}$ (Lemma 9) and we bound how much changing $\vec{w}$ can hurt centrality (Lemma 10). Together these lemmas will allow us to show that we can keep $\vec{w}$ close to $\vec{g}(\vec{x})$ while still improving centrality (Section 5.4).

**Lemma 9.** *For all $t\in[0,1]$, let $\vec{x}_t \stackrel{\mathrm{def}}{=} \vec{x}_0 + t\vec{\Delta}_x$ for $\vec{\Delta}_x\in\mathbb{R}^m$, $\vec{x}_t\in\Omega^0$, $\vec{g}_t=\vec{g}(\vec{x}_t)$ and $\epsilon = \left\Vert \sqrt{\vec{\phi}_0''}\vec{\Delta}_x\right\Vert_{\vec{g}_0+\infty}\leq 0.1$. Then*

$$\left\Vert \log\left(\vec{g}_1\right)-\log\left(\vec{g}_0\right)\right\Vert_{\vec{g}_0+\infty}\leq c_\delta\epsilon(1+4\epsilon)\leq 0.2$$

*and for all $s,t\in[0,1]$ and for all $\vec{y}\in\mathbb{R}^m$ we have*

$$\left\Vert \vec{y}\right\Vert_{\vec{g}_s+\infty}\leq(1+2\epsilon)\left\Vert \vec{y}\right\Vert_{\vec{g}_t+\infty}. \tag{5.1}$$

*Proof.* Let $\vec{q}:[0,1]\to\mathbb{R}^m$ be given by $\vec{q}(t)\stackrel{\mathrm{def}}{=}\log\left(\vec{g}_t\right)$ for all $t\in[0,1]$. Then, we have

$$\vec{q}'(t)=\mathbf{G}_t^{-1}\mathbf{G}_t'\vec{\Delta}_x.$$

Let $Q(t)\stackrel{\mathrm{def}}{=}\left\Vert \vec{q}(t)-\vec{q}(0)\right\Vert_{\vec{g}_0+\infty}$. Using Jensen's inequality we have that for all $u\in[0,1]$,

$$Q(u)\quad\leq\quad\overline{Q}(u)\stackrel{\mathrm{def}}{=}\int_0^u\left\Vert \mathbf{G}_t^{-1}\mathbf{G}_t'\left(\vec{\phi}_t''\right)^{-1/2}\right\Vert_{\vec{g}_0+\infty}\left\Vert \sqrt{\vec{\phi}_t''}\vec{\Delta}_x\right\Vert_{\vec{g}_0+\infty}dt.$$

Using Lemma 3 and $\epsilon\leq\frac{1}{10}$, we have for all $t\in[0,1]$,

$$\begin{aligned}
\left\Vert \sqrt{\vec{\phi}_t''}\vec{\Delta}_x\right\Vert_{\vec{g}_0+\infty} &\leq \left\Vert \sqrt{\vec{\phi}_t''}/\sqrt{\vec{\phi}_0''}\right\Vert_\infty\left\Vert \sqrt{\vec{\phi}_0''}\vec{\Delta}_x\right\Vert_{\vec{g}_0+\infty}\\
&\leq \left(1-\left\Vert \sqrt{\vec{\phi}_0''}\vec{\Delta}_x\right\Vert_\infty\right)^{-1}\left\Vert \sqrt{\vec{\phi}_0''}\vec{\Delta}_x\right\Vert_{\vec{g}_0+\infty}\\
&\leq \frac{\epsilon}{1-\epsilon}.
\end{aligned}$$

Thus, we have

$$\overline{Q}(u)\quad\leq\quad\frac{\epsilon}{1-\epsilon}\int_0^u\left\Vert \mathbf{G}_t^{-1}\mathbf{G}_t'\left(\vec{\phi}_t''\right)^{-1/2}\right\Vert_{\vec{g}_0+\infty}dt. \tag{5.2}$$

Note that $\overline{Q}$ is monotonically increasing. Let $\theta=\sup_{u\in[0,1]}\left\{\overline{Q}(u)\leq c_\delta\epsilon(1+4\epsilon)\right\}$. Since $\overline{Q}(\theta)\leq\frac{1}{2}$, we know that for all $s,t\in[0,\theta]$, we have

$$\left\Vert \frac{\vec{g}(\vec{x}_s)-\vec{g}(\vec{x}_t)}{\vec{g}(\vec{x}_t)}\right\Vert_\infty\leq\left\Vert \vec{q}(s)-\vec{q}(t)\right\Vert_\infty+\left\Vert \vec{q}(s)-\vec{q}(t)\right\Vert_\infty^2$$

and therefore

$$\left\Vert \vec{g}_s/\vec{g}_t\right\Vert_\infty\leq\left(1+\left\Vert \vec{q}(s)-\vec{q}(t)\right\Vert_\infty+\left\Vert \vec{q}(s)-\vec{q}(t)\right\Vert_\infty^2\right)^2\leq\left(1+c_\delta\epsilon(1+4\epsilon)\right)^2$$

Consequently,

$$\|\vec{y}\|_{\vec{g}_s+\infty} \leq (1+c_\delta\epsilon(1+4\epsilon))\,\|\vec{y}\|_{\vec{g}_t+\infty} \leq (1+2\epsilon)\,\|\vec{y}\|_{\vec{g}_t+\infty}\,.$$

Using (5.2), we have for all $u\in[0,\theta]$,

$$\begin{aligned}
Q(u)\leq\overline{Q}(u) &\leq \frac{\epsilon}{1-\epsilon}\int_0^u\left\|\mathbf{G}_t^{-1}\mathbf{G}_t'\left(\vec{\phi}_t''\right)^{-1/2}\right\|_{\vec{g}_0+\infty}dt\\
&\leq \frac{\epsilon}{1-\epsilon}\int_0^u(1+2\epsilon)\left\|\mathbf{G}_t^{-1}\mathbf{G}_t'\left(\vec{\phi}_t''\right)^{-1/2}\right\|_{\vec{g}_t+\infty}dt\\
&\leq \frac{\epsilon}{1-\epsilon}(1+2\epsilon)\,c_\delta\theta\\
&< c_\delta\epsilon(1+4\epsilon).
\end{aligned}$$

Consequently, we have that $\theta=1$ and we have the desired result with $Q(1)\leq c_\delta\epsilon(1+4\epsilon)<\frac{1}{5}$. $\square$

**Lemma 10.** *Let $\vec{v},\vec{w}\in\mathbb{R}^m_{>0}$ such that $\epsilon=\|\log(\vec{w})-\log(\vec{v})\|_{\vec{w}+\infty}\leq 0.1$. Then for $\vec{x}\in\Omega^0$ we have*

$$\delta_t(\vec{x},\vec{v})\leq(1+4\epsilon)(\delta_t(\vec{x},\vec{w})+\epsilon).$$

*Proof.* Let $\vec{\eta}_w$ be such that

$$\delta_t(\vec{x},\vec{w})=\left\|\frac{\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}_w}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty} \tag{5.3}$$

Furthermore, the assumption shows that $(1+\epsilon)^{-2}\vec{w}_i\leq\vec{v}_i\leq(1+\epsilon)^2\vec{w}_i$ for all $i$. Using these, we bound the energy with the new weights as follows

$$\begin{aligned}
\delta_t(\vec{x},\vec{v})=\min_\eta\left\|\frac{\vec{c}+\vec{v}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{v}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{v}+\infty} &\leq \left\|\frac{\vec{c}+\vec{v}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}_w}{\vec{v}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{v}+\infty}\\
&\leq(1+\epsilon)\left\|\frac{\vec{c}+\vec{v}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}_w}{\vec{v}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\\
&\leq(1+\epsilon)\cdot\left(\left\|\frac{\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}_w}{\vec{v}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}+\left\|\frac{(\vec{v}-\vec{w})\vec{\phi}'(\vec{x})}{\vec{v}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\right)\\
&\leq(1+\epsilon)^3\,\delta_t(\vec{x},\vec{w})+(1+\epsilon)\cdot\left\|\frac{\vec{\phi}'(\vec{x})}{\sqrt{\vec{\phi}''(\vec{x})}}\right\|_\infty\cdot\left\|\frac{(\vec{v}-\vec{w})}{\vec{v}}\right\|_{\vec{w}+\infty}
\end{aligned}$$

Using that $|\phi_i'(\vec{x})|\leq\sqrt{\phi_i''(\vec{x})}$ for all $i\in[m]$ by Definition 2 and using Lemma 37 we have that

$$\begin{aligned}
\delta_t(\vec{x},\vec{v}) &\leq (1+\epsilon)^3\delta_t(\vec{x},\vec{w})+(1+\epsilon)^2\epsilon\\
&\leq (1+4\epsilon)\left(\delta_t(\vec{x},\vec{w})+\epsilon\right).
\end{aligned}$$

$\square$

### 5.4 Centering

In the previous subsection, we saw how much the weight function, $\vec{g}(\vec{x})$, can change after a Newton step on $\vec{x}$ and we bounded how much we can move the weights without affecting centrality too much. Here we study how to correctly move the weights even when we cannot compute the weight function exactly. Our solution is based on "the chasing $\vec{0}$ game" defined in Part I [22]. We restate the main result from Part I [22] on this game and in Theorem 12 show how to use this result to improve centrality and maintain the weights even when we can only compute the weight function approximately.

**Theorem 11** ([22]). *For $\vec{x}_0 \in \mathbb{R}^m$ and $0 < \epsilon < \frac{1}{5}$, consider the two player game consisting of repeating the following for $k = 1, 2, \ldots$*

1. *The adversary chooses $U^{(k)} \subseteq \mathbb{R}^k$, $\vec{u}^{(k)} \in U^{(k)}$, and sets $\vec{y}^{(k)} = \vec{x}^{(k)} + \vec{u}^{(k)}$.*
2. *The adversary chooses $\vec{z}^{(k)}$ such that $\left\|\vec{z}^{(k)} - \vec{y}^{(k)}\right\|_\infty \leq R$*
3. *The adversary reveals $\vec{z}^{(k)}$ and $U^{(k)}$ to the player.*
4. *The player chooses $\vec{\Delta}^{(k)} \in (1+\epsilon)\, U^{(k)}$ and sets $\vec{x}^{(k+1)} = \vec{y}^{(k)} + \vec{\Delta}^{(k)}$.*

*Suppose that each $U^{(k)}$ is a symmetric convex set that contains an $\ell_\infty$ ball of radius $r_k$ and is contained in a $\ell_\infty$ ball of radius $R_k \leq R$ and consider the strategy*

$$\vec{\Delta}^{(k)} = (1+\epsilon) \operatorname*{arg\,min}_{\vec{\Delta} \in U^{(k)}} \left\langle \nabla \Phi_\mu(\vec{z}^{(k)}), \vec{\Delta} \right\rangle$$

*where $\mu = \frac{\epsilon}{12R}$ and $\Phi_\mu(\vec{x}) = \sum_i \left(e^{\mu x_i} + e^{-\mu x_i}\right)$. Let $\tau = \max_k \frac{R_k}{r_k}$ and suppose $\Phi_\mu(\vec{x}^{(0)}) \leq \frac{12 m \tau}{\epsilon}$. This strategy guarantees that for all $k$ we have*

$$\Phi_\mu(\vec{x}^{(k+1)}) \leq \left(1 - \frac{\epsilon^2 r_k}{24R}\right) \Phi_\mu(\vec{x}^{(k)}) + \epsilon m \frac{R_k}{2R} \leq \frac{12 m \tau}{\epsilon}.$$

*In particular, we have $\left\|\vec{x}^{(k)}\right\|_\infty \leq \frac{12R}{\epsilon} \log\left(\frac{12 m \tau}{\epsilon}\right)$.*

We can think updating weight is playing this game, we want to make sure the error between $\vec{w}$ and $\vec{g}(\vec{x})$ is close to 0 while the adversary control the next point $\vec{g}(\vec{x})$ and the noise in the approximate $\vec{g}(\vec{x})$. Theorem 11 shows that we can control the error to be small in $\ell_\infty$ if we can approximate $\vec{g}(\vec{x})$ with small $\ell_\infty$ error.

Formally, we will measure its distance from the optimal weights in log scale by

$$\vec{\Psi}(\vec{x}, \vec{w}) \stackrel{\text{def}}{=} \log(\vec{g}(\vec{x})) - \log(\vec{w}). \tag{5.4}$$

Our goal will be to keep $\left\|\vec{\Psi}(\vec{x}, \vec{w})\right\|_{\vec{w}+\infty} \leq K$ for some error $K$ that is just small enough to not impair our ability to decrease $\delta_t$ linearly and not to impair our ability to approximate $\vec{g}$. We will attempt to do this without moving $\vec{w}$ too much in $\|\cdot\|_{\vec{w}+\infty}$.

| $(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) = \texttt{centeringInexact}(\vec{x}, \vec{w}, K)$ |
|---|
| 1. $c_k = \frac{1}{1 - c_\delta c_\gamma}, R = \frac{K}{48 c_k \log(400m)}$, $\delta_t = \delta_t(\vec{x}, \vec{w})$ and $\epsilon = \frac{1}{2c_k}$. |
| 2. $\vec{x}^{(\text{new})} = \vec{x} - \frac{1}{\sqrt{\vec{\phi}''(\vec{x})}} \mathbf{P}_{\vec{x},\vec{w}} \left(\frac{t\vec{c} - \vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)$. |
| 3. Let $U = \{\vec{x} \in \mathbb{R}^m \mid \|\vec{x}\|_{\vec{w}+\infty} \leq \left(1 - \frac{7}{8c_k}\right) \delta_t\}$ |
| 4. Find $\vec{z}$ such that $\left\|\vec{z} - \log(\vec{g}(\vec{x}^{(\text{new})}))\right\|_\infty \leq R$. |
| 5. $\vec{w}^{(\text{new})} = \exp\left(\log(\vec{w}) + (1+\epsilon) \arg\min_{\vec{u} \in U} \left\langle \nabla \Phi_{\frac{\epsilon}{12R}}(\vec{z} - \log(\vec{w})), \vec{u} \right\rangle\right)$ |

The minimization problem in step 5 is simply a projection onto the convex set $U$ and it can be done in $\tilde{O}(1)$ depth and $\tilde{O}(m)$ work. See section B.2 for details.

**Theorem 12.** *Assume that* $24m^{1/4} \geq c_k \stackrel{\text{def}}{=} \frac{1}{1-c_\delta c_\gamma} \geq 5$, $1 \leq C_{norm} \leq 2c_k$ *and* $K \leq \frac{1}{20c_k}$. *Let* $\vec{\Psi}(\vec{x},\vec{w}) \stackrel{\text{def}}{=} \log(\vec{g}(\vec{x})) - \log(\vec{w})$. *Suppose that*

$$\delta \stackrel{\text{def}}{=} \delta_t(\vec{x},\vec{w}) \leq \frac{K}{48c_k \log(400m)} \quad \text{and} \quad \Phi_\mu(\vec{\Psi}(\vec{x},\vec{w})) \leq (400m)^2$$

*where* $\mu = \frac{\epsilon}{12R} = 2\log(400m)/K$. *Let* $(\vec{x}^{(new)}, \vec{w}^{(new)}) = \texttt{centeringInexact}(\vec{x},\vec{w},K)$, *then*

$$\delta_t(\vec{x}^{(new)}, \vec{w}^{(new)}) \leq \left(1 - \frac{1}{4c_k}\right)\delta \quad \text{and} \quad \Phi_\mu(\vec{\Psi}(\vec{x}^{(new)}, \vec{w}^{(new)})) \leq (400m)^2 .$$

*Also, we have* $\left\| \log(\vec{g}(\vec{x}^{(new)})) - \log(\vec{w}) \right\|_\infty \leq K$.

*Proof.* By Lemma 9, inequality (4.5), $c_\delta c_\gamma \leq 1$ and $c_\gamma \leq \frac{5}{4}$ (see Def 6), we have

$$\begin{aligned} \left\| \log\left(\vec{g}(\vec{x}^{(\text{new})})\right) - \log\left(\vec{g}(\vec{x})\right) \right\|_{\vec{g}(\vec{x})+\infty} &\leq c_\delta c_\gamma \delta (1 + 4c_\gamma \delta) \\ &\leq c_\delta c_\gamma \delta + 5\delta^2 \\ &\leq \left(1 - \frac{15}{16c_k}\right)\delta. \end{aligned}$$

Using $K \leq \frac{1}{20c_k}$, we have

$$\left\| \frac{\vec{w} - \vec{g}(\vec{x})}{\vec{g}(\vec{x})} \right\|_\infty \leq \left\| \log(\vec{w}) - \log(\vec{g}(\vec{x})) \right\|_\infty + \left\| \log(\vec{w}) - \log(\vec{g}(\vec{x})) \right\|_\infty^2 \leq \frac{21}{20}K \leq \frac{1}{16c_k}.$$

Hence, we have

$$\begin{aligned} \left\| \log\left(\vec{g}(\vec{x}^{(\text{new})})\right) - \log\left(\vec{g}(\vec{x})\right) \right\|_{\vec{w}+\infty} &\leq \left(1 + \frac{1}{16c_k}\right)\left(1 - \frac{15}{16c_k}\right)\delta \\ &\leq \left(1 - \frac{7}{8c_k}\right)\delta. \end{aligned}$$

Therefore, we know that for the Newton step, we have $\vec{\Psi}(\vec{x}^{(\text{new})}, \vec{w}) - \vec{\Psi}(\vec{x},\vec{w}) \in U$ where $U$ is the symmetric convex set given by

$$U \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{R}^n \mid \|\vec{x}\|_{\vec{w}+\infty} \leq C\}$$

where $C = \left(1 - \frac{7}{8c_k}\right)\delta$. Note that from our assumption on $\delta$, we have

$$C \leq \delta \leq \frac{K}{48c_k \log(400m)} = R.$$

It ensures that $U$ are contained in some $\ell_\infty$ ball of radius $R$. Therefore, we can play the chasing 0 game on $\vec{\Psi}(\vec{x},\vec{w})$ attempting to maintain the invariant that $\left\|\vec{\Psi}(\vec{x},\vec{w})\right\|_\infty \leq K$ without taking steps that are more than $1+\epsilon$ times the size of $U$ where we pick $\epsilon = \frac{1}{2c_k}$ so to not interfere with our ability to decrease $\delta_t$ linearly.

However, to do this with the chasing 0 game, we need to ensure that $R$ satisfying the following

$$\frac{12R}{\epsilon}\log\left(\frac{12m\tau}{\epsilon}\right)\leq K$$

where here $\tau$ is as defined in Theorem 11.

