Solving Linear Programs in the Current Matrix Multiplication Time

Michael B. Cohen*
MIT & Microsoft Research
micohen@mit.edu

Yin Tat Lee*
University of Washington & Microsoft Research
yintat@uw.edu

Zhao Song*
UT-Austin & University of Washington
zhaos@utexas.edu

Abstract

This paper shows how to solve linear programs of the form $\min_{Ax=b,x\geq 0} c^\top x$ with n variables in time

$$O^*((n^\omega + n^{2.5-\alpha/2} + n^{2+1/6}) \log(n/\delta))$$

where ω is the exponent of matrix multiplication, α is the dual exponent of matrix multiplication, and δ is the relative accuracy. For the current value of $\omega \sim 2.37$ and $\alpha \sim 0.31$, our algorithm takes $O^*(n^\omega \log(n/\delta))$ time. When $\omega = 2$, our algorithm takes $O^*(n^{2+1/6} \log(n/\delta))$ time.

Our algorithm utilizes several new concepts that we believe may be of independent interest:

- We define a stochastic central path method.
- We show how to maintain a projection matrix $\sqrt{W}A^T(AWA^\top)^{-1}A\sqrt{W}$ in sub-quadratic time under ℓ_2 multiplicative changes in the diagonal matrix W.

*Work was done while the first two authors were visiting Microsoft Research and hosted by Sébastien Bubeck, and the third author was visiting University of Washington. The authors would like to express their sincere gratitude to Rasmus Kyng for his questions about inverse maintenance that initiated this project.

1 Introduction

Linear programming is one of the key problems in computer science. In both theory and practice, many problems can be reformulated as linear programs to take advantage of fast algorithms. For an arbitrary linear program $\min_{Ax=b,x\geq 0} c^\top x$ with n variables and d constraints[1], the fastest algorithm takes $O^*(\sqrt{d}\cdot \mathrm{nnz}(A)+d^{2.5})$[2] where $\mathrm{nnz}(A)$ is the number of non-zeros in A [LS14, LS15].

For the generic case $d=\Omega(n)$ we focus in this paper, the current fastest runtime is dominated by $O^*(n^{2.5})$. This runtime has not be been improved since the result by Vaidya on 1989 [Vai87, Vai89b]. The $n^{2.5}$ bound originated from two factors: the cost per iteration n^2 and the number of iterations $\sqrt{n}$. The n^2 cost per iteration looks optimal because this is the cost to compute Ax for a dense A. Therefore, many efforts [Kar84, Ren88, NN89, Vai89a, LS14] have been focused on decreasing the number of iterations while maintaining the cost per iteration. As for many important linear programs (and convex programs), the number of iterations has been decreased, including maximum flow [Mad13, Mad16], minimum cost flow [CMSV17], geometric median [CLM$^+$16], matrix scaling and balancing [CMTV17], and ℓ_p regression [BCLL18]. Unfortunately, beating $\sqrt{n}$ iterations (or $\sqrt{d}$ when $d\ll n$) for the general case remains one of the biggest open problems in optimization.

Avoiding this open problem, this paper develops a stochastic central path method that has a runtime of $O^*(n^\omega+n^{2.5-\alpha/2}+n^{2+1/6})$, where ω is the exponent of matrix multiplication and α is the dual exponent of matrix multiplication[3]. For the current value of $\omega\sim 2.38$ and $\alpha\sim 0.31$, the runtime is simply $O^*(n^\omega)$. This achieves the natural barrier for solving linear programs because linear system is a special case of linear program and that the currently fastest way to solve general linear systems involves matrix multiplication. Despite the exact approach used in [CW87, Wil12, DS13, LG14] cannot give a bound on ω better than 2.3078 [AFLG15] and all known approaches cannot achieve the bound $\omega=2$ [AW18], it is still possible that $\omega=2.01$ using all known approaches. Therefore, we believe improving the additive $2+1/6$ term remains an interesting open problem.

Our method is a stochastic version of the short step central path method. This short step method takes $O^*(\sqrt{n})$ steps and each step decreases $x_i s_i$ by a $1-1/\sqrt{n}$ factor for all i where s is the dual variable [Ren88]. This results in $O^*(n^{1.5})$ coordinate updates and $O^*(n^{2.5})$ total time. Our method takes the same number of step but only updates $\widetilde{O}(\sqrt{n})$ coordinates each step. Therefore, we only update $O^*(n)$ coordinates in total, which is nearly optimal.

Our framework is efficient enough to take a much smaller step while maintaining the same running time. For the current value of $\omega\sim 2.38$, we show how to obtain the same runtime of $O^*(n^\omega)$ by taking $O^*(n)$ steps and $\widetilde{O}(1)$ coordinates update per steps. This is because the complexity of each step decreases proportionally when the step size decreases. Beyond the cost per iteration, we remark that our algorithm is one of the very few central path algorithms [PRT02, Mad13, Mad16] that does not maintain $x_i s_i$ close to some ideal vector in ℓ_2 norm. We are hopeful that our stochastic method and our proof will be useful for future research on interior point methods. In particular, it would be interesting to see how this can be combined with techniques in [Cla95, LS14] to get a faster algorithm for linear programs with $d\ll n$.

Besides the applications to linear programs, some of our techniques are probably useful for studying other important problems in convex optimization. In particular, our framework should be naturally extendable to a larger class of convex programs.

[1]Throughout this paper, we assume there is no redundant constraints and hence $n\geq d$. Note that papers in different communities uses different symbols to denote the number of variables and constraints in a linear program.

[2]We use O^* to hide all $n^{o(1)}$ and $\log^{O(1)}(1/\delta)$ factors in the introduction.

[3]The dual exponent of matrix multiplication α is the supremum among all $a\geq 0$ such that it takes $n^{2+o(1)}$ time to multiply an $n\times n$ matrix by an $n\times n^a$ matrix.

2 Results and Techniques

Theorem 2.1 (Main result)**.** *Given a linear program* $\min_{Ax=b,x\geq 0} c^\top x$ *with no redundant constraints. Assume that the polytope has diameter* R *in* ℓ_1 *norm, namely, for any* $x \geq 0$ *with* $Ax = b$*, we have* $\|x\|_1 \leq R$*.*

Then, for any $0 < \delta \leq 1$*,* $\textsc{Main}(A, b, c, \delta)$ *outputs* $x \geq 0$ *such that*

$$c^\top x \leq \min_{Ax=b,x\geq 0} c^\top x + \delta \cdot \|c\|_\infty R \quad \textit{and} \quad \|Ax - b\|_1 \leq \delta \cdot \left(R \sum_{i,j} |A_{i,j}| + \|b\|_1 \right)$$

in expected time

$$\left(n^{\omega+o(1)} + n^{2.5-\alpha/2+o(1)} + n^{2+1/6+o(1)} \right) \cdot \log(\frac{n}{\delta})$$

where ω *is the exponent of matrix multiplication,* α *is the dual exponent of matrix multiplication.*

For the current value of $\omega \sim 2.38$ *and* $\alpha \sim 0.31$*, the expected time is simply* $n^{\omega+o(1)} \log(\frac{n}{\delta})$*.*

Remark 2.2. *See [Ren88] and [LS13, Sec E, F] on the discussion on converting an approximation solution to an exact solution. For integral* A, b, c*, it suffices to pick* $\delta = 2^{-O(L)}$ *to get an exact solution where* $L = \log(1 + d_{\max} + \|c\|_\infty + \|b\|_\infty)$ *is the bit complexity and* $d_{\max}$ *is the largest absolute value of the determinant of a square sub-matrix of* A*. For many combinatorial problems,* $L = O(\log(n + \|b\|_\infty + \|c\|_\infty))$*.*

If $T(n)$ *is the current cost of matrix multiplication and inversion with* $T(n) \sim n^{2.38}$*, our runtime is simply* $O(T(n) \log n \log(\frac{n}{\delta}))$*. The* $\log(\frac{n}{\delta})$ *comes from iteration count and the* $\log n$ *factor comes from the doubling trick* ($|y_{\pi(1.5r)}| \geq (1 - 1/\log n)|y_{\pi(r)}|$) *in the projection maintenance section. We left the problem of obtaining* $O(T(n) \log(\frac{n}{\delta}))$ *as an open problem.*

2.1 Central Path Method

Our algorithm relies on two new ingredients: stochastic central path and projection maintenance. The central path method consider the linear program

$$\min_{Ax=b,x\geq 0} c^\top x \quad \text{(primal)} \quad \text{and} \quad \max_{A^\top y\leq c} b^\top y \quad \text{(dual)}$$

with $A \in \mathbb{R}^{d\times n}$. Any solution of the linear program satisfies the following optimality conditions:

$$\begin{aligned} x_i s_i &= 0 \text{ for all } i, \\ Ax &= b, \\ A^\top y + s &= c, \\ x_i, s_i &\geq 0 \text{ for all } i. \end{aligned}$$

We call (x, s, y) feasible if it satisfies the last three equations above. For any feasible (x, s, y), the duality gap is $\sum_i x_i s_i$. The central path method find a solution of the linear program by following the central path which uniformly decrease the duality gap. The central path $(x_t, s_t, y_t) \in \mathbb{R}^{n+n+d}$ is a path parameterized by t and defined by

$$\begin{aligned} x_{t,i} s_{t,i} &= t \text{ for all } i, \\ Ax_t &= b, \\ A^\top y_t + s_t &= c, \\ x_{t,i}, s_{t,i} &\geq 0 \text{ for all } i. \end{aligned}$$

It is known [YTM94] how to transform linear programs by adding $O(n)$ many variables and constraints so that:

- The optimal solution remains the same.
- The central path at $t = 1$ is near $(1_n, 1_n, 0_d)$ where 1_n and 0_d are all 1 and all 0 vectors with appropriate lengths.
- It is easy to covert an approximate solution of the transformed program to the original one.

For completeness, a theoretical version of such result is included in Lemma A.6. This result shows that it suffices to move gradually (x_1, s_1, y_1) to (x_t, s_t, y_t) for small enough t.

2.1.1 Short Step Central Path Method

The short step central path method maintains $x_i s_i = \mu_i$ for some vector μ such that

$$\sum_i (\mu_i - t)^2 = O(t^2) \quad \text{for some scalar } t > 0.$$

To move from μ to $\mu + \delta_\mu$ approximately, we approximate the term $(x+\delta_x)_i(s+\delta_s)_i$ by $x_i s_i + x_i \delta_{s,i} + s_i \delta_{x,i}$ and obtain the following system: system:

$$\begin{aligned} X\delta_s + S\delta_x &= \delta_\mu, \\ A\delta_x &= 0, \\ A^\top \delta_y + \delta_s &= 0, \end{aligned} \tag{1}$$

where $X = \mathrm{diag}(x)$ and $S = \mathrm{diag}(s)$. This equation is the linear approximation of the original goal (moving from μ to $\mu + \delta_\mu$), and that the step is explicitly given by the formula

$$\delta_x = \frac{X}{\sqrt{XS}}(I - P)\frac{1}{\sqrt{XS}}\delta_\mu \text{ and } \delta_s = \frac{S}{\sqrt{XS}}P\frac{1}{\sqrt{XS}}\delta_\mu \tag{2}$$

where $P = \sqrt{\frac{X}{S}}A^\top \left(A\frac{X}{S}A^\top\right)^{-1} A\sqrt{\frac{X}{S}}$ is an orthogonal projection and the formulas $\frac{X}{\sqrt{XS}}, \frac{X}{S}, \cdots$ are the diagonal matrices of the corresponding vectors.

A standard choice of $\delta_{\mu,i}$ is $-t/\sqrt{n}$ for all i and this requires $\widetilde{O}(\sqrt{n})$ iterations to converge. Combining this with the inverse maintenance technique [Vai87], this gives a total runtime of $n^{2.5}$. We remark that $\sum_i (\mu_i - t)^2 = O(t^2)$ is an invariant of the algorithm and the progress is measured by t because the duality gap is roughly nt.

2.1.2 Stochastic Central Path Method

This part discuss how to modify the short step central path to decrease the cost per iteration to roughly $n^{\omega - \frac{1}{2}}$. Since our goal is to implement a central path method in sub-quadratic time per iteration, we even do not have the budget to compute Ax every iterations. Therefore, instead of maintaining $\left(A\frac{X}{S}A^\top\right)^{-1}$ shown in previous papers, we will study the problem of maintaining a projection matrix $P = \sqrt{\frac{X}{S}}A^\top \left(A\frac{X}{S}A^\top\right)^{-1} A\sqrt{\frac{X}{S}}$ due to the formula of δ_x and δ_s (2).

However, even if the projection matrix P is given explicitly for free, it is difficult to multiply the dense projection matrix with a dense vector δ_μ in time $o(n^2)$. To avoid moving along a dense δ_μ, we move along an $O(k)$ sparse direction $\widetilde{\delta}_\mu$ defined by

$$\widetilde{\delta}_{\mu,i} = \begin{cases} \delta_{\mu,i}/p_i, & \text{with probability } p_i \overset{\text{def}}{=} k \cdot \left(\frac{\delta_{\mu,i}^2}{\sum_l \delta_{\mu,l}^2} + \frac{1}{n}\right); \\ 0, & \text{else.} \end{cases} \tag{3}$$

The sparse direction is defined so that we are moving in the same direction in expectation ($\mathbf{E}[\widetilde{\delta}_{\mu,i}] = \delta_{\mu,i}$) and that the direction has as small variance as possible ($\mathbf{E}[\widetilde{\delta}_{\mu,i}^2] \leq \frac{\sum_i \delta_{\mu,i}^2}{k}$). If the projection matrix is given explicitly, we can apply the projection matrix on $\widetilde{\delta}_\mu$ in time $O(nk)$. This paper picks $k \sim \sqrt{n}$ and the total cost of projection vector multiplications is about n^2.

During the whole algorithm, we maintain a projection matrix

$$\overline{P} = \sqrt{\frac{\overline{X}}{\overline{S}}} A^\top \left(A \frac{\overline{X}}{\overline{S}} A^\top \right)^{-1} A \sqrt{\frac{\overline{X}}{\overline{S}}}$$

for vectors $\overline{x}$ and $\overline{s}$ such that $\overline{x}_i = \Theta(x_i)$ and $\overline{s} = \Theta(s_i)$ for all i. Since we maintain the projection at a nearby point $(\overline{x}, \overline{s})$, our stochastic step $x \leftarrow x + \widetilde{\delta}_x$, $s \leftarrow s + \widetilde{\delta}_s$ and $y \leftarrow y + \widetilde{\delta}_y$ are defined by

$$\begin{aligned} \overline{X}\widetilde{\delta}_s + \overline{S}\widetilde{\delta}_x &= \widetilde{\delta}_\mu, \\ A\widetilde{\delta}_x &= 0, \\ A^\top \widetilde{\delta}_y + \widetilde{\delta}_s &= 0, \end{aligned} \tag{4}$$

which is different from (1) on both sides of the first equation. Similar to (2), Lemma 4.2 shows that

$$\widetilde{\delta}_x = \frac{\overline{X}}{\sqrt{\overline{XS}}}(I - \overline{P})\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu \text{ and } \widetilde{\delta}_s = \frac{\overline{S}}{\sqrt{\overline{XS}}}\overline{P}\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu. \tag{5}$$

The previously fastest algorithm involves maintaining the matrix inverse $(A\frac{X}{S}A^\top)^{-1}$ using subspace embedding techniques [Sar06, CW13, NN13] and leverage score sampling [SS11]. In this paper, we maintain the projection directly.

The key departure from the central path we present is that we can only maintain

$$0.9t \leq \mu_i = x_i s_i \leq 1.1t \quad \text{ for some } t > 0$$

instead of μ close to t in ℓ_2 norm. We will further explain the proof in Section 4.1.

2.2 Projection Maintenance

The projection matrix we maintain is of the form $\sqrt{W}A^\top \left(AWA^\top\right)^{-1} A\sqrt{W}$ where $W = \mathrm{diag}(x/s)$. For intuition, we only explain how to maintain the matrix $M_w \stackrel{\mathrm{def}}{=} A^\top(AWA^\top)^{-1}A$ for the short step central path step here. In this case, we have $\sum_i \left(\frac{w_i^{\mathrm{new}} - w_i}{w_i}\right)^2 = O(1)$ for each step.

If the changes of w is uniformly across all the coordinates, then $w_i^{\mathrm{new}} = (1 \pm \frac{1}{\sqrt{n}})w_i$ for all i. Since it takes $\sqrt{n}$ steps to change all coordinates by a constant factor and we only need to maintain M_v with $v_i = \Theta(w_i)$ for all i, we can update the matrix every $\sqrt{n}$ steps. Hence, the average cost of maintaining the projection matrix is $n^{\omega - \frac{1}{2}}$, which is exactly what we desired.

For the other extreme case that the "adversary" puts all of his ℓ_2 budget on few coordinates, only $\sqrt{n}$ coordinates are changed by a constant factor after $\sqrt{n}$ iterations. In this case, instead of updating M_w every step, we can compute $M_w h$ online by the woodbury matrix identity.

Fact 2.3. *The Woodbury matrix identity is*

$$(M + UCV)^{-1} = M^{-1} - M^{-1}U(C^{-1} + VM^{-1}U)^{-1}VM^{-1}.$$

Let $S \subset [n]$ denote the set of coordinates that is changed by more than a constant factor and $r = |S|$. Using the identity above, we have that

$$M_{w^{\text{new}}} = M_w - (M_w)_S(\Delta_{S,S}^{-1} + (M_w)_{S,S})^{-1}((M_w)_S)^\top \tag{6}$$

where $\Delta = \text{diag}(w^{\text{new}} - w)$, $(M_w)_S \in \mathbb{R}^{n\times r}$ is the r columns from S of M_w and $(M_w)_{S,S}, \Delta_{S,S} \in \mathbb{R}^{r\times r}$ are the r rows and columns from S of M_w and Δ.

As long as $v_i = \Theta(w_i)$ for all i except not too many coordinates, (6) can be applied online efficiently. In another case, we can use (6) instead to update the matrix M_w and the cost is dominated by multiplying a $n \times n$ matrix with a $n \times n^r$ matrix.

Theorem 2.4 (Rectangular matrix multiplication, [LGU18])**.** *Let the dual exponent of matrix multiplication α be the supremum among all $a \geq 0$ such that it takes $n^{2+o(1)}$ time to multiply an $n \times n$ matrix by an $n \times n^a$ matrix.*

Then, for any $n \geq r$, multiplying an $n \times r$ with an $r \times n$ matrix or $n \times n$ with $n \times r$ takes time

$$n^{2+o(1)} + r^{\frac{\omega-2}{1-\alpha}} n^{2-\frac{\alpha(\omega-2)}{1-\alpha}+o(1)}.$$

Furthermore, we have $\alpha > 0.31389$.

See Lemma A.5 for the origin of the formula. Since the cost of multiplying $n \times n$ matrix by a $n \times 1$ matrix is same as the cost for $n \times n$ with $n \times n^{0.31}$, (6) should be used to update at least $n^{0.31}$ coordinates. In the extreme case we are discussing, we only need to update the matrix $n^{\frac{1}{2}-0.31}$ times and each takes n^2 time, and hence the total cost is less than n^ω.