To bound $\tau$, we need to lower bound the radius of $\ell_\infty$ ball it contains. Since by assumption $\left\|\vec{g}(\vec{x})\right\|_\infty\leq 2$ and $\left\|\vec{\Psi}(\vec{x},\vec{w})\right\|_\infty\leq\frac{1}{8}$, we have that $\left\|\vec{w}\right\|_\infty\leq 3$. Hence, we have

$$\forall u\in\mathbb{R}^m\quad:\quad\left\|\vec{u}\right\|_\infty^2\geq\frac{1}{3m}\left\|\vec{u}\right\|_{\vec{w}}^2.$$

Consequently, if $\left\|\vec{u}\right\|_\infty\leq\frac{\delta}{5C_{\text{norm}}\sqrt{m}}$, then $\vec{u}\in U$. So, we have that $U$ contains a box of radius $\frac{\delta}{5C_{\text{norm}}\sqrt{m}}$ and since $U$ is contained in a box of radius $\delta$, we have that

$$\tau\quad\leq\quad 5C_{\text{norm}}\sqrt{m}\leq 10c_k\sqrt{m}.$$

Using $c_k\leq 24m^{1/4}$, we have

$$\begin{aligned}\frac{12R}{\epsilon}\log\left(\frac{12m\tau}{\epsilon}\right)&\leq 24c_kR\log\left(240m^{3/2}c_k^2\right)\\&\leq 48c_kR\log\left(400m\right)=K.\end{aligned}$$

and

$$\frac{12m\tau}{\epsilon}\leq 240m^{3/2}c_k^2\leq(400m)^2.$$

This proves that we meet the conditions of Theorem 11. Consequently, $\left\|\vec{\Psi}(\vec{x}^{(\text{new})},\vec{w}^{(\text{new})})\right\|_\infty\leq K$ and $\Phi_\alpha(\vec{\Psi}(\vec{x}^{(\text{new})},\vec{w}^{(\text{new})}))\leq(400m)^2$.

Since $K\leq\frac{1}{4}$, Lemma 8 shows that

$$\delta_t(\vec{x}^{(\text{new})},\vec{w})\quad\leq\quad 4\left(\delta_t(\vec{x},\vec{w})\right)^2.$$

The step 5 shows that

$$\begin{aligned}\left\|\log(\vec{w})-\log(\vec{w}^{(\text{new})})\right\|_{\vec{w}+\infty}&\leq\left(1+\frac{1}{2c_k}\right)\left(1-\frac{7}{8c_k}\right)\delta\\&\leq\left(1-\frac{3}{8c_k}\right)\delta.\end{aligned}$$

Using $\delta\leq\frac{1}{80c_k}$, the Lemma 10 shows that

$$\begin{aligned}\delta_t(\vec{x}^{(\text{new})},\vec{w}^{(\text{new})})&\leq\left(1+4\left(1-\frac{3}{8c_k}\right)\delta\right)\left(\delta_t(\vec{x}^{(\text{new})},\vec{w})+\left(1-\frac{3}{8c_k}\right)\delta\right)\\&\leq(1+4\delta)\left(4\delta^2+\left(1-\frac{3}{8c_k}\right)\delta\right)\\&\leq\left(1-\frac{3}{8c_k}\right)\delta+4\delta^2+16\delta^3+4\delta^2\\&\leq\left(1-\frac{1}{4c_k}\right)\delta.\end{aligned}$$

$\square$

## 6 Weight Function

In this section we present the weight function that we use to achieve our $\widetilde{O}(\sqrt{\text{rank}(\mathbf{A})}\log(U/\epsilon))$ iteration linear program solver. This weight function is similar to the one we used in Part I [22]. Due to subtle differences in the analysis we provide many of proofs of properties of the weight function in full. For further intuition on the weight function or proof details see Part I [22].

We define the weight function $\vec{g}:\Omega^0\rightarrow\mathbb{R}^m_{>0}$ for all $\vec{x}\in\mathbb{R}^m_{>0}$ as follows

$$\vec{g}(\vec{x})\stackrel{\text{def}}{=}\arg\min_{\vec{w}\in\mathbb{R}^m_{>0}}\hat{f}(\vec{x},\vec{w})\quad\text{where}\quad\hat{f}(\vec{x},\vec{w})\stackrel{\text{def}}{=}\vec{\mathbb{1}}^T\vec{w}+\frac{1}{\alpha}\log\det\left(\mathbf{A}_x^T\mathbf{W}^{-\alpha}\mathbf{A}_x\right)-\beta\sum_{i\in[m]}\log w_i. \tag{6.1}$$

where here and in the remainder of the subsection we let $\mathbf{A}_x\stackrel{\text{def}}{=}(\mathbf{\Phi}''(\vec{x}))^{-1/2}\mathbf{A}$ and the parameters $\alpha,\beta$ are chosen later such that the following hold

$$\alpha\in[1,2)\,,\ \beta\in(0,1)\,,\text{ and }\ \beta^{1-\alpha}\leq 2\,. \tag{6.2}$$

Here we choose $\beta$ small and $\alpha$ just slightly larger than 1.[9]

We start by computing the gradient and Hessian of $\hat{f}(\vec{x},\vec{w})$ with respect to $\vec{w}$.

**Lemma 13.** *For all $\vec{x}\in\Omega^0$ and $\vec{w}\in\mathbb{R}^m_{>0}$, we have*

$$\nabla_w\hat{f}(\vec{x},\vec{w})=\left(\mathbf{I}-\mathbf{\Sigma}\mathbf{W}^{-1}-\beta\mathbf{W}^{-1}\right)\vec{\mathbb{1}}\quad\text{and}\quad\nabla^2_{ww}\hat{f}(\vec{x},\vec{w})=\mathbf{W}^{-1}\left(\mathbf{\Sigma}+\beta\mathbf{I}+\alpha\mathbf{\Lambda}\right)\mathbf{W}^{-1}$$

*where $\mathbf{\Sigma}\stackrel{\text{def}}{=}\mathbf{\Sigma}_{\mathbf{A}_x}(\vec{w}^{-\alpha})$ and $\mathbf{\Lambda}\stackrel{\text{def}}{=}\mathbf{\Lambda}_{\mathbf{A}_x}(\vec{w}^{-\alpha})$.*

*Proof.* Using Lemma 38 and the chain rule we compute the gradient of $\nabla_w\hat{f}(\vec{x},\vec{w})$ as follows

$$\begin{aligned}\nabla_w\hat{f}(\vec{x},\vec{w}) &= \vec{\mathbb{1}}+\frac{1}{\alpha}\mathbf{\Sigma}\mathbf{W}^{\alpha}\left(-\alpha\mathbf{W}^{-\alpha-1}\right)-\beta\mathbf{W}^{-1}\vec{\mathbb{1}}\\ &= \left(\mathbf{I}-\mathbf{\Sigma}\mathbf{W}^{-1}-\beta\mathbf{W}^{-1}\right)\vec{\mathbb{1}}.\end{aligned}$$

Next, using Lemma 38 and chain rule, we compute the following for all $i,j\in[m]$

$$\begin{aligned}\frac{\partial(\nabla_w\hat{f}(\vec{x},\vec{w}))_i}{\partial\vec{w}_j} &= -\frac{\vec{w}_i\mathbf{\Lambda}_{ij}\vec{w}_j^{\alpha}\left(-\alpha\vec{w}_j^{-\alpha-1}\right)-\mathbf{\Sigma}_{ij}\vec{\mathbb{1}}_{i=j}}{\vec{w}_i^2}+\beta\vec{\mathbb{1}}_{i=j}\left\{\vec{w}_i^{-2}\right\}\\ &= \frac{\mathbf{\Sigma}_{ij}}{\vec{w}_i\vec{w}_j}+\alpha\frac{\mathbf{\Lambda}_{ij}}{\vec{w}_i\vec{w}_j}+\frac{\beta\vec{\mathbb{1}}_{i=j}}{\vec{w}_i^2}\quad. \qquad\text{(Using that }\mathbf{\Sigma}\text{is diagonal)}\end{aligned}$$

Consequently, $\nabla^2_{ww}\hat{f}(\vec{x},\vec{w})=\mathbf{W}^{-1}\left(\mathbf{\Sigma}+\beta\mathbf{I}+\alpha\mathbf{\Lambda}\right)\mathbf{W}^{-1}$ as desired. $\square$

**Lemma 14.** *For all $\vec{x}\in\Omega^0$, the weight function $\vec{g}(\vec{x})$ is a well defined with*

$$\beta\leq g_i(\vec{s})\leq 1+\beta\quad\text{and}\quad\left\Vert\vec{g}(\vec{x})\right\Vert_1=\text{rank}(\mathbf{A})+\beta\cdot m.$$

*Furthermore, for all $\vec{x}\in\Omega^0$, the weight function obeys the following equations*

$$\mathbf{G}(\vec{x})=\left(\mathbf{\Sigma}+\beta\mathbf{I}\right)\vec{\mathbb{1}}\quad,\text{ and}\quad\mathbf{G}'(\vec{x})=-\mathbf{G}(\vec{x})\left(\mathbf{G}(\vec{x})+\alpha\mathbf{\Lambda}\right)^{-1}\mathbf{\Lambda}\left(\mathbf{\Phi}''(\vec{x})\right)^{-1}\mathbf{\Phi}'''(\vec{x})$$

*where $\mathbf{\Sigma}\stackrel{\text{def}}{=}\mathbf{\Sigma}_{\mathbf{A}_x}(\vec{g}^{-\alpha}(\vec{x}))$, $\mathbf{\Lambda}\stackrel{\text{def}}{=}\mathbf{\Lambda}_{\mathbf{A}_x}(\vec{g}^{-\alpha}(\vec{x}))$, and $\mathbf{G}'(\vec{x})$ is the Jacobian matrix of $\vec{g}$ at $\vec{x}$.*

[9] Note that this formula is different than the formula we used in [22].

*Proof.* By Lemma 38 we have that $\mathbf{\Sigma} \succeq \mathbf{\Lambda} \succeq \mathbf{0}$. Therefore, by Lemma 13, we have that $\nabla^2_{ww}\hat{f}(\vec{x},\vec{w}) \succeq \beta\mathbf{W}^{-2}$ and $\hat{f}(\vec{x},\vec{w})$ is convex. Using the formula for the gradient in Lemma 13, we see that that for all $i \in [m]$ it is the case that

$$\left[\nabla_w \hat{f}(\vec{x},\vec{w})\right]_i = \frac{1}{w_i}\left(w_i - \mathbf{\Sigma}_{ii} - \beta\right).$$

Using that $0 \le \sigma_i \le 1$ for all $i$ by Lemma 38 and $\beta \in (0,1)$ by (6.2), we see that if $\vec{w}_i \in (0,\beta)$ then $\left[\nabla_w \hat{f}(\vec{x},\vec{w})\right]_i$ is strictly negative and if $\vec{w}_i \in (1+\beta,\infty)$ then $\left[\nabla_w \hat{f}(\vec{x},\vec{w})\right]_i$ is strictly positive. Therefore, for any $\vec{x} \in \Omega^0$, the $\vec{w}$ that minimizes this convex function $\hat{f}(\vec{x},\vec{w})$ lies between the box between $\beta$ to $1+\beta$. Since $\hat{f}$ is strongly convex in this region, the minimizer is unique.

The formula for $\mathbf{G}(\vec{x})$ follows by setting $\nabla_w \hat{f}(\vec{x},\vec{w}) = \vec{0}$ and the size of $g(\vec{x})$ follows from the fact that $\left\|\vec{\sigma}\right\|_1 = \mathrm{tr}\left(\mathbf{P}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)\right)$. Since $\mathbf{P}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)$ is a projection onto the image of $\mathbf{G}(\vec{x})^{-\alpha/2}\mathbf{A}_x$ and since $\vec{g}(\vec{x}) > \vec{0}$ and $\vec{\phi}''(\vec{x}) > \vec{0}$, we have that the dimension of the image of $\mathbf{G}(\vec{x})^{-\alpha/2}\mathbf{A}_x$ is the rank of $\mathbf{A}$. Hence, we have that $\left\|\vec{g}(\vec{x})\right\|_1 = \mathrm{rank}(\mathbf{A}) + \beta \cdot m$.

By Lemma 38 and chain rule, we get the following for all $i,j \in [m]$

$$\frac{\partial(\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w}))_i}{\partial \vec{x}_j} = -\vec{w}_i^{-1}\mathbf{\Lambda}_{ij}\vec{\phi}''_j(\vec{x})\left(-(\vec{\phi}''_j(\vec{x}))^{-2}\vec{\phi}'''_j(\vec{x})\right) = \vec{w}_i^{-1}\mathbf{\Lambda}_{ij}(\vec{\phi}''_j(\vec{x}))^{-1}\vec{\phi}'''_j(\vec{x}).$$

Consequently, $\mathbf{J}_{\vec{x}}(\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w})) = \mathbf{W}^{-1}\mathbf{\Lambda}\left(\mathbf{\Phi}''(\vec{x})\right)^{-1}\mathbf{\Phi}'''(\vec{x})$ where $\mathbf{J}_{\vec{x}}$ denotes the Jacobian matrix of the function $\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w})$ with respect to $\vec{x}$. Since we have already know that $\mathbf{J}_{\vec{w}}(\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w})) = \nabla^2_{\vec{w}\vec{w}}f_t(\vec{x},\vec{w}) = \mathbf{W}^{-1}\left(\mathbf{\Sigma} + \beta\mathbf{I} + \alpha\mathbf{\Lambda}\right)\mathbf{W}^{-1}$ is positive definite (and hence invertible), by applying the implicit function theorem to the specification of $\vec{g}(\vec{x})$ as the solution to $\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w}) = \vec{0}$, we have

$$\mathbf{G}'(\vec{x}) = -\left(\mathbf{J}_{\vec{w}}(\nabla_w \hat{f}(\vec{x},\vec{w}))\right)^{-1}\left(\mathbf{J}_{\vec{x}}(\nabla_{\vec{w}}\hat{f}(\vec{x},\vec{w}))\right) = -\mathbf{G}(\vec{x})\left(\mathbf{G}(\vec{x}) + \alpha\mathbf{\Lambda}\right)^{-1}\mathbf{\Lambda}\left(\mathbf{\Phi}''(\vec{x})\right)^{-1}\mathbf{\Phi}'''(\vec{x}).$$

$\square$

Now we show the step consistency of $\vec{g}$.

**Lemma 15** (Step Consistency)**.** *For all $\vec{x} \in \Omega^0$ and $\vec{y} \in \mathbb{R}^m$, and*

$$\mathbf{B} \stackrel{\text{def}}{=} \mathbf{G}(\vec{x})^{-1}\mathbf{G}'(\vec{x})(\vec{\phi}(\vec{x})'')^{-1/2},$$

*we have*

$$\left\|\mathbf{B}\vec{y}\right\|_{\mathbf{G}(\vec{x})} \le \frac{2}{1+\alpha}\left\|\vec{y}\right\|_{\mathbf{G}(\vec{x})} \quad \text{and} \quad \left\|\mathbf{B}\vec{y}\right\|_\infty \le \frac{2}{1+\alpha}\left(\left\|\vec{y}\right\|_\infty + \frac{1+2\alpha}{1+\alpha}\left\|\vec{y}\right\|_{\mathbf{G}(\vec{x})}\right).$$

*Therefore*

$$\left\|\mathbf{B}\right\|_{\vec{g}+\infty} \le \frac{2}{1+\alpha}\left(1 + \frac{2}{C_{norm}}\right).$$

*Proof.* Fix an arbitrary $\vec{x} \in \Omega^0$ and let $\vec{g} \stackrel{\text{def}}{=} \vec{g}(\vec{x})$, $\vec{\sigma} \stackrel{\text{def}}{=} \vec{\sigma}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)$, $\mathbf{\Sigma} \stackrel{\text{def}}{=} \mathbf{\Sigma}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)$, $\mathbf{P} \stackrel{\text{def}}{=} \mathbf{P}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)$, $\mathbf{\Lambda} \stackrel{\text{def}}{=} \mathbf{\Lambda}_{\mathbf{A}_x}\left(\vec{g}^{-\alpha}(\vec{x})\right)$. Also, fix an arbitrary $\vec{y} \in \mathbb{R}^m$ and let $\vec{z} \stackrel{\text{def}}{=} \mathbf{B}\vec{y}$.

By Lemma 14, $\mathbf{G}' = -\mathbf{G}\left(\mathbf{G} + \alpha\mathbf{\Lambda}\right)^{-1}\mathbf{\Lambda}\left(\mathbf{\Phi}''\right)^{-1}\mathbf{\Phi}'''$ and therefore

$$\begin{aligned}\mathbf{B} &= -\mathbf{G}^{-1}\left(\mathbf{G}\left(\mathbf{G} + \alpha\mathbf{\Lambda}\right)^{-1}\mathbf{\Lambda}\left(\mathbf{\Phi}''\right)^{-1}\mathbf{\Phi}'''\right)\left(\mathbf{\Phi}''\right)^{-1/2} \\ &= \left(\mathbf{G} + \alpha\mathbf{\Lambda}\right)^{-1}(2\mathbf{\Lambda})\,\mathbf{diag}\left(\frac{-\vec{\phi}'''}{2(\vec{\phi}'')^{3/2}}\right).\end{aligned}$$

Let $\mathbf{C} \stackrel{\text{def}}{=} (\mathbf{G} + \alpha\mathbf{\Lambda})^{-1}(2\mathbf{\Lambda})$ and let $\vec{y}' \stackrel{\text{def}}{=} \mathbf{diag}\left(\frac{-\vec{\phi}'''}{2(\vec{\phi}'')^{3/2}}\right)\vec{y}$. By the self concordance of $\vec{\phi}$ (Definition 2) we know that $\|\vec{y}'\| \leq \|\vec{y}\|$ for both $\|\cdot\|_{\mathbf{G}}$ and $\|\cdot\|_{\infty}$. Since $\vec{z} = \mathbf{B}\vec{y} = \mathbf{C}\vec{y}'$, it suffices to bound $\|\mathbf{C}\vec{y}'\|$ in terms of $\|\vec{y}'\|$ for the necessary norms.

Letting $\bar{\mathbf{\Lambda}} \stackrel{\text{def}}{=} \mathbf{G}^{-1/2}\mathbf{\Lambda}\mathbf{G}^{-1/2}$, we simplify the equation further and note that

$$\|\mathbf{C}\|_{\mathbf{G}} = \left\|\mathbf{G}^{1/2}(\mathbf{G} + \alpha\mathbf{\Lambda})^{-1}(2\mathbf{\Lambda})\mathbf{G}^{-1/2}\right\|_2 = \left\|\left(\mathbf{I} + \alpha\bar{\mathbf{\Lambda}}\right)^{-1}(2\bar{\mathbf{\Lambda}})\right\|_2.$$

Now, for any eigenvector, $\vec{v}$, of $\bar{\mathbf{\Lambda}}$ with eigenvalue $\lambda$, we see that $\vec{v}$ is an eigenvector of $(\mathbf{I}+\alpha\bar{\mathbf{\Lambda}})^{-1}(2\bar{\mathbf{\Lambda}})$ with eigenvalue $2\lambda/(1+\alpha\lambda)$. Furthermore, since $\mathbf{0} \preceq \bar{\mathbf{\Lambda}} \preceq \mathbf{I}$, we have that $\|\mathbf{C}\|_{\mathbf{G}} \leq 2/(1+\alpha)$ and hence $\|\vec{z}\|_{\mathbf{G}} \leq 2(1+\alpha)^{-1}\|\vec{y}'\|_{\mathbf{G}} \leq 2(1+\alpha)^{-1}\|\vec{y}\|_{\mathbf{G}}$ as desired.

To bound $\|\vec{z}\|_{\infty}$, we use that $(\mathbf{G} + \alpha\mathbf{\Lambda})\vec{z} = 2\mathbf{\Lambda}\vec{y}'$, $\mathbf{\Lambda} = \mathbf{\Sigma} - \mathbf{P}^{(2)}$, and $\mathbf{G} = \mathbf{\Sigma} + \beta\mathbf{I}$ to derive

$$(1+\alpha)\mathbf{\Sigma}\vec{z} + \beta\vec{z} - \alpha\mathbf{P}^{(2)}\vec{z} = 2\mathbf{\Sigma}\vec{y}' - 2\mathbf{P}^{(2)}\vec{y}'.$$

Looking at the $i^{th}$ coordinate of both sides and using that $\vec{\sigma}_i \geq 0$, we have

$$\begin{aligned}
&((1+\alpha)\vec{\sigma}_i + \beta)|\vec{z}_i| \\
\leq&\alpha\left|[\mathbf{P}^{(2)}\vec{z}]_i\right| + 2\vec{\sigma}_i\|\vec{y}'\|_{\infty} + 2\left|[\mathbf{P}^{(2)}\vec{y}']_i\right| \\
\leq&\alpha\vec{\sigma}_i\|\vec{z}\|_{\mathbf{\Sigma}} + 2\vec{\sigma}_i\|\vec{y}'\|_{\infty} + 2\vec{\sigma}_i\|\vec{y}'\|_{\mathbf{\Sigma}} && \text{(Lemma 38)} \\
\leq&2\vec{\sigma}_i\|\vec{y}'\|_{\infty} + \vec{\sigma}_i\left(\frac{2\alpha}{1+\alpha} + 2\right)\|\vec{y}'\|_{\mathbf{G}} && (\mathbf{\Sigma} \preceq \mathbf{G}\text{and } \|\vec{z}\|_{\mathbf{G}} \leq 2(1+\alpha)^{-1}\|\vec{y}'\|_{\mathbf{G}})
\end{aligned}$$

Hence, we have

$$\begin{aligned}
|\vec{z}_i| &\leq \frac{2}{1+\alpha}\|\vec{y}'\|_{\infty} + \frac{1}{1+\alpha}\left(\frac{2\alpha}{1+\alpha} + 2\right)\|\vec{y}'\|_{\mathbf{G}} \\
&\leq \frac{2}{1+\alpha}\left[\|\vec{y}'\|_{\infty} + 2\|\vec{y}'\|_{\mathbf{G}}\right].
\end{aligned}$$

Therefore, $\|\mathbf{B}\vec{y}\|_{\infty} = \|\vec{z}\|_{\infty} \leq 2(1+\alpha)^{-1}(\|\vec{y}'\|_{\infty} + 2\|\vec{y}'\|_{\mathbf{G}})$. Finally, we note that

$$\begin{aligned}
\|\mathbf{B}\vec{y}\|_{\vec{g}+\infty} &= \|\mathbf{B}\vec{y}\|_{\infty} + C_{\text{norm}}\|\mathbf{B}\vec{y}\|_{\mathbf{G}} && \text{(Definition)} \\
&\leq \frac{2}{1+\alpha}\|\vec{y}\|_{\infty} + \frac{2}{1+\alpha}\cdot 2\|\vec{y}\|_{\mathbf{G}} + \frac{2}{1+\alpha}C_{\text{norm}}\|\vec{y}\|_{\mathbf{G}} \\
&\leq \frac{2}{1+\alpha}\left(1 + \frac{2}{C_{\text{norm}}}\right)\|\vec{y}\|_{\vec{g}+\infty}.
\end{aligned}$$

□

**Theorem 16.** *Choosing parameters*

$$\alpha = 1 + \frac{1}{\log_2\left(\frac{2m}{\text{rank}(\mathbf{A})}\right)} \quad , \quad \beta = \frac{\text{rank}(\mathbf{A})}{2m} \quad , \textit{ and } \quad C_{norm} = 18\log_2\left(\frac{2m}{\text{rank}(\mathbf{A})}\right)$$

*yields*

$$c_1(\vec{g}) = 2\,\text{rank}(\mathbf{A}) \quad , \quad c_\gamma(\vec{g}) = 1 + \frac{1}{9\log_2\left(\frac{2m}{\text{rank}(\mathbf{A})}\right)} \quad , \textit{ and } \quad c_\delta(\vec{g}) = 1 - \frac{2}{9\log_2\left(\frac{2m}{\text{rank}(\mathbf{A})}\right)}.$$

*In particular, we have*

$$c_\gamma(\vec{g})c_\delta(\vec{g}) \leq 1 - \frac{1}{9\log_2\left(\frac{2m}{\text{rank}(\mathbf{A})}\right)}.$$

*Proof.* The bounds on $c_1(\vec{g})$ and $c_\delta(\vec{g})$ follow immediately from Lemma 14 and Lemma 15. Now, we estimate the $c_\gamma(\vec{g})$ and let $\frac{4}{5}\vec{g} \leq \vec{w} \leq \frac{5}{4}\vec{g}$. Fix an arbitrary $\vec{x} \in \Omega^0$ and let $\vec{g} \stackrel{\text{def}}{=} \vec{g}(\vec{x})$. Recall that by Lemma 14, we have $\vec{g} \geq \beta$. Furthermore, since $\vec{g}^{-1} = \vec{g}^{\alpha-1}\vec{g}^{-\alpha}$ and $\beta^{\alpha-1} \geq \frac{1}{2}$, the following holds

$$\frac{4}{10}\vec{g}_i^{-\alpha} \leq \frac{4}{5}\beta^{\alpha-1}\vec{g}_i^{-\alpha} \leq \frac{4}{10}\vec{g}_i^{-1} \leq \vec{w}_i^{-1} \tag{6.3}$$

for all $i$. Applying this and using the definition of $\mathbf{P}_{\mathbf{A}_x}$ yields

$$\mathbf{A}_x(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}\mathbf{A}_x^T \preceq \frac{10}{4}\mathbf{A}_x(\mathbf{A}_x^T\mathbf{G}^{-\alpha}\mathbf{A}_x)^{-1}\mathbf{A}_x^T = \frac{10}{4}\mathbf{G}^{\alpha/2}\mathbf{P}_{\mathbf{A}_x}(\vec{g}^{-\alpha})\mathbf{G}^{\alpha/2} \quad . \tag{6.4}$$

Hence, we have

$$\begin{aligned}
\frac{\vec{\sigma}_i\left(\frac{1}{\vec{w}\vec{\phi}''}\right)}{\vec{w}_i} &= \frac{\vec{\mathbb{1}}_i^T\mathbf{A}_x(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}\mathbf{A}_x^T\vec{\mathbb{1}}_i}{\vec{w}_i^2} \\
&\leq \frac{10}{4}\frac{\vec{\mathbb{1}}_i^T\mathbf{G}^{\alpha/2}\mathbf{P}_{\mathbf{A}_x}(\vec{g}^{-\alpha})\mathbf{G}^{\alpha/2}\vec{\mathbb{1}}_i}{\vec{w}_i^2} \\
&\leq \frac{10}{4}\left(\frac{5}{4}\right)^2\frac{\vec{\sigma}_i\left(\frac{1}{\vec{g}^{\alpha}\vec{\phi}''}\right)}{\vec{g}_i^{-2\alpha}} < 4.
\end{aligned}$$

Since $\mathbf{P}_{\vec{x},\vec{w}}$ is an orthogonal projection in $\|\cdot\|_{\vec{w}}$, we have $\left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}\to\vec{w}} = 1$. Let $\overline{\mathbf{P}}_{\vec{x},\vec{w}} \stackrel{\text{def}}{=} \mathbf{I} - \mathbf{P}_{\vec{x},\vec{w}}$, we have

$$\begin{aligned}
\left\|\overline{\mathbf{P}}_{\vec{x},\vec{w}}\right\|_{\vec{w}\to\infty} &= \max_{i\in[m]}\max_{\left\|\vec{y}\right\|_{\vec{w}}\leq 1}\vec{\mathbb{1}}_i^T\overline{\mathbf{P}}_{\vec{x},\vec{w}}\vec{y} \\
&\leq \max_{i\in[m]}\left\|(\vec{w})^{-1/2}\,\overline{\mathbf{P}}_{\vec{x},\vec{w}}^T\vec{\mathbb{1}}_i\right\|^2 \\
&= \max_{i\in[m]}\sqrt{\vec{\mathbb{1}}_i\mathbf{W}^{-1}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\mathbf{W}^{-1}\vec{\mathbb{1}}_i}. \\
&= \max_{i\in[m]}\sqrt{\frac{\sigma_i\left(\frac{1}{\vec{w}\vec{\phi}''}\right)}{w_i}} \leq 2.
\end{aligned}$$

For any $\vec{y}$, we have

$$\begin{aligned}
\left\|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\right\|_{\vec{w}+\infty} &\leq \left\|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\right\|_\infty + C_{\text{norm}}\left\|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\right\|_{\vec{w}} \\
&\leq \left\|\vec{y}\right\|_\infty + \left\|\overline{\mathbf{P}}_{\vec{x},\vec{w}}\vec{y}\right\|_\infty + C_{\text{norm}}\left\|\vec{y}\right\|_{\vec{w}} \\
&\leq \left\|\vec{y}\right\|_\infty + (2 + C_{\text{norm}})\left\|\vec{y}\right\|_{\vec{w}} \\
&\leq \frac{C_{\text{norm}} + 2}{C_{\text{norm}}}\left\|\vec{y}\right\|_{\vec{w}+\infty}.
\end{aligned}$$

Hence, we have $c_\gamma \leq \frac{C_{\text{norm}}+2}{C_{\text{norm}}}$. Thus, we have picked $C_{\text{norm}} = \frac{18}{\alpha-1}$ and have $c_\gamma \leq 1 + \frac{\alpha-1}{9}$. $\square$

### 6.1 Computing and Correcting Weight Function

Here we discuss how to compute the weight function using gradient descent and dimension reduction techniques as in [35] for approximately computing leverage scores. The algorithm and the proof is essentially the same as in Part I [22], modified to the subtle changes in the weight function.

**Theorem 17** (Weight Computation and Correction). *There is an algorithm,* `computeWeight`$(\vec{x}, \vec{w}^{(0)}, K)$*, that given* $K < 1$ *and* $\{\vec{x}^{(0)}, \vec{w}^{(0)}\} \in \{\Omega^0 \times \mathbb{R}^m_{>0}\}$ *such that* $\left\|\mathbf{W}_{(0)}^{-1}(\vec{g}(\vec{x}) - \vec{w}^{(0)})\right\|_\infty \leq \frac{1}{48}$ *the algorithm returns* $\vec{w} \in \mathbb{R}^m_{>0}$ *such that*

$$\left\|\mathbf{G}(\vec{x})^{-1}(\vec{g}(\vec{x}) - \vec{w})\right\|_\infty \leq K$$

*with probability* $(1 - \frac{1}{m})^{O(\log^2(m/K))}$ *using only* $\tilde{O}(\log^3(1/K)/K^2)$ *linear system solves.*

*Without the initial weight* $\vec{w}^{(0)}$*, there is an algorithm,* `computeInitialWeight`$(\vec{x}, K)$*, that returns a weight with same guarantee with constant probability using only* $\tilde{O}(\sqrt{\operatorname{rank}(\mathbf{A})}\log^3(1/K)/K^2)$ *times linear system solves.*

*Proof.* Let $Q = \{\vec{w} : \left\|\mathbf{W}_{(0)}^{-1}(\vec{w} - \vec{w}^{(0)})\right\|_\infty \leq \frac{1}{48}\}$. From our assumption, $\vec{g}(\vec{x}) \in Q$. For any $\vec{w} \in Q$, it is easy to see that

$$\frac{4}{5}\mathbf{W}^{-1} \preceq \nabla^2_{\vec{w}\vec{w}}\hat{f}(\vec{x}, \vec{w}) \preceq 4\mathbf{W}^{-1}.$$

Therefore, in this region $Q$, the function is well conditioned and gradient descent converges to the minimizer of $\hat{f}$ quickly. Note that a gradient descent step projected on $Q$ can be written as

$$\vec{w}^{(j)} = \texttt{median}\left(\left(1 - \frac{1}{48}\right)\vec{w}^{(0)}, \frac{3}{4}\vec{w}^{(j-1)} + \frac{1}{4}\vec{\sigma}_{\mathbf{A}_x}\left(\left(\vec{w}^{(k)}\right)^{-\alpha}\right) + \frac{\beta}{4}, \left(1 + \frac{1}{48}\right)\vec{w}^{(0)}\right).$$

Similarly to [22], one can show that the iteration is stable under noise induced by approximate leverage score computation and therefore yields the desired approximation of $\vec{g}(\vec{x})$ assuming we can compute $\vec{\sigma}_{\mathbf{A}_x}$ with small multiplicative $\ell_\infty$ error. Since such leverage scores can be computed with high probability by solving $\tilde{O}(1)$ linear systems [35] we have that there is an algorithm `computeWeight` as desired.

To compute the initial weight, we follow the approach in Part I [22]. Note that if $\beta = 100$, then $\vec{w} = 100$ is a good approximation of $\vec{g}(\vec{x})$. Consequently, we can repeatedly use `computeWeight` to compute the $\vec{g}(\vec{x})$ for a certain $\beta$ and then decrease $\beta$ by a factor of $1 - \sqrt{\operatorname{rank}\mathbf{A}}$. This algorithm converges in $\tilde{O}\left(\sqrt{\operatorname{rank}\mathbf{A}}\right)$ iterations yielding the desired result. □

# 7 The Algorithm

Here we show how to use the results of previous sections to solve (3.1) using exact linear system solver. In the next section we will discuss how to relax this assumption. The central goal of this section is to develop an algorithm, `LPSolve`, for which we can prove the following theorem

**Theorem 18.** *Suppose we have an interior point* $\vec{x}_0 \in \Omega^0$ *for the linear program* (3.1)*. Then, the algorithm* `LPSolve` *outputs* $\vec{x}$ *such that* $\vec{c}^T\vec{x} \leq OPT + \epsilon$ *in* $\tilde{O}\left(\sqrt{\operatorname{rank}(\mathbf{A})}\left(\mathcal{T}_w + \operatorname{nnz}(\mathbf{A})\right)\log\left(U/\epsilon\right)\right)$ *work and* $\tilde{O}\left(\sqrt{\operatorname{rank}(\mathbf{A})}\mathcal{T}_d\log\left(U/\epsilon\right)\right)$ *depth where* $U = \max\left(\left\|\frac{\vec{u}-\vec{l}}{\vec{u}-\vec{x}_0}\right\|_\infty, \left\|\frac{\vec{u}-\vec{l}}{\vec{x}_0-\vec{l}}\right\|_\infty, \left\|\vec{u}-\vec{l}\right\|_\infty, \left\|\vec{c}\right\|_\infty\right)$ *and* $\mathcal{T}_w$ *and* $\mathcal{T}_d$ *is the work and depth needed to compute* $\left(\mathbf{A}^T\mathbf{D}\mathbf{A}\right)^{-1}\vec{q}$ *for input positive definite diagonal matrix* $\mathbf{D}$ *and vector* $\vec{q}$*.*

We break this proof into several parts. First we provide Lemma 19, and adaptation of a proof from [26, Thm 4.2.7] that allows us to reason about the effects of making progress along the weighted central path. Then we provide Lemma 20 that we use to bound the distance to the weighted central path in terms of centrality. After that in Lemma 20, we analyze a subroutine, `pathFollowing`, for following the weighted central path. Using these lemmas we conclude by describing our `LPSolve` algorithm and proving Theorem 18.

**Lemma 19** ([26, Theorem 4.2.7]). *Let $x^* \in \mathbb{R}^m$ denote an optimal solution to (3.1) and $\vec{x}_t = \arg\min f_t(\vec{x}, \vec{w})$ for some $t > 0$ and $\vec{w} \in \mathbb{R}^m_{>0}$. Then the following holds*

$$\vec{c}^T\vec{x}_t(\vec{w}) - \vec{c}^T\vec{x}^* \leq \frac{\|\vec{w}\|_1}{t}.$$

*Proof.* By the optimality conditions of (3.1) we know that $\nabla_x f_t(\vec{x}_t(\vec{w})) = t \cdot \vec{c} + \vec{w}\vec{\phi}'(\vec{x}_t(\vec{w}))$ is orthogonal to the kernel of $\mathbf{A}^T$. Furthermore since $\vec{x}_t(\vec{w}) - \vec{x}^* \in \ker(\mathbf{A}^T)$ we have

$$\left(t \cdot \vec{c} + \vec{w}\vec{\phi}'(\vec{x}_t(\vec{w}))\right)^T (\vec{x}_t(\vec{w}) - \vec{x}^*) = 0.$$

Using that $\phi_i'(x_t(\vec{w})_i) \cdot (x_i^* - x_t(\vec{w})_i) \leq 1$ by Lemma 4 then yields

$$\vec{c}^T(\vec{x}_t(\vec{w}) - \vec{x}^*) = \frac{1}{t}\sum_{i\in[m]} w_i \cdot \phi_i'(x_t(\vec{w})_i) \cdot (x_i^* - x_t(\vec{w})_i) \leq \frac{\|\vec{w}\|_1}{t}.$$

□

**Lemma 20.** *For $\delta_t(\vec{x}^{(1)}, \vec{g}(\vec{x}^{(1)})) \leq \frac{1}{960c_k^2\log(400m)}$ and $\vec{x}_t \stackrel{\text{def}}{=} \arg\min f_t(\vec{x}, \vec{w})$ we have*

$$\left\|\sqrt{\vec{\phi}''(\vec{x}_t)}\left(\vec{x}^{(1)} - \vec{x}_t\right)\right\|_\infty \leq 16c_\gamma c_k \delta_t(\vec{x}^{(1)}, \vec{g}(\vec{x}^{(1)})).$$

*Proof.* We use Theorem 12 with exact weight computation and start with $\vec{x}^{(1)}$ and $\vec{w}^{(1)} = \vec{g}(\vec{x}^{(1)})$. In each iteration, $\delta_t$ is decreased by a factor of $\left(1 - \frac{1}{4c_k}\right)$. (4.5) shows that

$$\|\sqrt{\vec{\phi}''(\vec{x}^{(k)})}\left(\vec{x}^{(k+1)} - \vec{x}^{(k)}\right)\|_\infty \leq c_\gamma \delta_t(\vec{x}^{(k)}, \vec{w}^{(k)}).$$

The Lemma 3 shows that

$$\begin{aligned}\left\|\log\left(\vec{\phi}''(\vec{x}^{(k)})\right) - \log\left(\vec{\phi}''(\vec{x}^{(k+1)})\right)\right\|_\infty &\leq \left(1 - 2c_\gamma\delta_t(\vec{x}^{(k)}, \vec{w}^{(k)})\right)^{-1} \\ &\leq e^{4c_\gamma\delta_t(\vec{x}^{(k)}, \vec{w}^{(k)})}.\end{aligned}$$