In previous papers [Kar84, Vai89b, NN91, NN94, LS14, LS15], the matrix is updated in a fixed schedule independent of the input sequence w. This leads to sub-optimal bounds if used in this paper. We instead define a potential function to measure the distance between the approximate vector v and the target vector w. When there are more than n^α coordinates of v that is far from w, we update v by a certain greedy step. As in the extreme cases, the worst case of our algorithm is that the "adversary" puts his ℓ_2 budget across all coordinates uniformly and hence the worst case runtime is $n^{\omega-\frac{1}{2}}$ per iteration. We will further explain the potential in Section 5.1.

3 Preliminaries

For notation convenience, we assume the number of variables $n \geq 10$ and there is no redundant constraints. In particular, this implies that the constraint matrix A is full rank and $n \geq d$

For a positive integer n, let $[n]$ denote the set $\{1, 2, \cdots, n\}$.

For any function f, we define $\widetilde{O}(f)$ to be $f \cdot \log^{O(1)}(f)$. In addition to $O(\cdot)$ notation, for two functions f, g, we use the shorthand $f \lesssim g$ (resp. $\gtrsim$) to indicate that $f \leq Cg$ (resp. $\geq$) for some absolute constant C.

We use $\sinh x$ to denote $\frac{e^x - e^{-x}}{2}$ and $\cosh x$ to denote $\frac{e^x + e^{-x}}{2}$.

For vectors $a, b \in \mathbb{R}^n$ and accuracy parameter $\epsilon \in (0, 1)$, we use $a \approx_\epsilon b$ to denote that $(1-\epsilon)b_i \leq a_i \leq (1+\epsilon)b_i, \forall i \in [n]$. Similarly, for any scalar t, we use $a \approx_\epsilon t$ to denote that $(1-\epsilon)t \leq a_i \leq (1+\epsilon)t, \forall i \in [n]$

For a vector $x \in \mathbb{R}^n$ and $s \in \mathbb{R}^n$, we use xs to denote a length n vector with the i-th coordinate $(xs)_i$ is $x_i \cdot s_i$. Similarly, we extend other scalar operations to vector coordinate-wise.

Given vectors $x, s \in \mathbb{R}^n$, we use X and S to denote the diagonal matrix of those two vectors. We use $\frac{X}{S}$ to denote the diagonal matrix given $(\frac{X}{S})_{i,i} = x_i/s_i$. Similarly, we extend other scalar operations to diagonal matrix diagonal-wise. Note that matrix $\sqrt{\frac{X}{S}} A^\top (A \frac{X}{S} A^\top)^{-1} A \sqrt{\frac{X}{S}}$ is an orthogonal projection matrix.

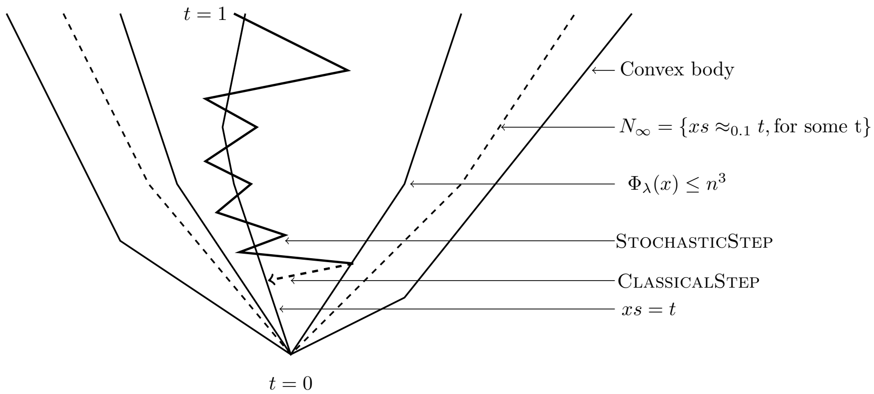

Figure 1: ClassicalStep happens with n^{-2} probability

4 Stochastic Central Path Method

4.1 Proof Outline

The short step central path method is defined using the approximation $(x + \delta_x)_i(s + \delta_s)_i \sim x_i s_i + x_i \delta_{s,i} + s_i \delta_{x,i}$. This approximate is only accurate if $\|X^{-1}\delta_x\|_\infty = O(1)$ and $\|S^{-1}\delta_s\|_\infty = O(1)$. For the δ_x step, we have

$$X^{-1}\delta_x = \frac{1}{\sqrt{XS}}(I - P)\frac{1}{\sqrt{XS}}\delta_\mu \sim \frac{1}{t}(I - P)\delta_\mu \tag{7}$$

where we used $x_i s_i \sim t$ for all i.

If we know that $\|\delta_\mu\|_2 \ll t$, then the ℓ_∞ norm can be bounded as follows:

$$\|X^{-1}\delta_x\|_\infty \leq \|X^{-1}\delta_x\|_2 \lesssim \frac{1}{t}\|(I - P)\delta_\mu\|_2 \leq \frac{1}{t}\|\delta_\mu\|_2 \ll 1$$

where we used that $I - P$ is an orthogonal projection matrix. This is the reason why a standard choice of $\delta_{\mu,i}$ is $-t/\sqrt{n}$ for all i.

For the stochastic step, $\widetilde{\delta}_{\mu,i} \sim -\frac{t}{\sqrt{n}}\frac{n}{k}$ for roughly k coordinates. Therefore, the ℓ_2 norm of $\widetilde{\delta}_\mu$ is very large ($\|\widetilde{\delta}_\mu\|_2 \sim t\sqrt{\frac{n}{k}}$). After the projection, we have $\|X^{-1}\delta_x\|_2 \sim \frac{1}{t}\|(I - P)\delta_\mu\|_2 \sim \sqrt{\frac{n}{k}}$. Hence, we cannot bound $\|X^{-1}\delta_x\|_\infty$ by $\|X^{-1}\delta_x\|_2$. To improve the bound, we use Chernoff bounds to estimate $\|X^{-1}\delta_x\|_\infty$.

Beside the ℓ_∞ norm bound, the proof sketch in (7) also requires using $x_i s_i \sim t$ for all i. The short step central path proof maintains an invariant that $\sum_i (x_i s_i - t)^2 = O(t^2)$. However, since our stochastic step has a stochastic noise with ℓ_2 norm as large as $t\sqrt{\frac{n}{k}}$, one cannot hope to maintain $x_i s_i$ close to t in ℓ_2 norm. Instead, we follow an idea in [LS14, LSW15] and maintain the following potential

$$\sum_{i=1}^n \cosh\left(\lambda\left(\frac{x_i s_i}{t} - 1\right)\right) = n^{O(1)}$$

with $\lambda = \Theta(\log n)$. Note that the potential bounded by $n^{O(1)}$ implies that $x_i s_i$ is a multiplicative approximation of t. To bound the potential, consider $r_i = \frac{x_i s_i}{t}$ and $\Phi(r)$ be the potential above. Then, we have that

$$\mathbf{E}[\Phi(r^{\text{new}})] \sim \Phi(r) + \langle \nabla\Phi(r), \mathbf{E}[r^{\text{new}} - r]\rangle + \frac{1}{2}\|r^{\text{new}} - r\|^2_{\nabla^2\Phi(r)}.$$

Algorithm 1

procedure StochasticStep$(\text{mp}, x, s, \delta_\mu, k, \epsilon)$ ▷ Lemma 4.2,4.3,4.8

  $w \leftarrow \frac{x}{s}$, $\widetilde{v} \leftarrow \text{mp.}\textsc{Update}(w)$ ▷ Algorithm 3

  $\overline{x} \leftarrow x\sqrt{\frac{\widetilde{v}}{w}}$, $\overline{s} \leftarrow s\sqrt{\frac{w}{\widetilde{v}}}$ ▷ It guarantees that $\frac{\overline{x}}{\overline{s}} = \widetilde{v}$ and $\overline{x s} = xs$

  repeat

    Generate $\widetilde{\delta}_\mu$ such that ▷ Compute a sparse direction

$$\widetilde{\delta}_{\mu,i} \leftarrow \begin{cases} \delta_{\mu,i}/p_i, & \text{with prob. } p_i = \min(1, k\cdot((\delta_{\mu,i}^2/\sum_{l=1}^n \delta_{\mu,l}^2) + 1/n)); \\ 0 & \text{else.} \end{cases}$$

    ▷ Compute an approximate step

    ▷ Find $(\widetilde{\delta}_x, \widetilde{\delta}_s, \widetilde{\delta}_y)$ such that these three equations hold

$$\overline{X}\widetilde{\delta}_s + \overline{S}\widetilde{\delta}_x = \widetilde{\delta}_\mu,$$
$$A\widetilde{\delta}_x = 0,$$
$$A^\top \widetilde{\delta}_y + \widetilde{\delta}_s = 0.$$

    $p_\mu \leftarrow \text{mp.}\textsc{Query}(\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu)$ ▷ Algorithm 3

    $\widetilde{\delta}_s \leftarrow \frac{\overline{S}}{\sqrt{\overline{XS}}} p_\mu$ ▷ According to (8)

    $\widetilde{\delta}_x \leftarrow \frac{1}{\overline{S}}\widetilde{\delta}_\mu - \frac{\overline{X}}{\sqrt{\overline{XS}}} p_\mu$ ▷ According to (9)

  until $\|\overline{s}^{-1}\widetilde{\delta}_s\|_\infty \leq \frac{1}{100\log n}$ and $\|\overline{x}^{-1}\widetilde{\delta}_x\|_\infty \leq \frac{1}{100\log n}$

  return $(x + \widetilde{\delta}_x, s + \widetilde{\delta}_s)$

end procedure

The first order term can be bounded efficiently because $\mathbf{E}[r^{\text{new}} - r]$ is close to the short step central path step. The second term is a variance term which scales like $1/k$ due to the k independent coordinates. Therefore, the potential changed by $1/k \sim 1/\sqrt{n}$ factor each step. Hence, we can maintain it for roughly $\sqrt{n}$ steps.

To make sure the potential Φ is bounded during the whole algorithm, our step is the mixtures of two steps of the form $\delta_\mu \sim -\frac{t}{\sqrt{n}} - t\frac{\nabla\Phi}{\|\nabla\Phi\|_2}$. The first term is to decrease t and the second term is to decrease Φ.

Since the algorithm is randomized, there is a tiny probability that Φ is large. In that case, we switch to a short step central path method. See Figure 1, Algorithm 1, and Algorithm 2. The first part of the proof involves bounding every quantity listed in Table 1. In the second part, we are using these quantities to bound the expectation of Φ.

To decouple the proof in both parts, we will make the following assumption in first part. It will be verified in the second part.

Assumption 4.1. *Assume the following for the input of the procedure* StochasticStep *(see Algorithm 1):*

- $xs \approx_{0.1} t$ *with* $t > 0$.
- *mp.*Update(w) *outputs* $\widetilde{v}$ *such that* $w \approx_{\epsilon_{mp}} \widetilde{v}$ *with* $\epsilon_{mp} \leq 1/40000$.
- $\|\delta_\mu\|_2 \leq \epsilon t$ *with* $0 < \epsilon < 1/(40000\log n)$.
- $k \geq 1000\epsilon\sqrt{n}\log^2 n/\epsilon_{mp}$.

Algorithm 2 Our main algorithm

procedure MAIN(A, b, c, δ) ▷ Theorem 2.1
    $\epsilon \leftarrow \frac{1}{40000 \log n}$, $\epsilon_{mp} \leftarrow \frac{1}{40000}$, $k \leftarrow \frac{1000\epsilon\sqrt{n}\log^2 n}{\epsilon_{mp}}$.
    $\lambda \leftarrow 40 \log n$, $\delta \leftarrow \min(\frac{\delta}{2}, \frac{1}{\lambda})$, $a \leftarrow \min(\alpha, 2/3)$.
    Modify the linear program and obtain an initial x and s according to Lemma A.6.
    MAINTAINPROJECTION mp
    mp.INITIALIZE$(A, \frac{x}{s}, \epsilon_{mp}, a)$ ▷ Algorithm 3
    $t \leftarrow 1$ ▷ Initialize t
    while $t > \delta^2/(2n)$ **do** ▷ We stopped once the precision is good
        $t^{\text{new}} \leftarrow (1 - \frac{\epsilon}{3\sqrt{n}})t$
        $\mu \leftarrow xs$
        $\delta_\mu \leftarrow (\frac{t^{\text{new}}}{t} - 1)xs - \frac{\epsilon}{2} \cdot t^{\text{new}} \cdot \frac{\nabla \Phi_\lambda(\mu/t-1)}{\|\nabla \Phi_\lambda(\mu/t-1)\|_2}$ ▷ Φ_λ is defined in Lemma 4.12
        $(x^{\text{new}}, s^{\text{new}}) \leftarrow$ STOCHASTICSTEP$(\text{mp}, x, s, \delta_\mu, k, \epsilon)$ ▷ Algorithm 1
        if $\Phi_\lambda(\mu^{\text{new}}/t^{\text{new}} - 1) > n^3$ **then** ▷ When potential function is large
            $(x^{\text{new}}, s^{\text{new}}) \leftarrow$ CLASSICALSTEP(x, s, t^{new}) ▷ Lemma A.2, [Vai89b]
            mp.INITIALIZE$(A, \frac{x^{\text{new}}}{s^{\text{new}}}, \epsilon_{mp}, a)$ ▷ Restart the data structure
        end if
        $(x, s) \leftarrow (x^{\text{new}}, s^{\text{new}})$, $t \leftarrow t^{\text{new}}$
    end while
    Return an approximate solution of the original linear program according to Lemma A.6.
end procedure

4.2 Bounding each quantity of stochastic step

First, we give an explicit formula for our step, which will be used in all subsequent calculations.

Lemma 4.2. *The procedure* STOCHASTICSTEP$(\text{mp}, x, s, \delta_\mu, k, \epsilon)$ *(see Algorithm 1) finds a solution* $\widetilde{\delta}_x$, $\widetilde{\delta}_s \in \mathbb{R}^n$ *to (4) by the formula*

$$\widetilde{\delta}_x = \frac{\overline{X}}{\sqrt{\overline{XS}}}(I - \overline{P})\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu \tag{8}$$

$$\widetilde{\delta}_s = \frac{\overline{S}}{\sqrt{\overline{XS}}}\overline{P}\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu \tag{9}$$

with

$$\overline{P} = \sqrt{\frac{\overline{X}}{\overline{S}}} A^\top \left(A\frac{\overline{X}}{\overline{S}}A^\top\right)^{-1} A\sqrt{\frac{\overline{X}}{\overline{S}}}. \tag{10}$$

Proof. For the first equation of (4), we multiply $A\overline{S}^{-1}$ on both sides,

$$A\overline{S}^{-1}\overline{X}\widetilde{\delta}_s + A\widetilde{\delta}_x = A\overline{S}^{-1}\widetilde{\delta}_\mu.$$

Since the second equation gives $A\widetilde{\delta}_x = 0$, then we know that $A\overline{S}^{-1}\overline{X}\widetilde{\delta}_s = A\overline{S}^{-1}\widetilde{\delta}_\mu$.

Multiplying $A\overline{S}^{-1}\overline{X}$ on both sides of the third equation of (4), we have

$$-A\overline{S}^{-1}\overline{X}A^\top\widetilde{\delta}_y = A\overline{S}^{-1}\overline{X}\widetilde{\delta}_s = A\overline{S}^{-1}\widetilde{\delta}_\mu.$$

Quantity	Bound	Place
$\\| \mathbf{E}[s^{-1}\widetilde{\delta}_s]\\|_2, \\| \mathbf{E}[x^{-1}\widetilde{\delta}_x]\\|_2, \\| \mathbf{E}[\mu^{-1}\widetilde{\delta}_\mu]\\|_2$	$O(\epsilon)$	Part 1, Lemma 4.3
$\\| \mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu)]\\|_2$	$O(\epsilon_{mp} \cdot \epsilon)$	Part 1, Lemma 4.8
$\\| \mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu)]\\|_2$	$O(\epsilon)$	Part 1, Lemma 4.8
$\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}], \mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}], \mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{\mu,i}]$	$O(\epsilon^2/k)$	Part 2, Lemma 4.3
$\mathbf{Var}[\mu_i^{-1}\mu^{\text{new}}]$	$O(\epsilon^2/k)$	Part 2, Lemma 4.8
$\\|s^{-1}\widetilde{\delta}_s\\|_\infty, \\|x^{-1}\widetilde{\delta}_x\\|_\infty, \\|\mu^{-1}\widetilde{\delta}_\mu\\|_\infty$	$O(1/\log n)$	Part 3, Lemma 4.3
$\\|\mu^{-1}(\mu^{\text{new}} - \mu)\\|_\infty$	$O(1/\log n)$	Part 3, Lemma 4.8

Table 1: The bound of each quantity under Assumption 4.1

Thus,

$$\begin{aligned}\widetilde{\delta}_y &= -(A\overline{S}^{-1}\overline{X}A^\top)^{-1}A\overline{S}^{-1}\widetilde{\delta}_\mu,\\ \widetilde{\delta}_s &= A^\top(A\overline{S}^{-1}\overline{X}A^\top)^{-1}A\overline{S}^{-1}\widetilde{\delta}_\mu,\\ \widetilde{\delta}_x &= \overline{S}^{-1}\widetilde{\delta}_\mu - \overline{S}^{-1}\overline{X}A^\top(A\overline{S}^{-1}\overline{X}A^\top)^{-1}A\overline{S}^{-1}\widetilde{\delta}_\mu.\end{aligned}$$

Recall we define $\overline{P}$ as (10), then we have

$$\widetilde{\delta}_s = \frac{\overline{S}}{\sqrt{\overline{XS}}} \cdot \sqrt{\frac{\overline{X}}{\overline{S}}}A^\top(A\frac{\overline{X}}{\overline{S}}A^\top)^{-1}\sqrt{\frac{\overline{X}}{\overline{S}}} \cdot \frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu = \frac{\overline{S}}{\sqrt{\overline{XS}}}\overline{P}\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu,$$

and

$$\widetilde{\delta}_x = \overline{S}^{-1}\widetilde{\delta}_\mu - \frac{\overline{X}}{\sqrt{\overline{XS}}} \cdot \sqrt{\frac{\overline{X}}{\overline{S}}}A^\top(A\frac{\overline{X}}{\overline{S}}A^\top)^{-1}\sqrt{\frac{\overline{X}}{\overline{S}}} \cdot \frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu = \frac{\overline{X}}{\sqrt{\overline{XS}}}(I - \overline{P})\frac{1}{\sqrt{\overline{XS}}}\widetilde{\delta}_\mu.$$

which are matching (8) and (9). □

Using the explicitly formula, we are ready to bound all quantities we needed in the following two subsubsections.