Therefore, for any $k$, we have

$$\begin{aligned}\left\|\log\left(\vec{\phi}''(\vec{x}^{(1)})\right) - \log\left(\vec{\phi}''(\vec{x}^{(k)})\right)\right\|_\infty &\leq e^{4c_\gamma\sum\delta_t(\vec{x}^{(k)}, \vec{w}^{(k)})} \\ &\leq e^{32c_kc_\gamma\delta_t(\vec{x}^{(1)}, \vec{g}(\vec{x}^{(1)}))} \\ &\leq 2.\end{aligned}$$

Hence, for any $k$, we have

$$\begin{aligned}\left\|\sqrt{\vec{\phi}''(\vec{x}_t)}\left(\vec{x}^{(1)} - \vec{x}^{(k)}\right)\right\|_\infty &\leq \sum 2c_\gamma\delta_t(\vec{x}^{(k)}, \vec{w}^{(k)}) \\ &\leq 16c_\gamma c_k\delta_t(\vec{x}^{(1)}, \vec{g}(\vec{x}^{(1)})).\end{aligned}$$

It is clear now $\vec{x}^{(k)}$ forms a Cauchy sequence and converges to $\vec{x}_t$ because $\delta_t$ continuous and $\vec{x}_t$ is the unique point such that $\delta_t = 0$. □

Next, we put together the results of Section 5 and analyze the following algorithm for following the weighted central path.

| $(\vec{x}^{(\textbf{final})}, \vec{w}^{(\textbf{new})}) = \texttt{pathFollowing}(\vec{x}, \vec{w}, t_{\textbf{start}}, t_{\textbf{end}}, \epsilon)$ |
|---|
| 1. $c_k = 9\log_2\left(\frac{2m}{\mathrm{rank}(\mathbf{A})}\right), t = t_{\text{start}}, K = \frac{1}{20c_k}$. |
| 2. While ($t < t_{end}$ if $t_{\text{start}} < t_{\text{end}}$) or ($t > t_{end}$ if $t_{\text{start}} > t_{\text{end}}$) |
| 2a. $(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) = \texttt{centeringInexact}(\vec{x}, \vec{w}, K)$<br>where it use the function `computeWeight` to find the approximation of $\vec{g}(\vec{x})$. |
| 2b. $t \leftarrow t\left(1 \pm \frac{1}{10^5 c_k^4 \log(400m)\sqrt{\mathrm{rank}(\mathbf{A})}}\right)$ where the sign of $\pm$ is the sign of $t_{\text{end}} - t_{\text{start}}$ |
| 2c. $\vec{x} \leftarrow \vec{x}^{(\text{final})}$, $\vec{w} \leftarrow \vec{w}^{(\text{new})}$. |
| 3. Repeat $4c_k \log(1/\epsilon)$ times |
| 3a. $(\vec{x}, \vec{w}) = \texttt{centeringInexact}(\vec{x}, \vec{w}, K)$<br>where it use the function `computeWeight` to find the approximation of $\vec{g}(\vec{x})$. |
| 4. Output $(\vec{x}, \vec{w})$. |

**Theorem 21.** *Suppose that*

$$\delta_{t_{start}}(\vec{x}, \vec{w}) \leq \frac{1}{960c_k^2 \log(400m)} \quad \textit{and} \quad \Phi_\mu(\vec{\Psi}(\vec{x}, \vec{w})) \leq (400m)^2 .$$

*where $\mu = 2\log(400m)/K$. Let $(\vec{x}^{(final)}, \vec{w}^{(new)}) = \texttt{pathFollowing}(\vec{x}, \vec{w}, t_{start}, t_{end})$, then*

$$\delta_{t_{end}}(\vec{x}^{(final)}, \vec{w}^{(new)}) \leq \epsilon \quad \textit{and} \quad \Phi_\mu(\vec{\Psi}(\vec{x}^{(final)}, \vec{w}^{(new)})) \leq (400m)^2 .$$

*Furthermore, $\texttt{pathFollowing}(\vec{x}, \vec{w}, t_{start}, t_{end})$ takes time $\tilde{O}\left(\sqrt{\mathrm{rank}(\mathbf{A})}\left(\left|\log\left(\frac{t_{end}}{t_{start}}\right)\right| + \log(1/\epsilon)\right)(\mathcal{T} + m)\right)$ where $\mathcal{T}$ is the time needed to solve on linear system.*

*Proof.* This algorithm maintains the invariant that $\delta_t(\vec{x}, \vec{w}) \leq \frac{1}{960c_k^2\log(400m)}$ and $\Phi_\alpha(\vec{\Psi}(\vec{x}, \vec{w})) \leq (400m)^2$ on each iteration in the beginning of the step (2a). Theorem 12 shows that

$$\left\| \log(\vec{g}(\vec{x}^{(\text{new})})) - \log(\vec{w}) \right\|_\infty \leq K \leq \frac{1}{20c_k}. \tag{7.1}$$

Thus, the weight satisfies the condition of Theorem 17 and the algorithm `centeringInexact` can use the function `computeWeight` to find the approximation of $\vec{g}(\vec{x}^{(\text{new})})$. Consequently,

$$\delta_t(\vec{x}^{(\text{final})}, \vec{w}^{(\text{new})}) \leq \left(1 - \frac{1}{4c_k}\right)\delta_t \quad \text{and} \quad \Phi_\alpha(\vec{\Psi}(\vec{x}^{(\text{final})}, \vec{w}^{(\text{new})})) \leq (400m)^2 .$$

Using Lemma 7, (7.1) and Theorem 16,we have

$$\delta_t(\vec{x}^{(\text{final})}, \vec{w}^{(\text{new})}) \leq \frac{1}{960c_k^2 \log(400m)}.$$

Hence, we proved that for every step (2c), we have the invariant. The $\delta_t < \epsilon$ bounds follows from the last loop. $\square$

| $\vec{x}^{(\textbf{final})} = \texttt{LPSolve}(\vec{x}, \epsilon)$ |
| --- |
| Input: an initial point $\vec{x}$. |
| 1. $\beta = \frac{\text{rank}(\mathbf{A})}{2m}$, $\vec{w} = \texttt{computeInitialWeight}(\vec{x}, \frac{1}{10^5 \log^5(400m)})$, $d = -\vec{w}_i \phi_i'(\vec{x})$. |
| 2. $t_1 = (10^{10} U^2 m^3)^{-1}$, $t_2 = 3m/\epsilon$, $\epsilon_1 = \frac{1}{2000 c_k^2 \log(400m)}$, $\epsilon_2 = \frac{\epsilon}{100^3 m^3 U^2}$. |
| 3. $(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) = \texttt{pathFollowing}(\vec{x}, \vec{w}, 1, t_1, \epsilon_1)$ with cost vector $\vec{d}$. |
| 4.$(\vec{x}^{(\text{final})}, \vec{w}^{(\text{final})}) = \texttt{pathFollowing}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}, t_1, t_2, \epsilon_2)$ with cost vector $\vec{c}$. |
| 5. Output $\vec{x}^{(\text{final})}$. |

*Proof of Theorem 18.* By Theorem 17, we know step 1 gives an weight

$$\left\|\mathbf{G}(\vec{x})^{-1}(\vec{g}(\vec{x}) - \vec{w})\right\|_\infty \leq \frac{1}{10^5 \log^5(400m)}.$$

By the definition of $\vec{d}$, we have $\vec{x}$ is the minimum of

$$\min \vec{d}^T \vec{x} - \sum \vec{w}_i \phi_i(\vec{x}) \text{ given } \mathbf{A}^T \vec{x} = \vec{b}.$$

Therefore, $(\vec{x}, \vec{w})$ satisfies the assumption of theorem 21 because $\delta_t = 0$ and $\Phi_\alpha$ is small enough. Hence, we have

$$\delta_{t_1}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) \leq \frac{1}{2000 c_k^2 \log(400m)} \quad \text{and} \quad \Phi_\alpha(\vec{\Psi}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})})) \leq (400m)^2.$$

Lemma 4 shows that $\left\|\phi_i'(\vec{x})\right\|_\infty \leq U$ and hence $\left\|\vec{c} - \vec{d}\right\|_\infty \leq 2U$. Also, Lemma 3 shows that $\min_{\vec{y}} \sqrt{\vec{\phi}''(\vec{y})} \geq \frac{1}{U}$. Therefore, we have

$$\begin{aligned}
\delta_{t_1}^{\vec{c}}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) &= \min_{\vec{\eta} \in \mathbb{R}^n} \left\| \frac{t_1 \vec{c} + \vec{w}\vec{\phi}'(\vec{x}^{(\text{new})}) - \mathbf{A}\vec{\eta}}{\vec{w}^{(\text{new})}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}} \right\|_{\vec{w}^{(\text{new})}+\infty} \\
&\leq \min_{\vec{\eta} \in \mathbb{R}^n} \left\| \frac{t_1 \vec{d} + \vec{w}\vec{\phi}'(\vec{x}^{(\text{new})}) - \mathbf{A}\vec{\eta}}{\vec{w}^{(\text{new})}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}} \right\|_{\vec{w}^{(\text{new})}+\infty} + t_1 \left\| \frac{\vec{c} - \vec{d}}{\vec{w}^{(\text{new})}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}} \right\|_{\vec{w}^{(\text{new})}+\infty} \\
&\leq \delta_{t_1}^{\vec{d}}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) + 4U^2 t_1 \left\|\vec{\mathbb{1}}\right\|_{\vec{w}+\infty} \\
&= \delta_{t_1}^{\vec{d}}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})}) + 100 m U^2 t_1.
\end{aligned}$$

Since we have chosen $t_1$ small enough, we have $\delta_{t_1}^{\vec{c}}(\vec{x}^{(\text{new})}, \vec{w}^{(\text{new})})$ is small enough to satisfy the assumption of Theorem 21. So, we only need to prove how large $t_2$ should be and how small $\epsilon_2$ should be in order to get $\vec{x}$ such that $\vec{c}^T\vec{x} \leq \text{OPT} + \epsilon$. By Lemma 19 and $\left\|\vec{w}^{(\text{final})}\right\| \leq 3m$, we have

$$\vec{c}^T \vec{x}_{t_2} \leq \text{OPT} + \frac{3m}{t_2}.$$

Also, Lemma 20 shows that we have

$$\left\|\sqrt{\vec{\phi}''(\vec{x}_{t_2})}\left(\vec{x}^{(\text{final})} - \vec{x}_{t_2}\right)\right\|_\infty \leq 32 \epsilon_2 c_k.$$

Using $\min_{\vec{y}} \sqrt{\vec{\phi}''(\vec{y})} \geq \frac{1}{U}$, we have $\left\|\vec{x}^{(\text{final})} - \vec{x}_{t_2}\right\|_\infty \leq 32 \epsilon_2 c_k U$ and hence our choice of $t_2$ and $\epsilon_2$ gives the result

$$\vec{c}^T \vec{x}^{(\text{final})} \leq \text{OPT} + \frac{3m}{t_2} + 32 \epsilon_2 c_k U^2 \leq \text{OPT} + \epsilon.$$

$\square$

# 8 Linear System Solver Requirements

Throughout our preceding analysis of weighted path finding we assumed that linear systems related to $\mathbf{A}$ could be solved exactly. In this section, we relax this assumption and discuss the effect of using inexact linear algebra in our algorithms.

Proving stability of the algorithms in this paper is more difficult than the "dual" algorithms in Part I [22] for two reasons. First, naively each iteration of interior point requires a linear system to be solved to to $\tilde{O}(\mathsf{poly}(\epsilon/U))$ accuracy and if we need to solve each linear system independently then the overall running time of our algorithm would depends on $\log^2(U/\epsilon)$ and improving this requires further insight. Second, here we need to maintain equality constraints which further complicates the analysis.

For the remainder of this section we assume that we have an algorithm $\mathtt{S}_{x,w}(\vec{q})$ such that for any vector $\vec{q}$ the algorithm $\mathtt{S}_{x,w}(\vec{q})$ outputs a vector in $\mathbb{R}^n$ such that

$$\left\|\mathtt{S}_{x,w}(\vec{q}) - \left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\vec{q}\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \leq \epsilon_{\mathtt{S}}\left\|\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\vec{q}\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x}$$

where $\epsilon_{\mathtt{S}} = 1/m^d$ for some sufficiently large, but fixed, $d$. Our goal in this section is is to show that implementing such a $\mathtt{S}_{x,w}(\vec{q})$ suffices for our algorithms (Section 8.1, 8.2, 8.3, 8.4). In Section 8.5, we show that the vector $\vec{q}$ satisfies some stability properties that allows us to construct efficient solver $\mathtt{S}_{x,w}(\vec{q})$ in later section.

## 8.1 The normal force $\mathbf{A}\vec{\eta}$.

To see the problem of using inexact linear system solvers more concretely, recall that we defined a Newton steps on $\vec{x} \in \Omega^0$ in Section 5.4 by

$$\begin{aligned}
\vec{x}^{(\text{new})} &:= \vec{x} - \frac{1}{\sqrt{\vec{\phi}''(\vec{x})}}\mathbf{P}_{\vec{x},\vec{w}}\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\\
&= \vec{x} - \frac{1}{\sqrt{\vec{\phi}''(\vec{x})}}\left(\mathbf{I} - \mathbf{W}^{-1}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\right)\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right).
\end{aligned}$$

One naive way to implement this step is to replace $\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}$ with the algorithm $\mathtt{S}_{x,w}$ . Unfortunately, this does not necessarily work well as the norm of the vector $(t\vec{c} - \vec{w}\vec{\phi}'(\vec{x}))/\vec{w}\sqrt{\vec{\phi}''(\vec{x})}$ can be as large as $\Omega(\log(U/\epsilon))$ because the current point $\vec{x}$ can be very close to boundary. For certain linear programs, the parameter $t$ need to be exponentially large and therefore for this approach to work we would need to use exponentially small $\epsilon_K$. The dual problem does not has this problem because the optimality conditions enforce $\nabla f$ is small. However, for the primal problem we are solving, the equality constraints puts a normal force into the systems. Therefore, even when we are very close to the optimal point, $\nabla f$ can be very large due to the normal force.

To circumvent this issue, we note that if we approximately know the normal force, then we can subtract it off from the system and only deal with a vector of reasonable size. In this section, we try to maintain such normal force $\mathbf{A}\vec{\eta}$. Recall that our algorithm measures the quality of $\vec{x}$ by

$$\delta_t(\vec{x},\vec{w}) \stackrel{\text{def}}{=} \min_{\vec{\eta}\in\mathbb{R}^n}\left\|\frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}.$$

We can think $\delta_t$ is the size of net force of the system, i.e. the result of subtracting the normal force $\mathbf{A}\vec{\eta}$ from the total force $\nabla f$. If $\delta_t$ is small, we know the contact force $\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}$ is small. Therefore, the following formula gives a more stable way to compute $\vec{x}^{(\text{new})}$:

$$\vec{x}^{(\text{new})} := \vec{x} - \left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\vec{\phi}''(\vec{x})}\right) + \frac{1}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\mathbf{A}_x \mathtt{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right).$$

Furthermore, since $\mathbf{P}_{\vec{x},\vec{w}}\mathbf{W}^{-1}\mathbf{A}_x = \mathbf{0}$, subtracting $\mathbf{A}\vec{\eta}$ from $\nabla_x f_t$ does not affect the step. Therefore, if we can find that $\vec{\eta}$, then we have a more stable algorithm.

First , we show that there is an explicit $\vec{\eta}^*$ that can be computed in polynomial time.

**Lemma 22** ($\vec{\eta}^*$ is good)**.** *For all $(\vec{x},\vec{w})$ in the algorithm and $t > 0$, we define the normal force*

$$\vec{\eta}_t^*(\vec{x},\vec{w}) = (\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}\mathbf{A}_x^T\mathbf{W}^{-1}\sqrt{\mathbf{\Phi}''(\vec{x})}^{-1}\nabla_x f_t(\vec{x},\vec{w}).$$

*Then, we have*

$$\left\|\frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}_t^*(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty} \le 2\delta_t(\vec{x},\vec{w}).$$

*Proof.* Using (4.5) we have

$$\left\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\right\|_{\vec{w}+\infty} \le \left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}+\infty}\cdot\delta_t(\vec{x},\vec{w}).$$

The result follows from the definition of $\vec{h}_t(\vec{x},\vec{w})$, i.e.

$$-\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w}) = \frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}_t^*(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}$$

and the fact the $\left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}+\infty} \le 2$ during the algorithm. □

The following lemma shows that we can improve $\vec{\eta}$ effectively using $\mathtt{S}_{x,w}$.

**Lemma 23** ($\vec{\eta}$ maintenance)**.** *For all $(\vec{x},\vec{w})$ appears in the algorithm and $t > 0$, we define*

$$\vec{\eta}^{(new)} = \vec{\eta} + \mathtt{S}_{x,w}\left(\mathbf{A}_x^T\mathbf{W}^{-1}\sqrt{\mathbf{\Phi}''(\vec{x})}^{-1}\left(\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}\right)\right)$$

*and*

$$\vec{\eta}_t^*(\vec{x},\vec{w}) = (\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}\mathbf{A}_x^T\mathbf{W}^{-1}\sqrt{\mathbf{\Phi}''(\vec{x})}^{-1}\nabla_x f_t(\vec{x},\vec{w}).$$

*If $\epsilon_{\mathtt{S}} \le \frac{1}{2}$, we have*

$$\left\|\frac{\mathbf{A}\left(\vec{\eta}^{(new)} - \vec{\eta}_t^*(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} \le \epsilon_{\mathtt{S}}\left\|\frac{\mathbf{A}\left(\vec{\eta} - \vec{\eta}_t^*(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}}.$$

*Proof.* By the definition of $\vec{\eta}^{(\text{new})}$ and $\vec{\eta}^{*}(\vec{x},\vec{w})$, we have

$$
\begin{aligned}
& \left\| \frac{\mathbf{A}\left(\vec{\eta}^{(\text{new})} - \vec{\eta}_t^{*}(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\mathbf{W}} \\
= & \left\| \vec{\eta}^{(\text{new})} - \vec{\eta}_t^{*}(\vec{x},\vec{w}) \right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
= & \left\| \vec{\eta} - \vec{\eta}_t^{*}(\vec{x},\vec{w}) + \mathtt{S}_{x,w}\left(\mathbf{A}_x^T\mathbf{W}^{-1}\sqrt{\mathbf{\Phi}''(\vec{x})}^{-1}\left(\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}\right)\right) \right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
= & \left\| \left(\vec{\eta}^{*}(\vec{x},\vec{w}) - \vec{\eta}\right) - \mathtt{S}_{x,w}\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\left(\vec{\eta}_t^{*}(\vec{x},\vec{w}) - \vec{\eta}\right)\right) \right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
\leq & \epsilon_{\mathtt{S}} \left\| \vec{\eta} - \vec{\eta}_t^{*}(\vec{x},\vec{w}) \right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} = \epsilon_{\mathtt{S}} \left\| \frac{\mathbf{A}\left(\vec{\eta} - \vec{\eta}_t^{*}(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\mathbf{W}} .
\end{aligned}
$$

□

Using the Lemma 23 we show how to maintain a good $\vec{\eta}$ throughout our algorithm `LPSolve`.

**Lemma 24** (Finding $\vec{\eta}$)**.** *Assume $\epsilon_{\mathtt{S}} = 1/m^d$ for a sufficiently large constant $d$. Throughout the algorithm we can maintain $\vec{\eta}$ such that*

$$
\left\| \frac{\mathbf{A}\left(\vec{\eta} - \vec{\eta}_t^{*}(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\mathbf{W}}^2 \leq 1 \quad .
$$

*by calling $\mathtt{S}_{x,w}$ an amortized constant number of times per iteration.*

*Proof.* We use $\vec{\eta}_t^{*}(\vec{x},\vec{w})$, defined in (22), as the initial $\vec{\eta}$. Since we compute this only once, we can compute a very precise $\vec{\eta}_t^{*}(\vec{x},\vec{w})$ for the initial points by gradient descent and preconditioning by $\mathtt{S}_{x,w}$.

Lemma 23 shows that we can move $\vec{\eta}$ closer to $\vec{\eta}_t^{*}(\vec{x},\vec{w})$ using $\mathtt{S}_{x,w}$. Therefore, it suffices to show that during each step of the algorithm, $\vec{\eta}$ does not move far from $\vec{\eta}_t^{*}(\vec{x},\vec{w})$ by $O(\mathsf{poly}(n))$, or if it does, we can find $\vec{\eta}^{(\text{new})}$ that does not.

We prove this by considering the three cases of changing of $t$, changing of $\vec{w}$ and changing of $\vec{x}$ separately.

For the changes of $t$, the proof of Lemma 7 shows that

$$
\begin{aligned}
\left\| \frac{\nabla_x f_{t(1+\alpha)}(\vec{x},\vec{w}) - (1+\alpha)\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\vec{w}+\infty} & \leq (1+\alpha) \left\| \frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\vec{w}+\infty} + \alpha\left(1 + C_{\text{norm}}\sqrt{\|\vec{w}\|_1}\right) \\
& \leq 2 \left\| \frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\vec{w}+\infty} + O(\mathsf{poly}(m)).
\end{aligned}
$$

Using the induction hypothesis

$$
\left\| \frac{\mathbf{A}\left(\vec{\eta} - \vec{\eta}_t^{*}(\vec{x},\vec{w})\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}} \right\|_{\mathbf{W}}^2 \leq 1 \quad ,
$$

we have

$$\left\|\frac{\nabla_x f_{t(1+\alpha)}(\vec{x},\vec{w})-(1+\alpha)\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\leq 2\left\|\frac{\nabla_x f_t(\vec{x},\vec{w})-\mathbf{A}\vec{\eta}_t^*(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}+O(\mathsf{poly}(m)) \quad .$$

Now, using Lemma 22, $\delta_t\leq 1$, we have

$$\left\|\frac{\nabla_x f_{t(1+\alpha)}(\vec{x},\vec{w})-(1+\alpha)\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}=O(\mathsf{poly}(m)).$$

Using Lemma 22 again, we have

$$\begin{aligned}
&\left\|\frac{\mathbf{A}\left(\vec{\eta}^*_{t(\text{new})}(\vec{x},\vec{w})-(1+\alpha)\vec{\eta}\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\\
\leq\;&\left\|\frac{\nabla_x f_{t(1+\alpha)}(\vec{x},\vec{w})-(1+\alpha)\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}+\left\|\frac{\nabla_x f_{t(1+\alpha)}(\vec{x},\vec{w})-\vec{\eta}^*_{t(\text{new})}(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\\
=\;&O(\mathsf{poly}(m)).
\end{aligned}$$

Therefore, we can set $\vec{\eta}^{(\text{new})}=(1+\alpha)\vec{\eta}$ and this yields $\vec{\eta}^*_{t(\text{new})}(\vec{x},\vec{w})$ is polynomial close to $\vec{\eta}^{(\text{new})}$.

For the changes of $\vec{w}$, the proof of Lemma 10 shows that

$$\left\|\frac{\nabla_x f_t(\vec{x},\vec{w}^{(\text{new})})-\mathbf{A}\vec{\eta}}{\vec{w}^{(\text{new})}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}\leq(1+4\epsilon)\left(\left\|\frac{\nabla_x f_t(\vec{x},\vec{w})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}+\epsilon\right).$$

Hence, by similar argument above, we have

$$\left\|\frac{\mathbf{A}\left(\vec{\eta}^*(\vec{x},\vec{w}^{(\text{new})})-\vec{\eta}\right)}{\vec{w}^{(\text{new})}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty}=O(\mathsf{poly}(m)).$$

Therefore, we can set $\vec{\eta}^{(\text{new})}=\vec{\eta}$ and this gives $\vec{\eta}^*(\vec{x},\vec{w}^{(\text{new})})$ is polynomial close to $\vec{\eta}^{(\text{new})}$.

For the changes of $\vec{x}$, the proof of Lemma 8 shows that

$$\left\|\frac{\nabla_x f_t(\vec{x}^{(\text{new})},\vec{w})-\mathbf{A}\vec{\eta}^*(\vec{x}^{(\text{new})},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}}\right\|_{\vec{w}+\infty}\leq 4\left(\delta_t(\vec{x},\vec{w})\right)^2=O(\mathsf{poly}(m)).$$

It is easy to show that

$$\begin{aligned}
&\left\|\frac{\mathbf{A}\left(\vec{\eta}^*(\vec{x}^{(\text{new})},\vec{w})-\vec{\eta}\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}}\right\|_{\vec{w}+\infty}\\
\leq\;&\left\|\frac{\nabla_x f_t(\vec{x}^{(\text{new})},\vec{w})-\mathbf{A}\vec{\eta}^*(\vec{x}^{(\text{new})},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}}\right\|_{\vec{w}+\infty}+\left\|\frac{\nabla_x f_t(\vec{x}^{(\text{new})},\vec{w})-\mathbf{A}\vec{\eta}^*(\vec{x}^{(\text{new})},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}}\right\|_{\vec{w}+\infty}\\
&+\left\|\frac{\mathbf{A}\left(\vec{\eta}^*(\vec{x}^{(\text{new})},\vec{w})-\vec{\eta}\right)}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\text{new})})}}\right\|_{\vec{w}+\infty}\\
\leq\;&\mathsf{poly}(m).
\end{aligned}$$

Therefore, we can set $\vec{\eta}^{(\text{new})} = \vec{\eta}$ and this gives $\vec{\eta}^*(\vec{x}^{(\text{new})}, \vec{w})$ is polynomial close to $\vec{\eta}^{(\text{new})}$.

Consequently, in all cases, we can find $\vec{\eta}^{(\text{new})}$ such that gives $\vec{\eta}^{*(\text{new})}$ is polynomial close to $\vec{\eta}^{(\text{new})}$. Applying Lemma 23, we can then obtain a $\vec{\eta}$ such that it is close to $\vec{\eta}^{(\text{new})}$ in $\left\|\cdot\right\|_w$ norm. Note that in the first iteration we need to call $\mathtt{S}_{x,w}$ $O(\log(U/\epsilon))$ time. Therefore, in average, we only call $\mathtt{S}_{x,w}$ constant many times in average per iteration. $\square$

### 8.2 An efficient $\vec{x}$ step

Having such "normal vector" $\vec{\eta}$, we can implement $\vec{x}$ step efficiently. Note that here we crucially use the assumption $\epsilon_{\mathtt{S}} < C/m^2$.

**Lemma 25** (Efficient $\vec{x}$ step)**.** *For all $(\vec{x}, \vec{w})$ in the algorithm and $t > 0$ let*

$$\vec{x}^{(new)} = \vec{x} - \frac{1}{\sqrt{\vec{\phi}''(\vec{x})}} \left(\mathbf{I} - \mathbf{W}^{-1}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\right)\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)$$

*and*

$$\vec{x}^{(apx)} = \vec{x} - \left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\vec{\phi}''(\vec{x})}\right) + \frac{1}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\mathbf{A}_x\mathtt{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right).$$

*We have*

$$\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(new)} - \vec{x}^{(apx)}\right)\right\|_{\vec{w}+\infty} \leq \tilde{O}\left(m\epsilon_{\mathtt{S}}\right)$$

*and*

$$\delta_t(\vec{x}^{(apx)}, \vec{w}) \leq \left(1 + \tilde{O}\left(m\epsilon_{\mathtt{S}}\right)\right)\delta_t(\vec{x}^{(new)}, \vec{w}) + \tilde{O}\left(m\epsilon_{\mathtt{S}}\right).$$

*Proof.* Note that

$$\begin{aligned}\mathbf{W}^{1/2}\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\text{new})} - \vec{x}^{(\text{apx})}\right) &= \mathbf{W}^{-1/2}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\\ &\quad - \mathbf{W}^{-1/2}\mathbf{A}_x\mathtt{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right)\end{aligned}$$

Therefore, we have

$$\begin{aligned}&\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\text{new})} - \vec{x}^{(\text{apx})}\right)\right\|_{\mathbf{W}}\\ \leq\ &\left\|\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right) - \mathtt{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right)\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x}\\ \leq\ &\epsilon_{\mathtt{S}}\left\|\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\mathbf{W}^{-1/2}\left(\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\sqrt{\vec{w}\vec{\phi}''(\vec{x})}}\right)\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x}\\ =\ &\epsilon_{\mathtt{S}}\left\|\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\sqrt{\vec{w}\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}^{-1/2}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T\mathbf{W}^{-1/2}} \leq \epsilon_{\mathtt{S}}\left\|\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}}\\ \leq\ &\epsilon_{\mathtt{S}}\left\|\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}^*}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} + \epsilon_{\mathtt{S}}\left\|\frac{\mathbf{A}(\vec{\eta}^* - \vec{\eta})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} \leq 2\epsilon_{\mathtt{S}}\delta_t(\vec{x}, \vec{w}) + \epsilon_{\mathtt{S}}\quad.\end{aligned}$$

where the last line comes from Lemma 22 and Lemma 24. Hence, we have

$$\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{new})}-\vec{x}^{(\mathrm{apx})}\right)\right\|_{\mathbf{W}}^{2}\leq 3\epsilon_{\mathtt{S}}.$$

Therefore, we have

$$\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{new})}-\vec{x}^{(\mathrm{apx})}\right)\right\|_{\vec{w}+\infty}^{2}\leq \tilde{O}\left(m\epsilon_{\mathtt{S}}\right).$$

For the last assertion, take $\vec{q}$ such that

$$\delta_t(\vec{x}^{(\mathrm{new})},\vec{w})=\left\|\frac{\nabla_x f_t(\vec{x}^{(\mathrm{new})},\vec{w})-\mathbf{A}\vec{q}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{new})})}}\right\|_{\vec{w}+\infty}.$$

Following similar analysis as in Lemma 8, we have

$$\begin{aligned}
\delta_t(\vec{x}^{(\mathrm{apx})},\vec{w}) &\leq \left\|\frac{\nabla_x f_t(\vec{x}^{(\mathrm{apx})},\vec{w})-\mathbf{A}\vec{q}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{apx})})}}\right\|_{\vec{w}+\infty}\\
&\leq \left\|\frac{\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{new})})}}{\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{apx})})}}\right\|_{\infty}\left\|\frac{\nabla_x f_t(\vec{x}^{(\mathrm{new})},\vec{w})-\mathbf{A}\vec{q}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{new})})}}\right\|_{\vec{w}+\infty}\\
&\quad+\left\|\frac{\nabla_x f_t(\vec{x}^{(\mathrm{new})},\vec{w})-\nabla_x f_t(\vec{x}^{(\mathrm{apx})},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x}^{(\mathrm{apx})})}}\right\|_{\vec{w}+\infty}\\
&= \left(1+\tilde{O}\left(m\epsilon_{\mathtt{S}}\right)\right)\delta_t(\vec{x}^{(\mathrm{new})},\vec{w})+\tilde{O}\left(m\epsilon_{\mathtt{S}}\right).
\end{aligned}$$

□

The above lemma shows that $\vec{x}^{(\mathrm{apx})}$ can be used to replace $\vec{x}^{(\mathrm{new})}$ without hurting $\delta_t$ too much. Also, the step size $\vec{x}^{(\mathrm{apx})}-\vec{x}$ is almost the same as the step size of $\vec{x}^{(\mathrm{new})}-\vec{x}$. Thus, we can implement the $\vec{x}$ step without using $\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}$ and using $\mathtt{S}_{x,w}$ instead.

Unfortunately, there is one additional problem with the this algorithm, it does not ensure $\mathbf{A}^T\vec{x}^{(\mathrm{apx})}=\vec{b}$. Therefore, we need to ensure $\mathbf{A}^T\vec{x}^{(\mathrm{apx})}\approx\vec{b}$ during the algorithm. Note that we cannot make $\mathbf{A}^T\vec{x}=\vec{b}$ exactly using this approach and consequently we need measure the infeasibility. We define

$$I(\vec{x},\vec{w})\stackrel{\mathrm{def}}{=}\left\|\mathbf{A}^T\vec{x}-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}}.$$

**Lemma 26.** *For all $(\vec{x},\vec{w})$ in the algorithm define $\vec{x}^{(apx)}$ as in Lemma 25. Then, we have*

$$I(\vec{x}^{(apx)},\vec{w})\leq 2I(\vec{x},\vec{w})+3\epsilon_{\mathtt{S}}.$$

*Proof.* Since $\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{apx})}-\vec{x}\right)\right\|_{\infty}$ is small, it is easy to show

$$\left\|\mathbf{A}^T\vec{x}^{(\mathrm{apx})}-\vec{b}\right\|_{\left(\mathbf{A}_{\vec{x}^{(\mathrm{apx})}}^T\mathbf{W}^{-1}\mathbf{A}_{\vec{x}^{(\mathrm{apx})}}\right)^{-1}}\leq 2\left\|\mathbf{A}\vec{x}^{(\mathrm{apx})}-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}}.$$

Then, note that

$$
\begin{aligned}
& \left\|\mathbf{A}^T\vec{x}^{(\mathrm{apx})}-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}} \\
= & \left\|\mathbf{A}^T\vec{x}-\mathbf{A}^T\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\vec{\phi}''(\vec{x})}\right)+\mathbf{A}_x^T\mathbf{W}\mathbf{A}_x\mathrm{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right)-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}} \\
\leq & \; I(\vec{x},\vec{w})+\left\|\left(\mathbf{A}_x^T\mathbf{W}\mathbf{A}_x\right)^{-1}\mathbf{A}^T\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\vec{\phi}''(\vec{x})}\right)-\mathrm{S}_{x,w}\left(\mathbf{A}_x^T\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right)\right)\right\|_{\mathbf{A}_x^T\mathbf{W}\mathbf{A}_x} \\
\leq & \; I(\vec{x},\vec{w})+\epsilon_{\mathrm{S}}\left\|\left(\mathbf{A}_x^T\mathbf{W}\mathbf{A}_x\right)^{-1}\mathbf{A}^T\left(\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\vec{\phi}''(\vec{x})}\right)\right\|_{\mathbf{A}_x^T\mathbf{W}\mathbf{A}_x} \\
\leq & \; I(\vec{x},\vec{w})+\epsilon_{\mathrm{S}}\left\|\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}}.
\end{aligned}
$$

Now, we bound the last term using Lemma 22 and 24 as follows

$$
\begin{aligned}
\left\|\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} & \leq \left\|\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}^{*}(\vec{x},\vec{w})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}}+\left\|\frac{\mathbf{A}(\vec{\eta}^{*}(\vec{x},\vec{w})-\vec{\eta})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} \\
& \leq 2\delta_t(\vec{x},\vec{w})+1\leq 3.
\end{aligned}
$$

□

Note that when we change $\vec{w}$ so long as no coordinate changes by more than a multiplicative constant then $I(\vec{x},\vec{w})$ changes by at most a multiplicative constant and thus no further proof on the stability of $I(\vec{x},\vec{w})$ with respect to $\vec{w}$ is needed.

Now, we show how to improve $I(\vec{x},\vec{w})$.

**Lemma 27** (Improve Feasibility). *Given $(\vec{x},\vec{w})$ appears in the algorithm. Define*

$$
\vec{x}^{(\mathit{fixed})}=\vec{x}-\sqrt{\mathbf{\Phi}''(\vec{x})}^{-1}\mathbf{W}^{-1}\mathbf{A}_x\mathrm{S}_{x,w}\left(\mathbf{A}\vec{x}-\vec{b}\right).
$$

*Assume that $I(\vec{x},\vec{w})\leq 0.01m^{-1}$, we have*

$$
I(\vec{x}^{(\mathit{fixed})},\vec{w})\leq 2\epsilon_{\mathrm{S}}I(\vec{x},\vec{w}).
$$

*Furthermore, $\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathit{fixed})}-\vec{x}\right)\right\|_{\vec{w}+\infty}\leq O\left(mI(\vec{x},\vec{w})\right)$.*

*Proof.* Note that

$$
\begin{aligned}
\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{fixed})}-\vec{x}\right)\right\|_{\mathbf{W}} & = \left\|\mathrm{S}_{x,w}\left(\mathbf{A}\vec{x}-\vec{b}\right)\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
& \leq (1+\epsilon_{\mathrm{S}})\left\|\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\left(\mathbf{A}\vec{x}-\vec{b}\right)\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
& \leq 2I(\vec{x},\vec{w}).
\end{aligned}
$$

Hence, $\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{fixed})}-\vec{x}\right)\right\|_{\infty}\leq 2mI(\vec{x},\vec{w})$. By the assumption, $\left\|\left(\mathbf{\Phi}''(\vec{x})\right)^{1/2}\left(\vec{x}^{(\mathrm{fixed})}-\vec{x}\right)\right\|_{\infty}$ is very small and hence one can show that

$$
\left\|\mathbf{A}^T\vec{x}^{(\mathrm{fixed})}-\vec{b}\right\|_{\left(\mathbf{A}_{\vec{x}^{(\mathrm{fixed})}}^T\mathbf{W}^{-1}\mathbf{A}_{\vec{x}^{(\mathrm{fixed})}}\right)^{-1}}\leq 2\left\|\mathbf{A}^T\vec{x}^{(\mathrm{fixed})}-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}}.
$$

Now, we note that

$$\begin{aligned}
& \left\|\mathbf{A}^T\vec{x}^{(\text{fixed})}-\vec{b}\right\|_{\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}} \\
= \ & \left\|\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\left(\mathbf{A}^T\vec{x}-\vec{b}\right)-\mathtt{S}_{x,w}\left(\mathbf{A}^T\vec{x}-\vec{b}\right)\right\|_{\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x} \\
\leq \ & \epsilon_{\mathtt{S}}I(\vec{x},\vec{w}).
\end{aligned}$$

□

Since $\epsilon_S$ is sufficiently small this lemma implies that the given step improves feasibility by much more than it hurts centrality. Therefore, by applying this step periodically throughout our algorithm we can maintain the invariant the the infeasibility is small.