4.2.1 Bounding $\widetilde{\delta}_s$, $\widetilde{\delta}_x$ and $\widetilde{\delta}_\mu$

Lemma 4.3. *Under the Assumption 4.1, the two vectors $\widetilde{\delta}_x$ and $\widetilde{\delta}_s$ found by* StochasticStep *satisfy :*

1. $\\| \mathbf{E}[\overline{s}^{-1}\widetilde{\delta}_s]\\|_2 \leq 2\epsilon, \\| \mathbf{E}[\overline{x}^{-1}\widetilde{\delta}_x]\\|_2 \leq 2\epsilon, \\| \mathbf{E}[s^{-1}\widetilde{\delta}_s]\\|_2 \leq 2\epsilon, \\| \mathbf{E}[x^{-1}\widetilde{\delta}_x]\\|_2 \leq 2\epsilon, \\| \mathbf{E}[\mu^{-1}\widetilde{\delta}_\mu]\\|_2 \leq 4\epsilon.$

2. $\mathbf{Var}[\frac{\widetilde{\delta}_{s,i}}{\overline{s}_i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\frac{\widetilde{\delta}_{x,i}}{\overline{x}_i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\frac{\widetilde{\delta}_{s,i}}{s_i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\frac{\widetilde{\delta}_{x,i}}{x_i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\frac{\widetilde{\delta}_{\mu,i}}{\mu_i}] \leq \frac{8\epsilon^2}{k}.$

3. $\\|\overline{s}^{-1}\widetilde{\delta}_s\\|_\infty \leq \frac{0.01}{\log n}, \\|s^{-1}\widetilde{\delta}_s\\|_\infty \leq \frac{0.02}{\log n}, \\|\overline{x}^{-1}\widetilde{\delta}_x\\|_\infty \leq \frac{0.01}{\log n}, \\|x^{-1}\widetilde{\delta}_x\\|_\infty \leq \frac{0.02}{\log n}, \\|\mu^{-1}\widetilde{\delta}_\mu\\|_\infty \leq \frac{0.02}{\log n}.$

Remark 4.4. *For notational simplicity, the* **E** *and* **Var** *in the proof are for the case without re-sampling (Line 12). Since the all the additional terms due to resampling are polynomially bounded and since we can set failure probability to an arbitrarily small inverse polynomial (see Claim 4.7), the proof does not change and the result remains the same.*

Proof.

Claim 4.5 (Part 1, bounding the ℓ_2 norm of expectation)**.**

$$\|\mathbf{E}[\overline{s}^{-1}\widetilde{\delta}_s]\|_2 \leq 2\epsilon, \|\mathbf{E}[\overline{x}^{-1}\widetilde{\delta}_x]\|_2 \leq 2\epsilon, \|\mathbf{E}[s^{-1}\widetilde{\delta}_s]\|_2 \leq 2\epsilon, \|\mathbf{E}[x^{-1}\widetilde{\delta}_x]\|_2 \leq 2\epsilon, \|\mathbf{E}[\mu^{-1}\widetilde{\delta}_\mu]\|_2 \leq 4\epsilon.$$

Proof. For $\|\overline{s}^{-1}\widetilde{\delta}_s\|_\infty$, we consider the i-th coordinate of the vector

$$\overline{s}_i^{-1}\widetilde{\delta}_{s,i} = \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\sum_{j=1}^n \overline{P}_{i,j}\frac{\widetilde{\delta}_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}.$$

Then, we have

$$\mathbf{E}\left[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}\right] = \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\sum_{j=1}^n \overline{P}_{i,j}\frac{\mathbf{E}[\widetilde{\delta}_{\mu,j}]}{\sqrt{\overline{x}_j\overline{s}_j}} = \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\sum_{j=1}^n \overline{P}_{i,j}\frac{\delta_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}$$

Since $xs \approx_{0.1} t$ and $\|\delta_\mu\| \leq \epsilon t$, we have $\|\frac{\delta_\mu}{\sqrt{xs}}\|_2 \leq \frac{1.1\epsilon t}{\sqrt{t}}$. Since $\overline{P}$ is an orthogonal projection matrix, we have $\|\overline{P}\frac{\delta_\mu}{\sqrt{\overline{xs}}}\|_2 \leq \|\frac{\delta_\mu}{\sqrt{\overline{xs}}}\|_2$. Putting all the above facts and $xs = \overline{xs}$, we can show

$$\begin{aligned}
\left\|\mathbf{E}[\overline{s}^{-1}\widetilde{\delta}_s]\right\|_2^2 = & \sum_{i=1}^n \left(\frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\sum_{j=1}^n \overline{P}_{i,j}\frac{\delta_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}\right)^2 = \sum_{i=1}^n \frac{1}{\overline{x}_i\overline{s}_i}\left(\sum_{j=1}^n \overline{P}_{i,j}\frac{\delta_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}\right)^2 \\
\leq & \frac{1}{0.9t}\sum_{i=1}^n\left(\sum_{j=1}^n \overline{P}_{i,j}\frac{\delta_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}\right)^2 = \frac{1}{0.9t}\|\overline{P}\frac{\delta_\mu}{\sqrt{\overline{xs}}}\|_2^2 \\
\leq & \frac{1}{0.9t}\|\frac{\delta_\mu}{\sqrt{\overline{xs}}}\|_2^2 \leq \frac{(1.1)^2}{0.9t}\cdot\frac{(\epsilon t)^2}{t} \leq 1.4\epsilon^2
\end{aligned}$$

which implies that

$$\left\|\mathbf{E}[\overline{s}^{-1}\widetilde{\delta}_s]\right\|_2 \leq 1.2\epsilon. \tag{11}$$

Notice that the proof for x is identical to the proof for s because $(I - \overline{P})$ is also a projection matrix. Since $\overline{s} \approx_{0.1} s$ and $\overline{x} \approx_{0.1} x$, then we can also prove the next two inequalities in the Claim statement.

Now, we are ready to bound $\|\mathbf{E}[\mu^{-1}\widetilde{\delta}_\mu]\|_2$

$$\|\mathbf{E}[\mu^{-1}\widetilde{\delta}_\mu]\|_2 = \|\mathbf{E}[\overline{s}^{-1}\overline{x}^{-1}(\overline{x}\widetilde{\delta}_s + \overline{s}\widetilde{\delta}_x)]\|_2 \leq \|\mathbf{E}[\overline{s}^{-1}\widetilde{\delta}_s] + \|\mathbf{E}[\overline{x}^{-1}\widetilde{\delta}_x]\|_2 \leq 4\epsilon.$$

by using $\mu = xs = \overline{xs}$ and $\overline{x}\widetilde{\delta}_s + \overline{s}\widetilde{\delta}_x = \widetilde{\delta}_\mu$ from (4). ☐

Claim 4.6 (Part 2, bounding the variance per coordinate)**.**

$$\mathbf{Var}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\overline{x}_i^{-1}\widetilde{\delta}_{x,i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}] \leq \frac{2\epsilon^2}{k}, \mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{\mu,i}] \leq \frac{8\epsilon^2}{k}.$$

Proof. For $\|\overline{s}^{-1}\widetilde{\delta}_s\|_\infty$, we consider the i-th coordinate of the vector

$$\overline{s}_i^{-1}\widetilde{\delta}_{s,i} = \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\sum_{j=1}^n \overline{P}_{i,j}\frac{\widetilde{\delta}_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}.$$

For variance of $\overline{s}_i^{-1}\widetilde{\delta}_{s,i}$, we have

$$\begin{aligned}
\mathbf{Var}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] &= \frac{1}{\overline{x}_i\overline{s}_i}\sum_{j=1}^n \frac{\overline{P}_{i,j}^2}{\overline{x}_j\overline{s}_j}\mathbf{Var}[\widetilde{\delta}_{\mu,j}] && \text{by all } \widetilde{\delta}_{\mu,j} \text{ are independent} \\
&\leq \frac{1}{\overline{x}_i\overline{s}_i}\sum_{j=1}^n \frac{\overline{P}_{i,j}^2}{\overline{x}_j\overline{s}_j}\frac{1}{k}\frac{\delta_{\mu,j}^2}{\frac{\delta_{\mu,j}^2}{\sum_{l=1}^n \delta_{\mu,l}^2}+\frac{1}{n}} && \text{by (3)} \\
&\leq \frac{1}{\overline{x}_i\overline{s}_i}\sum_{j=1}^n \frac{\overline{P}_{i,j}^2}{\overline{x}_j\overline{s}_j}\frac{1}{k}\sum_{l=1}^n \delta_{\mu,l}^2 \\
&\leq \frac{1.3}{t^2}\sum_{j=1}^n \overline{P}_{i,j}^2\frac{1}{k}\sum_{l=1}^n \delta_{\mu,l}^2 \leq \frac{1.3\epsilon^2}{k}, && \text{by } \overline{x}_i\overline{s}_i = x_is_i \approx_{1/10} t
\end{aligned}$$

where we used that $\sum_{j=1}^n \overline{P}_{i,j}^2 = \overline{P}_{i,i} \leq 1$, $\|\delta_\mu\|_2 \leq \epsilon t$ at the end.

The proof for the other three inequalities in the Claim statement are identical to this one. We omit here.

For the variance of $\mu_i^{-1}\widetilde{\delta}_{\mu,i}$,

$$\begin{aligned}
\mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{\mu,i}] = &\ \mathbf{Var}[\overline{x}_i^{-1}\overline{s}_i^{-1}(\overline{x}_i\widetilde{\delta}_{s,i}+\overline{s}_i\widetilde{\delta}_{x,i})] \\
&\leq 2\,\mathbf{Var}[\overline{x}_i^{-1}\overline{x}_i\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] + 2\,\mathbf{Var}[\overline{s}_i^{-1}\overline{s}_i\overline{x}_i^{-1}\widetilde{\delta}_{x,i}] \\
&= 2\,\mathbf{Var}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] + 2\,\mathbf{Var}[\overline{x}_i^{-1}\widetilde{\delta}_{x,i}] \leq 8\epsilon^2/k.
\end{aligned}$$

where the first step follows by definition of $\mu = xs = \overline{xs}$ and (4), the second step follows by triangle inequality and, the last step follows by $\mathbf{Var}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}], \mathbf{Var}[\overline{x}_i^{-1}\widetilde{\delta}_{x,i}] \leq 2\epsilon^2/k$ □

Claim 4.7 (Part 3, bounding the probability of success)**.** *Without resampling, the following holds with probability* $1 - 2n\exp(-\frac{0.003k}{\epsilon\sqrt{n}\log n})$.

$$\|\overline{s}^{-1}\widetilde{\delta}_s\|_\infty \leq \frac{0.01}{\log n}, \|s^{-1}\widetilde{\delta}_s\|_\infty \leq \frac{0.02}{\log n}, \|\overline{x}^{-1}\widetilde{\delta}_x\|_\infty \leq \frac{0.01}{\log n}, \|x^{-1}\widetilde{\delta}_x\|_\infty \leq \frac{0.02}{\log n}, \|\mu^{-1}\widetilde{\delta}_\mu\|_\infty \leq \frac{0.02}{\log n}.$$

With resampling, it always holds.

Proof. We can write $\overline{s}_i^{-1}\widetilde{\delta}_{s,i} - \mathbf{E}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] = \sum_j Y_j$ where Y_j are independent random variables defined by

$$Y_j = \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\overline{P}_{i,j}\frac{\widetilde{\delta}_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}} - \frac{1}{\sqrt{\overline{x}_i\overline{s}_i}}\overline{P}_{i,j}\frac{\delta_{\mu,j}}{\sqrt{\overline{x}_j\overline{s}_j}}.$$

We bound the sum using Bernstein inequality. Note that Y_j are mean 0 and that Claim 4.6 shows that $\sum_{j=1}^n \mathbf{E}[Y_j^2] = \mathbf{Var}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}] \leq \frac{2\epsilon^2}{k}$. We also need to give an upper bound for Y_j

$$\begin{aligned}
|Y_j| &= \left| \frac{1}{\sqrt{\overline{x}_i \overline{s}_i}} \overline{P}_{i,j} \left(\frac{\widetilde{\delta}_{\mu,j} - \delta_{\mu,j}}{\sqrt{\overline{x}_j \overline{s}_j}} \right) \right| \\
&\leq \frac{1.2}{t} |\widetilde{\delta}_{\mu,j} - \delta_{\mu,j}| && \text{by } |\overline{P}_{i,j}| \leq 1, x_i s_i \approx_{1/10} t \\
&\leq \frac{1.2}{t} |\delta_{\mu,j}/p_j| && \text{by } \widetilde{\delta}_{\mu,j} \in [0, \delta_{\mu,j}/p_j] \\
&= \frac{1.2}{t} \frac{1}{k} \frac{1}{(\frac{\delta_{\mu,i}}{\sum_{l=1}^n \delta_{\mu,l}^2} + \frac{1}{n\delta_{\mu,i}})} && \text{by } (3) \\
&\leq \frac{0.6}{t} \frac{1}{k} \left(n \sum_{l=1}^n \delta_{\mu,l}^2 \right)^{1/2} && \text{by } a^2 + b^2 \geq 2ab \\
&\leq \frac{0.6\epsilon\sqrt{n}}{k} \stackrel{\text{def}}{=} M. && \text{by } \|\delta_\mu\|_2 \leq \epsilon t
\end{aligned}$$

Now, we can apply Bernstein inequality

$$\begin{aligned}
\Pr\left[\left| \sum_{j=1}^n Y_j \right| > b \right] &\leq 2\exp\left(-\frac{b^2/2}{\sum_{j=1}^n \mathbf{E}[Y_j^2] + Mb/3} \right) \\
&\leq 2\exp\left(-\frac{b^2/2}{2\epsilon^2/k + (0.6\epsilon\sqrt{n}/k)\cdot b/3} \right).
\end{aligned}$$

We choose $b = \frac{0.005}{\log n}$ and use $\epsilon \leq \frac{1}{400 \log n}$ and $n \geq 10$ to get

$$\Pr\left[\left| \sum_{j=1}^n Y_j \right| \geq \frac{0.05}{\log n} \right] \leq 2\exp\left(-\frac{0.003k}{\epsilon\sqrt{n}\log n} \right).$$

Since $\|\mathbf{E}[\overline{s}_i^{-1}\widetilde{\delta}_{s,i}]\|_2 \leq 2\epsilon \leq \frac{0.005}{\log n}$, we have that $|\overline{s}_i^{-1}\widetilde{\delta}_{s,i}| \leq \frac{0.01}{\log n}$ with probability $1 - 2\exp(-\frac{0.003k}{\epsilon\sqrt{n}\log n})$. Taking a union bound, we have that $\|\overline{s}^{-1}\widetilde{\delta}_s\|_\infty \leq \frac{0.01}{\log n}$ with probability $1 - 2n\exp(-\frac{0.003k}{\epsilon\sqrt{n}\log n})$. Similarly, this holds for the other 3 terms.

Now, the last term follows by

$$|\mu_i^{-1}\widetilde{\delta}_{\mu,i}| = |\overline{x}_i^{-1}\overline{s}_i^{-1}(\overline{x}_i\widetilde{\delta}_{s,i} + \overline{s}_i\widetilde{\delta}_{x,i})| = |\overline{s}_i^{-1}\widetilde{\delta}_{s,i}| + |\overline{x}_i^{-1}\widetilde{\delta}_{x,i}| \leq \frac{0.02}{\log n}.$$

□

□

4.2.2 Bounding $\mu^{\text{new}} - \mu$

Lemma 4.8. *Under the Assumption 4.1, the vector* $\mu_i^{\text{new}} \stackrel{\text{def}}{=} (x_i + \widetilde{\delta}_{x,i})(s_i + \widetilde{\delta}_{s,i})$ *satisfies*

1. $\|\mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu)]\|_2 \leq 10\epsilon_{mp} \cdot \epsilon$ *and* $\|\mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu)]\|_2 \leq 5\epsilon$.
2. $\mathbf{Var}[\mu_i^{-1}\mu_i^{\text{new}}] \leq 50\epsilon^2/k$ *for all* i.
3. $\|\mu^{-1}(\mu^{\text{new}} - \mu)\|_\infty \leq \frac{0.021}{\log n}$.

Claim 4.9 (Part 1 of Lemma 4.8)**.**

$$\|\mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu)]\|_2 \le 10\epsilon_{mp} \cdot \epsilon, \;\; \textit{and} \; \|\mathbf{E}[\mu^{-1}(\mu^{\text{new}} - \mu)]\|_2 \le 5\epsilon.$$

Proof.

$$\mu^{\text{new}} = (x + \widetilde{\delta}_x)(s + \widetilde{\delta}_s) = \mu + x\widetilde{\delta}_s + s\widetilde{\delta}_x + \widetilde{\delta}_x\widetilde{\delta}_s = \mu + \underbrace{\overline{x}\widetilde{\delta}_s + \overline{s}\widetilde{\delta}_x}_{\widetilde{\delta}_\mu} + \underbrace{(x-\overline{x})\widetilde{\delta}_s + (s-\overline{s})\widetilde{\delta}_x + \widetilde{\delta}_x\widetilde{\delta}_s}_{\widehat{\delta}_\mu}.$$

Taking the expectation on both sides, we have

$$\mathbf{E}[\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu] = (x - \overline{x})\,\mathbf{E}[\widetilde{\delta}_s] + (s - \overline{s})\,\mathbf{E}[\widetilde{\delta}_x] + \mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s].$$

Hence, we have that

$$\begin{aligned}
&\|\mu^{-1}\,\mathbf{E}[\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu]\|_2 \\
\le\; &\|\mu^{-1}(x-\overline{x})s \cdot s^{-1}\,\mathbf{E}[\widetilde{\delta}_s]\|_2 + \|\mu^{-1}(s-\overline{s})x \cdot x^{-1}\,\mathbf{E}[\widetilde{\delta}_x]\|_2 + \|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2 \\
\le\; &\|\mu^{-1}(x-\overline{x})s\|_\infty \cdot \|s^{-1}\,\mathbf{E}[\widetilde{\delta}_s]\|_2 + \|\mu^{-1}(s-\overline{s})x\|_\infty \cdot \|x^{-1}\,\mathbf{E}[\widetilde{\delta}_x]\|_2 + \|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2 \\
\le\; &\epsilon_{mp} \cdot \|s^{-1}\,\mathbf{E}[\widetilde{\delta}_s]\|_2 + \epsilon_{mp} \cdot \|x^{-1}\,\mathbf{E}[\widetilde{\delta}_x]\|_2 + \|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2 \\
\le\; &4\epsilon_{mp} \cdot \epsilon + \|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2, \qquad (12)
\end{aligned}$$

where the first step follows by triangle inequality, the second step follows by $\|ab\|_2 \le \|a\|_\infty \cdot \|b\|_2$, the third step follows by $\|\mu^{-1}(x-\overline{x})s\|_\infty \le \epsilon_{mp}$ and $\|\mu^{-1}(s-\overline{s})x\|_\infty \le \epsilon_{mp}$ (since $\overline{x} \approx_{\epsilon_{mp}} x$, $\overline{s} \approx_{\epsilon_{mp}} s$), the last step follows by $\|\mathbf{E}[s^{-1}\widetilde{\delta}_s]\|_2 \le 2\epsilon$ and $\|\mathbf{E}[x^{-1}\widetilde{\delta}_x]\|_2 \le 2\epsilon$ (Part 1 of Lemma 4.3).

To bound the last term, using $\mathbf{E}[\widetilde{\delta}_s] = \delta_s$ and $\mathbf{E}[\widetilde{\delta}_x] = \delta_x$, we note that

$$\mathbf{E}[\widetilde{\delta}_{x,i}\widetilde{\delta}_{s,i}] = \delta_{x,i}\delta_{s,i} + \mathbf{E}[(\widetilde{\delta}_{x,i} - \delta_{x,i})(\widetilde{\delta}_{s,i} - \delta_{s,i})].$$

Hence, we have

$$\begin{aligned}
\|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2 &\le \|\mu^{-1}\delta_x\delta_s\|_2 + \left(\sum_{i=1}^n \left(\mathbf{E}\left[x_i^{-1}(\widetilde{\delta}_{x,i} - \delta_{x,i}) \cdot s_i^{-1}(\widetilde{\delta}_{s,i} - \delta_{s,i})\right]\right)^2\right)^{1/2} \\
&\le 4\epsilon^2 + \frac{1}{2}\left(\sum_{i=1}^n \left(\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}] + \mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}]\right)^2\right)^{1/2} \\
&\le 4\epsilon^2 + \frac{1}{2}\left(\sum_{i=1}^n 2(\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}])^2 + 2(\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}])^2\right)^{1/2} \\
&\le 4\epsilon^2 + 2\sqrt{n \cdot \epsilon^4/k^2} \le 4\epsilon^2 + 2\epsilon \cdot \epsilon_{mp} \le 6\epsilon \cdot \epsilon_{mp}, \qquad (13)
\end{aligned}$$

where the second step follows by $\|\mu^{-1}\delta_x\delta_s\|_2 \le \|x^{-1}\delta_x\|_2 \cdot \|s^{-1}\delta_s\|_2 \le 4\epsilon^2$ (Part 1 of Lemma 4.3) and $2ab \le a^2 + b^2$, the third step follows by $(a+b)^2 \le 2a^2 + 2b^2$, the fourth step follows by $\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}] \le 2\epsilon^2/k$ and $\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] \le 2\epsilon^2/k$ (Part 2 of Lemma 4.3), the last step follows by $k \ge \frac{\epsilon\sqrt{n}}{\epsilon_{mp}}$.

Combining (12) and (13), we have that

$$\|\mu^{-1}(\mathbf{E}[\mu^{\text{new}} - \mu - \widetilde{\delta}_\mu])\|_2 \le 4\epsilon_{mp} \cdot \epsilon + \|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_x\widetilde{\delta}_s]\|_2 \le 10\epsilon_{mp} \cdot \epsilon.$$

where we used $\epsilon \le \epsilon_{mp}$.

From Part 1 of Lemma 4.3, we know that $\|\mu^{-1}\,\mathbf{E}[\widetilde{\delta}_\mu]\|_2 \le 4\epsilon$. Thus using triangle inequality, we know

$$\|\mu^{-1}(\mathbf{E}[\mu^{\text{new}} - \mu])\|_2 \le 10\epsilon_{mp} \cdot \epsilon + 4\epsilon \le 5\epsilon.$$

□

Claim 4.10 (Part 2 of Lemma 4.8)**.** $\mathbf{Var}[\mu_i^{-1}\mu_i^{\text{new}}] \le 50\epsilon^2/k$ *for all* i.

Proof. Recall that

$$\mu^{\text{new}} = \mu + \widetilde{\delta}_\mu + (x - \overline{x})\widetilde{\delta}_s + (s - \overline{s})\widetilde{\delta}_x + \widetilde{\delta}_x\widetilde{\delta}_s.$$

We can upper bound the variance of $\mu_i^{-1}\mu_i^{\text{new}}$,

$$\begin{aligned}
\mathbf{Var}[\mu_i^{-1}\mu_i^{\text{new}}] &\le 4\,\mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{\mu,i}] + 4\,\mathbf{Var}[\mu_i^{-1}(x_i - \overline{x}_i)\widetilde{\delta}_{s,i}] + 4\,\mathbf{Var}[\mu_i^{-1}(s_i - \overline{s}_i)\widetilde{\delta}_{x,i}] + 4\,\mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{x,i}\widetilde{\delta}_{s,i}] \\
&\le 32\frac{\epsilon^2}{k} + 4\frac{\epsilon^2}{k} + 4\frac{\epsilon^2}{k} + \mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{x,i}\widetilde{\delta}_{s,i}] \\
&= 40\frac{\epsilon^2}{k} + \mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i} \cdot s_i^{-1}\widetilde{\delta}_{s,i}] \\
&\le 40\frac{\epsilon^2}{k} + 2\,\mathbf{Sup}[(x_i^{-1}\widetilde{\delta}_{x,i})^2] \cdot \mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] + 2\,\mathbf{Sup}[(y_i^{-1}\widetilde{\delta}_{y,i})^2] \cdot \mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}] \\
&\le 40\frac{\epsilon^2}{k} + 2 \cdot (\frac{0.02}{\log n})^2 \cdot \frac{\epsilon^2}{k} + 2 \cdot (\frac{0.02}{\log n})^2 \cdot \frac{\epsilon^2}{k} \le 50\frac{\epsilon^2}{k}.
\end{aligned}$$

where the second step follows by $\mathbf{Var}[\mu_i^{-1}\widetilde{\delta}_{\mu,i}] \le 8\epsilon^2/k$ (Part 2 of Lemma 4.3),

$$\mathbf{Var}[\mu_i^{-1}(x_i - \overline{x}_i)\widetilde{\delta}_{s,i}] = \mathbf{Var}[x_i^{-1}(x_i - \overline{x}_i)s_i^{-1}\widetilde{\delta}_{s,i}] \le 2\epsilon_{mp}^2\,\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] \le \epsilon^2/k.$$

and a similar inequality $\mathbf{Var}[\mu_i^{-1}(s_i - \overline{s}_i)\widetilde{\delta}_{x,i}] \le \epsilon^2/k$, the third step follows by $\mu = xs$, the fourth step follows by $\mathbf{Var}[xy] \le 2\,\mathbf{Sup}[x^2]\,\mathbf{Var}[y] + 2\,\mathbf{Sup}[y^2]\,\mathbf{Var}[x]$ (Lemma A.1) with **Sup** denoting the deterministic maximum of the random variable, the fifth step follows by $\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] \le 2\epsilon^2/k$ and $\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}] \le 2\epsilon^2/k$ (Part 2 of Lemma 4.3). □

Claim 4.11 (Part 3 of Lemma 4.8)**.** $\|\mu^{-1}(\mu^{\text{new}} - \mu)\|_\infty \le \frac{0.021}{\log n}$.

Proof. We again note that

$$\mu^{\text{new}} = \mu + \widetilde{\delta}_\mu + (x - \overline{x})\widetilde{\delta}_s + (s - \overline{s})\widetilde{\delta}_x + \widetilde{\delta}_x\widetilde{\delta}_s.$$

Hence, we have

$$\begin{aligned}
&|\mu_i^{-1}(\mu_i^{\text{new}} - \mu_i - \widetilde{\delta}_{\mu,i})| \\
\le\ &|(x - \overline{x})_i\mu_i^{-1}\widetilde{\delta}_{s,i}| + |(s - \overline{s})_i\mu_i^{-1}\widetilde{\delta}_{x,i}| + |\mu_i^{-1}\widetilde{\delta}_{x,i}\widetilde{\delta}_{s,i}| \\
=\ &|(x - \overline{x})_i x_i^{-1}| \cdot |s_i^{-1}\widetilde{\delta}_{s,i}| + |(s - \overline{s})_i s_i^{-1}| \cdot |x_i^{-1}\widetilde{\delta}_{x,i}| + |x_i^{-1}\widetilde{\delta}_{x,i}| \cdot |s_i^{-1}\widetilde{\delta}_{s,i}| \\
\le\ &\epsilon_{mp}|s_i^{-1}\widetilde{\delta}_{s,i}| + \epsilon_{mp}|x_i^{-1}\widetilde{\delta}_{x,i}| + |s_i^{-1}\widetilde{\delta}_{s,i}||x_i^{-1}\widetilde{\delta}_{x,i}| \\
\le\ &\epsilon_{mp} \cdot \frac{0.2}{\log n} + \epsilon_{mp} \cdot \frac{0.02}{\log n} + (\frac{0.02}{\log n})^2 \le \frac{1}{1000\log n},
\end{aligned}$$

where the first step follows by triangle inequality, the second step follows by $\mu_i = x_i s_i$, the third step follows by $x \approx_{\epsilon_{mp}} \overline{x}$ and $s \approx_{\epsilon_{mp}} \overline{s}$, the fifth step follows by $|s_i^{-1}\widetilde{\delta}_{s,i}| \leq \frac{0.02}{\log n}$ and $|x_i^{-1}\widetilde{\delta}_{x,i}| \leq \frac{0.02}{\log n}$ (Part 3 of Lemma 4.3).

Since we know that $|\mu_i^{-1}\widetilde{\delta}_{\mu,i}| \leq \frac{0.02}{\log n}$ (Part 3 of Lemma 4.3), we have

$$|\mu_i^{-1}(\mu_i^{\text{new}} - \mu_i)| \leq \frac{1}{1000\log n} + \frac{0.02}{\log n} \leq \frac{0.021}{\log n}.$$

□

4.3 Stochastic central path

Now, we are ready to prove $x_i s_i \approx_{0.1} t$ during the whole algorithm. As explained in the proof outline (see Section 4.1), we will prove this bound by analyzing the potential $\Phi_\lambda(\mu/t - 1)$ where $\Phi_\lambda(r) = \sum_{i=1}^n \cosh(\lambda r_i)$.

First, we give some basic properties of Φ_λ.

Lemma 4.12 (Basic properties of potential function)**.** *Let $\Phi_\lambda(r) = \sum_{i=1}^n \cosh(\lambda r_i)$ for some $\lambda > 0$. For any vector $r \in \mathbb{R}^n$,*

1. For any vector $\|v\|_\infty \leq 1/\lambda$, we have that

$$\Phi_\lambda(r+v) \leq \Phi_\lambda(r) + \langle \nabla\Phi_\lambda(r), v\rangle + 2\|v\|^2_{\nabla^2\Phi_\lambda(r)}.$$

2. $\|\nabla\Phi_\lambda(r)\|_2 \geq \frac{\lambda}{\sqrt{n}}(\Phi_\lambda(r) - n)$.

3. $\left(\sum_{i=1}^n \lambda^2\cosh^2(\lambda r_i)\right)^{1/2} \leq \lambda\sqrt{n} + \|\nabla\Phi_\lambda(r)\|_2$.

Proof. For each $i \in [n]$, we use r_i to denote the i-th coordinate of vector r.

Proof of Part 1. Since $\|v\|_\infty \leq \frac{1}{2\lambda}$, we have that

$$\cosh(\lambda(r_i+v_i)) = \cosh(\lambda r_i) + \lambda\sinh(\lambda r_i)v_i + \frac{\lambda^2}{2}\cosh(\zeta_i)v_i^2,$$

where ζ_i is between λr_i and $\lambda(r_i+v_i)$. By definition of cosh, we have that

$$\cosh(\zeta_i) = \frac{1}{2}\exp(\zeta_i) + \frac{1}{2}\exp(-\zeta_i) \leq \exp(1)\cdot\frac{1}{2}(\exp(\lambda r_i) + \exp(-\lambda r_i)) \leq 3\cosh(\lambda r_i).$$

Hence, we have

$$\cosh(\lambda(r_i+v_i)) \leq \cosh(\lambda r_i) + \lambda\sinh(\lambda r_i)v_i + 2\lambda^2\cosh(\lambda r_i)v_i^2.$$

Summing over all the coordinates gives

$$\begin{aligned}
&\sum_{i=1}^n \cosh(\lambda(r_i+v_i)) \leq \sum_{i=1}^n \left[\cosh(\lambda r_i) + 2\lambda\sinh(\lambda r_i)v_i + \lambda^2\cosh(\lambda r_i)v_i^2\right] \\
\implies\ & \Phi_\lambda(r+v) \leq \Phi_\lambda(r) + \langle\nabla\Phi_\lambda(r), v\rangle + 2\|v\|^2_{\nabla^2\Phi_\lambda(r)}.
\end{aligned}$$

Proof of Part 2. Since $\Phi_\lambda(r) = \sum_{i=1}^n \cosh(\lambda r_i)$, then

$$\nabla\Phi_\lambda(r) = \begin{bmatrix}\lambda\sinh(\lambda r_1) & \lambda\sinh(\lambda r_2) & \cdots & \lambda\sinh(\lambda r_n)\end{bmatrix}^\top.$$

Thus, we can lower bound $\|\nabla\Phi_\lambda(r)\|_2$ in the following way,

$$\begin{aligned}
\|\nabla\Phi_\lambda(r)\|_2 &= \left(\sum_{i=1}^n \lambda^2 \sinh^2(\lambda r_i)\right)^{1/2} \\
&= \left(\sum_{i=1}^n \lambda^2 (\cosh^2(\lambda r_i) - 1)\right)^{1/2} && \text{by } \cosh^2(y) - \sinh^2(y) = 1, \forall y \\
&\geq \frac{\lambda}{\sqrt{n}} \sum_{i=1}^n \sqrt{\cosh^2(\lambda r_i) - 1} && \text{by } \|\cdot\|_2 \geq \frac{1}{\sqrt{n}}\|\cdot\|_1 \\
&\geq \frac{\lambda}{\sqrt{n}} \sum_{i=1}^n (\cosh(\lambda r_i) - 1) && \text{by } \cosh(\lambda r_i) \geq 1 \\
&= \frac{\lambda}{\sqrt{n}} (\Phi_\lambda(r) - n) && \text{by def of } \Phi(r)
\end{aligned}$$

Proof of Part 3.

$$\begin{aligned}
\left(\sum_{i=1}^n \lambda^2 \cosh^2(\lambda r_i)\right)^{1/2} &= \left(\sum_{i=1}^n \lambda^2 + \lambda^2 \sinh^2(\lambda r_i)\right)^{1/2} && \text{by } \cosh^2(y) - \sinh^2(y) = 1, \forall y \\
&\leq (n\lambda^2)^{1/2} + \left(\sum_{i=1}^n \lambda^2 \sinh^2(\lambda r_i)\right)^{1/2} \\
&= \lambda\sqrt{n} + \|\nabla\Phi_\lambda(r)\|_2.
\end{aligned}$$

□

The following lemma shows that the potential Φ is decreasing in expectation when Φ is large.

Lemma 4.13. *Under the Assumption 4.1, we have*

$$\mathbf{E}\left[\Phi_\lambda\left(\frac{\mu^{\text{new}}}{t^{\text{new}}} - 1\right)\right] \leq \Phi_\lambda\left(\frac{\mu}{t} - 1\right) - \frac{\lambda\epsilon}{15\sqrt{n}}\left(\Phi_\lambda\left(\frac{\mu}{t} - 1\right) - 10n\right).$$

Proof. Let $\widehat{\delta}_\mu = \mu^{\text{new}} - \mu - \widetilde{\delta}_\mu$. From the definition, we have

$$\mu^{\text{new}} - t^{\text{new}} = \mu + \widetilde{\delta}_\mu + \widehat{\delta}_\mu - t^{\text{new}}$$

which implies

$$\begin{aligned}
\frac{\mu^{\text{new}}}{t^{\text{new}}} - 1 &= \frac{\mu}{t^{\text{new}}} + \frac{1}{t^{\text{new}}}(\widetilde{\delta}_\mu + \widehat{\delta}_\mu) - 1 \\
&= \frac{\mu}{t}\frac{t}{t^{\text{new}}} + \frac{1}{t^{\text{new}}}(\widetilde{\delta}_\mu + \widehat{\delta}_\mu) - 1 \\
&= \frac{\mu}{t} + \frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1) + \frac{1}{t^{\text{new}}}(\widetilde{\delta}_\mu + \widehat{\delta}_\mu) - 1 \\
&= \frac{\mu}{t} - 1 + \underbrace{\frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1) + \frac{1}{t^{\text{new}}}(\widetilde{\delta}_\mu + \widehat{\delta}_\mu)}_{v}.
\end{aligned} \tag{14}$$

To apply Lemma 4.12 with $r = \mu/t - 1$ and $r + v = \mu^{\text{new}}/t^{\text{new}} - 1$, we first compute the expectation of v

$$\begin{aligned}
\mathbf{E}[v] &= \frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1) + \frac{1}{t^{\text{new}}}(\mathbf{E}[\widetilde{\delta}_\mu] + \mathbf{E}[\widehat{\delta}_\mu]) \\
&= \frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1) + \frac{1}{t^{\text{new}}}(\delta_\mu + \mathbf{E}[\widehat{\delta}_\mu]) \\
&= \frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1) + \frac{1}{t^{\text{new}}}\left(\left((\frac{t^{\text{new}}}{t} - 1)\mu - \frac{\epsilon}{2}t^{\text{new}}\frac{\nabla\Phi_\lambda(\mu/t-1)}{\|\nabla\Phi_\lambda(\mu/t-1)\|_2}\right) + \mathbf{E}[\widehat{\delta}_\mu]\right) \\
&= -\frac{\epsilon}{2}\frac{\nabla\Phi_\lambda(\mu/t-1)}{\|\nabla\Phi_\lambda(\mu/t-1)\|_2} + \frac{1}{t^{\text{new}}}\mathbf{E}[\widehat{\delta}_\mu] \qquad (15)
\end{aligned}$$

where the third step follows by definition of δ_μ.

Next, we bound the $\|v\|_\infty$ as follows

$$\begin{aligned}
\|v\|_\infty &\le \left\|\frac{\mu}{t}(\frac{t}{t^{\text{new}}} - 1)\right\|_\infty + \left\|\frac{1}{t^{\text{new}}}(\widetilde{\delta}_\mu + \widehat{\delta}_\mu)\right\|_\infty \le \frac{\epsilon}{\sqrt{n}} + \frac{\|\mu^{-1}(\mu^{\text{new}} - \mu)\|_\infty}{0.9} \\
&\le \frac{\epsilon}{\sqrt{n}} + \frac{0.021}{0.9\log n} \le \frac{1}{\lambda}.
\end{aligned}$$

where we used Part 3 of Lemma 4.8 and $\epsilon \le \frac{1}{400\log n}$.