### 8.3 An efficient $\vec{w}$ step

There are two computations performed by our algorithm involving the weights. The first is in the "chasing 0" game for centering we are given approximate weights and then need to change the weights. However, here there is no linear system that is solved. The second place, is in the computing of these approximate weights. However, here we just need to use approximate linear system solvers to approximate leverage scores and we discussed how to do this in Part I [22, ArXiv v3, Section D].

### 8.4 The stable algorithm

We summarize the section as follows:

**Theorem 28.** *Suppose we have an interior point $\vec{x}\in\Omega^0$ for the for the linear program (3.1)and suppose that for any diagonal positive definite matrix $\mathbf{D}$ and vector $\vec{q}$, we can find $\vec{x}$ in $\mathcal{T}_w$ work and $\mathcal{T}_d$ depth such that*

$$\left\|\vec{x}-\left(\mathbf{A}^T\mathbf{D}\mathbf{A}\right)^{-1}\vec{q}\right\|_{\mathbf{A}^T\mathbf{D}\mathbf{A}}\leq\epsilon_{\mathtt{S}}\left\|\left(\mathbf{A}^T\mathbf{D}\mathbf{A}\right)^{-1}\vec{q}\right\|_{\mathbf{A}^T\mathbf{D}\mathbf{A}}$$

*for $\epsilon_{\mathtt{S}}=1/m^k$ for some large constant $k$. Then, using `LPSolve` we can compute $\vec{x}$ such that $\vec{c}^T\vec{x}\leq OPT+\epsilon$, $\|\mathbf{A}^T\vec{x}-\vec{b}\|_{\mathbf{A}^T\mathbf{S}^{-2}\mathbf{A}}\leq\epsilon$, and for all $i\in[n]$ $l_i\leq x_i\leq u_i$ in $\tilde{O}\left(\sqrt{\text{rank}(\mathbf{A})}\left(\mathcal{T}_w+\text{nnz}(\mathbf{A})\right)\log\left(U/\epsilon\right)\right)$ work and $\tilde{O}\left(\sqrt{\text{rank}(\mathbf{A})}\mathcal{T}_d\log\left(U/\epsilon\right)\right)$ depth where $U\stackrel{\text{def}}{=}\max\left(\left\|\frac{\vec{u}-\vec{l}}{\vec{u}-\vec{x}_0}\right\|_\infty,\left\|\frac{\vec{u}-\vec{l}}{\vec{x}_0-\vec{l}}\right\|_\infty,\|\vec{u}-\vec{l}\|_\infty,\|\vec{c}\|_\infty\right)$ and $\mathbf{S}$ is a diagonal matrix with $\mathbf{S}_{ii}=\min(x_i-l_i,u_i-x_i)$.*

*Proof.* Lemma 24 shows that we can maintain $\vec{\eta}$ which is close to $\vec{\eta}^*$ defined in Lemma 22. Lemma 25 shows that using this $\vec{\eta}$, we can compute a more numerically stable step $\vec{x}^{(\text{apx})}$. Hence, this gives us a way to implement $\vec{x}$ step using $\mathtt{S}_{x,w}$. In the previous subsection, we explained how to implement $\vec{w}$ step using $\mathtt{S}_{x,w}$.

To deal with infeasibility, Lemma 26 shows that the stable step $\vec{x}^{(\text{apx})}$ does not hurt the infeasibility $I(\vec{x},\vec{w})$ too much. It is also easy to show the step for $\vec{w}$ does not hurt the infeasibility too much. Whenever $I(\vec{x},\vec{w})>1/m^2$, we improve the feasibility using Lemma 27. This decreases the infeasibility a lot while only taking a very small step as shown in Lemma 27 and consequently it does not hurt the progress $\delta_t$ and $\Phi_\mu$.

Therefore, Theorem 18 can be implemented using the necessary inexact linear algebra. To get the bound on $\|\mathbf{A}^T\vec{x}-\vec{b}\|_{\mathbf{A}^T\mathbf{S}^{-2}\mathbf{A}}$, we use Lemma 3 to show that $\mathbf{S}\preceq\Phi''(\vec{x})$, therefore $\|\mathbf{A}^T\vec{x}-\vec{b}\|_{\mathbf{A}^T\mathbf{S}^{-2}\mathbf{A}}\leq\|\mathbf{A}^T\vec{x}-\vec{b}\|_{\mathbf{A}_x^T\mathbf{A}_x}=I(\vec{x},\vec{w})$. □

For some problems, we need a dual solution instead of the primal. We prove how to do this in the following theorem. In the proof we essentially show that the normal force we maintain for numerical stability is essentially a dual solution.

**Theorem 29.** *Suppose we have an initial $\vec{x}_0$ such that $\mathbf{A}^T\vec{x}_0 = \vec{b}$ and $-1 \leq [\vec{x}_0]_i \leq 1$ and suppose that for any diagonal positive definite matrices $\mathbf{D}$ and vectors $\vec{q}$, we can find $\vec{x}$ from such that*

$$\left\|\vec{x} - \left(\mathbf{A}^T\mathbf{D}\mathbf{A}\right)^{-1}\vec{q}\right\|_{\mathbf{A}^T\mathbf{D}\mathbf{A}} \leq \epsilon_{\mathsf{S}} \left\|\left(\mathbf{A}^T\mathbf{D}\mathbf{A}\right)^{-1}\vec{q}\right\|_{\mathbf{A}^T\mathbf{D}\mathbf{A}}$$

*for $\epsilon_{\mathsf{S}} = 1/m^k$ for sufficiently large constant $k$ in $\mathcal{T}_w$ work and $\mathcal{T}_d$ depth. Then, there is an algorithm that compute $\vec{y}$ such that*

$$\vec{b}^T\vec{y} + \left\|\mathbf{A}\vec{y} + \vec{c}\right\|_1 \leq \min_{\vec{y}} \left(\vec{b}^T\vec{y} + \left\|\mathbf{A}\vec{y} + \vec{c}\right\|_1\right) + \epsilon.$$

*in $\tilde{O}(\sqrt{\mathrm{rank}(\mathbf{A})}\,(\mathcal{T}_w + \mathrm{nnz}(\mathbf{A}))\log(U/\epsilon))$ work and $\tilde{O}(\sqrt{\mathrm{rank}(\mathbf{A})}\mathcal{T}_d\log(U/\epsilon))$ depth where $U \stackrel{\mathrm{def}}{=} \max\left(\left\|\frac{2}{1-\vec{x}}\right\|_\infty, \left\|\frac{2}{\vec{x}+1}\right\|_\infty, \left\|\vec{c}\right\|_\infty\right)$.*

*Proof.* We can use our algorithm to solve the following linear program

$$\min_{\mathbf{A}^T\vec{x}=\vec{b}, -1\leq x_i\leq 1} \vec{c}^T\vec{x}$$

and find $(\vec{x}, \vec{w}, \vec{\eta})$ such that

$$\left\|\frac{t\vec{c} + \vec{w}\vec{\phi}'(\vec{x}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty} \leq \delta \tag{8.1}$$

and

$$I(\vec{x}, \vec{b}) = \left\|\mathbf{A}^T\vec{x} - \vec{b}\right\|_{(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}} \leq \delta$$

for some small $\delta$ and large $t$. To use this to derive a dual solution it seems we need to be very close to the central path. Thus we use our algorithm to compute a central path point for a particular $t$ and then, we do an extra $\tilde{O}(\log(mU/\epsilon))$ iterations to ensure that the error $\delta$ is as small as $1/\mathsf{poly}(mU/\epsilon)$.Let $\vec{y} = -\vec{\eta}/t$, $\vec{\lambda} = \vec{c} + \mathbf{A}\vec{y}$ and $\vec{\tau} = \vec{\lambda} + \frac{1}{t}\vec{w}\vec{\phi}'(\vec{x})$ where $\phi(x) = -\log\cos(\frac{\pi x}{2})$ because the constraints are all $-1 < x_i < 1$. By the definition of $\vec{\lambda}$, we have

$$\left\langle\vec{\lambda}, \vec{x}\right\rangle = \left\langle\vec{c}, \vec{x}\right\rangle + \left\langle\mathbf{A}\vec{y}, \vec{x}\right\rangle.$$

We claim that :

1. $\left|\left\langle\mathbf{A}\vec{y}, \vec{x}\right\rangle - \left\langle\vec{y}, \vec{b}\right\rangle\right| \leq \left(\frac{\delta+2m}{t} + 2mU\right)\delta$.
2. $\left|\left\langle\vec{c}, \vec{x}\right\rangle + \min_{\vec{y}}\left(\left\langle\vec{b}, \vec{y}\right\rangle + \left\|\vec{c} + \mathbf{A}\vec{y}\right\|_1\right)\right| \leq \left(\frac{1}{t} + \delta\right)\mathsf{poly}(mU)$.
3. $\left|\left\langle\vec{\lambda}, \vec{x}\right\rangle + \left\|\vec{c} + \mathbf{A}\vec{y}\right\|_1\right| \leq \left(\delta + \frac{1}{t}\right)\mathsf{poly}(m)$.

Using these claims we can compute a very centered point for $t = \frac{1}{\delta} = (mU)^k/\epsilon$ for sufficiently large $k$ and get the result

$$\left\langle\vec{y}, \vec{b}\right\rangle + \left\|\vec{c} + \mathbf{A}\vec{y}\right\|_1 \leq \min_{\vec{y}}\left(\left\langle\vec{b}, \vec{y}\right\rangle + \left\|\vec{c} + \mathbf{A}\vec{y}\right\|_1\right) + \epsilon.$$

Claim (1): Note that

$$
\begin{aligned}
& \left|\langle \mathbf{A}\vec{y},\vec{x}\rangle - \left\langle \vec{y},\vec{b}\right\rangle\right| \\
\leq\ & \|\vec{y}\|_{(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)} \left\|\mathbf{A}^T\vec{x}-\vec{b}\right\|_{(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x)^{-1}} \\
=\ & \frac{1}{t}\left\|\frac{\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} I(\vec{x},\vec{b}) \\
\leq\ & \frac{1}{t}\left(\left\|\frac{t\vec{c}+\vec{w}\vec{\phi}'(\vec{x})-\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} + \left\|\frac{t\vec{c}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} + \left\|\frac{\vec{w}\vec{\phi}'(\vec{x})}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}}\right)\delta \\
\leq\ & \left(\frac{\delta}{t} + \left\|\frac{\vec{c}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\mathbf{W}} + \frac{\left\|\vec{\mathbb{1}}\right\|_{\mathbf{W}}}{t}\right)\delta.
\end{aligned}
$$

Since $\phi(x) = -\log\cos(\frac{\pi x}{2})$, $\phi''(x) \geq \pi^2/4$. Thus, we have

$$\left|\langle \mathbf{A}\vec{y},\vec{x}\rangle - \left\langle \vec{y},\vec{b}\right\rangle\right| \leq \left(\frac{\delta+2m}{t} + 2mU\right)\delta.$$

Claim (2): From the proof of Theorem 21, we see that

$$\left|\vec{c}^T\vec{x} - \min_{\mathbf{A}^T\vec{x}=\vec{b},-1\leq x_i\leq 1} \vec{c}^T\vec{x}\right| \leq \mathsf{poly}(mU)\left(\frac{1}{t}+\delta\right).$$

Since there is an interior point for $\{\mathbf{A}^T\vec{x}=\vec{b}, -1\leq x_i\leq 1\}$ and the set is bounded, the strong duality shows that

$$
\begin{aligned}
& \min_{\mathbf{A}^T\vec{x}=\vec{b},-1\leq x_i\leq 1} \vec{c}^T\vec{x} \\
=\ & \min_{\vec{x}} \max_{\vec{\lambda}^{(1)}\geq 0,\vec{\lambda}^{(2)}\geq 0,\vec{y}} \vec{c}^T\vec{x} + \left\langle \vec{y},\mathbf{A}^T\vec{x}-\vec{b}\right\rangle + \left\langle \vec{\lambda}^{(1)},\vec{x}-\vec{\mathbb{1}}\right\rangle + \left\langle \vec{\lambda}^{(2)},-\vec{\mathbb{1}}-\vec{x}\right\rangle \\
=\ & \max_{\vec{y},\vec{\lambda}^{(1)}\geq 0,\vec{\lambda}^{(2)}\geq 0} \min_{\vec{x}} \left\langle \vec{c}+\mathbf{A}\vec{y}+\vec{\lambda}^{(1)}-\vec{\lambda}^{(2)},\vec{x}\right\rangle - \left\langle \vec{b},\vec{y}\right\rangle - \left\langle \vec{\lambda}^{(1)}+\vec{\lambda}^{(2)},\vec{\mathbb{1}}\right\rangle \\
=\ & -\min_{\vec{y}} \left\langle \vec{b},\vec{y}\right\rangle + \left\|\vec{c}+\mathbf{A}\vec{y}\right\|_1
\end{aligned}
$$

yielding the claim.

Claim (3): Recall that $\vec{\tau} = \vec{\lambda} + \frac{1}{t}\vec{w}\vec{\phi}'(\vec{x})$. Hence, we have

$$\tau_i = \lambda_i + \frac{\pi}{2t}w_i\tan(\frac{\pi}{2}x_i).$$

Therefore, we have

$$x_i = \frac{2}{\pi}\tan^{-1}\left(\frac{2t}{\pi w_i}(\tau_i-\lambda_i)\right).$$

Thus, we have

$$
\begin{aligned}
\left\langle \vec{\lambda},\vec{x}\right\rangle &= \sum_i \lambda_i\frac{2}{\pi}\tan^{-1}\left(\frac{2t}{\pi w_i}(\tau_i-\lambda_i)\right) \\
&= -\frac{2}{\pi}\sum_i \lambda_i\tan^{-1}\left(\frac{2t}{\pi w_i}(\lambda_i-\tau_i)\right).
\end{aligned}
$$

Thus, Lemma 39 shows that

$$-\left\|\vec{\lambda}\right\|_1 \leq \left\langle \vec{\lambda}, \vec{x} \right\rangle \leq -\left\|\vec{\lambda}\right\|_1 + 2\left\|\vec{\tau}\right\|_1 + \frac{\left\|\vec{w}\right\|_1}{t}.$$

Recall that $\left\|\vec{w}\right\|_1 = O(m)$. Also, the bound (8.1) is equivalent to

$$|\tau_i| \leq \frac{\delta w_i \sqrt{\phi''(x_i)}}{t} \leq \frac{\delta \sqrt{\phi''(x_i)}}{t}.$$

Lemma 31 shows that the slack of central point is larger than $\mathsf{poly}(m)/t$. Therefore, Lemma 3 shows that $\frac{\delta\sqrt{\phi''(x_i)}}{t} \leq \mathsf{poly}(m)\delta$. Therefore $\left\|\vec{\tau}\right\|_1 = \mathsf{poly}(m)\delta$. Using $\vec{\lambda} = \vec{c} + \mathbf{A}\vec{y}$, we have $\left|\left\langle \vec{\lambda}, \vec{x} \right\rangle + \left\|\vec{c} + \mathbf{A}\vec{y}\right\|_1\right| \leq \mathsf{poly}(m)\left(\delta + \frac{1}{t}\right).$ □

*Remark* 30. Note how this algorithm uses the initial point to certify that $\min_{\vec{y}} \vec{b}^T\vec{y} + \left\|\mathbf{A}\vec{y} + \vec{c}\right\|_1$ is bounded. As usual, one can use standard technique to avoid the requirement on the initial point.

### 8.5 Well conditioned

For many problems, the running time of linear system solvers depend on the condition number and/or how fast the linear systems change from iteration to iteration. The following lemma shows that our interior point method enjoys many properties frequently exploited in other interior point methods and therefore is amenable to different techniques for improving iteration costs. In particular, here we bound the condition number of the matrices involved which in turn, allows us to use the fast M matrix solver in next section.

There are two key lemmas we prove in this section. First, in Lemma 31 we bound how close the weighted central path can go to the boundary of the polytope. This allows us to reason about how ill-conditioned the linear system we need to solver become over the course of the algorithm. Weshows that if the slacks, i.e. distances to the boundary of the polytope, of the initial point are polynomially bounded below and if we only change the weight multiplicatively by a polynomial factor, then the slacks of the new weighted central path point is still polynomially bounded below. Second, in Lemma 32 we bound how much the linear systems can change over the course of our algorithm.