Since $\|v\|_\infty \le \frac{1}{\lambda}$, we can apply Part 1 of Lemma 4.12 and get

$$\begin{aligned}
&\mathbf{E}[\Phi_\lambda(\mu/t + v - 1)] \\
\le\ & \Phi_\lambda(\mu/t-1) + \langle\nabla\Phi_\lambda(\mu/t-1), \mathbf{E}[v]\rangle + 2\|\,\mathbf{E}[v]\|^2_{\nabla^2\Phi_\lambda(\mu/t+v-1)} \\
=\ & \Phi_\lambda(\mu/t-1) - \frac{\epsilon}{2}\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + \frac{t}{t^{\text{new}}}\langle\nabla\Phi_\lambda(\mu/t-1), \mathbf{E}[t^{-1}\widehat{\delta}_\mu]\rangle + 2\|\,\mathbf{E}[v]\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)} \\
\le\ & \Phi_\lambda(\mu/t-1) - \frac{\epsilon}{2}\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + \frac{t}{t^{\text{new}}}\|\nabla\Phi_\lambda(\mu/t-1)\|_2 \cdot \|\,\mathbf{E}[t^{-1}\widehat{\delta}_\mu]\|_2 + 2\|\,\mathbf{E}[v]\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)} \\
\le\ & \Phi_\lambda(\mu/t-1) - \frac{\epsilon}{2}\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + 10\epsilon_{mp}\cdot\epsilon\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + 2\,\mathbf{E}[\|v\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)}],
\end{aligned}$$

where the second step follows by substituting $\mathbf{E}[v]$ by (15), the third step follows by $\langle a, b\rangle \le \|a\|_2 \cdot \|b\|_2$, the fourth step follows by $\|\,\mathbf{E}[t^{-1}\widehat{\delta}_\mu]\|_2 \le 10\epsilon_{mp}\cdot\epsilon$ (from Part 1 of Lemma 4.8 and $\mu \approx_{0.1} t$).

We still need to bound $\mathbf{E}[\|v\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)}]$. Before bounding it, we first bound $\mathbf{E}[v_i^2]$,

$$\begin{aligned}
\mathbf{E}[v_i^2] &\le 2\,\mathbf{E}\left[\left(\frac{\mu_i}{t}(\frac{t}{t^{\text{new}}} - 1)\right)^2\right] + 2\,\mathbf{E}\left[\left(\frac{1}{t^{\text{new}}}(\widetilde{\delta}_{\mu,i} + \widehat{\delta}_{\mu,i})\right)^2\right] \\
&\le \epsilon^2/n + 2.5\,\mathbf{E}\left[((\mu_i^{\text{new}} - \mu_i)/\mu_i)^2\right] \\
&= \epsilon^2/n + 2.5\,\mathbf{Var}[(\mu_i^{\text{new}} - \mu_i)/\mu_i] + 2.5(\mathbf{E}[(\mu_i^{\text{new}} - \mu_i)/\mu_i])^2 \\
&\le \epsilon^2/n + 125\epsilon^2/k + 2.5(\mathbf{E}[(\mu_i^{\text{new}} - \mu_i)/\mu_i])^2 \\
&\le 126\epsilon^2/k + 3(\mathbf{E}[(\mu_i^{\text{new}} - \mu_i)/\mu_i])^2 \qquad (16)
\end{aligned}$$

where the first step follows by definition of v (see (14)), the second step follows by $\mu \approx_{0.1} t$ and $(t/t^{\text{new}} - 1)^2 \le \epsilon^2/(4n)$, the third step follows by $\mathbf{E}[x^2] = \mathbf{Var}[x] + (\mathbf{E}[x])^2$, the fourth step follows by Part 2 of Lemma 4.8, and the last step follows by $n \ge k$.

Now, we are ready to bound $\mathbf{E}[\|v\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)}]$

$$
\begin{aligned}
&\mathbf{E}[\|v\|^2_{\nabla^2\Phi_\lambda(\mu/t-1)}] \\
=& \lambda^2 \sum_{i=1}^n \mathbf{E}[\Phi_\lambda(\mu/t-1)_i v_i^2] \\
\leq& \lambda^2 \sum_{i=1}^n \Phi_\lambda(\mu/t-1)_i \cdot (126\epsilon^2/k + 3(\mathbf{E}[(\mu_i^{\text{new}}-\mu_i)/\mu_i])^2) \\
=& 126\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) + 3\lambda^2\sum_{i=1}^n \Phi_\lambda(\mu/t-1)_i \cdot (\mathbf{E}[(\mu_i^{\text{new}}-\mu_i)/\mu_i])^2 \\
\leq& 126\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) + 3\lambda\left(\sum_{i=1}^n \lambda^2\Phi_\lambda(\mu/t-1)_i^2\right)^{1/2} \cdot \|\mathbf{E}[\mu^{-1}(\mu^{\text{new}}-\mu)]\|_4^2 \\
\leq& 126\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) + 3\lambda\left(\lambda\sqrt{n} + \|\nabla\Phi_\lambda(\mu/t-1)\|_2\right)\cdot(5\epsilon)^2,
\end{aligned}
$$

where the first step follows by defining $\Phi_\lambda(x)_i = \cosh(\lambda x_i)$, the second step follows from (16), the fourth step follows from Cauchy-Schwarz inequality, the fifth step follows from Part 3 of Lemma 4.12 and the fact that $\|\mathbf{E}[\mu^{-1}(\mu^{\text{new}}-\mu)]\|_4^2 \leq \|\mathbf{E}[\mu^{-1}(\mu^{\text{new}}-\mu)]\|_2^2 \leq (5\epsilon)^2$ (Lemma 4.8).

Then,

$$
\begin{aligned}
&\mathbf{E}[\Phi_\lambda(\mu/t+v-1)] \\
\leq& \Phi_\lambda(\mu/t-1) - (\frac{\epsilon}{2} - 10\epsilon_{mp}\cdot\epsilon)\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + 252\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) \\
&+ 150\lambda^2\epsilon^2\sqrt{n} + 150\lambda\epsilon^2\|\Phi_\lambda(\mu/t-1)\|_2 \\
\leq& \Phi_\lambda(\mu/t-1) - \frac{\epsilon}{3}\|\nabla\Phi_\lambda(\mu/t-1)\|_2 + 252\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) + 150\lambda^2\epsilon^2\sqrt{n} \\
\leq& \Phi_\lambda(\mu/t-1) - \frac{\lambda\epsilon}{3\sqrt{n}}(\Phi_\lambda(\mu/t-1)-n) + 252\frac{\lambda^2\epsilon^2}{k}\Phi_\lambda(\mu/t-1) + 150\lambda^2\epsilon^2\sqrt{n} \\
\leq& \Phi_\lambda(\mu/t-1) - \frac{\lambda\epsilon}{3\sqrt{n}}(\Phi_\lambda(\mu/t-1)/5 - 2n),
\end{aligned}
$$

where the second step follows from $1000\lambda\epsilon \leq 1$ and $1000\epsilon_{mp} \leq 1$, the third step follows from Part 2 of Lemma 4.12, and the last step follows from $1000\lambda\epsilon_{mp} \leq \log n$ and $k \geq \frac{\sqrt{n}\epsilon\log n}{\epsilon_{mp}}$. ☐

As a corollary, we have the following:

Lemma 4.14. *During the* Main *algorithm, Assumption 4.1 is always satisfied. Furthermore, the* ClassicalStep *happens with probability $O(\frac{1}{n^2})$ each step.*

Proof. The second and the fourth assumptions simply follow from the choice of ϵ_{mp} and k.

Let $\Phi^{(k)}$ be the potential at the k-th iteration of the Main. The ClassicalStep ensures that $\Phi^{(k)} \leq n^3$ at the end of each iteration. By the definition of Φ and the choice of λ in Main, we have that

$$\left\|\frac{xs}{t} - 1\right\|_\infty \leq \frac{\ln(2n^3)}{\lambda} \leq 0.1.$$

This proves the first assumption $xs \approx_{0.1} t$ with $t > 0$.

For the third assumption, we note that

$$\begin{aligned}\|\delta_\mu\|_2 &= \left\| \left(\frac{t^{\text{new}}}{t} - 1\right) xs - \frac{\epsilon}{2} \cdot t^{\text{new}} \cdot \frac{\nabla \Phi_\lambda(\mu/t-1)}{\|\nabla \Phi_\lambda(\mu/t-1)\|_2} \right\|_2 \\ &\le \left| \frac{t^{\text{new}}}{t} - 1 \right| \|xs\|_2 + \frac{\epsilon}{2} t^{\text{new}} \\ &\le \frac{\epsilon}{3\sqrt{n}} \cdot 1.1\sqrt{n}t + 1.01 \cdot \frac{\epsilon}{2} t \le \epsilon t,\end{aligned}$$

where we used $xs \approx_{0.1} t$ and the formula of t^{new}. Hence, we proved all assumptions in Assumption 4.1.

Now, we bound the probability that ClassicalStep happens. In the beginning of the Main, Lemma A.6 is used to modify the linear program with parameter $\min(\frac{\delta}{2}, \frac{1}{\lambda})$. Hence, the initial point x and s satisfies $xs \approx_{1/\lambda} 1$. Therefore, we have $\Phi^{(0)} \le 10n$. Lemma 4.13 shows $\mathbf{E}[\Phi^{(k+1)}] \le (1 - \frac{\lambda\epsilon}{15\sqrt{n}}) \mathbf{E}[\Phi^{(k)}] + \frac{\lambda\epsilon}{15\sqrt{n}} 10n$. By induction, we have that $\mathbf{E}[\Phi^{(k)}] \le 10n$ for all k. Since the potential is positive, Markov inequality shows that for any k, $\Phi^{(k)} \ge n^3$ with probability at most $O(\frac{1}{n^2})$. □

4.4 Analysis of cost per iteration

To apply the data structure for projection maintenance (Theorem 5.1), we need to first prove the input vector w does not change too much for each step.

Lemma 4.15. *Let $x^{\text{new}} = x + \widetilde{\delta}_x$ and $s^{\text{new}} = s + \widetilde{\delta}_s$. Let $w = \frac{x}{s}$ and $w^{\text{new}} = \frac{x^{\text{new}}}{s^{\text{new}}}$. Then we have*

$$\sum_{i=1}^n \left(\frac{\mathbf{E}[w_i^{\text{new}}] - w_i}{w_i} \right)^2 \le 64\epsilon^2, \sum_{i=1}^n \left(\mathbf{E}\left[\left(\frac{w_i^{\text{new}} - w_i}{w_i} \right)^2 \right] \right)^2 \le 1000\epsilon^2, \left| \frac{w_i^{\text{new}} - w_i}{w_i} \right| \le 0.1.$$

Proof. From the definition, we know that

$$\frac{w_i^{\text{new}}}{w_i} = \frac{1}{s_i^{-1} x_i} \frac{x_i + \widetilde{\delta}_{x,i}}{s_i + \widetilde{\delta}_{s,i}} = \frac{1 + x_i^{-1}\widetilde{\delta}_{x,i}}{1 + s_i^{-1}\widetilde{\delta}_{s,i}}.$$

Part 1. For each $i \in [n]$, we have

$$\begin{aligned}\frac{\mathbf{E}[w_i^{\text{new}}]}{w_i} - 1 &= \mathbf{E}\left[\frac{1 + x_i^{-1}\widetilde{\delta}_{x,i}}{1 + s_i^{-1}\widetilde{\delta}_{s,i}} \right] - 1 \\ &= \mathbf{E}\left[\frac{x_i^{-1}\widetilde{\delta}_{x,i} - s_i^{-1}\widetilde{\delta}_{s,i}}{1 + s_i^{-1}\widetilde{\delta}_{s,i}} \right] \\ &\le 2 |\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i} - s_i^{-1}\widetilde{\delta}_{s,i}]| && \text{by } |s_i^{-1}\widetilde{\delta}_{s,i}| \le 0.2, \text{ part 3 of Lemma 4.3} \\ &\le 2 |\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i}]| + 2 |\mathbf{E}[s_i^{-1}\widetilde{\delta}_{s,i}]|. && \text{by triangle inequality}\end{aligned}$$

Thus, summing over all the coordinates gives

$$\sum_{i=1}^n \left(\frac{\mathbf{E}[w_i^{\text{new}}] - w_i}{w_i} \right)^2 \le \sum_{i=1}^n 8(\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i}])^2 + 8(\mathbf{E}[s_i^{-1}\widetilde{\delta}_{s,i}])^2 \le 64\epsilon^2.$$

where the first step follows by triangle inequality, the last step follows by $\|\mathbf{E}[s^{-1}\widetilde{\delta}_s]\|_2^2, \|\mathbf{E}[x^{-1}\widetilde{\delta}_x]\|_2^2 \le 4\epsilon^2$ (Part 1 of Lemma 4.3).

Part 2. For each $i \in [n]$, we have

$$\begin{aligned}
\mathbf{E}\left[\left(\frac{w_i^{\text{new}}}{w_i}-1\right)^2\right] &= \mathbf{E}\left[\left(\frac{x_i^{-1}\widetilde{\delta}_{x,i}-s_i^{-1}\widetilde{\delta}_{s,i}}{1+s_i^{-1}\widetilde{\delta}_{s,i}}\right)^2\right] \\
&\leq 2\,\mathbf{E}[(x_i^{-1}\widetilde{\delta}_{x,i}-s_i^{-1}\widetilde{\delta}_{s,i})^2] \\
&\leq 2\,\mathbf{E}[2(x_i^{-1}\widetilde{\delta}_{x,i})^2+2(s_i^{-1}\widetilde{\delta}_{s,i})^2] \\
&= 4\,\mathbf{E}[(x_i^{-1}\widetilde{\delta}_{x,i})^2]+4\,\mathbf{E}[(s_i^{-1}\widetilde{\delta}_{s,i})^2] \\
&= 4\,\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}]+4(\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i}])^2+4\,\mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}]+4(\mathbf{E}[s_i^{-1}\widetilde{\delta}_{s,i}])^2 \\
&\leq 16\epsilon^2/k+4(\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i}])^2+4(\mathbf{E}[s_i^{-1}\widetilde{\delta}_{s,i}])^2,
\end{aligned}$$

where the last step follows by $\mathbf{Var}[x_i^{-1}\widetilde{\delta}_{x,i}], \mathbf{Var}[s_i^{-1}\widetilde{\delta}_{s,i}] \leq 2\epsilon^2/k$ (Part 2 of Lemma 4.3).

Thus summing over all the coordinates

$$\begin{aligned}
\sum_{i=1}^n \left(\mathbf{E}\left[\left(\frac{w_i^{\text{new}}}{w_i}-1\right)^2\right]\right)^2 &\leq \frac{512n\epsilon^4}{k^2}+64\sum_{i=1}^n\left((\mathbf{E}[x_i^{-1}\widetilde{\delta}_{x,i}])^4+(\mathbf{E}[s_i^{-1}\widetilde{\delta}_{s,i}])^4\right) \\
&\leq \frac{512n\epsilon^4}{k^2}+2048\epsilon^4 \leq 1000\epsilon^2,
\end{aligned}$$

where the last step follows by $\|\mathbf{E}[s^{-1}\widetilde{\delta}_s]\|_2^2, \|\mathbf{E}[x^{-1}\widetilde{\delta}_x]\|_2^2 \leq 4\epsilon^2$ and $k \geq \sqrt{n}\epsilon$.

Part 3. For each $i \in [n]$

$$\left|\frac{w_i^{\text{new}}}{w_i}-1\right| = \left|\frac{1+x_i^{-1}\widetilde{\delta}_{x,i}}{1+s_i^{-1}\widetilde{\delta}_{s,i}}-1\right| \leq \left|\frac{1+0.02}{1-0.02}-1\right| \leq 0.1.$$

where the second step follows by $|x_i^{-1}\widetilde{\delta}_{x,i}| \leq 0.02$ and $|s_i^{-1}\widetilde{\delta}_{s,i}| \leq 0.02$ (Part 3 of Lemma 4.3). □

Now, we analyze the cost per iteration in procedure MAIN. This is a direct application of our projection maintenance result.

Lemma 4.16. *For $\epsilon \geq \frac{1}{\sqrt{n}}$, each iteration of* MAIN *(Algorithm 2) takes*

$$n^{1+a+o(1)}+\epsilon\cdot(n^{\omega-1/2+o(1)}+n^{2-a/2+o(1)})$$

expected time per iteration in amortized.

Proof. Lemma 4.14 shows that CLASSICALSTEP happens with only $O(1/n^2)$ probability each step. Since the cost of each step only takes $\widetilde{O}(n^{2.5})$, the expected cost is only $\widetilde{O}(n^{0.5})$.

Lemma 4.15 shows that the conditions in Theorem 5.1 holds with the parameter $C_1 = O(\epsilon), C_2 = O(\epsilon), \epsilon_{mp} = \Theta(1)$.

In the procedure STOCHASTICSTEP, Theorem 5.1 shows that the amortized time per iteration is mainly dominated by two steps:

1. mp.UPDATE(w): $O(\epsilon\cdot(n^{\omega-1/2+o(1)}+n^{2-a/2+o(1)}))$.
2. mp.QUERY($\frac{1}{\sqrt{XS}}\widetilde{\delta}_\mu$): $O(n\cdot\|\widetilde{\delta}_\mu\|_0+n^{1+a+o(1)})$.

Combining both running time and using $\mathbf{E}[\|\widetilde{\delta}_\mu\|_0] = O(1+k) = O(\epsilon\sqrt{n}\log^2 n)$ (according to the probability of success in Claim 4.7 and matching Assumption 4.1), we have the result.

□

4.5 Main result

Proof of Theorem 2.1. In the beginning of the MAIN algorithm, Lemma A.6 is called to modify the linear program. Then, we run the stochastic central path method on this modified linear program.

When the algorithm stops, we obtain a vector x and s such that $xs \approx_{0.1} t$ with $t \leq \frac{\delta^2}{2n}$. Hence, the duality gap is bounded by $\sum_i x_i s_i \leq \delta^2$. Lemma A.6 shows how to obtain an approximate solution of the original linear program with the guarantee needed using the x and s we just found.

Since t is decreased by $1 - \frac{\epsilon}{3\sqrt{n}}$ factor each iteration, it takes $O(\frac{\sqrt{n}}{\epsilon} \cdot \log(\frac{n}{\delta}))$ iterations in total. In Lemma 4.16, we proved that each iteration takes

$$n^{1+a+o(1)} + \epsilon \cdot (n^{\omega-1/2+o(1)} + n^{2-a/2+o(1)}).$$

and hence the total runtime is

$$O(n^{2.5-a/2+o(1)} + n^{\omega-1/2+o(1)} + \frac{n^{1.5+a+o(1)}}{\epsilon}) \cdot \log(\frac{n}{\delta}).$$

Since $\epsilon = \Theta(\frac{1}{\log n})$, the total runtime is

$$O(n^{2.5-a/2+o(1)} + n^{\omega-1/2+o(1)} + n^{1.5+a+o(1)}) \cdot \log(\frac{n}{\delta}).$$

Finally, we note that the optimal choice of a is $\min(\frac{2}{3}, \alpha)$, which gives the promised runtime. □

Using the same proof, but different choice of the parameters, we can analyze the ultra short step stochastic central path method, where each step involves sampling only polylogarithmic coordinates. As we mentioned before, the runtime is still around n^ω.

Corollary 4.17. *Under the same assumption as Theorem 2.1, if we choose $\epsilon = \Theta(1/\sqrt{n})$ and $a = \min(\frac{1}{3}, \alpha)$, the expected time of* MAIN *(Algorithm 2) is*

$$\left(n^{\omega+o(1)} + n^{2.5-\alpha/2+o(1)} + n^{2+1/3+o(1)}\right) \cdot \log(\frac{n}{\delta}).$$

5 Projection Maintenance

The goal of this section is to prove the following theorem:

Theorem 5.1 (Projection maintenance)**.** *Given a full rank matrix $A \in \mathbb{R}^{d\times n}$ with $n \geq d$ and a tolerance parameter $0 < \epsilon_{mp} < 1/4$. Given any positive number a such that $a \leq \alpha$ where α is the duel exponent of matrix multiplication. There is a deterministic data structure (Algorithm 3) that approximately maintains the projection matrices*

$$\sqrt{W}A^\top(AWA^\top)^{-1}A\sqrt{W}$$

for positive diagonal matrices W through the following two operations:

1. Update(w)*: Output a vector $\widetilde{v}$ such that for all i,*

$$(1-\epsilon_{mp})\widetilde{v}_i \leq w_i \leq (1+\epsilon_{mp})\widetilde{v}_i.$$

2. Query(h)*: Output $\sqrt{\widetilde{V}}A^\top(A\widetilde{V}A^\top)^{-1}A\sqrt{\widetilde{V}}h$ for the $\widetilde{v}$ outputted by the last call to* Update*.*

The data structure takes $n^2 d^{\omega-2}$ time to initialize and each call of Query(h) *takes time*

$$n \cdot \|h\|_0 + n^{1+a+o(1)}.$$

Furthermore, if the initial vector $w^{(0)}$ and the (random) update sequence $w^{(1)}, \cdots, w^{(T)} \in \mathbb{R}^n$ satisfies

$$\sum_{i=1}^n \left(\frac{\mathbf{E}[w_i^{(k+1)}] - w_i^{(k)}}{w_i^{(k)}}\right)^2 \leq C_1^2, \sum_{i=1}^n \left(\mathbf{E}\left[\left(\frac{w_i^{(k+1)} - w_i^{(k)}}{w_i^{(k)}}\right)^2\right]\right)^2 \leq C_2^2, \left|\frac{w_i^{(k+1)} - w_i^{(k)}}{w_i^{(k)}}\right| \leq \frac{1}{4}.$$

with the expectation is conditional on $w_i^{(k)}$ for all $k = 0, 1, \cdots, T-1$. Then, the amortized expected time[4] *per call of* Update(w) *is*

$$(C_1/\epsilon_{mp} + C_2/\epsilon_{mp}^2) \cdot (n^{\omega-1/2+o(1)} + n^{2-a/2+o(1)}).$$

Remark 5.2. *For our linear program algorithm, we have $C_1 = O(1/\log n)$, $C_2 = O(1/\log n)$ and $\epsilon_{mp} = O(1)$. See Lemma 4.15.*

5.1 Proof outline

For intuition, we consider the case $C_1 = \Theta(1)$, $C_2 = \Theta(1)$, and $\epsilon_{mp} = \Theta(1)$ in this explanation. The correctness of the data structure directly follows from Woodbury matrix identity. The amortized time analysis is based on a potential function that measures the distance of the approximate vector v and the target vector w. We will show that

- The cost to update the projection M is proportional to the decrease of the potential.
- Each call to query increase the potential by a fixed amount.

Combining both together gives the amortized runtime bound of our data structure.

Now, we explain the definition of the potential. Consider the k-th round of the algorithm. For all $i \in [n]$, we define $x_i^{(k)} = \frac{w_i^{(k)}}{v_i^{(k)}} - 1$. Note that $|x_i^{(k)}|$ measures the relative distance between $w_i^{(k)}$ and

[4] If the input is deterministic, so is the output and the runtime.

Algorithm 3 Projection Maintenance Data Structure

datastructure MAINTAINPROJECTION ▷ Theorem 5.1

members
  $w \in \mathbb{R}^n$ ▷ Target vector
  $v, \widetilde{v} \in \mathbb{R}^n$ ▷ Approximate vector
  $A \in \mathbb{R}^{d \times n}$
  $M \in \mathbb{R}^{n \times n}$ ▷ Approximate Projection Matrix
  $\epsilon_{mp} \in (0, 1/4)$ ▷ Tolerance
  $a \in (0, \alpha]$ ▷ Batch Size for Update (n^a)
end members

procedure INITIALIZE(A, w, ϵ_{mp}, a) ▷ Lemma 5.3
  $w \leftarrow w$, $v \leftarrow w$, $\epsilon_{mp} \leftarrow \epsilon_{mp}$, $A \leftarrow A$, $a \leftarrow a$
  $M \leftarrow A^\top (AVA^\top)^{-1} A$
end procedure

procedure UPDATE(w^{new}) ▷ Lemma 5.4
  $y_i \leftarrow w_i^{\text{new}} / v_i - 1$, $\forall i \in [n]$
  $r \leftarrow$ the number of indices i such that $|y_i| \geq \epsilon_{mp}$.
  if $r < n^a$ **then**
    $v^{\text{new}} \leftarrow v$
    $M^{\text{new}} \leftarrow M$
  else
    Let $\pi : [n] \to [n]$ be a sorting permutation such that $|y_{\pi(i)}| \geq |y_{\pi(i+1)}|$
    while $1.5 \cdot r < n$ and $|y_{\pi(1.5r)}| \geq (1 - 1/\log n)|y_{\pi(r)}|$ **do**
      $r \leftarrow \min(\lceil 1.5 \cdot r \rceil, n)$
    end while
    $v_{\pi(i)}^{\text{new}} \leftarrow \begin{cases} w_{\pi(i)}^{\text{new}} & i \in \{1, 2, \cdots, r\} \\ v_{\pi(i)} & i \in \{r+1, \cdots, n\} \end{cases}$

    ▷ Compute $M^{\text{new}} = A^\top (AV^{\text{new}} A^\top)^{-1} A$ via Matrix Woodbury
    $\Delta \leftarrow \mathrm{diag}(v^{\text{new}} - v)$ ▷ $\Delta \in \mathbb{R}^{n \times n}$ and $\|\Delta\|_0 = r$
    Let $S \leftarrow \pi([r])$ be the first r indices in the permutation.
    Let $M_S \in \mathbb{R}^{n \times r}$ be the r columns from S of M.
    Let $M_{S,S}, \Delta_{S,S} \in \mathbb{R}^{r \times r}$ be the r rows and columns from S of M and Δ.
    $M^{\text{new}} \leftarrow M - M_S \cdot (\Delta_{S,S}^{-1} + M_{S,S})^{-1} \cdot (M_S)^\top$
  end if
  $w \leftarrow w^{\text{new}}$, $v \leftarrow v^{\text{new}}$, $M \leftarrow M^{\text{new}}$
  $\widetilde{v}_i \leftarrow \begin{cases} v_i & \text{if } (1 - \epsilon_{mp}) v_i \leq w_i \leq (1 + \epsilon_{mp}) v_i \\ w_i & \text{otherwise} \end{cases}$
  return $\widetilde{v}$
end procedure

procedure QUERY(h) ▷ Lemma 5.5
  Let $\widetilde{S}$ be the indices i such that $(1 - \epsilon_{mp}) v_i \leq w_i \leq (1 + \epsilon_{mp}) v_i$ is false.
  return $\sqrt{\widetilde{V}} \cdot (M \cdot (\sqrt{\widetilde{V}} \cdot h)) - \sqrt{\widetilde{V}} \cdot (M_{\widetilde{S}} \cdot ((\widetilde{\Delta}_{\widetilde{S},\widetilde{S}}^{-1} + M_{\widetilde{S},\widetilde{S}})^{-1} \cdot (M_{\widetilde{S}}^\top \sqrt{\widetilde{V}} h)))$
end procedure

end datastructure

$v_i^{(k)}$. Our algorithm fixes the indices with largest error $x_i^{(k)}$. To capture the fact that updating in a larger batch is more efficient, we define the potential as a weighted combination of the error where we put more weight to higher $x_i^{(k)}$. Formally, we sort the coordinates of $x^{(k)}$ such that $|x_i^{(k)}| \geq |x_{i+1}^{(k)}|$ and define the potential by

$$\Phi_k = \sum_{i=1}^{n} g_i \cdot \psi(x_i^{(k)}).$$

where g_i are positive decreasing numbers to be chosen and ψ is a symmetric ($\psi(x) = \psi(-x)$) positive function that increases on both sides. For intuition, one can think $\psi(x)$ behaves roughly like $|x|$.

Each iteration we update the projection matrix such that the error of $|x_1|, \cdots, |x_r|$ drops from roughly ϵ_{mp} to 0. This decreases the potential of $\psi(x_i^{(k)})$ by $\Omega(\epsilon_{mp})$ from $i = 1, \cdots, r$. Therefore, the whole potential decreases by $\Omega(\epsilon_{mp} \sum_{i=1}^{r} g_i)$. To make the term $\sum_{i=1}^{r} g_i$ proportional to the time to update a rank r part of the projection matrix, we set

$$g_i = \begin{cases} n^{-a}, & \text{if } i < n^a; \\ i^{\frac{\omega-2}{1-a}-1} n^{-\frac{a(\omega-2)}{1-a}}, & \text{otherwise.} \end{cases} \tag{17}$$

where ω is the exponent of matrix multiplication and a is any positive number less than or equals to the dual exponent of matrix multiplication. Lemma A.4 shows that g is indeed non-increasing and Lemma 5.4 shows that the update time of data-structure is indeed $O(r g_r n^{2+o(1)}) = O(\sum_{i=1}^{r} g_i n^{2+o(1)})$ for any $r \geq n^a$.

Each call to QUERY, the expectation of the vector $x^{(k)}$ moves roughly in an unit ℓ_2 ball. Therefore, the changes of the potential is roughly upper bounded $(\sum_{i=1}^{n} g_i^2)^{1/2} \approx n^{\omega-5/2}$. Since it takes us $n^{2+o(1)}$ time to decrease the potential by roughly 1 in the update step, the total time is roughly $n^{\omega-1/2}$.

For the case of stochastic central path, we note that the variance of the vector x is quite small. By choosing a smooth potential function ψ (see (18)), we can essentially give the same result as if there is no variance.

5.2 Proof of Theorem 5.1

Now, we give the proof of Theorem 5.1. We will defer some simple calculations into later sections.

Proof of Theorem 5.1.

Proof of Correctness. The definition of $\widetilde{v}$ in Line 38 ensures that $(1-\epsilon_{mp})\widetilde{v}_i \leq w_i \leq (1+\epsilon_{mp})\widetilde{v}_i$.

Using the Matrix Woodbury formula, one can verify that the update rule in Line 35 correctly maintains $M = A^\top (AVA^\top)^{-1} A$. See the deviation of the formula in Lemma 5.3. By the same reasoning, the Line 44 outputs the vector $\sqrt{\widetilde{V}} A^\top (A\widetilde{V}A^\top)^{-1} A \sqrt{\widetilde{V}} h$. This completes the proof of correctness.

Definition of x and y. Consider the k-th round of the algorithm. For all $i \in [n]$, we define $x_i^{(k)}$, $x_i^{(k+1)}$ and $y_i^{(k)}$ as follows:

$$x_i^{(k)} = \frac{w_i^{(k)}}{v_i^{(k)}} - 1, y_i^{(k)} = \frac{w_i^{(k+1)}}{v_i^{(k)}} - 1, x_i^{(k+1)} = \frac{w_i^{(k+1)}}{v_i^{(k+1)}} - 1.$$

Note that the difference between $x_i^{(k)}$ and $y_i^{(k)}$ is that w is changing. The difference between $y_i^{(k)}$ and $x_i^{(k+1)}$ is that v is changing. For simplicity, we define $\beta_i = (\mathbf{E}[w_i^{(k+1)}] - w_i^{(k)})/w_i^{(k)}$, then one of assumption becomes $\sum_{i=1}^{n} \beta_i^2 \leq C_1$.

Assume sorting. Assume the coordinates of vector $x^{(k)} \in \mathbb{R}^n$ are sorted such that $|x_i^{(k)}| \geq |x_{i+1}^{(k)}|$. Let τ and π are permutations such that $|x_{\tau(i)}^{(k+1)}| \geq |x_{\tau(i+1)}^{(k+1)}|$ and $|y_{\pi(i)}^{(k)}| \geq |y_{\pi(i+1)}^{(k)}|$.

Definition of Potential function. Let g be defined in (17). Let $\psi : \mathbb{R} \to \mathbb{R}$ be defined by

$$\psi(x) = \begin{cases} \frac{|x|^2}{2\epsilon_{mp}} & |x| \in [0, \epsilon_{mp}] \\ \epsilon_{mp} - \frac{(2\epsilon_{mp} - |x|)^2}{2\epsilon_{mp}} & |x| \in (\epsilon_{mp}, 2\epsilon_{mp}] \\ \epsilon_{mp} & |x| \in (2\epsilon_{mp}, +\infty) \end{cases} \tag{18}$$

We define the potential at the k-th round by

$$\Phi_k = \sum_{i=1}^n g_i \cdot \psi(x_{\tau_k(i)}^{(k)}).$$

where $\tau_k(i)$ is the permutation such that $x_{\tau_k(i)}^{(k)} \geq x_{\tau_k(i+1)}^{(k)}$.

Bounding the potential.

We can express $\Phi_{k+1} - \Phi_k$ as follows:

$$\begin{aligned} \Phi_{k+1} - \Phi_k = & \sum_{i=1}^n g_i \cdot \left(\psi(x_{\tau(i)}^{(k+1)}) - \psi(x_i^{(k)}) \right) \\ = & \sum_{i=1}^n g_i \cdot \underbrace{\left(\psi(y_{\pi(i)}^{(k)}) - \psi(x_i^{(k)}) \right)}_{w \text{ move}} - \sum_{i=1}^n g_i \cdot \underbrace{\left(\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\tau(i)}^{(k+1)}) \right)}_{v \text{ move}} \end{aligned} \tag{19}$$

Now, using Lemma 5.6 and 5.9, and the fact that $\Phi_0 = 0$ and $\Phi_T \geq 0$, with (19), we get

$$\begin{aligned} 0 \leq \Phi_T - \Phi_0 = & \sum_{k=0}^{T-1} (\Phi_{k+1} - \Phi_k) \\ \leq & \sum_{k=0}^{T-1} \left(O(C_1 + C_2/\epsilon_{mp}) \cdot \sqrt{\log n} \cdot (n^{-a/2} + n^{\omega - 5/2}) - \Omega(\epsilon_{mp} r_k g_{r_k} / \log n) \right) \\ = & T \cdot O(C_1 + C_2/\epsilon_{mp}) \cdot \sqrt{\log n} \cdot (n^{-a/2} + n^{\omega-5/2}) - \sum_{k=1}^{T} \Omega(\epsilon_{mp} r_k g_{r_k} / \log n) \end{aligned}$$

where the third step follows by Lemma 5.6 and Lemma 5.9 and r_k is the number of coordinates we update during that iteration.

Therefore, we get,

$$\sum_{k=1}^{T} r_k g_{r_k} = O\left(T \cdot (C_1/\epsilon_{mp} + C_2/\epsilon_{mp}^2) \cdot \log^{3/2} n \cdot (n^{\omega-5/2} + n^{-a/2}) \right).$$

Proof of running time. See the Section 5.3. □

5.3 Initialization time, update time, query time

To formalize the amortized runtime proof, we first analyze the initialization time (Lemma 5.3), update time (Lemma 5.4), and query time (Lemma 5.5) of our projection maintenance data-structure.

Lemma 5.3 (Initialization time)**.** *The initialization time of data-structure* MaintainProjection *(Algorithm 3) is* $O(n^2 d^{\omega-2})$.

Proof. Given matrix $A \in \mathbb{R}^{d\times n}$ and diagonal matrix $V \in \mathbb{R}^{n\times n}$, computing $A^\top (AVA^\top)^{-1}A$ takes $O(n^2 d^{\omega-2})$. □

Lemma 5.4 (Update time)**.** *The update time of data-structure* MaintainProjection *(Algorithm 3) is* $O(rg_r n^{2+o(1)})$ *where* r *is the number of indices we updated in* v.

Proof. Let $A_S \in \mathbb{R}^{d\times r}$ be the r columns from S of A. From k-th query to $(k+1)$-th query, we have

$$\begin{aligned}
& A^\top (AV^{(k+1)}A^\top)^{-1}A \\
= & A^\top (A(V^{(k)}+\Delta)A^\top)^{-1}A \\
= & A^\top \left((AV^{(k)}A^\top)^{-1} - (AV^{(k)}A^\top)^{-1}A_S(\Delta_{S,S}^{-1} + A_S^\top (AV^{(k)}A^\top)^{-1}A_S)^{-1}A_S^\top (AV^{(k)}A^\top)^{-1} \right) A \\
= & A^\top (AV^{(k)}A^\top)^{-1}A - A^\top (AV^{(k)}A^\top)^{-1}A_S(\Delta_{S,S}^{-1} + A_S^\top (AV^{(k)}A^\top)^{-1}A_S)^{-1}A_S^\top (AV^{(k)}A^\top)^{-1}A \\
= & M^{(k)} - M_S^{(k)}(\Delta_{S,S}^{-1} + M_{S,S}^{(k)})^{-1}(M_S^{(k)})^\top,
\end{aligned}$$

where the second step follows by Matrix Woodbury Identity and the last step follows by definition of $M^{(k)} \in \mathbb{R}^{n\times n}$.

Thus the update rule of matrix $M^{(k+1)} \in \mathbb{R}^{n\times n}$ can be written as

$$M^{(k+1)} = M^{(k)} - M_S^{(k)}(\Delta_{S,S}^{-1} + (M^{(k)})_{S,S})^{-1}(M_S^{(k)})^\top.$$

The updates in round k can be splitted into four parts:

1. Adding two $r\times r$ matrices takes $O(r^2)$ time.
2. Computing the inverse of an $r \times r$ matrix takes $O(r^{\omega+o(1)})$ time.
3. Computing the matrix multiplication of a $n\times r$ and $r\times n$ matrix takes $O(rg_r \cdot n^{2+o(1)})$ time where we used that $r \ge n^a$ (Lemma 2.4).
4. Adding two $n \times n$ matrices together takes $O(n^2)$ time.

Hence, the total cost is

$$O(r^2 + r^{\omega+o(1)} + rg_r \cdot n^{2+o(1)} + n^2) = O(r^2 + r^{\omega+o(1)} + rg_r\cdot n^{2+o(1)}) = O(rg_r \cdot n^{2+o(1)}).$$

where the first step follows by $rg_r \ge 1$ for all $r \ge n^a$ and the last step follows by the calculations. □

Lemma 5.5 (Query time)**.** *The query time of data-structure* MaintainProjection *(Algorithm 3) is* $O(n\cdot \|h\|_0 + n^{1+a+o(1)})$.

Proof. Let $\widetilde{\Delta}$ satisfies $\widetilde{V} = V + \widetilde{\Delta}$. Let $\widetilde{S} \subset [n]$ denote the support of $\widetilde{\Delta}$ and then $|\widetilde{S}| \le n^a$. Let $\widetilde{r}$ denote $|\widetilde{S}|$. We abuse the notation here, $\widetilde{\Delta}$ denotes both $n\times n$ diagonal matrix and a length n vector.