**Lemma 31.** *For all $\vec{w} \in \mathbb{R}^m_{>0}$ and $t > 0$ let $\vec{x}_{t,\vec{w}} = \arg\min f_t(\vec{x}, \vec{w})$. For all $\vec{x} \in \Omega^0$ and $i \in [m]$ let $s_i(\vec{x})$ denote the slack of constraint $i$, i.e. $s_i(\vec{x}) \stackrel{\text{def}}{=} \min\{u_i - x_i, x_i - l_i\}$. For any $a, b > 0$ and $\vec{w}^{(1)}, \vec{w}^{(2)} \in \mathbb{R}^m_{>0}$ and $i \in [m]$ we have*

$$s_i\left(\vec{x}_{b,\vec{w}^{(2)}}\right) \geq \min\left\{\frac{\left(\min_{j\in[m]} w_j^{(2)}\right)\cdot\left(\min_{j\in[m]} s_j\left(\vec{x}_{b,\vec{w}^{(1)}}\right)\right)}{2\left(\frac{b}{a}\left\|\vec{w}^{(1)}\right\|_1 + \left\|\vec{w}^{(2)}\right\|_1\right)}, 1\right\} s_i\left(\vec{x}_{a,\vec{w}^{(1)}}\right). \tag{8.2}$$

*Proof.* Fix an arbitrary $i \in [m]$ and consider the straight line from $\vec{x}_{a,\vec{w}^{(1)}}$ to $\vec{x}_{b,\vec{w}^{(2)}}$. If this line never reaches a point $\vec{y}$ such that $s_i(\vec{y}) = 0$ then $s_i(\vec{x}_{b,\vec{w}^{(2)}}) \geq s_i(\vec{x}_{a,\vec{w}^{(2)}})$ and clearly (8.2). Otherwise, we can parameterize the the straight line by $\vec{p}(t)$ such that $\vec{p}(-1) = \vec{x}_{a,\vec{w}^{(1)}}$, $s_i(\vec{p}(0)) = 0$, and $\vec{p}(-\theta) = \vec{x}_{b,\vec{w}^{(2)}}$ for some $\theta \in [0, 1]$. Since $\phi_i(p(t)) \to \infty$ as $t \to 0$, Lemma 3 shows that

$$\left.\frac{d^2\phi_i}{dt^2}\right|_t \geq \frac{1}{t^2}.$$

Integrating then yields that.

$$\begin{aligned}\left.\frac{d\phi_i}{dt}\right|_{t=-\theta} &\geq \left.\frac{d\phi_i}{dt}\right|_{t=-1}+\int_{-1}^{-\theta}\frac{1}{t^2}dt\\ &= \left.\frac{d\phi_i}{dt}\right|_{t=-1}+\left(\frac{1}{\theta}-1\right).\end{aligned}$$

Since each of the $\phi_j$ is convex, we have

$$\sum_{j\in[m]} w_j^{(2)}\left.\frac{d\phi_j}{dt}\right|_{t=-\theta} \geq \sum_{j\in[m]} w_j^{(2)}\left.\frac{d\phi_j}{dt}\right|_{t=-1}+\left(\min_{j\in[m]} w_j^{(2)}\right)\cdot\left(\frac{1}{\theta}-1\right).$$

Using the optimality condition of $\vec{x}_{b,\vec{w}^{(2)}}$ and the optimality condition of $\vec{x}_{a,\vec{w}^{(1)}}$, we have

$$\frac{b}{a}\sum_{j\in[m]} w_j^{(1)}\left.\frac{d\phi_i}{dt}\right|_{t=-1} \geq \sum_{j\in[m]} w_j^{(2)}\left.\frac{d\phi_i}{dt}\right|_{t=-1}+\left(\min_{j\in[m]} w_j^{(2)}\right)\left(\frac{1}{\theta}-1\right).$$

Hence,

$$\left(\frac{b}{a}\left\|\vec{w}^{(1)}\right\|_1+\left\|\vec{w}^{(2)}\right\|_1\right)\max_{j\in[m]}\left|\left(\left.\frac{d\phi_j}{dt}\right|_{t=-1}\right)\right| \geq \left(\min_{j\in[m]} w_j^{(2)}\right)\left(\frac{1}{\theta}-1\right).$$

Applying Lemma 4 again yields that for all $j\in[m]$

$$\left|\left(\left.\frac{d\phi_j}{dt}\right|_{t=-1}\right)\right| s_j\left(\vec{x}_{b,\vec{w}^{(1)}}\right)\leq 1.$$

Thus, we have

$$\frac{b}{a}\left\|\vec{w}^{(1)}\right\|_1+\left\|\vec{w}^{(2)}\right\|_1 \geq \left(\min_{j\in[m]} w_j^{(2)}\right)\left(\frac{1}{\theta}-1\right)\min_j s_j\left(\vec{x}_{b,\vec{w}^{(1)}}\right).$$

Hence,

$$\theta \geq \frac{\left(\min_{j\in[m]} w_j^{(2)}\right)\cdot\left(\min_{j\in[m]} s_j\left(\vec{x}_{b,\vec{w}^{(1)}}\right)\right)}{2\left(\frac{b}{a}\left\|\vec{w}^{(1)}\right\|_1+\left\|\vec{w}^{(2)}\right\|_1\right)}.$$

Since $i\in[m]$ was arbitrary we have the desired result. □

**Lemma 32.** *Using the notations and assumptions in Theorem 28 or Theorem 29 let $\mathbf{A}^T\mathbf{D}_k\mathbf{A}$ be the $k^{th}$ linear system that is used in the algorithm* `LPSolve`*. For all $k\geq 1$, we have the following:*

1. *The condition number of $\mathbf{D}_k$ is bounded by $\mathsf{poly}(mU/\epsilon)$, i.e., $\mathsf{poly}(\epsilon/(mU))\mathbf{A}^T\mathbf{A}\preceq\mathbf{A}^T\mathbf{D}_k\mathbf{A}\preceq\mathsf{poly}(mU/\epsilon)\mathbf{A}^T\mathbf{A}$*
2. *$\left\|\log(\mathbf{D}_{k+1})-\log(\mathbf{D}_k)\right\|_\infty\leq 1/10$.*
3. *$\left\|\log(\mathbf{D}_{k+1})-\log(\mathbf{D}_k)\right\|_{\mathbf{\Sigma}_{\mathbf{A}}(\vec{d}_k)}\leq 1/10$.*

*Proof.* During the algorithm, the matrix we need to solve is of the form $\mathbf{A}^T\mathbf{D}\mathbf{A}$ where $\mathbf{D}=\mathbf{W}^{-1}\mathbf{\Phi}''(\vec{x})^{-1}$. We know that $\frac{n}{2m}\leq\vec{w}_i\leq 3$. In the proof of Theorem 18, we showed that $\vec{\phi}_i''(\vec{x})\geq\frac{1}{U^2}$. Also, Lemma 31 shows that the slacks is never too small and hence $\vec{\phi}_i''(\vec{x})$ is upper bounded by $\mathsf{poly}(mU/\epsilon)$. Thus, the condition number of $\mathbf{D}$ is bounded by $\mathsf{poly}(mU/\epsilon)$.

Now, we bound the changes of $\mathbf{D}$ by bound the changes of $\mathbf{\Phi}''(\vec{x})$ and the changes of $\mathbf{W}$ separately. For the changes of $\mathbf{\Phi}''(\vec{x})$, (4.5) shows that $\left\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\right\|_{\vec{w}+\infty} \leq \left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}+\infty}\delta_t$. Since $\left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}+\infty} \leq 2$ and $\delta_t \leq 1/80$, we have

$$\left\|\sqrt{\vec{\phi}''(\vec{x})}(\vec{x}^{(\mathrm{new})}-\vec{x})\right\|_{\vec{w}+\infty} = \left\|\sqrt{\vec{\phi}''(\vec{x})}\vec{h}_t(\vec{x},\vec{w})\right\|_{\vec{w}+\infty} \leq 1/40.$$

Using this on Lemma 3, we have

$$\begin{aligned}\left\|\log\left(\vec{\phi}''(\vec{x}^{(\mathrm{new})})\right)-\log\left(\vec{\phi}''(\vec{x})\right)\right\|_{\vec{w}+\infty} &\leq \left(1-\left\|\sqrt{\vec{\phi}''(\vec{x})}(\vec{x}^{(\mathrm{new})}-\vec{x})\right\|_{\vec{w}+\infty}\right)^{-1}-1\\ &\leq 1/36.\end{aligned}$$

Since $\vec{w}_i \geq \frac{1}{2}\vec{\sigma}_i$ for all $i$, we have

$$\left\|\log\left(\vec{\phi}''(\vec{x}^{(\mathrm{new})})\right)-\log\left(\vec{\phi}''(\vec{x})\right)\right\|_{\vec{\sigma}+\infty} \leq 1/20. \tag{8.3}$$

For the changes of $\mathbf{W}$, we look at the description of `centeringInexact`. The algorithm ensures the changes of $\log(\vec{w})$ is in $(1+\epsilon)U$ where $U=\{\vec{x}\in\mathbb{R}^m \mid \|\vec{x}\|_{\vec{w}+\infty} \leq \left(1-\frac{7}{8c_k}\right)\delta_t\}$. Since $\delta_t \leq 1/80$ and $\vec{w}_i \geq \frac{1}{2}\vec{\sigma}_i$ for all $i$, we get that

$$\left\|\log\left(\vec{w}^{(\mathrm{new})}\right)-\log\left(\vec{w}\right)\right\|_{\vec{\sigma}+\infty} \leq 1/20. \tag{8.4}$$

The assertion (2) and (3) follows from (8.3) and (8.4). $\square$

# 9 Generalized Minimum Cost Flow

In this section we show how to use the interior point method in Section 7 to solve the maximum flow problem in time $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$, to solve the minimum cost flow problem in time, $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$, and to compute $\epsilon$-approximate solutions to the lossy generalized minimum cost flow problem in time $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$. Our algorithm for the generalized minimum cost flow problem is essentially the same as our algorithm for the simpler specific case of minimum cost flow and maximum flow and therefore, we present the algorithm for the generalized minimum cost flow problem directly. [10]

The generalized minimum cost flow problem [5] is as follows. Let $G=(V,E)$ be a connected directed graph where each edge $e$ has capacity $c_e>0$ and multiplier $1\geq\gamma_e>0$. For each edge $e$, there can be only at most $c_e$ units of flow on that edge and the flow on that edge must be non-negative. Also, for each unit of flow entering edge $e$, there are only $\gamma_e$ units of flow going out. The generalized maximum flow problem is to compute how much flow can be sent into $t$ given a unlimited source $s$. The generalized minimum cost flow is to ask what is the minimum cost of sending the maximum flow given the cost of each edge is $q_e$. The maximum flow and the minimum cost flow are the case with $\gamma_e=1$ for all edges $e$.

Since the generalized minimum cost flow includes all of these cases, we focus on this general formulation. The problem can be written as the following linear program

$$\min_{\vec{0}\leq\vec{x}\leq\vec{c}} \vec{q}^T\vec{x} \text{ such that } \mathbf{A}\vec{x}=F\vec{1}_t$$

[10] Our algorithm could be simplified slightly for the simpler cases and the dependence on polylogarithmic factors for these problems could possibly be improved.

where $F$ is the generalized maximum flow value, $\vec{\mathbb{1}}_t$ is a indicator vector of size $(n-1)$ that is non-zero at vertices $t$ and $\mathbf{A}$ is a $|V\backslash\{s\}| \times |E|$ matrix such that for each edge $e$, we have

$$\begin{aligned}\mathbf{A}(e_{head}, e) &= \gamma(e),\\ \mathbf{A}(e_{tail}, e) &= -1.\end{aligned}$$

In order words, the constraint $\mathbf{A}x = F\vec{\mathbb{1}}_t$ requires the flow to satisfies the flow conversation at all vertices except $s$ and $t$ and requires it flows $F$ unit of flow to $t$. We assume $c_e$ are integer and $\gamma_e$ is a rational number. Let $U$ be the maximum of $c_e$, $q_e$, the numerator of $\gamma_e$ and the denominator of $\gamma_e$. For the generalized flow problems, getting an efficient exact algorithm is difficult and we aim for approximation algorithms only.

**Definition 33.** We call a flow an $\epsilon-$approximate generalized maximum flow if it is a flow satisfies the flow conservation and the flow value is larger than maximum flow value minus $\epsilon$. We call a flow is an $\epsilon-$approximate generalized minimum cost maximum flow if it is an $\epsilon$-approximate maximum flow and has cost not greater than the minimum cost maximum flow value.

Note that $\operatorname{rank}(\mathbf{A}) = n-1$ because the graph is connected and hence our algorithm takes only $\tilde{O}(\sqrt{n}L)$ iterations. Therefore, the problems remaining are to compute $L$ and bound how much time is required to solve the linear systems involved. However, $L$ is large in the most general setting and hence we cannot use the standard theory to say how to get the initial point, how to round to the vertex. Furthermore, the condition number of $\mathbf{A}^T\mathbf{A}$ can be very bad.

In [5], they used dual path following to solve the generalized minimum cost flow problem with the caveats that the dual polytope is not bounded, the problem of getting the initial flow, the problem of rounding it to the a feasible flow. We use there analysis to formulate the problem in a manner more amenable to our algorithms. Since we are doing the primal path following, we will state a reformulation of the LP slightly different.

**Theorem 34** ([5])**.** *Given a directed graph $G$. We can find a new directed graph $\tilde{G}$ with $O(m)$ edges and $O(n)$ vertices in $\tilde{O}(m)$ time such that the modified linear program*

$$\min_{0\leq x_i\leq c_i, 0\leq y_i\leq 4mU^2, 0\leq z_i\leq 4mU^2} \vec{q}^T\vec{x} + \frac{256m^5U^5}{\epsilon^2}\left(\vec{\mathbb{1}}^T\vec{y} + \vec{\mathbb{1}}^T\vec{z}\right) \text{ such that } \mathbf{A}\vec{x} + \vec{y} - \vec{z} = F\vec{\mathbb{1}}_t$$

*satisfies the following conditions:*

1. $\vec{x} = \frac{c}{2}\vec{\mathbb{1}}$, $\vec{y} = 2mU^2\vec{\mathbb{1}} - (\mathbf{A}\frac{c}{2}\vec{\mathbb{1}})^- + F\vec{\mathbb{1}}_t$, $z = 2mU^2\vec{\mathbb{1}} + (\mathbf{A}\frac{c}{2}\vec{\mathbb{1}})^+$ *is an interior point of the linear program.*
2. *Given any $(\vec{x}, \vec{y}, \vec{z})$ such that $\left\|\mathbf{A}\vec{x} + \vec{y} - \vec{z}\right\|_2 \leq \frac{\epsilon^2}{128m^2n^2U^3}$ and with cost value within $\frac{\epsilon^2}{128m^2n^2U^3}$ of the optimum. Then, one can compute an $\epsilon$-approximate minimum cost maximum flow for graph $G$ in time $\tilde{O}(m)$.*
3. *The linear system of the linear program is well-conditioned, i.e., the condition number of $\begin{bmatrix} \mathbf{A} & \mathbf{I} & -\mathbf{I} \end{bmatrix}\begin{bmatrix} \mathbf{A}^T \\ \mathbf{I} \\ -\mathbf{I} \end{bmatrix}$ is $O(mU)$.*
4. *The linear system of the linear program can be solve in nearly linear time, i.e. for any diagonal matrix $\mathbf{S}$ with condition number $\kappa$ and vector $b$, it takes $\tilde{O}\left(m\log\left(\frac{\kappa U}{\delta}\right)\right)$ time to find $x$ such that*

$$\left\|x - \mathbf{L}^{-1}b\right\|_{\mathbf{L}} \leq \delta\left\|x\right\|_{\mathbf{L}}$$

*where* $\mathbf{L} = \begin{bmatrix} \mathbf{A} & \mathbf{I} & -\mathbf{I} \end{bmatrix} \mathbf{S} \begin{bmatrix} \mathbf{A}^T \\ \mathbf{I} \\ -\mathbf{I} \end{bmatrix}$.

The main difference between what stated in [5] and here is that

1. Our linear program solver can support constraint $l_i \leq x_i \leq u_i$ and hence we do not need to split the flow variable to positive part and negative part.

2. Our linear program solver is primal path following and hence we add the constraint $y_i \leq 4mU^2$ and $z_i \leq 4mU^2$. Since the maximum flow value is at most $mU^2$, it does not affect the optimal solution of the linear program.

3. We remove the variable $\mathbf{x}_3$ in [5] because the purpose of that is to make the dual polytope is bounded and we do not need it here.

Using the reduction mentioned above, one can obtain the promised generalized minimum cost flow algorithm.

**Theorem 35.** *There is a randomized algorithm to compute an $\epsilon-$approximate generalized minimum cost maximum flow in $\tilde{O}(\sqrt{n}\log^{O(1)}(U/\epsilon))$ depth $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U/\epsilon))$ total work (see Definition 33). Furthermore, there is an algorithm to compute an exact standard minimum cost maximum flow in $\tilde{O}(\sqrt{n}\log^{O(1)}(U))$ depth and $\widetilde{O}(m\sqrt{n}\log^{O(1)}(U))$ total work.*

*Proof.* Using the reduction above and Theorem 18, we get an algorithm of generalized minimum cost flow by solving $\widetilde{O}(\sqrt{n})$ linear systems to $\tilde{\mathcal{O}}(1)$ bit accuracy and the condition number of those systems are $\mathsf{poly}(mU/\epsilon)$. In [5], they showed that the linear system involved can be reduced to $\tilde{O}(\log(U/\epsilon))$ many Laplacian systems and hence we can use a recent nearly linear work polylogarithmic depth Laplacian system solver of Spielman and Peng [31]. In total, it takes $\widetilde{O}(m\log^{O(1)}(\frac{U}{\epsilon}))$ time to solve each systems.

For the standard minimum cost maximum flow problem, it is known that the solution set is a convex polytope with integer coordinates and we can use Isolation lemma to make sure there is unique minimum. Hence, we only need to take $\epsilon = \mathsf{poly}(1/mU)$ and round the solution to the closest integer. See Section 3.5 in [5] for details. $\square$

## 10 Acknowledgments

We thank Yan Kit Chim, Andreea Gane, and Jonathan A. Kelner for many helpful conversations. This work was partially supported by NSF awards 0843915 and 1111109, NSF Graduate Research Fellowship (grant no. 1122374) and Hong Kong RGC grant 2150701.

## References

[1] Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. Network flows: theory, algorithms, and applications. 1993.

[2] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[3] András A Benczúr and David R Karger. Approximating st minimum cuts in õ (n 2) time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 47–55. ACM, 1996.

[4] Paul Christiano, Jonathan A Kelner, Aleksander Madry, Daniel A Spielman, and Shang-Hua Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 273–282. ACM, 2011.

[5] Samuel I Daitch and Daniel A Spielman. Faster approximate lossy generalized flow via interior point algorithms. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 451–460. ACM, 2008.

[6] Jack Edmonds and Richard M Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

[7] Shimon Even and R Endre Tarjan. Network flow and testing graph connectivity. *SIAM journal on computing*, 4(4):507–518, 1975.

[8] Robert M Freund and Michael J Todd. Barrier functions and interior-point algorithms for linear programming with zero-, one-, or two-sided bounds on the variables. *Mathematics of Operations Research*, 20(2):415–440, 1995.

[9] Zvi Galil and Éva Tardos. An o (n 2 (m+ n log n) log n) min-cost flow algorithm. *Journal of the ACM (JACM)*, 35(2):374–386, 1988.

[10] Andrew V. Goldberg and Satish Rao. Beyond the flow decomposition barrier. *J. ACM*, 45(5):783–797, 1998.

[11] Andrew V Goldberg and Robert E Tarjan. Finding minimum-cost circulations by successive approximation. *Mathematics of Operations Research*, 15(3):430–466, 1990.

[12] David Karger and Matthew Levine. Random sampling in residual graphs. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 63–66. ACM, 2002.

[13] David R Karger. Better random sampling algorithms for flows in undirected graphs. In *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 490–499. Society for Industrial and Applied Mathematics, 1998.

[14] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[15] Alexander V Karzanov. On finding a maximum flow in a network with special structure and some applications. *Matematicheskie Voprosy Upravleniya Proizvodstvom*, 5:81–94, 1973.

[16] Jonathan A Kelner, Yin Tat Lee, Lorenzo Orecchia, and Aaron Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. In *SODA*, pages 217–226. SIAM, 2014.

[17] Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A Simple, Combinatorial Algorithm for Solving SDD Systems in Nearly-Linear Time. January 2013.

[18] Ioannis Koutis, Gary L. Miller, and Richard Peng. Approaching optimality for solving SDD systems. In *Proceedings of the 51st Annual Symposium on Foundations of Computer Science*, 2010.

[19] Ioannis Koutis, Gary L. Miller, and Richard Peng. A nearly-m log n time solver for sdd linear systems. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 590 –598, oct. 2011.

[20] Yin Tat Lee, Satish Rao, and Nikhil Srivastava. A new approach to computing maximum flows using electrical flows. In *The 45th ACM Symposium on Theory of Computing (STOC)*, pages 755–764, 2013.

[21] Yin Tat Lee and Aaron Sidford. Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems. In *The 54th Annual Symposium on Foundations of Computer Science (FOCS)*, 2013.

[22] Yin Tat Lee and Aaron Sidford. Path finding i: Solving linear programs with \˜ o (sqrt(rank)) linear system solves. *arXiv preprint arXiv:1312.6677*, 2013.