Using Matrix Woodbury Identity and definition of M, a same proof as Update time (Lemma 5.4) shows

$$A^\top (A\widetilde{V}A^\top)^{-1}A = M + M_{\widetilde{S}} \left(\widetilde{\Delta}_{\widetilde{S},\widetilde{S}}^{-1} + M_{\widetilde{S},\widetilde{S}} \right)^{-1} M_{\widetilde{S}}^\top,$$

where $\widetilde{\Delta}_{\widetilde{S}\times\widetilde{S}}$ has size $\widetilde{r}\times\widetilde{r}$, $M_{\widetilde{S},\widetilde{S}}$ has size $\widetilde{r}\times\widetilde{r}$ and $M_{\widetilde{S}}$ has size $n\times\widetilde{r}$.

To compute $\sqrt{\widetilde{V}}A^\top(A\widetilde{V}A^\top)^{-1}A\sqrt{\widetilde{V}}h$, we just need to compute

$$\sqrt{\widetilde{V}}M\sqrt{\widetilde{V}}h + \sqrt{\widetilde{V}}M_{\widetilde{S}}(\widetilde{\Delta}^{-1}_{\widetilde{S},\widetilde{S}} + M_{\widetilde{S},\widetilde{S}})^{-1}M_{\widetilde{S}}^\top\sqrt{\widetilde{V}}h.$$

Note the running time of computing the first term of the above equation only takes $O(n \cdot \|h\|_0)$ time.

Next, we analyze the cost of computing the second term of the above equation. It contains several parts:

1. Computing $\widetilde{M}_{\widetilde{S}}^\top \cdot (\sqrt{\widetilde{V}} \cdot h) \in \mathbb{R}^{\widetilde{r}}$ takes $\widetilde{r}\|h\|_0$ time.
2. Computing $(\widetilde{\Delta}^{-1}_{\widetilde{S},\widetilde{S}} + M_{\widetilde{S},\widetilde{S}})^{-1} \in \mathbb{R}^{\widetilde{r}\times\widetilde{r}}$ that is the inverse of a $\widetilde{r}\times\widetilde{r}$ matrix takes $\widetilde{r}^{\omega+o(1)}$ time.
3. Computing matrix-vector multiplication between $\widetilde{r}\times\widetilde{r}$ matrix $((\widetilde{\Delta}^{-1}_{\widetilde{S},\widetilde{S}} + M_{\widetilde{S},\widetilde{S}})^{-1})$ and $\widetilde{r}\times 1$ vector $(\widetilde{M}_{\widetilde{S}}^\top\sqrt{\widetilde{V}}h)$ takes $O(\widetilde{r}^2)$ time.
4. Computing matrix-vector multiplication between $n\times\widetilde{r}$ matrix $(M_{\widetilde{S}})$ and $\widetilde{r}\times 1$ vector $((\widetilde{\Delta}^{-1}_{\widetilde{S},\widetilde{S}} + M_{\widetilde{S},\widetilde{S}})^{-1}M_{\widetilde{S}}^\top\sqrt{\widetilde{V}}h)$ takes $O(n\widetilde{r})$ time.
5. Computing the entry-wise product of two n vectors takes $O(n)$ time

Thus, overall the running time is

$$O(\widetilde{r}\|h\|_0 + \widetilde{r}^{\omega+o(1)} + \widetilde{r}^2 + n\widetilde{r} + n) = O(\widetilde{r}^{\omega+o(1)} + n\widetilde{r}) = O(n^{a\cdot\omega+o(1)} + n^{1+a}).$$

Finally, we note that $\omega \le 3-\alpha \le 3-a$ (Lemma A.4) and hence $a\cdot\omega \le a(3-a) \le 1+a$. Therefore, the runtime is $n^{1+a+o(1)}$. □

5.4 Bounding w move

The goal of this section is to prove Lemma 5.6.

Lemma 5.6 (w move)**.** *We have*

$$\sum_{i=1}^n g_i \cdot \mathbf{E}\left[\psi(y_{\pi(i)}^{(k)}) - \psi(x_i^{(k)})\right] \le O(C_1 + C_2/\epsilon_{mp}) \cdot \sqrt{\log n} \cdot (n^{-a/2} + n^{\omega-5/2}).$$

Proof. Observe that since the errors $|x_i^{(k)}|$ are sorted in descending order, and $\psi(x)$ is symmetric and non-decreasing function for $x \ge 0$, thus $\psi(x_i^{(k)})$ is also in decreasing order. In addition, note that g is decreasing, we have

$$\sum_{i=1}^n g_i\psi(x_{\pi(i)}^{(k)}) \le \sum_{i=1}^n g_i\psi(x_i^{(k)}). \tag{20}$$

Hence the first term in (19) can be upper bounded as follows:

$$\begin{aligned}
\mathbf{E}\left[\sum_{i=1}^n g_i \cdot \left(\psi(y_{\pi(i)}^{(k)}) - \psi(x_i^{(k)})\right)\right] &\le \mathbf{E}\left[\sum_{i=1}^n g_i \cdot \left(\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\pi(i)}^{(k)})\right)\right] && \text{by (20)}\\
&= \sum_{i=1}^n g_i \cdot \mathbf{E}[\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\pi(i)}^{(k)})]\\
&= O(C_1 + C_2/\epsilon_{mp}) \cdot \sqrt{\log n} \cdot (n^{-a/2} + n^{\omega-5/2}). && \text{by Lemma 5.7}
\end{aligned}$$

Thus, we complete the proof of w move Lemma. □

It remains to prove the following Lemma,

Lemma 5.7.

$$\sum_{i=1}^{n} g_i \cdot \mathbf{E}[\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\pi(i)}^{(k)})] = O(C_1 + C_2/\epsilon_{mp}) \cdot \sqrt{\log n} \cdot (n^{-a/2} + n^{\omega-5/2}).$$

Proof. Let I be the set of indices such that $|x_i^{(k)}| \leq 1$. We separate the term into two:

$$\sum_{i=1}^{n} g_i \cdot \mathbf{E}[\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\pi(i)}^{(k)})] = \sum_{i \in I} g_{\pi^{-1}(i)} \cdot \mathbf{E}[\psi(y_i^{(k)}) - \psi(x_i^{(k)})] + \sum_{i \in I^c} g_{\pi^{-1}(i)} \cdot \mathbf{E}[\psi(y_i^{(k)}) - \psi(x_i^{(k)})].$$

Case 1: Terms from I Mean value theorem shows that

$$\begin{aligned}
\psi(y_i^{(k)}) - \psi(x_i^{(k)}) &= \psi'(x_i^{(k)})(y_i^{(k)} - x_i^{(k)}) + \frac{1}{2}\psi''(\zeta)(y_i^{(k)} - x_i^{(k)})^2 \\
&\leq \psi'(x_i^{(k)}) \frac{w_i^{(k+1)} - w_i^{(k)}}{v_i^{(k)}} + \frac{L_2}{2}\left(\frac{w_i^{(k+1)} - w_i^{(k)}}{v_i^{(k)}}\right)^2,
\end{aligned}$$

where $L_2 = \max_x \psi''(x)$. Taking conditional expectation given $w^{(k)}$ on both sides

$$\begin{aligned}
\mathbf{E}[\psi(y_i^{(k)}) - \psi(x_i^{(k)})] &\leq \psi'(x_i^{(k)}) \cdot \frac{\mathbf{E}[w_i^{(k+1)}] - w_i^{(k)}}{v_i^{(k)}} + \frac{L_2}{2}\frac{1}{(v_i^{(k)})^2}\mathbf{E}[(w_i^{(k+1)} - w_i^{(k)})^2] \\
&= \psi'(x_i^{(k)}) \cdot \frac{w_i^{(k)}}{v_i^{(k)}}\beta_i + \frac{L_2}{2}\frac{(w_i^{(k)})^2}{(v_i^{(k)})^2}\gamma_i,
\end{aligned}$$

where $\beta_i = \frac{\mathbf{E}[w_i^{(k+1)}] - w_i^{(k)}}{w_i^{(k)}}$ and $\gamma_i = \mathbf{E}\left[\left(\frac{w_i^{(k+1)} - w_i^{(k)}}{w_i^{(k)}}\right)^2\right]$.

To bound $\sum_{i \in I} g_{\pi^{-1}(i)} \mathbf{E}[\psi(y_i^{(k)}) - \psi(x_i^{(k)})]$, we need to bound the following two terms,

$$\sum_{i \in I} g_{\pi^{-1}(i)} \psi'(x_i^{(k)}) \frac{w_i^{(k)}}{v_i^{(k)}}\beta_i, \text{ and } \sum_{i \in I} g_{\pi^{-1}(i)} \frac{L_2}{2}\frac{(w_i^{(k)})^2}{(v_i^{(k)})^2}\gamma_i. \tag{21}$$

For the term $\frac{w_i^{(k)}}{v_i^{(k)}}$, we note that for $i \in I$, we have $\left|\frac{w_i^{(k)}}{v_i^{(k)}}\right| \leq \left|x_i^{(k)}\right| + 1 \leq 2$. Using this, we can bound the first term by

$$\begin{aligned}
\sum_{i \in I} g_{\pi^{-1}(i)} \psi'(x_i^{(k)}) \frac{w_i^{(k)}}{v_i^{(k)}}\beta_i &\leq \left(\sum_{i \in I}\left(g_{\pi^{-1}(i)}\psi'(x_i^{(k)})\frac{w_i^{(k)}}{v_i^{(k)}}\right)^2 \sum_{i \in I}\beta_i^2\right)^{1/2} \\
&\leq O(L_1)\left(\sum_{i=1}^{n} g_i^2 \cdot C_1^2\right)^{1/2} = O(C_1 L_1 \|g\|_2).
\end{aligned} \tag{22}$$

where $L_1 = \max_x |\psi'(x)|$, the first step follows by Cauchy-Schwarz inequality and the second step follows by $\left|\psi'(x_i^{(k)}) \cdot w_i^{(k)}/v_i^{(k)}\right| \leq 2L_1$ and $\sum_{i=1}^{n}\beta_i^2 \leq C_1^2$.

For the second term, we have

$$\sum_{i\in I} g_{\pi^{-1}(i)} \frac{L_2}{2}\frac{(w_i^{(k)})^2}{(v_i^{(k)})^2}\gamma_i \le O(L_2)\cdot\sum_{i=1}^{n} g_i\cdot\gamma_i = O(C_2L_2\|g\|_2). \tag{23}$$

Now, combining (22) and (23) and using that $L_1 = O(1)$, $L_2 = O(1/\epsilon_{mp})$ (from part 4 of Lemma 5.10) and $\|g\|_2 \le \sqrt{\log n}\cdot O(n^{-a/2}+n^{\omega-5/2})$ (from Lemma 5.8), we have that

$$\sum_{i\in I} g_{\pi^{-1}(i)}\cdot \mathbf{E}[\psi(y_i^{(k)})-\psi(x_i^{(k)})] \le O(C_1+C_2/\epsilon_{mp})\cdot\sqrt{\log n}\cdot(n^{-a/2}+n^{\omega-5/2}).$$

Case 2: Terms from I^c

For all $i\in I^c$, we have $|x_i^{(k)}|\ge 1$. Note that $\psi(x)$ is a constant for $x\ge 2\epsilon_{mp}$ and that $\epsilon_{mp}\le 1/4$. Therefore, if $|y_i^{(k)}|\ge 1/2$, we have that $\psi(y_i^{(k)})-\psi(x_i^{(k)})=0$. Hence, we only need to consider the $i\in I^c$ such that $|y_i^{(k)}|<1/2$. For these i, we have that

$$\frac{1}{2} < |y_i^{(k)}-x_i^{(k)}| = \left|\frac{w_i^{(k+1)}-w_i^{(k)}}{v_i^{(k)}}\right| = \left|\frac{w_i^{(k+1)}}{v_i^{(k)}}\right|\left|\frac{w_i^{(k+1)}-w_i^{(k)}}{w_i^{(k+1)}}\right| \le \frac{3}{2}\left|\frac{w_i^{(k+1)}-w_i^{(k)}}{w_i^{(k+1)}}\right|,$$

where we used that $\left|y_i^{(k)}\right| = \left|\frac{w_i^{(k+1)}}{v_i^{(k)}}-1\right| \le 1/2$. Hence, we have that $\left|\frac{w_i^{(k+1)}-w_i^{(k)}}{w_i^{(k+1)}}\right| > 1/3$ and hence $\left|\frac{w_i^{(k+1)}-w_i^{(k)}}{w_i^{(k)}}\right| > 1/4$, which is impossible.

Hence, we have

$$\sum_{i\in I^c} g_{\pi^{-1}(i)}\cdot\mathbf{E}[\psi(y_i^{(k)})-\psi(x_i^{(k)})] = 0.$$

Combining both cases, we have the result. ◻

Lemma 5.8.

$$\left(\sum_{i=1}^{n} g_i^2\right)^{1/2} \le \sqrt{\log n}\cdot O(n^{-a/2}+n^{\omega-5/2}).$$

Proof. Since function g behaves differently when $i\le n^a$ and $i>n^a$. We will the sum into two parts.

For the first part, we have

$$\sum_{i=1}^{n^a} g_i^2 = \sum_{i=1}^{n^a} n^{-2a} = n^{-a}.$$

For the second part, we have

$$\sum_{i=n^a}^{n} g_i^2 = \sum_{i=n^a}^{n} i^{\frac{2(\omega-2)}{1-a}-2} n^{-\frac{2a(\omega-2)}{1-a}} = \sum_{i=n^a}^{n}\frac{1}{i}\cdot i^{\frac{2(\omega-2)}{1-a}-1} n^{-\frac{2a(\omega-2)}{1-a}}.$$

Note that

$$\max_{i\in[n^a,n]} i^{\frac{2(\omega-2)}{1-a}-1} n^{-\frac{2a(\omega-2)}{1-a}} = \max(n^{a\frac{2(\omega-2)}{1-a}-a} n^{-\frac{2a(\omega-2)}{1-a}}, n^{\frac{2(\omega-2)}{1-a}-1} n^{-\frac{2a(\omega-2)}{1-a}}) = \max(n^{-a}, n^{2\omega-5}).$$

Thus, the second part is

$$\sum_{i=n^a}^{n} g_i^2 = \sum_{i=n^a}^{n}\frac{1}{i}\cdot\max(n^{-a},n^{2\omega-5}) = O(\log n)\cdot\max(n^{-a},n^{2\omega-5}).$$

Combining the first part and the second part completes the proof. ◻

5.5 Bounding v move

Lemma 5.9 (v move)**.** *We have,*

$$\sum_{i=1}^{n} g_i \cdot \left(\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\tau(i)}^{(k+1)}) \right) \geq \Omega(\epsilon_{mp} r_k g_{r_k} / \log n).$$

Proof. We split the proof into two cases.

We first understand some simple facts which are useful in the later proof. Note that from definition of $x_i^{(k+1)}$, we know that $x^{(k+1)}$ has r_k coordinates are 0. Basically, $\|y^{(k)} - x^{(k+1)}\|_0 = r_k$. The difference between those vectors is, for the largest r_k coordinates in $y^{(k)}$, we erase them in $x^{(k+1)}$. Then for each $i \in [n - r_k]$, $x_{\tau(i)}^{(k+1)} = y_{\pi(i+r_k)}^{(k)}$. For convenience, we define $y_{\pi(n+i)}^{(k)} = 0$, $\forall i \in [r_k]$.

Case 1. We exit the while loop when $1.5 r_k \geq n$.

Let u^* denote the largest u s.t. $|y_{\pi(u)}^{(k)}| \geq \epsilon_{mp}$. If $u^* = r_k$, we have that $|y_{\pi(r_k)}^{(k)}| \geq \epsilon_{mp} \geq \epsilon_{mp}/100$. Otherwise, the condition of the loop shows that

$$|y_{\pi(r_k)}^{(k)}| \geq (1 - 1/\log n)^{\log_{1.5} r_k - \log_{1.5} u^*} |y_{\pi(u^*)}^{(k)}| \geq (1 - 1/\log n)^{\log_{1.5} n} \epsilon_{mp} \geq \epsilon_{mp}/100.$$

where we used that $n \geq 4$.

According to definition of $x_{\tau(i)}^{(k+1)}$, we have

$$\begin{aligned} \sum_{i=1}^{n} g_i(\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\tau(i)}^{(k+1)})) = & \sum_{i=1}^{n} g_i(\psi(y_{\pi(i)}^{(k)}) - \psi(y_{\pi(i+r_k)}^{(k)})) \geq \sum_{i=n/3+1}^{n} g_i(\psi(y_{\pi(i)}^{(k)}) - \psi(y_{\pi(i+r_k)}^{(k)})) \\ \geq & \sum_{i=n/3+1}^{n} g_i(\psi(y_{\pi(i)}^{(k)})) \geq \sum_{i=n/3+1}^{2n/3} g_i \psi(\epsilon_{mp}/100) \geq \Omega(r_k g_{r_k} \epsilon_{mp}). \end{aligned}$$

where the first step follows from $x_{\tau(i)}^{(k+1)} = y_{\pi(i+r_k)}^{(k)}$, the second step follows from $\psi(|x|)$ is non-decreasing (part 2 of Lemma 5.10) and $|y_{\pi(i)}^{(k)}|$ is non-increasing, the third step follows from $1.5 r_k > n$ and hence $\psi(y_{\pi(i+r_k)}^{(k)}) = 0$ for $i \geq n/3 + 1$, the fourth step follows from ψ is non-decreasing and $|y_{\pi(i)}^{(k)}| \geq |y_{\pi(r_k)}^{(k)}| \geq \epsilon_{mp}/100$ for all $i < 2n/3$, and the last step follows by g is decreasing and part 3 of Lemma 5.10.

Case 2. We exit the while loop when $1.5 r_k < n$ and $|y_{\pi(1.5 r_k)}^{(k)}| < (1 - 1/\log n) |y_{\pi(r_k)}^{(k)}|$.

By the same argument as Case 1, we have that $|y_{\pi(r_k)}^{(k)}| \geq \epsilon_{mp}/100$. Part 3 of Lemma 5.10 together with the fact

$$|y_{\pi(1.5r)}^{(k)}| < \min(\epsilon_{mp}, |y_{\pi(r)}^{(k)}| \cdot (1 - 1/\log n)),$$

shows that

$$\psi(|y_{\pi(1.5r)}^{(k)}|) - \psi(|y_{\pi(r)}^{(k)}|) = \Omega(\epsilon_{mp}/\log n). \tag{24}$$

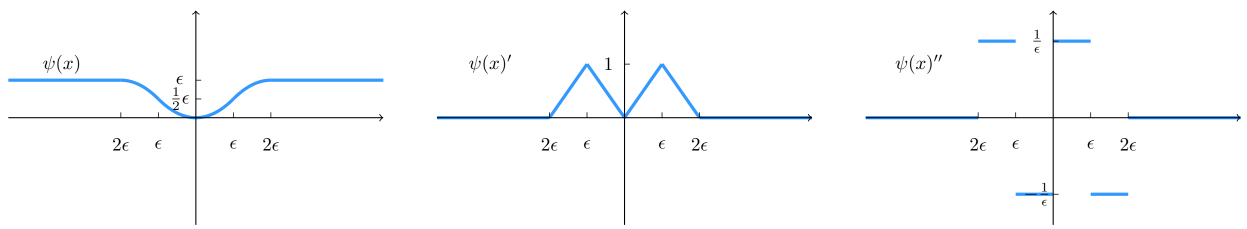

Figure 2: $\psi(x)$, $\psi(x)'$ and $\psi(x)''$. For $\epsilon_{mp} \in (0,1)$.

Putting it all together, we have

$$\begin{aligned}
&\sum_{i=1}^{n} g_i \cdot (\psi(y_{\pi(i)}^{(k)}) - \psi(x_{\tau(i)}^{(k+1)})) \\
= &\sum_{i=1}^{n} g_i \cdot (\psi(y_{\pi(i)}^{(k)}) - \psi(y_{\pi(i+r_k)}^{(k)})) && \text{by } x_{\tau(i)}^{(k+1)} = y_{\pi(i+r_k)}^{(k)} \\
\geq &\sum_{i=r_k/2}^{r_k} g_i \cdot (\psi(y_{\pi(i)}^{(k)}) - \psi(y_{\pi(i+r_k)}^{(k)})) && \text{by } \psi(y_{\pi(i)}^{(k)}) - \psi(y_{\pi(i+r_k)}^{(k)}) \geq 0 \\
\geq &\sum_{i=r_k/2}^{r_k} g_i \cdot (\psi(y_{\pi(r_k)}^{(k)}) - \psi(y_{\pi(1.5r_k)}^{(k)})) \\
\geq &\sum_{i=r_k/2}^{r_k} g_i \cdot \Omega(\frac{\epsilon_{mp}}{\log n}) && \text{by } (24) \\
\geq &\sum_{i=r_k/2}^{r_k} g_{r_k} \cdot \Omega(\frac{\epsilon_{mp}}{\log n}) && \text{by } g_i \text{ is decreasing} \\
= &\,\Omega\left(\epsilon_{mp} r_k g_{r_k} / \log n\right),
\end{aligned}$$

where the third step follows by $|y_{\pi(i)}^{(k)}|$ is decreasing and ψ is non-decreasing (from part 2 of Lemma 5.10). □

5.6 Potential function ψ

Lemma 5.10 (Properties of function ψ). *Let function ψ be defined in (18). Then function ψ satisfies the following properties:*
1. Symmetric ($\psi(-x) = \psi(x)$) and $\psi(0) = 0$;
2. $\psi(|x|)$ is non-decreasing;
3. $|\psi'(x)| = \Omega(1), \forall |x| \leq 1.5\epsilon_{mp}$;
4. $L_1 \stackrel{\text{def}}{=} \max_x \psi'(x) = 1$ and $L_2 \stackrel{\text{def}}{=} \max_x \psi''(x) = 1/\epsilon_{mp}$;
5. $\psi(x)$ is a constant for $|x| \geq 2\epsilon_{mp}$.

Proof. We can see that

$$\psi(x)' = \begin{cases} \frac{|x|}{\epsilon_{mp}} & |x| \in [0, \epsilon_{mp}] \\ \frac{2\epsilon_{mp} - |x|}{\epsilon_{mp}} & |x| \in (\epsilon_{mp}, 2\epsilon_{mp}] \\ 0 & x \in (2\epsilon_{mp}, +\infty) \end{cases} \quad \text{and} \quad \psi(x)'' = \begin{cases} \frac{1}{\epsilon_{mp}} & x \in [0, \epsilon_{mp}] \cup [-2\epsilon_{mp}, -\epsilon_{mp}] \\ -\frac{1}{\epsilon_{mp}} & x \in (\epsilon_{mp}, 2\epsilon_{mp}] \cup [-\epsilon_{mp}, 0] \\ 0 & x \in (2\epsilon_{mp}, +\infty) \end{cases}$$

From the $\psi(x)'$ and $\psi(x)''$, it is not hard to see that ψ satisfies the properties needed. □

Acknowledgement

This work was supported in part by NSF Awards CCF-1740551, CCF-1749609, and DMS-1839116. We thank Sébastien Bubeck and Aaron Sidford for helpful discussions. We thank Rasmus Kyng for bringing up the question and providing some fixes in the proof of projection maintenance. We thank Josh Alman for some useful discussions about matrix multiplication. We thank Eric Price for the suggestion of the title of this paper.

Finally, we are honored and blessed to have collaborated with Michael Cohen. Arguably, this project is a simple corollary for his beautiful algorithm and proof for the inverse maintenance problem, described in the Figure 3. We believe that his enthusiasm in finding the proofs from The Book will be remembered.

Michael Cohen
so consider the simplest possible version of the inverse maintainence thing
which is
n = O(d)
AND
you are promised l_2 of the update is small

Michael Cohen
I think I can achieve this (up to polylog factor), deterministically, using matrix multiplication in d^{2.5} time
though i'm very likely screwing up

Michael Cohen
OK so
we're deferring updates to the weights
so for each weight we have a "current" weight and a "desired multiplication"
all "desired multiplications" are between e.g. 1/2 and 3/2
i'll refer to "desired update" = "desired multiplication"-1, so they're between -1/2 and 1/2
makes sense so far?

Michael Cohen
so each step updates the "desired updates"
as long as all of them stay < 1/2 in absolute value, we do nothing
if the biggest has absolute value > 1/2
we find the first i
such that the (2i)th largest (in absolute value) desired update
is smaller by at least a factor of 1-1/(log n)
than the i'th
then we fix all of the biggest (2i) desired updates, with a rank-(2i) update

Michael Cohen
that's the whole algorithm
potential function is
sum over i of (i'th largest (absolute) desired update) / sqrt{i}

Figure 3: The message that starts this paper.

References

[AFLG15] Andris Ambainis, Yuval Filmus, and François Le Gall. Fast matrix multiplication: limitations of the coppersmith-winograd method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing(STOC)*, pages 585–593. ACM, 2015.

[AW18] Josh Alman and Virginia Vassilevska Williams. Limits on all known (and some unknown) approaches to matrix multiplication. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018.

[BCLL18] Sébastien Bubeck, Michael B Cohen, Yin Tat Lee, and Yuanzhi Li. An homotopy method for ℓ_p regression provably beyond self-concordance and in input-sparsity time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1130–1137. ACM, 2018.

[Cla95] Kenneth L Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM (JACM)*, 42(2):488–499, 1995.

[CLM$^+$16] Michael B Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (STOC)*, pages 9–21. ACM, 2016.

[CMSV17] Michael B Cohen, Aleksander Mądry, Piotr Sankowski, and Adrian Vladu. Negative-weight shortest paths and unit capacity minimum cost flow in $\widetilde{O}(m^{10/7} \log W)$ time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms(SODA)*, pages 752–771. SIAM, 2017.

[CMTV17] Michael B Cohen, Aleksander Madry, Dimitris Tsipras, and Adrian Vladu. Matrix scaling and balancing via box constrained newton's method and interior point methods. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 902–913. IEEE, 2017.

[CW87] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing(STOC)*, pages 1–6. ACM, 1987.

[CW13] Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 81–90. https://arxiv.org/pdf/1207.6365, 2013.

[DS13] Alexander Munro Davie and Andrew James Stothers. Improved bound for complexity of matrix multiplication. *Proceedings of the Royal Society of Edinburgh Section A: Mathematics*, 143(2):351–369, 2013.

[Kar84] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing(STOC)*, pages 302–311. ACM, 1984.

[LG14] François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation(ISSAC)*, pages 296–303. ACM, 2014.

[LGU18] Francois Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms(SODA)*, pages 1029–1046. SIAM, 2018.

[LS13] Yin Tat Lee and Aaron Sidford. Path finding I: Solving linear programs with $\widetilde{O}(\sqrt{rank})$ linear system solves. *arXiv preprint arXiv:1312.6677*, 2013.

[LS14] Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: Solving linear programs in $O(\sqrt{rank})$ iterations and faster algorithms for maximum flow. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 424–433. IEEE, 2014.

[LS15] Yin Tat Lee and Aaron Sidford. Efficient inverse maintenance and faster algorithms for linear programming. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 230–249. IEEE, 2015.

[LSW15] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1049–1065. IEEE, 2015.

[Mad13] Aleksander Madry. Navigating central path with electrical flows: From flows to matchings, and back. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 253–262. IEEE, 2013.

[Mad16] Aleksander Madry. Computing maximum flow with augmenting electrical flows. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 593–602. IEEE, 2016.

[NN89] Yu Nesterov and Arkadi Nemirovsky. Self-concordant functions and polynomial-time methods in convex programming. *Report, Central Economic and Mathematic Institute, USSR Acad. Sci*, 1989.

[NN91] Yu Nesterov and Arkadi Nemirovsky. Acceleration and parallelization of the path-following interior point method for a linearly constrained convex quadratic problem. *SIAM Journal on Optimization*, 1(4):548–564, 1991.

[NN94] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Siam, 1994.

[NN13] Jelani Nelson and Huy L Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 117–126. IEEE, `https://arxiv.org/pdf/1211.1002`, 2013.

[PRT02] Jiming Peng, Cornelis Roos, and Tamás Terlaky. Self-regular functions and new search directions for linear and semidefinite optimization. *Mathematical Programming*, 93(1):129–171, 2002.

[Ren88] James Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Mathematical Programming*, 40(1-3):59–93, 1988.

[Sar06] Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS) , 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 143–152, 2006.

[SS11] Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

[Vai87] Pravin M Vaidya. An algorithm for linear programming which requires $O(((m+n)n^2 + (m+n)^{1.5}n)L)$arithmetic operations. In *FOCS*. IEEE, 1987.

[Vai89a] Pravin M Vaidya. A new algorithm for minimizing convex functions over convex sets. In *30th Annual Symposium on Foundations of Computer Science(FOCS)*, pages 338–343. IEEE, 1989.

[Vai89b] Pravin M Vaidya. Speeding-up linear programming using fast matrix multiplication. In *FOCS*. IEEE, 1989.

[Wil12] Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing (STOC)*, pages 887–898. ACM, 2012.

[YTM94] Yinyu Ye, Michael J Todd, and Shinji Mizuno. An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67, 1994.

A Appendix

Lemma A.1. *Let x and y are (possibly dependent) random variables such that $|x| \leq c_x$ and $|y| \leq c_y$ almost surely. Then, we have*

$$\mathbf{Var}[xy] \leq 2c_x^2 \cdot \mathbf{Var}[y] + 2c_y^2 \cdot \mathbf{Var}[x].$$

Proof. Recall that $\mathbf{Var}[xy] \leq \mathbf{E}[(xy - t)^2]$ for any scalar t. Hence,

$$\begin{aligned}
\mathbf{Var}[xy] &\leq \mathbf{E}[(xy - \mathbf{E}[x]\,\mathbf{E}[y])^2] = \mathbf{E}[(xy - x\,\mathbf{E}[y] + x\,\mathbf{E}[y] - \mathbf{E}[x]\,\mathbf{E}[y])^2] \\
&\leq 2\,\mathbf{E}[(xy - x\,\mathbf{E}[y])^2] + 2\,\mathbf{E}[(x\,\mathbf{E}[y] - \mathbf{E}[x]\,\mathbf{E}[y])^2] \\
&\leq 2c_x^2 \cdot \mathbf{Var}[y] + 2c_y^2 \cdot \mathbf{Var}[x].
\end{aligned}$$

□

Lemma A.2 ([Vai89b])**.** *Given a matrix $A \in \mathbb{R}^{d \times n}$, vectors $b \in \mathbb{R}^d, c \in \mathbb{R}^n$. Suppose $x, s, y \in \mathbb{R}^n$ satisfy that $xs \approx_{0.1} t$, $Ax = b$ and $A^\top y + s = c$ for some $t > 0$. For any $\epsilon \in (0, 1/2]$, in $\widetilde{O}(n^{2.5}\log(n/\epsilon))$ time, we can find vectors $x^{\mathrm{new}}, s^{\mathrm{new}} \in \mathbb{R}^n$ and $y^{\mathrm{new}} \in \mathbb{R}^d$ such that*

$$\begin{aligned}
\|x^{\mathrm{new}} s^{\mathrm{new}} - t\|_2 &\leq \epsilon, \\
A x^{\mathrm{new}} &= b, \\
A^\top y^{\mathrm{new}} + s &= c.
\end{aligned}$$

Remark A.3. *Instead of using this, one can also run our algorithm with $k = n$ for $O(\sqrt{n}\log n)$ iterations. Since $k = n$, there is no randomness involves and hence Φ will decrease deterministically to $O(n)$.*

Lemma A.4. $\omega \le 3-\alpha$.

Proof. We consider a $n \times n$ matrix A multiply another $n \times n$ B, we split A into $n^{1-\alpha}$ fat matrices where each of them has size $n^{\alpha} \times n$. Since ω is the best exponent of matrix multiplication, thus we know

$$n^{\omega+o(1)} \le n^{1-\alpha} \cdot n^{2+o(1)}$$

which implies $\omega \le 3-\alpha$. □

Lemma A.5 (Rectangular matrix multiplication)**.** *For any $n \ge r$, multiplying an $n \times r$ with an $r \times n$ matrix or $n \times n$ with $n \times r$ takes time*

$$n^{2+o(1)} + r^{\frac{\omega-2}{1-\alpha}} n^{2-\frac{\alpha(\omega-2)}{1-\alpha}+o(1)}.$$

Proof. The cost for multiplying a $n \times n$ and a $n \times r$ matrix is the same as multiplying a $n \times r$ and a $r \times n$ matrix. So, we focus on the later case.

For the case $r \le n^{\alpha}$, it follows from the rectangular matrix multiplication result in [LGU18].

For the case $r \ge n^{\alpha}$, we let $k = (n/r)^{\frac{1}{1-\alpha}}$. We can view the problem as multiplying a $k \times k^{\alpha}$ and a $k^{\alpha} \times k$ block matrices and each block has size $\frac{n}{k} \times \frac{n}{k}$ size. Therefore, the total cost is

$$k^{2+o(1)} \times (\frac{n}{k})^{\omega+o(1)} = r^{\frac{\omega-2}{1-\alpha}} n^{2-\frac{\alpha(\omega-2)}{1-\alpha}+o(1)}.$$

□

Lemma A.6. *Consider a linear program $\min_{Ax=b,x\ge 0} c^{\top}x$ with n variables and d constraints. Assume that*

1. *Diameter of the polytope: For any $x \ge 0$ with $Ax=b$, we have that $\|x\|_1 \le R$.*
2. *Lipschitz constant of the linear program: $\|c\|_{\infty} \le L$.*

For any $0 < \delta \le 1$, the modified linear program $\min_{\overline{A}\overline{x}=\overline{b},\overline{x}\ge 0} \overline{c}^{\top}\overline{x}$ with

$$\overline{A} = \begin{bmatrix} A & 0 & \frac{1}{R}b - A1_n \\ 1_n^{\top} & 1 & 0 \\ -1_n^{\top} & -1 & 0 \end{bmatrix}, \overline{b} = \begin{bmatrix} \frac{1}{R}b \\ n+1 \\ -(n+1) \end{bmatrix}, \text{ and } \overline{c} = \begin{bmatrix} \delta/L \cdot c \\ 0 \\ 1 \end{bmatrix}$$

satisfies the following:

1. $\overline{x} = \begin{bmatrix} 1_n \\ 1 \\ 1 \end{bmatrix}$, $\overline{y} = \begin{bmatrix} 0_d \\ 0 \\ 1 \end{bmatrix}$ *and* $\overline{s} = \begin{bmatrix} 1_n + \frac{\delta}{L} \cdot c \\ 1 \\ 1 \end{bmatrix}$ *are feasible primal dual vectors.*
2. *For any feasible primal dual vectors $(\overline{x}, \overline{y}, \overline{s})$ with duality gap $\le \delta^2$, consider the vector $\widehat{x} = R \cdot \overline{x}_{1:n}$ ($\overline{x}_{1:n}$ is the first n coordinates of $\overline{x}$) is an approximate solution to the original linear program in the following sense*

$$c^{\top}\widehat{x} \le \min_{Ax=b,x\ge 0} c^{\top}x + LR \cdot \delta,$$

$$\|A\widehat{x} - b\|_1 \le 2\delta \cdot \left(R \sum_{i,j} |A_{i,j}| + \|b\|_1 \right),$$

$$\widehat{x} \ge 0.$$

Proof. For the first result, using $\delta < 1$, straightforward calculations show that $(\overline{x}, \overline{y}, \overline{s})$ are feasible.

For the second result, we let $\mathrm{OPT} = \min_{Ax=b, x\geq 0} c^\top x$ and $\overline{\mathrm{OPT}} = \min_{\overline{A}\overline{x}=\overline{b}, \overline{x}\geq 0} \overline{c}^\top \overline{x}$. For any feasible x in the original LP, $\overline{x} = \begin{bmatrix} x/R \\ 0 \\ 0 \end{bmatrix}$ is a feasible in the modified LP. Therefore, we have that

$$\overline{\mathrm{OPT}} \leq \frac{\delta}{L} \cdot c^\top (x/R) = \frac{\delta}{LR} \cdot \mathrm{OPT}.$$

Given a feasible $(\overline{x}, \overline{y}, \overline{s})$ with duality gap δ^2. Write $\overline{x} = \begin{bmatrix} \overline{x}_{1:n} \\ \tau \\ \theta \end{bmatrix}$ for some $\tau \geq 0, \theta \geq 0$. We can compute $\overline{c}^\top \overline{x}$ which is $\frac{\delta}{L} \cdot c^\top \overline{x}_{1:n} + \theta$. Then, we have

$$\frac{\delta}{L} \cdot c^\top \overline{x}_{1:n} + \theta \leq \overline{\mathrm{OPT}} + \delta^2 \leq \frac{\delta}{LR} \cdot \mathrm{OPT} + \delta^2. \tag{25}$$

Hence, we can upper bound the OPT of the transformed program as follows:

$$c^\top \widehat{x} = R \cdot c^\top \overline{x}_{1:n} = \frac{RL}{\delta} \cdot \frac{\delta}{L} c^\top \overline{x}_{1:n} \leq \frac{RL}{\delta} \left(\frac{\delta}{LR} \cdot \mathrm{OPT} + \delta^2 \right) = \mathrm{OPT} + LR \cdot \delta,$$

where the first step follows by $\widehat{x} = R \cdot \overline{x}_{1:n}$, the third step follow by (25).

For the feasibility, we have that $\theta \leq \frac{\delta}{LR} \cdot \mathrm{OPT} + \delta^2 \leq 2\delta$ because $\mathrm{OPT} = \min_{Ax=b, x\geq 0} c^\top x \leq LR$. The constraint in the new polytope shows that

$$A\overline{x}_{1:n} + (\frac{1}{R} b - A 1_n)\theta = \frac{1}{R} b.$$

Rewriting it, we have $A\widehat{x} - b = (RA1_n - b)\theta$ and hence

$$\|A\widehat{x} - b\|_1 \leq \left(R \sum_{i,j} |A_{ij}| + \|b\|_1 \right) \theta \leq 2\delta \cdot \left(R \sum_{i,j} |A_{ij}| + \|b\|_1 \right).$$

□