[23] Mu Li, Gary L Miller, and Richard Peng. Iterative row sampling. 2012.

[24] Aleksander Madry. Fast approximation algorithms for cut-based problems in undirected graphs. In *FOCS*, pages 245–254, 2010.

[25] Aleksander Madry. Navigating central path with electrical flows: from flows to matchings, and back. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science*, 2013.

[26] Yu Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume I. 2003.

[27] Yurii Nesterov and Arkadii Semenovich Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Society for Industrial and Applied Mathematics, 1994.

[28] James B Orlin. Genuinely polynominal simplex and non-simplex algorithms for the minimum cost flow problem. 1984.

[29] James B Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations research*, 41(2):338–350, 1993.

[30] James B Orlin. Max flows in o (nm) time, or better. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 765–774. ACM, 2013.

[31] Richard Peng and Daniel A Spielman. An efficient parallel solver for sdd linear systems. *arXiv preprint arXiv:1311.3286*, 2013.

[32] Alexander Schrijver. On the history of the transportation and maximum flow problems. *Mathematical Programming*, 91(3):437–445, 2002.

[33] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

[34] Jonah Sherman. Nearly maximum flows in nearly linear time. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science*, 2013.

[35] Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

[36] Daniel A Spielman and Shang-Hua 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*, pages 81–90. ACM, 2004.

[37] Éva Tardos. A strongly polynomial minimum cost circulation algorithm. *Combinatorica*, 5(3):247–255, 1985.

# A Glossary

Here we summarize various linear programming specific notation that we use throughout the paper. For many quantities we included the typical order of magnitude as they appear during our algorithms.

- Linear program related: constraint matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ , cost vector $\vec{c} \in \mathbb{R}^m$, constraint vector $\vec{b} \in \mathbb{R}^n$, solution $\vec{x} \in \mathbb{R}^m$, weights of constraints $\vec{w} \in \mathbb{R}^m$ where $m$ is the number of variables and $n$ is the number of constraints.

- Matrix version of variables: $\mathbf{S}$ is the diagonal matrix corresponds to $\vec{s}$, $\mathbf{W}$ corresponds to $\vec{w}$, $\mathbf{\Phi}$ corresponds to $\phi$.

- Penalized objective function (4.1): $f_t(\vec{x},\vec{w}) = t\cdot \vec{c}^T\vec{x} + \sum_{i\in[m]} \vec{w}_i\phi_i(\vec{x}_i)$.

- Barrier functions (Sec 3.1): For $[l,\infty)$, we use $\phi(x) = -\log(x-l)$. For $(-\infty,u]$, we use $\phi(x) = -\log(u-x)$. For $[l,u]$, we use $\phi(x) = -\log(ax+b)$ where $a = \frac{\pi}{u-l}$ and $b = -\frac{\pi}{2}\frac{u+l}{u-l}$.

- The projection matrix $\mathbf{P}_{\vec{x},\vec{w}}$ (4.3): $\mathbf{P}_{\vec{x},\vec{w}} = \mathbf{I} - \mathbf{W}^{-1}\mathbf{A}_x\left(\mathbf{A}_x^T\mathbf{W}^{-1}\mathbf{A}_x\right)^{-1}\mathbf{A}_x^T$ where $\mathbf{A}_x \stackrel{\text{def}}{=} \mathbf{\Phi}''(\vec{x})^{-1/2}\mathbf{A}$.

- Newton step (4.2): $\vec{h}_t(\vec{x},\vec{w}) = -\mathbf{\Phi}''(\vec{x})^{-1/2}\mathbf{P}_{\vec{x},\vec{w}}\mathbf{W}^{-1}\mathbf{\Phi}''(\vec{x})^{-1/2}\nabla_x f_t(\vec{x},\vec{w})$.

- The mixed norm (4.4): $\|\vec{y}\|_{\vec{w}+\infty} = \left\|\vec{y}\right\|_\infty + C_{\text{norm}}\left\|\vec{y}\right\|_{\mathbf{W}}$ where $C_{\text{norm}} \approx \mathsf{polylog}(m)$.

- Centrality (4.6): $\delta_t(\vec{x},\vec{w}) = \min_{\vec{\eta}\in\mathbb{R}^n}\left\|\frac{\nabla_x f_t(\vec{x},\vec{w}) - \mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|_{\vec{w}+\infty} \approx \frac{1}{\mathsf{polylog}(m)}$.

- Properties of weight function (Def 6): **size** $c_1(\vec{g}) = \left\|\vec{g}(\vec{x})\right\|_1 \approx \operatorname{rank}(\mathbf{A})$, **slack sensitivity** $c_\gamma(\vec{g}) = \left\|\mathbf{P}_{\vec{x},\vec{w}}\right\|_{\vec{w}+\infty} \approx 1 + \frac{1}{\mathsf{polylog}(m)}$, **step consistency** $c_\delta(\vec{g}) \approx 1 - \frac{1}{\mathsf{polylog}(m)}$.

- Difference between $\vec{g}$ and $\vec{w}$ (5.4): $\vec{\Psi}(\vec{x},\vec{w}) = \log(\vec{g}(\vec{x})) - \log(\vec{w})$.

- Potential function for tracing 0 (Thm 11): $\Phi_\mu(\vec{x}) = e^{\mu x} + e^{-\mu x} \approx \mathsf{poly}(m)$.

- The weight function proposed (6.1):

$$\vec{g}(\vec{x}) = \operatorname*{arg\,min}_{\vec{w}\in\mathbb{R}^m_{>0}} \hat{f}(\vec{x},\vec{w}) \quad \text{where} \quad \hat{f}(\vec{x},\vec{w}) = \vec{\mathbb{1}}^T\vec{w} + \frac{1}{\alpha}\log\det\left(\mathbf{A}_x^T\mathbf{W}^{-\alpha}\mathbf{A}_x\right) - \beta\sum_i \log w_i$$

  where $\mathbf{A}_x = (\mathbf{\Phi}''(\vec{x}))^{-1/2}\mathbf{A}$, $\alpha \approx 1 + 1/\log_2\left(\frac{m}{\operatorname{rank}(\mathbf{A})}\right)$, $\beta \approx \operatorname{rank}(\mathbf{A})/m$.

## B Appendix

### B.1 Technical Lemmas

**Lemma 36.** *For any norm* $\|\cdot\|$ *and* $\|\vec{y}\|_Q \stackrel{\text{def}}{=} \min_{\vec{\eta}\in\mathbb{R}^n}\left\|\vec{y}-\frac{\mathbf{A}\vec{\eta}}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|$, *we have*

$$\|\vec{y}\|_Q \leq \|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\| \leq \|\mathbf{P}_{\vec{x},\vec{w}}\| \cdot \|\vec{y}\|_Q .$$

*Proof.* By definition $\mathbf{P}_{\vec{x},\vec{w}}\vec{y} = \vec{y} - \frac{\mathbf{A}\vec{\eta}_y}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}$ for some $\vec{\eta}_y \in \mathbb{R}^n$. Consequently,

$$\|\vec{y}\|_Q \quad = \quad \min_{\vec{\eta}\in\mathbb{R}^n}\left\|\vec{y}-\frac{\mathbf{A}\eta}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\| \leq \|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\| .$$

On the other hand, let $\vec{\eta}_q$ by such that such that $\|\vec{y}\|_Q = \left\|\vec{y}-\frac{\mathbf{A}\vec{\eta}_q}{\vec{w}\sqrt{\vec{\phi}''(\vec{x})}}\right\|$. Then, since $\mathbf{P}_{\vec{x},\vec{w}}\mathbf{W}^{-1}(\mathbf{\Phi}'')^{-1/2}\mathbf{A} = \mathbf{0}$, we have

$$\|\mathbf{P}_{\vec{x},\vec{w}}\vec{y}\| \quad = \quad \left\|\mathbf{P}_{\vec{x},\vec{w}}\left(\vec{y}-\frac{\mathbf{A}\vec{\eta}_q}{\vec{w}\sqrt{\vec{\phi}''}}\right)\right\| \leq \|\mathbf{P}_{\vec{x},\vec{w}}\| \cdot \left\|\vec{y}-\frac{\mathbf{A}\vec{\eta}_q}{\vec{w}\sqrt{\vec{\phi}''}}\right\| = \|\mathbf{P}_{\vec{x},\vec{w}}\| \cdot \|\vec{y}\|_Q .$$

□

**Lemma 37** (Log Notation [22, Appendix])**.** *Suppose* $|\log(a) - \log(b)| = \epsilon \leq 1/2$ *then* $\left|\frac{a-b}{b}\right| \leq \epsilon+\epsilon^2$. *If* $\left|\frac{a-b}{b}\right| = \epsilon \leq 1/2$, *then* $|\log(a) - \log(b)| \leq \epsilon + \epsilon^2$.

**Lemma 38** ([22, Appendix])**.** *For any projection matrix* $\mathbf{P} \in \mathbb{R}^{m\times m}$, $\mathbf{\Sigma} = \mathbf{diag}(\mathbf{P})$, $i, j \in [m]$, $\vec{x} \in \mathbb{R}^m$, *and* $\vec{w} \in \mathbb{R}^m_{>0}$ *we have*

- $\mathbf{\Sigma}_{ii} = \sum_{j\in[m]} \mathbf{P}_{ij}^{(2)}$,
- $\mathbf{0} \preceq \mathbf{P}^{(2)} \preceq \mathbf{\Sigma} \preceq \mathbf{I}$,
- $\mathbf{P}_{ij}^{(2)} \leq \mathbf{\Sigma}_{ii}\mathbf{\Sigma}_{jj}$,
- $|\vec{\mathbb{1}}_i^T \mathbf{P}^{(2)}\vec{x}| \leq \mathbf{\Sigma}_{ii}\|\vec{x}\|_{\mathbf{\Sigma}}$.
- $\nabla_{\vec{w}} \log\det(\mathbf{A}^T\mathbf{W}\mathbf{A}) = \mathbf{\Sigma}_{\mathbf{A}}(\vec{w})\vec{w}^{-1}$.
- $\mathbf{J}_{\vec{w}}(\vec{\sigma}_{\mathbf{A}}(\vec{w})) = \mathbf{\Lambda}_{\mathbf{A}}(\vec{w})\mathbf{W}^{-1}$.

**Lemma 39.** *For any* $x, \epsilon$ *and* $\lambda > 0$, *we have*

$$\frac{\pi}{2}|x| - \pi\epsilon - \frac{1}{\lambda} \leq x\tan^{-1}(\lambda(x+\epsilon)) \leq \frac{\pi}{2}|x| .$$

*Proof.* We first consider the case $\epsilon = 0$. Note that

$$x\tan^{-1}(\lambda x) \leq \frac{\pi}{2}|x| .$$

Also, we note that

$$x\tan^{-1}(\lambda x) \geq |x|\left(\frac{\pi}{2} - \frac{1}{\lambda|x|}\right)$$

because

$$\left|\tan(\frac{\pi}{2} - \frac{1}{\lambda|x|})\right| = \left|\frac{\cos(\frac{1}{\lambda|x|})}{\sin(\frac{1}{\lambda|x|})}\right| \leq \lambda|x|.$$

Hence, we have

$$\frac{\pi}{2}|x| - \frac{1}{\lambda} \leq x\tan^{-1}(\lambda x) \leq \frac{\pi}{2}|x|.$$

For $\epsilon \neq 0$, we have

$$\frac{\pi}{2}|x+\epsilon| - \frac{1}{\lambda} \leq (x+\epsilon)\tan^{-1}(\lambda(x+\epsilon)) \leq \frac{\pi}{2}|x+\epsilon|.$$

Thus, we have

$$\frac{\pi}{2}|x| - \pi\epsilon - \frac{1}{\lambda} \leq x\tan^{-1}(\lambda(x+\epsilon)) \leq \frac{\pi}{2}|x|.$$

□

## B.2 Projection on Mixed Norm Ball

In the [22], we studied the following problem:

$$\max_{\|\vec{x}\|_2 \leq 1, -l_i \leq x_i \leq l_i} \langle \vec{a}, \vec{x} \rangle \tag{B.1}$$

for some given vector $\vec{a}$ and $\vec{l}$ in $\mathbb{R}^m$. We proved that the following algorithm outputs a solution of (B.1) in depth $\tilde{O}(1)$ and work $\tilde{O}(m)$.

| $\vec{x} = \texttt{projectOntoBallBoxParallel}(\vec{a}, \vec{l})$ |
|---|
| 1. Set $\vec{a} = \vec{a}/\|\vec{a}\|_2$. |
| 2. Sort the coordinate such that $|a_i|/l_i$ is in descending order. |
| 3. Precompute $\sum_{k=0}^{i} l_k^2$ and $\sum_{k=0}^{i} a_k^2$ for all $i$. |
| 4. Find the first $i$ such that $\frac{1-\sum_{k=0}^{i} l_k^2}{1-\sum_{k=0}^{i} a_k^2} \leq \frac{l_i^2}{a_i^2}$. |
| 5. Output $\vec{x}_j = \begin{cases} \mathrm{sign}(a_j)\, l_j & \text{if } j \in \{1,2,\cdots,i\} \\ \sqrt{\frac{1-\sum_{k=0}^{i} l_k^2}{1-\sum_{k=0}^{i} a_k^2}}\,\vec{a}_j & \text{otherwise} \end{cases}$. |

In this section, we show that the algorithm above can be transformed to solve the problem

$$\max_{\|\vec{x}\|_{\vec{w}} + \|\vec{x}\|_\infty \leq 1} \langle \vec{a}, \vec{x} \rangle \tag{B.2}$$

for some given vector $\vec{a}$ and $\vec{w} > 0$. To do this, let study (B.1) more closely. Without loss of generality, we can assume $\|\vec{a}\|_2 = 1$ and $|a_i|/l_i$ is in descending order. The key consequence of `projectOntoBallBoxParallel` is that the problem (B.1) always has a solution of the form

$$\vec{x}_{l,a}^{(i_t)} = \begin{cases} \mathrm{sign}(a_j)\, l_j & \text{if } j \in \{1,2,\cdots,i_t\} \\ \sqrt{\frac{1-\sum_{k=0}^{i_t} l_k^2}{1-\sum_{k=0}^{i_t} a_k^2}}\,\vec{a}_j & \text{otherwise} \end{cases}. \tag{B.3}$$

where $i_t$ be the first coordinate such that

$$\frac{1-t^2\sum_{k=0}^{i}l_k^2}{1-\sum_{k=0}^{i}a_k^2}\leq\frac{t^2l_i^2}{a_i^2}.$$

Note that $i_t \geq i_s$ if $t \leq s$. Therefore, we have that the set of $t$ such that $i_t = j$ is simply[11]

$$\frac{|a_j|}{\sqrt{l_j^2\left(1-\sum_{k=0}^{j}a_k^2\right)+a_j^2\sum_{k=0}^{j}l_k^2}}\leq t<\frac{|a_{j-1}|}{\sqrt{l_{j-1}^2\left(1-\sum_{k=0}^{j-1}a_k^2\right)+a_{j-1}^2\sum_{k=0}^{j-1}l_k^2}}. \tag{B.4}$$

Define the function $f$ by

$$f(t)=\max_{\left\|\vec{x}\right\|_2\leq 1,-tl_i\leq x_i\leq tl_i}\left\langle\vec{a},\vec{x}\right\rangle.$$

We know that

$$\begin{aligned} f(t) &= \left\langle\vec{a},\vec{x}_{tl,a}^{(i_t)}\right\rangle \\ &= t\sum_{j=1}^{i_t}|a_j|\,|l_j|+\sqrt{1-t^2\sum_{k=0}^{i_t}l_k^2}\sqrt{1-\sum_{k=0}^{i_t}a_k^2}. \end{aligned}$$

Therefore, we have

$$\begin{aligned} \max_{\left\|\vec{x}\right\|_2+\left\|\vec{l}^{-1}\vec{x}\right\|_\infty\leq 1}\left\langle\vec{a},\vec{x}\right\rangle &= \max_{0\leq t\leq 1}\max_{\left\|\vec{x}\right\|_2\leq 1-t\text{ and }-tl_i\leq x_i\leq tl_i}\left\langle\vec{a},\vec{x}\right\rangle \\ &= \max_{0\leq t\leq 1}(1-t)\max_{\left\|\vec{x}\right\|_2\leq 1\text{ and }-\frac{t}{1-t}l_i\leq x_i\leq\frac{t}{1-t}l_i}\left\langle\vec{a},\vec{x}\right\rangle \\ &= \max_{0\leq t\leq 1}(1-t)f\left(\frac{t}{1-t}\right) \\ &= \max_{0\leq t\leq 1}t\sum_{j=1}^{i_t}|a_j|\,|l_j|+\sqrt{(1-t)^2-t^2\sum_{k=0}^{i_t}l_k^2}\sqrt{1-\sum_{k=0}^{i_t}a_k^2}. \end{aligned}$$

Note that the function $t\sum_{j=1}^{i}|a_j|\,|l_j|+\sqrt{(1-t)^2-t^2\sum_{k=0}^{i}l_k^2}\sqrt{1-\sum_{k=0}^{i}a_k^2}$ is concave and the solution has a close form. Therefore, one can compute the maximum value for each interval of $t$ (B.4) and find which is the best. Hence, we get the following algorithm.

| $\vec{x}=$ `projectOntoMixedNormBallParallel`$(\vec{a},\vec{l})$ |
|---|
| 1. Set $\vec{a}=\vec{a}/\left\|\vec{a}\right\|_2$. |
| 2. Sort the coordinate such that $\left\|a_i\right\|/l_i$ is in descending order. |
| 3. Precompute $\sum_{k=0}^{i}l_k^2$, $\sum_{k=0}^{i}a_k^2$ and $\sum_{j=1}^{i}\left\|a_j\right\|\left\|l_j\right\|$ for all $i$. |
| 4. Let $g_i(t)=t\sum_{j=1}^{i}\left\|a_j\right\|\left\|l_j\right\|+\sqrt{(1-t)^2-t^2\sum_{k=0}^{i}l_k^2}\sqrt{1-\sum_{k=0}^{i}a_k^2}$. |
| 5. For each $j\in\{1,\cdots,n\}$, Find $t_j=\arg\max_{i_t=j}g_j(t)$ using (B.4) |
| 6. Find $i=\arg\max_i g_i(t_i)$. |
| 7. Output $(1-t_i)\vec{x}^{(i)}_{\frac{t_i}{1-t_i}l,a}$ defined by (B.3). |

[11] There are some boundary cases we ignored for simplicity.

The discussion above leads to the following theorem. The problem in the from (B.2) can be solved by `projectOntoMixedNormBallParallel` and a change of variables.

**Theorem 40.** *The algorithm* `projectOntoMixedNormBallParallel` outputs a solution to

$$\max_{\left\|\vec{x}\right\|_2+\left\|\vec{l}^{-1}\vec{x}\right\|_\infty\leq 1} \langle \vec{a}, \vec{x}\rangle$$

in total work $\tilde{O}(m)$ and depth $\tilde{O}(1)$.